UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

TOWARD A GENERAL SCIENCE OF SECURE PROTOCOL NETWORKS: MEASURING AND IMPROVING PRIVACY, SECURITY, AND RESILIENCE IN PLANET-SCALE DECENTRALIZED SYSTEMS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By SCOTT T. SEIDENBERGER

Norman, Oklahoma

2026

TOWARD A GENERAL SCIENCE OF SECURE PROTOCOL NETWORKS:
MEASURING AND IMPROVING PRIVACY, SECURITY, AND RESILIENCE IN PLANET-SCALE DECENTRALIZED SYSTEMS

A DISSERTATION APPROVED FOR THE
GALLOGLY COLLEGE OF ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Anindya Maiti, Chair

Dr. Sina Khanmohammadi

Dr. Richard Veras

Dr. Joseph Ripberger

Dr. Murtuza Jadliwala

The first publicly available version of ChatGPT arrived at the beginning of my PhD. It was immediately clear to me that research and science communication had crossed a threshold, and that we would not be going back. I believe artificial intelligence will continue to reshape how science is done and how it is communicated. As intelligence becomes more commoditized, the burden shifts to each of us to sharpen our critical thinking, and to scientists and researchers to curate knowledge in a way that is accessible and reliable.

Through this work, I hope to have contributed to that curation for decentralized systems and the emerging field of protocol science. I dedicate this dissertation to those who read it with the goal of learning to think in protocols, and with an appreciation for the series of consecutive miracles required for planet-scale decentralized systems to function at all.

This dissertation is further dedicated to the village that put me here. To my family, my friends, and the institutions and mentors who enabled me to pursue this PhD and complete this work, I give my deepest thanks. I try each day to live up to the responsibility that comes with the tools and opportunities I have been given.

Acknowledgments

I extend my sincere gratitude to my advisor and committee chair, Dr. Anindya Maiti, for his guidance, support, and steady encouragement throughout this journey. Our work together took us far beyond the lab—Thailand, Denmark, Vietnam, Italy, Spain, and of course Oklahoma. I am grateful for the opportunities, trust, and perspective that came with building a shared path and vision for this work together.

To my committee members—Dr. Sina Khanmohammadi, Dr. Richard Veras, Dr. Murtuza Jadliwala, and Dr. Joseph Ripberger—thank you for your thoughtful questions, careful feedback, and consistent investment in the development of this research over the years. Your insights strengthened both the work and my approach to doing it.

I am grateful to the Ethereum Foundation for supporting this research through grants and collaborations. I have deep respect for their long-term commitment to building and sustaining public infrastructure, and for taking on some of humanity's hardest coordination problems with seriousness and care.

Finally, to the Secret Service of SecretLab, thank you for being exceptional colleagues, co-authors, and friends. You made my time at OU both more engaging and a lot more fun. This work is better because I got to do it alongside you.

Contents

List Of Tables

List Of Figures

Abstract

This dissertation develops a measurement-first science of secure protocol networks. These are planet-scale, decentralized, sociotechnical systems that coordinate behavior through standardized message formats and economic incentives. Protocol networks such as Ethereum, BitTorrent, DNS, and SMTP, function as hidden critical infrastructure that powers numerous applications and communities, yet their distributed control, fragmented telemetry, and multilayer ownership leave major observability gaps in their security posture. The dissertation's central claim is that these systems cannot have robust security and resilience guarantees without first measuring the structures and dynamics that shape their risk. Three research questions organize the work: how to build credible longitudinal multi-vantage telescopes for protocol networks, which systemic mechanisms shape privacy, security, and resilience across protocol families, and which interventions remain practical, composable, and evaluable under real constraints.

To answer those questions, the dissertation develops a set of reusable empirical instruments and analysis methods. MagnetDB is a 300-week longitudinal telescope for the BitTorrent DHT that captures 28.6 million torrents and 950.6 million files. EtherBee is a multimodal Ethereum node observatory deployed across ten geographically distributed vantage points, recording 130.9 million attack events and 45.4 million P2P sessions. An email inbox sensor captured email provenance over 361 days across 150 services, and NinjaDoH provides a build–measure–evaluate loop for adversarial intervention testing in censorship-resistant DNS. These artifacts are paired with a methodological framework built around

multi-vantage observation, paired deployments, cross-layer panels, and longitudinal collection, as well as the Dataset Value Taxonomy (DVT) for reasoning about dataset maturity and stewardship.

Across the empirical chapters, four recurring mechanisms emerge. First, visibility functions as an attack surface: a paired Ethereum deployment shows that beacon nodes receive roughly three times more hostile traffic than matched controls. This is a significant risk, as 25.68% of publicly active beacon nodes on the Ethereum mainnet expose auxiliary services beyond their P2P ports. Second, de facto centralization is observed in protocol networks that are supposed to be logically decentralized. In email, the top eight autonomous systems carry 89.35% of observed email volume. Third, overlay networks inherit physical chokepoints: Ethereum's traffic-weighted peer focal points converge toward the North America–Europe corridor, leaving peripheral regions such as South America effectively tethered to those core routes. Fourth, temporal dynamics shape security outcomes. In a study of the distribution of pirated media content on BitTorrent, torrents that appear on the network before the official release date of a media title carry a fourteen-fold elevation in suspicious executables.

The dissertation then evaluates both authenticated overlay connectivity hardening and out-of-band snapshot recovery for Ethereum node operators. Using a discrete-time network simulator with 16,000 paired runs per scenario grounded in real-world incidents and stressors, the combined defense reduces median recovery time from approximately 3.4 hours to 0.6 hours in resynchronization scenarios and lowers tail-event exceedance rates from 10% to 3.0% under significant disruptions to Internet routing.

Taken together, the dissertation contributes a cross-protocol taxonomy that links node visibility, de facto centralization, physical chokepoints, and temporal tail risk. It also contributes methodological advances in telescope-building and dataset stewardship formalized through the DVT, long-horizon empirical instruments, a modeling toolkit for inference under partial observability, and practical interventions. The broader conclusion is that secure protocol networks require measuring what is otherwise hidden, identifying the mechanisms that

recur across ecosystems, and evaluating interventions against the same empirical baselines used to diagnose the problem.

Part I

Foundations: Telescopes for Protocol Networks

Chapter 1

Protocol Networks as Critical Infrastructure

1.1 Planet-scale coordination systems

Many of the most consequential systems on the Internet are not services with a single operator, but shared coordination layers that make independent systems interoperable. Sending email, resolving domain names, discovering and fetching content, and reaching agreement on a public ledger all rely on common message formats and conventions that are implemented by many parties with varying incentives. These coordination layers are easy to overlook precisely because they are successful, disappearing behind applications and APIs until they fail, are abused, or become contested infrastructure (Star and Ruhleder 1994).

In this dissertation, the term *protocol network* is used to denote a sociotechnical system in which coordination emerges from a standardized protocol that is deployed by heterogeneous actors at Internet scale. A protocol network has (at least) three defining characteristics. First, the protocol specifies an interface (message formats, state transitions, and validity rules) that allows independently operated components to compose a shared system. Second, participation is not fully mediated by a single authority. Operators join, leave, upgrade, and configure software under their own constraints, which makes the system dynamic and only partially observable from any single vantage point. Third, the protocol is inseparable from

its surrounding incentives and governance. In practice, performance trade-offs, economic rewards, operational costs, and policy pressures shape which implementations and deployment patterns become dominant.

Protocol networks are a subset of *decentralized systems*. Decentralization is a set of design commitments that can be measured. Troncoso et al. (2017) define decentralized systems as a subset of distributed systems in which multiple authorities control different components and no authority is fully trusted by all; as a result, any component must be treated as potentially adversarial. This perspective is especially useful for protocol networks, where the operational reality is an ecosystem of mutually suspicious parties. Clients do not fully trust servers, nodes do not fully trust peers, and operators do not fully trust the broader environment of counterparties and other stakeholders. Empirical studies of cryptocurrency peer-to-peer networks illustrate that decentralization is not a binary property, and that connectivity, geography, and infrastructure dependencies can exhibit significant concentration in practice (Gencer et al. 2018; Kim et al. 2018; Wang et al. 2021). Decentralization can improve privacy, integrity, and availability by distributing trust and reducing single points of control, but it also introduces new failure modes and new costs. For example, greater system complexity, weaker global coordination, and sharper trade-offs between performance, usability, and security properties can be byproducts of decentralization (Troncoso et al. 2017).

The practical question, therefore, is rarely "centralized or decentralized?" but rather *how* decentralization is realized and *where* centralization persists. A system may decentralize computation while centralizing peering discovery; decentralize storage while centralizing naming; decentralize governance in principle while concentrating operations in a small number of providers. Out-of-band incentives can also make centralization emerge even when the protocol itself appears logically decentralized, as work on Ethereum validator economics illustrates (Cortes-Goicoechea et al. 2023). Troncoso et al. (2017) frame this design space with a set of questions that also guide the dissertation. These questions include how the system is decentralized, what is gained, what is lost, what privacy or security properties are

supported, and what remains centralized. Throughout the chapters that follow, protocol networks are treated as moving targets within this multi-dimensional space rather than as fixed archetypes.

The dissertation focuses on four protocol families that exemplify planet-scale coordination under heterogeneous incentives. *Blockchains*, such as Ethereum, implement a global state machine whose correctness depends on decentralized agreement, but whose operational footprint depends on ordinary Internet infrastructure and is shaped by economic incentives (Gencer et al. 2018; Cortes-Goicoechea et al. 2023). *BitTorrent*, and its DHT-based discovery mechanisms, coordinate the supply and retrieval of content without a central catalog, yet remain sensitive to churn, enumeration, and sampling bias in measurement (Pouwelse et al. 2005; Loewenstern and Norberg 2008; Hazel and Norberg 2008; Wolchok and Halderman 2010; Wang and Kangasharju 2013; Zhang et al. 2010). *DNS*, and evolutions such as DNS-over-HTTPS, form the naming and discovery layer that many other systems assume to be "just there." Encrypted DNS variants such as DoH improve resilience to censors, but they also make visibility into this system more opaque for regulators and network operators (Hu et al. 2016; Hoffman and McManus 2018; Böttger et al. 2019; Niaki et al. 2020; Hoang et al. 2021; Master 2023). *Email* is a protocol network with decades of deployment history. It is open by design, but in practice dominated by large providers and a complex pipeline of intermediaries (Englehardt et al. 2018; Hoofnagle et al. 2012; Acquisti et al. 2015; Plice et al. 2008).

It is important to distinguish protocol networks from *platforms*. Platforms are typically characterized by a coherent administrative boundary. A single operator or tightly coupled set of operators controls the service, defines access, and can directly influence or compel certain user behavior (Van Dijck et al. 2018). Additionally, platforms often expose centralized logs, which allows for more direct and comprehensive measurement. Protocol networks, by contrast, are multi-actor, multi-layer, and only partially governed. They span different client implementations, overlay networks, and intermediaries, with critical dependencies that are

easy to miss when looking only at any one layer. Because no single entity "owns" the whole, protocol networks are often under-instrumented by default and adversarially measured in practice, which motivates the dissertation's emphasis on building measurement "telescopes" that make these coordination systems legible. The telescope designs draw on Internet-scale measurement traditions, including honeynets and darknet sensors, while also using protocol-native observability hooks (Barford et al. 2010; Fachkha and Debbabi 2015; Alata et al. 2006).

1.2 Structurally under-observed infrastructure

Protocol networks are designed as background infrastructure. Their primary interfaces are machine-to-machine rather than user-facing products, and their operation is distributed across organizations that are largely invisible to end users. Protocol networks are structurally difficult to observe because of scale, heterogeneity, distributed control, and a lack of unified vantage points. These challenges follow from the way protocol networks distribute control, split responsibility across layers, and evolve through heterogeneous deployments, and they are visible in practice in efforts to measure decentralized overlays and internet censorship at scale (Wang and Kangasharju 2013; Niaki et al. 2020).

Distributed control is a primary cause of under-observation. In a platform setting, a single administrator can understand the end-to-end behavior of a service by retaining detailed logs and enforcing consistent telemetry. In a protocol network, operational authority is dispersed across independent parties that do not share a common view, and incentives to share data are often weak. Each party observes only its local slice of the system, and local observations are rarely comparable. Globally generalizable claims therefore require combining evidence across many vantage points rather than extrapolating from a single measurement site. This multi-vantage design pattern appears in both overlay network measurement and censorship measurement studies (Zhang et al. 2010; Niaki et al. 2020).

Telemetry is also fragmented across layers. Protocol networks span client software, overlay protocols, naming and discovery mechanisms, and physical infrastructure such as AS-level routing and cloud hosting. Each layer produces different artifacts. Some are readily observable, such as protocol-level messages or peer lists, while others are private, transient, or encrypted (Hu et al. 2016; Hoffman and McManus 2018). Routing and middlebox behavior can shape which peers can connect, yet these effects are often difficult to trace (Honda et al. 2011). Discovery and rendezvous mechanisms determine who can find whom, yet the measurement surface may be intentionally limited to resist enumeration and abuse, as BitTorrent DHT measurement illustrates (Loewenstern and Norberg 2008; Wolchok and Halderman 2010; Wang and Kangasharju 2013). The signals that matter most for security and resilience are therefore distributed across layers that are instrumented by different communities with different tools.

Heterogeneity compounds these structural challenges. Client implementations differ, deployed versions drift, and configurations vary widely across operators. Even when a protocol specification is stable, the deployed system is shaped by given defaults, heuristics, and operational shortcuts. The resulting behavior can be a larger determinant of security posture than the protocol itself. A measurement strategy that assumes a single implementation or a single deployment environment can mischaracterize the system, particularly when adversaries and operators adapt in response to observed threats (Durumeric et al. 2014b; Fiebig et al. 2016; Ye et al. 2024).

These conditions yield recurring observability failures that motivate this dissertation's measurement-first stance. Two measurement failures are especially common for protocol networks. First, single-vantage measurements rarely capture a generalizable global view. Many components of protocol networks depend on geography and topology. Measurements from one region or one network can miss concentration, asymmetries, and failure modes that appear only when multiple vantages are compared. Without multi-vantage observation, global claims about system behavior are unreliable (Zhang et al. 2010; Niaki et al. 2020).

Second, snapshot measurements hide temporal dynamics such as shocks and drift. Protocol networks evolve through software upgrades and shifting incentives, and these changes reshape a system's structure over time. A one-shot crawl or short-lived snapshot can therefore mischaracterize the system by treating transient observations as stable, and by missing rare but consequential events. Longitudinal measurement makes it possible to establish baselines, detect drift, and evaluate whether interventions persist beyond the immediate aftermath of a shock (Wang and Kangasharju 2013; Niaki et al. 2020; Seidenberger and Maiti 2025a).

1.3 The observability gap as security bottleneck

This dissertation advances a claim that protocol network security is often constrained by a lack of system-level observability. The common maxim that "what cannot be measured cannot be secured" is a driver for this dissertation's research agenda and aligns with prior measurement work in both Internet-scale security monitoring and protocol network measurement (Barford et al. 2010; Wang and Kangasharju 2013).

The observability gap matters because designing for long-term security depends on reliable answers to basic questions that are difficult to obtain at scale. These questions include which components are exposed to which adversaries, which dependencies form single points of failure, whether a change in the system is local or global, and whether a mitigation actually reduces real-world risk. In protocol networks, these questions are hard because no single operator can see the full system, and the signals that determine security outcomes are distributed across technical layers and organizational boundaries. Many of these signals are sociotechnical in nature, and prior work on peer-to-peer (P2P) network measurement illustrates the need for active, multi-vantage observation to answer them credibly (Gencer et al. 2018; Kim et al. 2018; Wang and Kangasharju 2013).

This gap also creates asymmetric visibility. Many threats in protocol networks begin with low-cost reconnaissance that is cheap to perform at scale. Commodity scanning, enumeration, and inference exploit the fact that defenders typically observe only the traffic and failures that reach their own nodes, while adversaries can probe the system broadly and repeatedly from many vantages (Barford et al. 2010; Fachkha and Debbabi 2015; Imamura and Omote 2019; Seidenberger and Maiti 2025c; Chin and Omote 2021). When threat surfaces are poorly measured, defense is reactive and often focuses on the symptoms visible in local logs rather than the ecosystem-level structure that makes those symptoms repeat (Anderson 2001).

Attacks and failures concentrate in the interfaces between layers. Discovery and rendezvous mechanisms determine who can find whom, and those mechanisms can be used to enumerate participants or to bias who peers with whom, as both BitTorrent DHT crawling and Ethereum peer measurement demonstrate (Wolchok and Halderman 2010; Kim et al. 2018). Infrastructure choices determine reachability and latency, but those choices are often made for convenience or cost and can create implicit chokepoints. Recent studies of Ethereum connectivity show how performance-driven peer selection and infrastructure dependence can concentrate connectivity along a small number of corridors (Cortes-Goicoechea et al. 2023; Seidenberger and Maiti 2025a). A layer-by-layer security analysis is therefore, on its own, insufficient when the most important risks arise from cross-layer coupling (discovery $\leftrightarrow$ overlay $\leftrightarrow$ infrastructure $\leftrightarrow$ application) (Brumley and Boneh 2005).

For these reasons, this dissertation treats measurement as a prerequisite for credible security claims. Observability enables risk assessments and performance evaluation. It supports the identification of recurring mechanisms, such as visibility as attack surface, incentive-driven drift toward centralization, and co-dependencies that amplify correlated failure. It also makes it possible to evaluate interventions under real constraints, where adversaries adapt and systems change over time. As analytic tasks become cheaper and more automated, the availability and stewardship of high-value datasets increasingly determine what can be studied and secured (Seidenberger and Maiti 2025b; Maslej et al. 2025).

This dissertation introduces "telescopes" as a recurring methodological metaphor. A telescope is a measurement instrument that makes previously hidden dynamics observable at the scale and time horizon on which protocol networks operate. Each telescope is designed to (i) produce repeatable, multi-vantage observations, (ii) support model building under partial observability, and (iii) enable intervention evaluation by providing baselines and comparable metrics. The chapters that follow use these instruments to measure, reason about, and improve the security and resilience of protocol networks.

1.4 Motivating research questions

The dissertation is organized around three motivating research questions that connect the measurements taken in Part I, the mechanisms explored in Part II, and the interventions proposed in Part III. Together, these questions provide a consistent frame for evaluating protocol networks that differ in governance, incentives, and technical design.

1. **RQ1 (Measurement).** How can credible, longitudinal, multi-vantage telescopes be built for protocol networks that enable reasoning about the system as a whole?

2. **RQ2 (Mechanisms).** Which systemic mechanisms commonly shape privacy, security, and resilience across protocol families?

3. **RQ3 (Interventions).** Which interventions are practical, composable, and evaluable under real constraints?

RQ1 frames protocol-network security as an instrument-building problem. Credible answers require multi-vantage observation, longitudinal baselines, and explicit accounting of bias and drift. RQ2 frames protocol-network security as a mechanism discovery problem. The goal is to identify recurring structures that appear across protocol families. RQ3 frames protocol-network security as an intervention evaluation problem. Interventions are treated as

hypotheses that can be tested against measured outcomes rather than as purely conceptual improvements.

1.5 Threat model philosophy

This dissertation adopts an adversary model that is *ecological*. Protocol networks co-evolve with their adversaries and with the operational environments in which they are deployed (Anderson 2001). A single attacker model is not sufficient because the relevant pressures include intentional attacks, large-scale abuse, and incentive-driven dynamics that reshape the system over time (Florêncio and Herley 2012).

Three categories of adversarial pressure recur across the protocol families studied in this dissertation.

- *Ordinary adversaries* are those that engage in spam, fraud, and opportunistic exploitation. These adversaries are persistent and their effects are observable at scale through monitoring approaches such as honeynets and darknet sensors (Barford et al. 2010; Fachkha and Debbabi 2015). Incentives also sustain abuse ecosystems, including unsolicited commercial messaging (Plice et al. 2008). They provide a baseline threat environment.

- *Strategic adversaries* include censors, advanced persistent threats, and actors that can significantly influence an entire subsystem or stakeholder group. These adversaries can be selective about where and when to apply pressure, and they often induce cross-layer effects that are difficult to diagnose without a sufficiently global view (Niaki et al. 2020; Hoang et al. 2021; Master 2023; MontazeriShatoori et al. 2020).

- *Structural adversaries* are not intentionally malicious to the system. They are optimizers that pursue their own incentives, including economic profit, convenience, or performance. These adversaries concentrate control, reduce ecosystem diversity, and

create chokepoints even when no single actor intends to harm the network. Structural adversaries therefore present a class of failure modes in which the system becomes fragile as a byproduct of rational local optimization (Cortes-Goicoechea et al. 2023; Englehardt et al. 2018).

This ecological threat model anchors the rest of the dissertation. The next chapter turns from the threat environment to the measurement problem itself, showing how longitudinal, multi-vantage telescopes can make these systems observable enough to study credibly.

Chapter 2

Telescopes and Dataset Value

2.1 Measuring under constraints

Chapter 1 frames protocol networks as under-instrumented critical infrastructure. Across the protocol families studied in this dissertation, the common obstacle is that these systems are only partially observed, change over time, and are measured in environments where the act of observation itself can alter behavior. This chapter establishes the methodological backbone for the empirical claims that follow. It introduces recurring measurement constraints, design patterns for working within them, and the concept of dataset value as a way to reason about the quality and utility of collected data.

The telescopes in this dissertation are inspired by how measurement works in the physical sciences, where observation is often constrained by the environment. In astronomy, observations are limited by the physics of optics, including occlusion and atmospheric distortion (Léna et al. 2013). In oceanography, pressure and corrosion make sensors difficult to deploy and maintain, and dense sensor coverage is rarely feasible at scale (Skålvik et al. 2023). Protocol networks face their own comparable frictions. Vantage points must be deployed in the right locations, which means obtaining servers across regions and providers. Data collection pipelines must handle the heterogeneity of the system while ingesting, filtering, and storing

large volumes of data. As in other domains, maintaining longitudinality requires keeping these pipelines running stably over long horizons, often for months or years.

2.1.1 Recurring constraints

Six constraints recur across the telescopes in this dissertation.

- *Partial observability.* A measurement vantage point sees projections of the system rather than global state. A DHT crawler sees discovered identifiers rather than a complete catalog. A blockchain node sees the peers it connects to rather than the full overlay. An email inbox sees received messages rather than the full sending ecosystem.

- *Non-stationarity and drift.* Topology, traffic patterns, and infrastructure dependencies change over time. A snapshot can be directionally correct yet misleading if drift is the dominant phenomenon.

- *Heterogeneous actors.* Protocol networks are operated by parties with different incentives and constraints. Client implementations, configurations, and hosting providers differ. These differences become confounders if they are not measured and controlled where possible.

- *Ethical and governance constraints.* Measurement must minimize harm, avoid enabling abuse, and manage privacy. These constraints shape what is collected, how long the data are retained, and what data can be responsibly released.

- *Measurement reactivity.* Even observation can change the system. Joining an overlay changes its connectivity. Scanning can add load or attract attention.

- *Adversarial adaptation.* Scanners, spammers, and censors respond to being measured. In protocol networks, the act of observation can create an incentive to evade observation, which in turn creates bias if it is not explicitly scoped.

These constraints make protocol-network measurement resemble inference under a persistent missing-data process. Measurement must therefore make its blind spots explicit and must prefer longitudinal baselines and multi-vantage triangulation over single snapshots.

2.1.2 Reusable measurement design patterns

The telescopes in this dissertation reuse a small set of design patterns that can be applied across protocols.

- *Multi-vantage observation.* Measurements are collected from geographically and topologically diverse locations to provide a more complete picture of the system. Multi-vantage designs are critical for quantifying infrastructure concentration and shared failure domains, and for separating vantage-specific artifacts from ecosystem-wide structure.

- *Paired deployments and controls.* Observational studies limit the causal claims that can be made; therefore, where possible experiments should be designed to compare the behavior of the instrumented system to a control group. This is important for quantifying deviations from baseline behavior that are later used to evaluate interventions (Seidenberger and Maiti 2025c).

- *Cross-layer panels.* Telescopes are built to connect measurements from different layers of the system. For example, we link how resources are discovered (e.g., peer discovery or naming) to what happens next in the overlay (who connects to whom, how traffic flows), and we link overlay behavior to where it is actually hosted (ASNs, cloud providers, and regions). These cross-layer links are needed in later chapters to identify shared dependencies and explain how failures can propagate across layers.

- *Longitudinal collection.* Telescopes are run continuously over long periods so we can measure how the system changes over time. This makes it possible to observe drift

(slow changes in infrastructure and behavior), seasonality (recurring patterns), and rare events (bursts, outages, or attack spikes). Because harm in adversarial ecosystems is often concentrated in these rare periods, the time span of the dataset is a core part of the measurement design.

These telescope design patterns are chosen to make the dissertation's claims as generalizable as possible by explicitly addressing internal, external, and construct validity in a setting where full observability and randomized experiments are rarely available. For internal validity, paired deployments and matched controls reduce confounding from provider/region effects and background Internet noise, while stable pipeline versioning and consistent collection procedures reduce the risk that apparent differences are artifacts of instrumentation changes; when interference cannot be eliminated (e.g., scanners reacting to measurement), it is treated as part of the system and the scope of inference is stated accordingly. For external validity, multi-vantage observation and longitudinal collection reduce the chance that results are specific to a single location, provider, client mix, or time window by enabling cross-vantage replication and multi-period analyses; at the same time, the telescopes are not assumed to represent all implementations or all regions without qualification, and the limits of coverage are reported as part of the measurement. For construct validity, cross-layer panels help prevent over-interpreting any single observation by linking several different measurements together, and by making clear when a proxy is only a partial view of the concept of interest. Throughout, operational definitions are stated explicitly and, where feasible, multiple proxies are used to triangulate the underlying mechanism.

2.2 Dataset Value Taxonomy (DVT)

The telescopes in this dissertation produce datasets that are costly to build and costly to maintain, not because analysis is intrinsically hard, but because *collection* at planet-scale is constrained by where one can observe, what can be stored and transported, and what can be

released responsibly. At the same time, AI is commoditizing the cost of downstream analysis. AI can write code to analyze data, and even generate draft manuscripts and visualizations in minutes.

The "unreasonable effectiveness of data" perspective already emphasized that, across many empirical tasks, data availability can dominate algorithmic novelty (Halevy et al. 2009). In an AI-enabled research environment, this dynamic becomes more pronounced as modeling and analytic workflows are increasingly automated. The limiting factor in AI-enabled empirical research then shifts upstream to instrument-building, experimental design, and data collection (Seidenberger and Maiti 2025b; Maslej et al. 2025).

This shift also clarifies why many "big data" taxonomies of the last decade were incomplete for the problems this dissertation targets (Laney 2001; Gandomi and Haider 2015). Volume was often treated as a surrogate for quality: "more data" was implicitly equated with "better data" (Leonelli 2014). That heuristic made sense when easily collected data at reasonable scale allowed researchers to tackle the low-hanging empirical questions. However, as those opportunities have been substantially mined, the remaining questions in protocol-network security and resilience are increasingly about understanding how complex systems behave—the kinds of questions that *cannot* be answered by scale alone.

To reason about these tradeoffs explicitly, this dissertation uses the Dataset Value Taxonomy (DVT) as a supporting evaluative lens for telescope-building (Seidenberger and Maiti 2025b). The DVT provides a vocabulary for explaining why certain datasets are more enabling for research than others, and for making dataset design choices. In a context where analysis is becoming commoditized, the DVT helps formalize the claim that dataset construction and stewardship are core scientific contributions rather than mere details of implementation.

At a high level, the DVT emphasizes three value dimensions: *temporal investment*, *scale and breadth*, and *accessibility* (Seidenberger and Maiti 2025b). Temporal investment captures the extent to which value depends on sustained collection and maintenance over time. Scale

and breadth capture whether a dataset spans enough of the system—across vantages, regions, and layers—to support mechanism-level claims. Accessibility captures the degree of logistical and administrative difficulty of obtaining, storing, and releasing the data. This framing is well matched to protocol-network telescopes, where the highest-value datasets in this dissertation are those that combine long-running collection with multi-vantage coverage and carefully designed release strategies that preserve scientific utility while managing privacy and misuse risk.

2.2.1 Dataset State Space

In applying DVT to the study of decentralized protocol networks, the *dataset state space* becomes a set of design choices that determine which phenomena become observable and which claims become defensible. Four dimensions are especially consequential in this dissertation.

- *Coverage.* The set of actors, regions, and infrastructure that can be observed. Coverage determines whether infrastructure concentration and shared dependencies can be measured rather than assumed.
- *Breadth.* The number of modalities and layers captured. Greater breadth makes it possible to ask how one layer shapes another.
- *Depth.* The granularity of events and the richness of metadata. Depth determines whether analysis can separate typical behavior from tail behavior and whether it can support intervention evaluation rather than descriptive summaries.
- *Temporal span.* How long the observation window lasts. Longer spans make it possible to distinguish lasting trends from transient events.

These dimensions jointly shape the tradeoff between scientific value and governance risk. Increasing coverage, breadth, depth, or temporal span can enable new inferences, but it can also increase linkage risk and the potential for operational harm if data are mishandled or released inappropriately (Boyd and Crawford 2012; Acquisti et al. 2015).

2.3 Ethics and Risk Management

Protocol-network measurement is conducted on live, globally shared infrastructure. The ethical burden therefore extends beyond protecting a small set of participants to protecting the integrity of the system (Bailey et al. 2012). Measurement can create operational risk for system operators, it can inadvertently increase privacy risk, and it can unintentionally enable abuse if results are haphazardly disclosed. Because the networks studied in this dissertation are open and permissionless, risk management is a prerequisite concept that is discussed in the context of each telescope and dataset design.

2.3.1 Measurement in Contested Environments

A practical way to reason about measurement risk is to distinguish whether an instrument is *observing* the system or *stimulating* it. In classic Internet measurement terms, *passive observation* means logging events that we can already see "on the wire" at a vantage point that arrive as a byproduct of normal operation. Passive measurement can create sensitive data in aggregate, because long retention windows and cross-dataset joins can turn seemingly innocuous metadata into high-resolution traces that enable re-identification, profiling, and linkage across contexts.

By contrast, *active measurement* means injecting signals into the system to elicit a response and thereby reveal structure that would otherwise be unobservable. Active methods carry additional responsibilities: they can impose load, they can create new attack surfaces if poorly designed, and they can violate privacy if not carefully scoped (Barford et al. 2010).

2.3.2 Stakeholder harms and mitigation strategies

Ethical risk becomes easier to manage when it is framed in terms of who can be harmed and by what mechanism. First, *participant harms* affect the operators and services that are directly measured (e.g., node operators, relay operators, or service providers). These

harms include reputational exposure, operational disruption, and privacy risk, including cases where the collected data are "only" network-layer metadata. Second, *bystander harms* affect individuals who did not choose to participate but can be indirectly affected, such as end users. Third, *ecosystem harms* refer to second-order effects on the wider network that come from releasing measurement outputs. High-resolution disclosures can lower the cost of abuse, improve targeting for monitoring or blocking, and induce strategic adaptation in response to what is measured and shared (Bailey et al. 2012).

Mitigations in this dissertation are designed to be repeatable across systems. We prioritize data minimization (collect the least identifying signal compatible with the research question), explicit retention limits, and secure stewardship (encryption at rest and in transit, access controls, and auditable handling procedures) (Cavoukian et al. 2009). When results are disseminated, we prefer aggregated or derived artifacts over raw identifiers, and we treat the release plan as part of the threat model: the question is not only what is collected, but what an adversary could do with what is published. Where scientific evaluation requires raw traces, we favor controlled access and reproducible pipelines that allow verification of claims without broad redistribution of high-risk data.

2.4 Core Telescopes

This dissertation builds three core empirical telescopes that are reused throughout later chapters. Each telescope is described in terms of its design goal, observation plane, vantage points, main biases, and governance handling. These instruments are complemented by an adversarial evaluation artifact that is introduced in the final subsection of this section.

Table 2.1: Summary of the telescopes reused across the dissertation

Instrument	**Primary observation plane**	**Canonical instantiation**
T1 DHT discovery telescope	BitTorrent DHT	MagnetDB (Seidenberger et al. 2025b)
T2 multimodal node observatory	Blockchain overlay and underlay	EtherBee (Seidenberger and Maiti 2025a)
T3 inbox sensor telescope	Email provenance and intermediation	Inbox audit dataset (Seidenberger et al. 2025a)

2.4.1 T1 longitudinal discovery telescope for the BitTorrent DHT

The first telescope measures the supply-side of torrents in decentralized content ecosystems by measuring BitTorrent's distributed hash table (DHT). The resulting dataset, MagnetDB, comes from this telescope over a multi-year period (Seidenberger et al. 2025b). It continuously crawled the BitTorrent DHT, recorded when a torrent first became discoverable, and retrieved torrent metadata so that torrent availability timing and subsequent diffusion dynamics could be studied.

MagnetDB operates as a long-lived DHT indexer deployed from a self-hosted vantage point, configured to track and index torrents from up to 10,000 peers simultaneously (Seidenberger et al. 2025b). Over a 300-week collection period (2018-12-30 through 2024-09-29), it discovered 28.6M torrents comprising 950.6M files. After an initial burn-in backfilling period during which it discovered torrents that were already available on the network, it entered a stable, long-run discovery period and discovered new torrents as they appeared on the network (Seidenberger et al. 2025b).

Observation plane. The telescope observes the BitTorrent DHT (and the metadata exchange used to resolve torrent contents) rather than a single tracker or catalog, yielding a protocol-native view of what becomes discoverable in trackerless BitTorrent.

Vantage strategy and sampling. The instrument continuously traverses the DHT, recursively peering and aggregating torrents discovered from each peer node. Longitudinality is treated as the primary bias-reduction mechanism: rather than relying on a one-shot crawl, the system ran for years with explicit accounting for downtime and degraded periods, and with substantial bandwidth provisioning (approximately 30 TB/month) to maintain reach (Seidenberger et al. 2025b; Wolchok and Halderman 2010; Wang and Kangasharju 2013; Zhang et al. 2010).

Data processing and enrichment. After discovery, torrents were processed to extract file-level metadata, identify video files by extension, parse standardized naming conventions, and link titles to IMDb via BM25-based retrieval over a custom Elasticsearch index. The resulting dataset includes an IMDb-matched subset (about 1.56M high-confidence matched video files, corresponding to ∼751K movies and ∼811K TV episodes), enabling title-resolved analyses and linkage to external covariates such as streaming availability (Seidenberger et al. 2025b).

Biases and blind spots. DHT churn and locality imply that a single vantage point can miss torrents with extremely short lifespans or that remain confined to small, weakly connected regions of the network. Any downtime that occurs also affects the apparent first-seen times of those torrents that appear during that downtime. Private-tracker-only swarms and out-of-band distribution channels are out of scope.

Governance handling. Because torrents often serve as vehicles for illicit content distribution, MagnetDB's public release withholds direct magnet links. It provides rich metadata in

common formats (SQLite and CSV) that enable further research, but releases the full data only via authenticated request for academic research (Seidenberger et al. 2025b).

Metrics enabled. BitTorrent is a particularly informative protocol network to study because it is (i) large-scale and Internet-exposed, (ii) decentralized and permissionless, and (iii) shaped by strong, observable incentives in a multistakeholder, adversarial ecosystem. As described in the MagnetDB paper's overview of BitTorrent and the DHT, the DHT provides a protocol-native mechanism for discovering torrents as they become visible to the network, rather than relying on curated tracker or index-site views (Seidenberger et al. 2025b). This design enables event-time measurement of *supply emergence*: when a torrent first becomes discoverable, how quickly new supply accumulates, and how concentrated that supply is across producers. Because these first-appearance signals are upstream of user-visible demand, they support clean early-window indicators that can be aligned to exogenous datetimes (e.g., media release schedules and law enforcement activities). This allows for cross-layer analyses that link supply-side shocks to downstream swarm diffusion and user risk.

2.4.2 T2 multimodal node observatory for blockchain nodes

Public blockchains are, in practice, overlay networks that run atop the Internet. Blockchain networks are designed to maintain certain security and liveness guarantees to support their application layer, so nodes must be discoverable, reachable, and long-lived for the stability of the system. That same reachability, however, also makes node infrastructure an exposed and economically meaningful target. The second telescope in this dissertation therefore treats blockchain nodes as a coupled system: a protocol participant at the overlay layer and a conventional Internet host with a physical infrastructure footprint. We measure the largest proof-of-stake blockchain, Ethereum, by deploying a multimodal observatory that measures node operation, network session metadata, and honeypot interactions. Over the course of

several months, data from ten geographically distributed vantage points were collected and processed to create the EtherBee dataset (Seidenberger and Maiti 2025a).

EtherBee couples three modalities collected concurrently over a multi-month window from ten geographically distributed vantage points. First, it records fine-grained node telemetry from production Ethereum clients (consensus and execution), including health and performance signals that reflect whether a node is keeping up with the chain and maintaining stable connectivity. Second, it captures full network session metadata using packet-capture indexing, producing a protocol-agnostic view of every TCP/UDP session to and from the host, including timestamps, 5-tuples, byte counts, and selected protocol headers. Third, it logs honeypot interactions from a broad-coverage sensor suite, providing additional evidence of inbound scanning and exploit-like behavior directed at the same hosts that run the nodes. The practical consequence of this multimodal design is that events observed "on the wire" can be aligned directly with node-level outcomes (e.g., peer loss or degraded performance), rather than inferred indirectly.

Observation plane. The observatory operates at the overlay–infrastructure boundary. It observes the P2P ports and discovery surfaces that make nodes usable to the protocol, while simultaneously capturing non-P2P services and background traffic that reflect the host's general Internet exposure. This boundary view is what enables later analyses to distinguish "overlay behavior" (who a node peers with, for how long, and with what traffic intensity) from "host pressure" (what the Internet sends to that same machine simply because it is visible and attractive).

Vantage strategy and paired design. The telescope is explicitly multi-vantage: it deploys across globally distributed regions and runs continuously long enough to observe drift rather than only snapshots (Seidenberger and Maiti 2025a). Where the key variable is visibility as a node, EtherBee also uses a paired design. Within each region, an experimental host runs a mainnet node with the honeypots, while a matched control host runs the same

sensors without a blockchain node. This pairing supports estimation of the incremental risk of node participation and visibility while holding constant geography, cloud provider, and sensor configuration (Seidenberger and Maiti 2025c). In other words, it lets us separate "what happens to any reachable server in this region" from "what happens because this server is running a blockchain node."

Biases and blind spots. As with any live Internet measurement, observability is shaped by where we stand and what we can instrument. Cloud vantage points can attract more baseline probing than residential or enterprise networks, encrypted transports limit payload-level inspection, and joining an overlay can itself influence peering dynamics. EtherBee mitigates these limits by holding configurations stable, using geographically diverse replication, and leveraging the regional control hosts to identify patterns that persist beyond local background noise. These choices do not eliminate bias, but they make it explicit and measurable, which is critical for later claims about targeting and topology drift.

Governance handling. Because node observatories can expose operator practices and operational vulnerabilities, EtherBee adopts a conservative release stance (Seidenberger and Maiti 2025a). Analyses are designed to work on derived signals and aggregate distributions whenever possible, and sensitive identifiers are redacted or anonymized in shared artifacts. Where raw session metadata is necessary for scientific evaluation, it is treated as restricted and handled under strict retention and access controls, with the goal of enabling verification without turning the dataset into a resource that could be used by would-be adversaries.

Metrics enabled. This dataset design supports metrics that appear repeatedly throughout the dissertation. It enables "visibility premium" estimates that quantify how much additional hostile activity is associated with running a node relative to a matched control (Seidenberger and Maiti 2025c). It supports peer churn and tenure measurements grounded in full-session metadata, and it enables cross-layer concentration and drift metrics that map

overlay dependence onto infrastructure corridors (e.g., AS-level and geography-weighted connectivity). These outputs are reusable ideas for later chapters on exposure, incentives, drift, chokepoints, and cascading failure mechanisms.

2.4.3 T3 inbox sensor telescope for email ecosystems

The third telescope uses inboxes as empirical sensors for privacy lineage and intermediation in email ecosystems. Email is a global protocol network that appears decentralized at the application layer but is heavily mediated by a concentrated set of infrastructure providers and marketing intermediaries. The inbox telescope measures this intermediation by collecting and classifying inbound mail for accounts created during controlled sign-up workflows (Seidenberger et al. 2025a).

The canonical instantiation of this telescope measures email received after sign-ups to 150 widely used apps and services and collects longitudinal evidence about sending infrastructure, frequency, and delegated sending authority (Seidenberger et al. 2025a).

Observation plane. The telescope focuses on message provenance and metadata surfaces that are visible in headers and associated infrastructure signals. The goal is to measure how apps and services delegate their email sending authority to intermediaries and identify any temporal patterns.

Classification and linkage. The inbox telescope classifies sources into internal senders, authorized third parties, and unknown third parties (Seidenberger et al. 2025a). This taxonomy supports ecosystem-level concentration analyses and enables measurement of how delegated custody expands the attack and privacy surface beyond the originating service.

Biases and blind spots. The inbox observes only what arrives. It cannot fully attribute intent, it does not provide a complete tracking graph, and it cannot observe mail that is blocked or filtered upstream. For these reasons, the telescope focuses on evidence present in

headers and on conservative claims about provenance and intermediation rather than claims about user intent.

Governance handling. Email content can be highly sensitive. The telescope emphasizes minimization by focusing on metadata and by redacting or excluding personal content where possible. Release strategies favor derived counts, concentration metrics, and de-identified provenance summaries.

Metrics enabled. The inbox telescope supports heavy-tail and concentration metrics for sender ecosystems, measures of delegated custody surfaces, and temporal burst signatures that can be compared across services and time windows. These metrics feed later chapters on visibility and de facto centralization through intermediaries.

2.4.4 Evaluation artifacts and design measurement loops

In addition to telescopes that observe existing protocol networks, this dissertation uses a measurement-first intervention artifact that is designed to measure its own properties under adversarial evaluation. This distinction matters because a telescope aims to observe an ecosystem with minimal disturbance, while an intervention artifact is designed to change the ecosystem or to change an end-user security property.

NinjaDoH is the canonical example of a design measurement loop in this dissertation (Seidenberger et al. 2026a). It is a moving-target DNS-over-HTTPS protocol intended to raise the cost of censorship by continually changing rendezvous identifiers through a combination of IPNS and public cloud infrastructure. The evaluation focus is not a single performance metric. It includes latency overhead, financial cost, and detectability against commercial firewall products and learning-based detectors. As a result, NinjaDoH motivates the intervention evaluation stance later adopted in Chapter 8. Measurements are used to evaluate practical feasibility under adversarial pressure rather than to claim a static security property in the abstract.

2.5 Scoping Claims

Measuring live protocol networks always involves partial observability. A telescope does not reveal the full system. It provides a well-defined view from a particular vantage point, collected under specific operating conditions. The goal is not to claim complete visibility, but to state clearly what can be learned from each measurement design and what cannot.

For that reason, this dissertation separates three kinds of claims, each with a different standard of support. Descriptive claims summarize what is directly observed, including distributions, concentrations, and temporal patterns in the data. These are the baseline outputs of the telescopes. Comparative claims ask whether two conditions differ. For example, one region versus another or one time period versus another. These claims require careful matching, sensitivity analysis, and an explicit discussion of what differences might still be explained by unmeasured factors. Mechanistic claims go one step further and propose a plausible pathway that connects observations to more universal truths. Because the full system cannot be observed, mechanistic claims are presented as evidence-based explanations supported by multiple measurements.

These limitations also shape how we think about representativeness. What we observe depends on where the telescope is placed and how often it samples. A vantage point can over-represent some populations and under-represent others. A sampling schedule can miss short-lived but important events. Some behaviors can be hidden or can change in response to being measured. Throughout the dissertation, these risks are addressed by using multiple vantage points where possible, combining complementary measurement modalities, checking whether findings hold across different networks, and keeping a clear distinction between descriptive measurements and causal explanations.

With these constraints in view, Chapter 3 introduces the modeling language used throughout the dissertation to draw inferences from incomplete observations. Recurring quantitative

motifs provide a common set of language and tools for moving from raw outputs to discovering common mechanisms affecting security and privacy across the studied networks, and later to evaluating interventions.

Chapter 3

From Measurement to Modeling

Chapter 2 introduced several telescopes as the instruments that enable empirical security claims in protocol networks. The telescopes produce longitudinal, multi-vantage datasets that support claims about the system as a whole. This chapter establishes the modeling language used throughout the remainder of this dissertation to convert these incomplete observations into decision-relevant signals. The goal here is not to provide a single unified model of every protocol network studied. The goal is to introduce a small set of quantitative motifs that recur across the case studies and interventions.

The chapter is organized into three sections. Section 3.1 describes a standardized way of modeling and understanding protocol networks, situating it within relevant measurement and complex systems literature. Section 3.2 introduces a set of core quantitative motifs that recur when studying privacy, security, and resilience in protocol networks. Section 3.3 shows how those motifs, together with cross-layer linkage, credible baselines, and stress-testing, support mechanism claims under partial observability.

3.1 Modeling Protocol Networks

Distributed systems differ in how observable they are, largely because control varies across architectures. In centralized distributed systems, a single operator can often collect logs and model the system comprehensively. In decentralized systems, control is distributed among

multiple actors, so no one actor has complete visibility. Protocol networks are a subset of these decentralized systems: stakeholders build and operate components collaboratively, based on incentives and constraints, while following a shared protocol.

This makes protocol networks sociotechnical ecosystems whose behavior is only partially visible from any given vantage point, whose structure changes as incentives shape the behavior of stakeholders, and whose security outcomes are often driven by tail events. These conditions align protocol-network measurement with long-standing traditions in Internet measurement, where inference is made from incomplete and biased samples collected on live infrastructure (Paxson 1997; Barford et al. 2010; Fachkha and Debbabi 2015; Wang and Kangasharju 2013; Niaki et al. 2020). The modeling stance adopted throughout this dissertation treats models as tools for designing interventions that improve some aspect of the security or resilience of the system rather than as perfect reconstructions of the system. This stance matches the operational reality described in Chapter 1. No single actor owns a protocol network end-to-end, and key system dynamics often emerge at the interfaces between layers. Therefore, we begin by identifying recurring quantitative motifs that have been found to be useful in studying these systems.

3.1.1 Empirical Regularities in Data-Generating Processes

Across Internet measurement and complex-systems research, decentralized networks are rarely well captured by simple, stationary statistical models. Instead, the data typically come from generative processes with a few recurring features. These features include heavy-tailed distributions, temporally clustered activity, time-varying participation and topology, and dependence between nodes that can amplify shocks to the system. Practically, this means that "typical" behavior is often not represented by averages because variability can be dominated by rare events, and the effective data-generating process can change over time.

- *Heavy tails.* Many Internet-scale networks exhibit strongly skewed distributions of activity and connectivity. A small fraction of nodes can contribute a disproportionate

share of messages, flows, peers, or content. In these settings, means and variances can be unstable or unrepresentative, and careful reporting (e.g., quantiles, tail rates, concentration measures) is often more informative than single "average" summaries (Crovella and Bestavros 1997; Clauset et al. 2009; Newman 2010).

- *Burstiness.* Network activity commonly arrives in clusters rather than at a steady rate. Long quiet periods are punctuated by short, high-intensity bursts that account for much of the volume or harm (e.g., scanning spikes, censorship events, release-driven surges). Bursts also appear as step changes associated with upgrades, interventions, or outages, which violates constant-rate assumptions (Chandola et al. 2009).

- *Churn.* Many protocol networks include large populations of short-lived participants, ephemeral sessions, and unstable peer sets. Churn complicates both measurement and modeling, since the observed topology is a time-averaged projection of an underlying process that is evolving during measurement. This makes estimates sensitive to the sampling window and complicates inference about long-run structure (Wang and Kangasharju 2013; Loewenstern and Norberg 2008).

- *Correlated shocks and failure modes.* Failures and attacks are frequently dependent rather than independent. Shared software implementations, shared hosting providers, common routing corridors, or common governance pressures can create systemic risk across nodes. This dependence increases the probability of cascades even when nodes are individually redundant (Buldyrev et al. 2010).

- *Feedback loops.* A feedback loop is a recurring pattern where the network's current state shapes how the graph evolves, and those graph changes then reshape the network's future state. In decentralized systems, this typically occurs because discovery and peer-selection mechanisms use observed signals (e.g., latency, availability, or recent responsiveness) to form and retain edges. For example, if clients preferentially keep connections to peers that respond quickly and remain online, those peers tend

to accumulate more inbound edges and carry more traffic, which further increases their visibility in discovery and reinforces their centrality. Conversely, peer scoring algorithms can provide negative feedback by downgrading poorly performing links and periodically reshuffling connections, which can redistribute load and partially stabilize an evolving graph. Either way, feedback implies a nonstationary data-generating process in which connectivity and performance distributions can shift endogenously over time (Cortes-Goicoechea et al. 2023).

These regularities motivate modeling choices that are robust to skew, nonstationarity, and dependence. In practice, this means emphasizing distribution-aware summaries (e.g., quantiles and concentration measures) rather than relying on means, using time-indexed analyses that can detect bursts and regime shifts, and explicitly linking observations across layers so that changes in one plane can be interpreted in terms of another. They also motivate a focus on tail risk, because many security failures are driven by rare, high-impact episodes rather than by typical conditions.

3.1.2 Inference under partial observability

Decentralized protocol networks do not offer a single, authoritative view of their own state. Participation is distributed across independent actors, much of the activity is not globally broadcast, and what any observer can see depends on where they stand and what the protocol reveals. The measurement instruments used in this dissertation (the "telescopes") therefore record *partial views* of a system, not a complete snapshot of the full network.

This shifts the modeling task. The central problem is not classical parameter estimation under full observability. It is inference in the presence of *persistent missing data*. Moreover, the missingness is often *structural* (Little and Rubin 2019). It is induced by design and governance choices (who controls what), by encryption and privacy mechanisms (what is intentionally hidden), and by measurement constraints (how many vantage points can be deployed, and where). The approach in this dissertation treats model construction as a

disciplined way to connect what is observable to the decision or question of interest, while keeping the limits of visibility explicit. Several recurring principles guide that process.

Begin with an explicit observation model. Each analysis starts by stating what the telescope can observe, what it cannot, and what selection effects are built into the measurement process. This includes the scope of coverage (which parts of the network are reachable), the unit of observation (nodes, sessions, messages, routes), and the mechanisms that occlude its visibility (access controls, encryption, rate limits, and vantage point placement). Making these constraints explicit helps ensure that later claims are aligned with the data that could plausibly support them.

Prefer quantities that remain meaningful under partial visibility. When coverage is incomplete, the most informative targets are often *relative* or *distribution-aware* quantities rather than absolute totals. Later chapters therefore emphasize statistics that are robust to sampling bias, such as concentration and inequality measures, ranks and quantiles, ratios, and deviations from local baselines. Paired comparisons and within-vantage contrasts are used when unmeasured heterogeneity across factors is likely to dominate. The goal is to focus on invariants of the data-generating process that can be estimated credibly from projections.

Use proxies deliberately and triangulate when possible. Key variables in protocol networks are frequently latent or only indirectly observable, so analyses often rely on proxies. Throughout the dissertation, a proxy is treated as an instrument with a stated scope and known failure modes, not as a substitute for ground truth. When more than one proxy is available, the preferred strategy is triangulation. We ask whether independent measurements, with different biases, point to the same qualitative conclusion. Disagreement across proxies is treated as evidence about the limits of observability.

Match claims to the strength of identification. Because many analyses are observational, it is important to separate descriptive observations from causal claims. Models are interpreted as characterizing conditional associations rather than identifying structural causal effects. Causal language is reserved for settings with a credible identification story, and otherwise the emphasis is on transparent, decision-relevant characterization of relationships in the observed data.

Treat adaptation to measurement as part of the system. Decentralized networks can react to being observed. Adversaries may change tactics, operators may alter configurations, and client software may update defaults in response to public measurement. Rather than treating this reactivity as noise, we treat it as part of the data-generating process. Shifts that occur after measurement changes can be informative signals about the effects of incentives and defenses, and help interpret apparent nonstationarity in the observed series. Taken together, these principles provide a consistent way to reason from partial observations to defensible conclusions about decentralized protocol networks, while keeping the boundaries of what the data can support clear.

3.1.3 Graphs and Time Series

The motifs reused throughout this dissertation rely on two complementary mathematical lenses. The first lens models protocol networks as graphs. The second lens models system behavior as time series. These lenses are coupled views of the same system.

Graph representations. A natural way to reason about and model protocol networks is through graphs. Nodes are actors or resources and edges are relationships such as peering, communication, delegation, or shared infrastructure dependence. Graph theory provides a vocabulary for describing topology and concentration, including degree distributions, clustering, path lengths, core structure, and centrality measures (Newman 2010). In protocol

network security, graph metrics become directly interpretable as risk metrics because they quantify chokepoints, exposure surfaces, and the size of reachable sets under failure or attack.

Graph representations also support the analysis of layered infrastructure dependence, and in many cases a *hypergraph* is the more natural object. Protocol networks are overlays on top of the Internet, and the Internet itself can be thought of as a network atop physical infrastructure networks such as fiber lines and power grids. These dependencies are not purely pairwise. A single provider, region, routing corridor, or physical site can simultaneously support *many* protocol nodes. Hypergraphs can represent protocol nodes as vertices and shared dependencies as hyperedges that connect sets of nodes across layers. This view provides a compact way to reason about nested failure domains and correlated shocks, and it can be projected back into derived graphs (e.g., node–node adjacency by shared dependencies) to quantify how apparent decentralization at the protocol layer can coexist with concentration and cascade risk in hosting, routing, geography, or physical infrastructure (Rinaldi et al. 2001).

Time series representations. Many security and resilience questions are fundamentally temporal. Attacks arrive in asynchronous bursts, nodes churn, and exogenous events affect the incentives that shape stakeholder behavior on a network. Therefore, security and resilience interventions should be evaluated against baselines. Time series analysis provides a vocabulary for separating baseline behavior from trend, seasonality, and transient shocks (Chandola et al. 2009). In this dissertation, time series methods are used primarily for three purposes.

First, they establish baselines that enable deviation-based metrics, which are essential when absolute levels can vary significantly across vantage points. Second, they allow for measuring drift, which is essential when a single snapshot view can mischaracterize an evolving ecosystem (especially if this evolution is slow). Third, by treating key metrics as time series, we can quantify the system's dynamic response to stressors, including the magnitude and

duration of departures from baseline and the rate at which the system returns toward its pre-perturbation regime, enabling resilience metrics that can be compared across interventions and scenarios.

A pragmatic stance is often necessary when working with time series from decentralized networks. These data are frequently noisy, heavy-tailed, and nonstationary, so classical assumptions behind purely parametric forecasting can be brittle. We therefore rely on robust summaries such as medians and quantiles, event-aligned windows, and change-focused metrics to characterize dynamics in a way that remains interpretable under bursts, churn, and idiosyncratic regimes.

3.1.4 Safety and Liveness

Protocol networks are often evaluated using a mix of privacy, security, and resilience objectives. For systems that maintain shared state or coordinate across mutually distrusting counterparties, two classic distributed systems concepts provide a unifying vocabulary. *Safety* captures the requirement that bad states do not occur, whereas *Liveness* captures the requirement that progress toward the agreed-upon state eventually occurs (Alpern and Schneider 1985). These notions are most explicit in consensus-based systems, particularly blockchain networks, where safety and liveness map directly onto the correctness and availability of the chain. However, the safety–liveness lens generalizes beyond blockchains.

Not all decentralized protocols aim to maintain a single, total order or a single global state. Many systems are deliberately multi-view and are not global state machines. Different participants may hold different subsets of information, observe events in different orders, or cache different versions, and the protocol may promise correctness only with respect to a particular operation (e.g., resolve a name, deliver a message, retrieve a file) rather than convergence to one shared state. In these settings, safety and liveness are best understood as properties of the protocol's *interface semantics*. Safety specifies which incorrect outcomes must be ruled out for the operation, while liveness specifies what progress the operation

should make, and under what assumptions, even under churn, partitions, and adversarial conditions.

This framing makes the concepts comparable across protocol families. For naming systems, safety corresponds to correct and authentic resolution, while liveness corresponds to the ability to resolve names despite failures or censorship. For email, safety includes authenticity and integrity properties of message provenance, while liveness corresponds to deliverability and timely receipt. For decentralized content distribution, safety includes resistance to poisoning and misattribution, while liveness corresponds to the ability to discover and retrieve content when availability is intermittent.

Safety and liveness are also a useful way to reason about design tradeoffs. Protocols that aim for stronger safety guarantees typically impose stricter rules on when outcomes are accepted, often requiring more coordination or stricter validation. These choices can reduce throughput, increase end-to-end latency, or sacrifice availability under partitions or adversarial conditions. Protocols that prioritize liveness broaden the conditions under which the system can continue to process and complete client operations, which can improve availability, but may tolerate inconsistency across views, provide stale information, or lead to outcomes that later require reconciliation. In practice, protocol design is largely about where to place this boundary and what guarantees can be maintained under realistic operational and threat assumptions.

Operationally, these choices show up in measurable system behavior, especially under stress. Stronger safety tends to manifest as lower rates of incorrect outcomes, but it can also manifest as slower progress or delayed completion during disruptions. Stronger liveness tends to manifest as continued service during disruptions, but with a greater reliance on mechanisms that detect and correct inconsistencies, retry or reroute around failures, and converge back toward normal operation once conditions improve.

3.2 Core quantitative motifs

Building on the background and modeling principles introduced in Section 3.1, this section presents a set of quantitative motifs that recur across the decentralized protocol families studied in this work. Each motif names common empirical patterns and measurements for them.

The motifs are not specific to any one dataset. They are reused across MagnetDB (Seidenberger et al. 2025b), EtherBee (Seidenberger and Maiti 2025a), the email inbox audit dataset (Seidenberger et al. 2025a), and intervention evaluations such as NinjaDoH (Seidenberger et al. 2026a). The motifs also appear in the empirical modeling of piracy timing and volume in decentralized discovery ecosystems (Seidenberger et al. 2026b) and in the modeling and simulation of blockchain resilience under correlated shocks and recovery interventions (Seidenberger et al. 2026c).

3.2.1 Heavy tails and concentration

Heavy-tailed distributions are widely prevalent in Internet-scale systems. Network traffic, file sizes, node degrees, and many security signals exhibit large variance, rare extreme values, and long upper tails (Crovella and Bestavros 1997; Clauset et al. 2009). In protocol networks, heavy tails are often evidence of structural centralization and concentrated attack surfaces.

Heavy-tail analysis serves two complementary roles in understanding decentralized systems. First, it provides a descriptive lens on inequality, showing how strongly activity or connectivity is concentrated across actors. Second, it supports mechanistic explanations by linking concentration to fragility. When participation or load is concentrated, the system's effective redundancy is smaller and shared failure domains are larger, which increases exposure to chokepoints and amplifies the impact of targeted attacks or correlated outages.

Top-k shares and Pareto effects. A basic concentration signal is the fraction of activity accounted for by the top k entities. Protocol ecosystems often exhibit Pareto-like structure,

where a small share of entities account for a large share of outcomes. Top-k metrics are especially useful when absolute totals are biased by sampling, because relative shares can remain stable under incomplete coverage when bias is not strongly correlated with rank.

Lorenz curves and Gini coefficients. The Lorenz curve $L(p)$ describes the cumulative share of activity accounted for by the bottom p fraction of entities, after sorting by activity. The Gini coefficient summarizes this inequality as a scalar in $[0, 1]$, where larger values indicate greater concentration. A discrete form used throughout the dissertation is:

$$G = \frac{1}{2n^2\mu} \sum_{i=1}^{n} \sum_{j=1}^{n} \left| x_i - x_j \right|$$

where x_i is activity for entity i, n is the number of entities, and μ is the mean activity. Gini-style summaries are repeatedly used later to quantify the concentration of large infrastructure intermediaries, to separate targeted attack concentration from diffuse background noise, and to characterize the prevalence of high-degree "super-nodes" in torrent ecosystems.

Concentration indices. When the unit of analysis is an infrastructure grouping such as an ASN or hosting provider, concentration can be interpreted as the extent to which activity depends on a small number of intermediaries, and therefore as a proxy for shared failure domain size. A common scalar summary is the Herfindahl–Hirschman Index (HHI), defined as the sum of squared shares across groups, $H = \sum_i s_i^2$. The index is small when activity is spread across many groups and large when a few groups dominate, because squaring emphasizes large shares.

Complementary tools compare *entire distributions* rather than collapsing them to a single number. KL divergence measures how different one distribution is from a reference distribution, with larger values indicating greater mismatch and an interpretation as "excess surprise" when modeling one distribution as if it were the other. Earth Mover's Distance measures how much probability mass must be moved, and how far, to transform one distribution into

the other, which is useful when the distributions have inherent geometry (e.g., geographic regions). These indices are useful for quantifying dependence on large infrastructure intermediaries, comparing distributions of inbound network attacks, and when comparing overlay connectivity patterns (who connects to whom at the protocol layer) to the concentration of the underlying infrastructure those connections rely on (which ASNs, providers, regions, or routing corridors host and carry them).

Heavy tails also change how results should be summarized. When a small number of extreme values dominate, averages can be misleading and measures of spread can be driven almost entirely by rare events, so they do not reflect typical behavior. Simple parametric fits can also fail when the observed distribution is produced by multiple underlying mechanisms. A more reliable approach is to report quantiles and tail-focused measures and to treat the tail itself as a central feature of the system rather than as noise to be smoothed away.

3.2.2 Timing and hazards

Many security questions in protocol networks can be posed as time-to-event problems. Examples include time to first appearance of a resource, time to first compromise attempt, time to disconnection from a peer, time to fault detection, or time to recovery after a shock. These problems fit naturally into survival analysis, where the key object is the hazard function, which captures the instantaneous risk of event occurrence conditional on survival up to time t (Cox 1972).

Survival analysis is particularly useful in protocol networks because observations are often *incomplete in time*. A measurement vantage typically covers only a bounded window, so events that do not occur before the window ends are treated as *right-censored* (we only know they did not happen *yet*, not that they never happen). *Delayed entry* also arises when an entity existed before observation began but is only observable, or only becomes at risk, after measurement starts.

A useful model in survival analyses is the Cox proportional hazards model:

$$h(t \mid X) = h_0(t) \exp\Big(\beta^\top X\Big)$$

where $h_0(t)$ is an unspecified baseline hazard and X is a vector of covariates. The Cox framework is useful because the baseline hazard in open ecosystems is rarely known a priori and can take complex shapes over time. The model yields hazard ratios that are interpretable as multiplicative changes in risk associated with covariates, holding other factors constant.

In practice, hazard models support two recurring uses. First, they help identify *early-warning windows* when risk is front-loaded. The period immediately after an exogenous event, such as a configuration change or a visibility event, can have disproportionate risk, and hazard models quantify how covariates shift the timing of onset. Second, they provide a principled way to study *tenure and churn dynamics*. Peer lifetimes and session durations can be modeled as survival processes, compared across regions, client types, or policy regimes, and related to covariates such as geography, ASN, or observed traffic intensity. These timing differences matter operationally because churn and short peer tenures can make the network less stable.

Hazard ratios should be interpreted cautiously in observational settings. When covariates are influenced by the same incentives or strategic choices that also affect outcomes, the estimated effects are treated as conditional associations rather than as causal effects, since assignment is not controlled.

3.2.3 Diffusion dynamics

Diffusion is a key way by which information flows through decentralized systems. Information is spread through local interactions rather than through a single coordinating authority. In practice, this includes the propagation of new client software versions, routing information, as well as adversarial techniques and mitigations. Some of the original academic literature on diffusion mechanics comes from epidemiology, which studies diffusion as a population process driven by exposure and social network structure (Rogers 2003; Bass 1969; Coleman

et al. 1966). In distributed systems and network science, related models formalize diffusion as contagion, gossip, or cascade dynamics on a graph (Kermack and McKendrick 1927; Pastor-Satorras and Vespignani 2001; Newman 2002; Granovetter 1978; Watts 2002).

Across many settings, diffusion is often summarized by an S-shaped cumulative adoption pattern. Adoption begins slowly when awareness is limited, accelerates as exposure increases, and then slows as the reachable population saturates. This empirical regularity motivates compact growth models that make diffusion comparable across systems and interventions, even when the underlying mechanisms are heterogeneous.

Logistic growth. A logistic curve is a simple way to describe adoption that starts slowly, speeds up, and then levels off as it approaches a maximum. It models the cumulative number of adopters $N(t)$ as:

$$N(t) = \frac{K}{1 + \exp(-r\,(t - t_0))}$$

where K is a carrying capacity, r is a growth rate, and t_0 is an inflection time. Logistic fits are commonly used to summarize and compare how quickly and how strongly adoption happens across settings, without claiming the model captures the true underlying mechanism (Rogers 2003).

Bass diffusion. The Bass model decomposes adoption into an exogenous component (innovation) and an endogenous component (imitation), often written as:

$$\frac{dN(t)}{dt} = \left(p + q\frac{N(t)}{K}\right)(K - N(t))\,,$$

where p reflects an innovation rate and q reflects imitation (Bass 1969). This decomposition is useful as a conceptual bridge between diffusion driven by external triggers (e.g., coordinated announcements to take a certain collective action or protocol changes) and diffusion driven by within-network reinforcement (e.g., herd behavior). It is useful for separating diffusion that looks mostly externally driven from diffusion that looks mostly driven by within-network

feedback, but in open systems these parameters are usually treated as descriptive signals unless there is stronger evidence for causality (Rogers 2003).

Diffusion on networks. When diffusion depends on who interacts with whom, the network's structure shapes how quickly and how far something spreads. Epidemic-style models treat diffusion as repeated exposure and transmission along edges (Kermack and McKendrick 1927; Pastor-Satorras and Vespignani 2001; Newman 2002). Threshold and cascade models instead assume adoption often requires reinforcement, coordination, or compatibility, which helps explain why some shocks fade out while others trigger large cascades (Granovetter 1978; Watts 2002). These perspectives matter in decentralized systems because hubs, clustering, and community structure can speed diffusion within parts of the network while still leaving bottlenecks and fragmented reachability across the whole system.

Event-time analysis. Many diffusion processes are also organized around discrete external shocks, such as releases, policy changes, outages, takedowns, or enforcement actions. Event-time analysis aligns observations to a shared event time and compares trajectories across groups or contexts, which separates event-driven dynamics from noisy calendar-time variation (Chandola et al. 2009). This alignment is often necessary in decentralized environments because adoption can be bursty, geographically staggered, and mediated by heterogeneous discovery and caching paths.

Appearance then adoption. A recurring practical distinction is between the *appearance* of a resource (when it becomes visible to measurement or discoverable in the ecosystem) and its *downstream adoption* (how widely it is fetched, used, replicated, or referenced). The two can have different drivers and time scales. For example, a resource can become visible quickly but diffuse slowly if retrieval is costly or trust is low, or it can remain hidden for long periods and then spread rapidly once a discovery channel amplifies it. Treating these as

separate stages helps interpret diffusion curves and prevents conflating visibility mechanisms with adoption mechanisms.

Diffusion models can support many different types of analysis, so it is important to be clear on how they are being used in a given context. They can summarize patterns in timing and saturation, support comparisons in speed or reach across settings, or serve as hypotheses about whether a process is driven mainly by external triggers (exogenous) or by within-network reinforcement (endogenous). In observational settings, they are typically strongest as descriptive and comparative tools, while mechanistic interpretations are treated as suggestive unless supported by stronger designs or triangulation across measurements (Rogers 2003; Watts 2002).

3.2.4 Churn and tenure

Churn and tenure are the temporal complements to the heavy-tail and diffusion motifs: they describe *who is present when*, *for how long*, and *how stable the effective topology is* over the measurement window. Because protocol networks are only partially observable, these quantities must be defined *relative to an observation rule* (what counts as "present") and a *time window* (the resolution at which entry/exit is detected).

Definitions. We use the terms as follows, in line with prior P2P and Internet measurement studies (Stutzbach and Rejaie 2006; Wang and Kangasharju 2013):

- *Tenure (lifetime).* The tenure of an entity is the elapsed time it remains continuously *observable* under a stated criterion. The entity can be a session, a peer, a sender domain, or a resource (e.g., a torrent). Tenure is therefore an operational definition: it depends on the detection rule (heartbeat, traffic threshold, successful handshake, first/last seen in a crawl) and can differ from the entity's true lifetime.

- *Churn (turnover process).* Churn is the process by which the set of observable entities changes over time due to entry, exit, and intermittency. Churn can be defined at

multiple levels: *population churn* (entities enter/exit the observed set), *edge churn* (relationships change), and *session churn* (connections begin/end).

Two clarifications matter for later chapters. First, churn is not synonymous with randomness: it can be driven by diurnal usage (Stutzbach and Rejaie 2006; Wang and Kangasharju 2013), client update cycles, load balancing, censorship responses, or adversarial pressure. Second, churn is not automatically "good" or "bad." In some settings, churn improves security by making sustained targeting harder (a moving-target effect); in others, it can *hurt* security and resilience by reducing stability, because links do not persist long enough for routing, replication, or coordination to take hold.

Session churn and session tenure. At the lowest level, churn appears as session initiation and termination. Given a sessionized record (e.g., from flow/session metadata), we define a session tenure random variable T as the duration between a session's start and end time. Tenure distributions are typically heavy-tailed and are most interpretable via survival summaries (e.g., medians, upper quantiles, and hazard-of-disconnect curves) rather than means. In EtherBee, for example, peer-session lifetimes are a direct stability signal for the overlay: bursty disconnections and short tenures are observable precursors to degraded peer counts and reduced validator-relevant performance proxies under stress (Seidenberger and Maiti 2025a; Seidenberger et al. 2026c).

Population churn and set stability. At the population level, churn appears as entry/exit of unique entities such as peers observed in a peer table, senders observed in an email inbox, or content identifiers (e.g., torrent infohashes or IPFS CIDs) seen in discovery crawl logs. Let V_t denote the set of entities observed in window t. A minimal stability statistic is the overlap between consecutive windows, for example the Jaccard similarity:

$$J_t = \frac{|V_t \cap V_{t-1}|}{|V_t \cup V_{t-1}|}, \quad \text{with turnover } \tau_t = 1 - J_t. \tag{3.1}$$

These overlap-based statistics are robust under partial observability because they do not require estimating the absolute population size; they quantify *relative* stability given the vantage and windowing rule.

When risk is concentrated in the top of a distribution (heavy tails), churn should also be measured *in rank space*. A practical metric reused later is turnover in the top-k set:

$$\text{Turnover}_t^{(k)} = 1 - \frac{|\text{Top}_k(t) \cap \text{Top}_k(t-1)|}{k}, \tag{3.2}$$

which distinguishes ecosystems where the apparent leaders are stable (structural concentration) from those where leadership is ephemeral (rotating attention or load).

Churn becomes especially diagnostic when it is *endogenously induced* in response to stressors or protocol changes. In that case, churn is not noise; it is part of the failure pathway. In the blockchain case studies presented later, bursts of hostile traffic can trigger peer loss and reconnection attempts that compound resource load, producing a feedback loop between exposure and stability (Seidenberger et al. 2026c). In decentralized discovery ecosystems, short tenures and rapid swarm turnover create narrow, high-risk windows where early participants face elevated security risk and where first-mover dynamics dominate diffusion (Seidenberger et al. 2026b, 2025b). More broadly, churn and tenure make "a graph" an evolving object rather than a static one, and they mediate interpretability: many security and resilience outcomes depend less on the *existence* of redundancy and more on the *persistence* of that redundancy under stress. In later chapters, churn metrics therefore appear both descriptively (how stable is the system in baseline operation?) and mechanistically (does an exposure burst measurably shorten peer tenures or increase top-k turnover?).

3.2.5 Correlated failures and cascades

Many of the most consequential failures in protocol networks are systemic because they are *dependent*. Protocol networks rarely fail one node at a time. Instead, failures tend to be

linked, so disruption in one place makes disruption elsewhere more likely. This subsection tightens terminology around dependence and systemic risk, clarifying the distinction between *correlated failures* and *cascades*, which are related but not interchangeable concepts.

In dependable and distributed systems research, a *fault* is a cause, an *error* is an incorrect internal state, and a *failure* is an externally visible breakdown in the service the system is meant to provide (Avizienis et al. 2004). In protocol networks, we use *shock* to denote an exogenous perturbation (e.g., a routing event, client bug, outage, or attack burst) that may induce faults and failures across many entities at once.

Correlated failures (or "common-cause" or "fate-sharing" failures) occur when multiple components fail with statistical dependence because they share a cause or dependency (Avizienis et al. 2004). Section 3.1.3 presented a useful mental model of thinking in terms of an *overlay hypergraph* of shared dependencies, where each node participates in multiple "shared-fate" groups defined by common software implementations, common cloud providers or regions, common routing corridors, common governance constraints, or common physical infrastructure. When a shared dependency experiences a fault or degradation, many nodes incident to that same dependency can enter error states or fail in the same window, creating dependence in outcomes even if components do not directly cause failures to propagate to one another through node-to-node interaction. The key feature is that failures are not independent across nodes, even without any failure propagation between nodes.

Whereas a correlated failure is a *shared-cause* or *co-failure*, a *cascade* is a *time-ordered propagation process* in which a failure in one part of the system increases the likelihood of further failures (Watts 2002; Buldyrev et al. 2010). The defining feature is *endogenous amplification*. Examples include load redistribution (remaining nodes become overloaded), reachability loss (partitions increase hop lengths and delay), or protocol-level reactions (such as loss of consensus in proof-of-stake blockchains) that compound stress.

Correlated failures and cascades differ along two axes:

- *Source of dependence.* Correlated failures arise from *shared cause* (common dependency). Cascades arise from *state-dependent propagation* (one failure increases stress on others).

- *Temporal structure.* Correlated failures can be simultaneous (many nodes fail at once). Cascades are sequential (failures accumulate through time as conditions evolve).

In practice, correlated failures can cause cascades. A correlated shock (e.g., a provider outage) can remove a large fraction of nodes at once, and the resulting load and connectivity changes can then trigger a cascade in the remaining network. Conversely, cascades need not begin with a common-cause event because a localized, initially independent failure can trigger system-wide disruption if the system is actually more centralized.

1. *Initiating event (shock or fault).* A node-level or dependency-level impairment occurs (attack burst, software fault, routing event, provider outage).

2. *Overlay response.* Connectivity and propagation change: peers disconnect and reconnect, discovery degrades, path lengths increase, or throughput collapses.

3. *Application-layer outcome.* The overlay degradation yields decision-relevant outcomes (missed attestations, delayed finality, degraded deliverability, reduced content availability).

In the blockchain case studies, this sequence appears when a disruption of Internet fiber lines (concentration risk) leads to peering instability, which then degrades validator performance (Seidenberger and Maiti 2025a; Seidenberger et al. 2026c).

Measuring resilience in the face of cascades. We can quantify the effects of cascades on system performance through a time series. Let $P(t)$ denote a proxy for system-level performance (availability, deliverability, consensus effectiveness, or retrieval success), and let P_{pre} denote a baseline level measured before a perturbation. We reuse standard resilience

summaries that separate *severity* from *duration* and support comparison across scenarios (Bruneau et al. 2003; Sterbenz et al. 2010):

- *Drawdown (depth of degradation).* The worst relative performance loss over an evaluation window:
$$d = \frac{P_{\text{pre}} - \min_t P(t)}{P_{\text{pre}}}.$$

- *Recovery time.* Time to return above a fraction α of baseline (right-censored if not reached within the horizon):
$$T_{\text{rec}}(\alpha) = \min\{t : P(t) \geq \alpha P_{\text{pre}}\}.$$

- *Deficit area (severity $\times$ duration).* The accumulated shortfall below baseline over the horizon:
$$L = \sum_t \max(0, P_{\text{pre}} - P(t)),$$
often normalized to compare across systems and windows. We use discrete time because most measurements from protocol networks are sampled at fixed intervals (e.g., per-block, per-crawl, or time-binned windows), not continuously observed.

- *Completeness and rebound rate.* Completeness is the fraction of baseline recovered by the end of the horizon; rebound rate summarizes how quickly performance improves conditional on recovery.

These metrics are broadly useful for measuring systems under cascading failures. First, they are computed as time series over windows, so they capture changing conditions and bursts instead of assuming steady-state behavior. Second, when changes are tested with paired comparisons, this focus on within-pair differences cancels much of the background variability and makes the effect of the change easier to detect. Finally, cascade analysis should be tail-oriented. Dependence makes rare, severe outcomes more likely, so evaluation should emphasize worst-case degradation and recovery (e.g., upper-quantile loss, time-to-recover, time spent in a degraded regime), not only means or medians.

3.3 From telescopes to mechanism claims

The motifs introduced in Section 3.2 are intentionally generic. Heavy tails, hazards, diffusion, churn, and correlated failure are not tied to a single protocol family, but rather are helpful concepts for describing the networks under study. What makes them most useful in this context is how they let us assemble partial observations into defensible, decision-relevant stories about security and resilience in live ecosystems. This section describes how the remainder of the dissertation turns telescope outputs into mechanism claims *without* assuming a complete view of the system.

A recurring theme in the papers underlying this dissertation is that the main difficulty is not choosing a sophisticated model. The difficulty is deciding what the model is allowed to claim given (i) partial observability, (ii) temporal drift, (iii) heterogeneity of data-generating processes, and (iv) adversaries that respond to being measured. The practical response is to treat three steps as inseparable: *linkage* (connecting layers), *comparison* (creating credible baselines), and *stress-testing* (checking that a conclusion survives plausible alternative explanations).

Cross-layer linkage is the bridge between what we can "see" and what we want to "explain." In protocol networks, the relevant causal pathways almost always cross layers: discovery mechanisms affect exposure, infrastructure placement shapes reachability and correlated failure domains, and application-level outcomes (deliverability, availability, correctness) are downstream of overlay behavior. But the process of linking data across time and modalities is also where hidden assumptions enter. Any join key (an infohash, a domain, an IP, a public key) encodes what we are treating as "the same entity" across time and modalities. In settings with NAT, multi-tenant hosting, and network identifier rotation, entity resolution becomes a problem unto itself. Therefore, entity identification rules have to be explicit modeling assumptions that are stated and defended.

Two concrete examples show why this matters. In MagnetDB, the stable unit is the BitTorrent content identifier (the infohash) observed through DHT discovery over a multi-year

window, which enables event-time alignment and hazard modeling of a torrent's first appearance on the network while remaining agnostic about downstream demand for that torrent (Seidenberger et al. 2025b). In the email inbox audit work, the stable unit is not a user identity but header- and infrastructure-level provenance features such as sender domains, which supports ecosystem-level claims about intermediation and concentration without requiring the actual message content (Seidenberger et al. 2025a). In both cases, the choice of join key is what makes the modeling feasible in the first place.

Linking entity identification data across datasets also raises governance risks that are easy to understate. Combining modalities can create "mosaic" effects where seemingly benign fields become identifying once fused (Ohm 2009). The Dataset Value Taxonomy work frames this as a general property of modern empirical research: as downstream analysis becomes cheap, scientific leverage shifts upstream to temporally deep, broad, and responsibly accessible datasets, and governance must explicitly account for linkage risk (Seidenberger and Maiti 2025b).

The concept of time. Protocol networks rarely provide a single trusted notion of global time. Nodes run on local clocks that can drift, and protocols often define progress using endogenous markers such as block height, epoch number, or sequence numbers (nonces), rather than wall-clock timestamps. As a result, any temporal analysis must begin by choosing a *clock*: calendar time from the measurement vantage point, a protocol-defined notion of progress, or an external event used as a reference. This dissertation focuses on the concept of *event time* as many protocol-network phenomena are driven by discrete events. Event-time alignment reduces confounding because it compares trajectories relative to a shared reference point rather than mixing incomparable phases of a process.

The BitTorrent work is an illustration of this. The study constructs a macro panel linking legitimate availability signals to the *hazard of first pre-release torrent appearance* and then a micro panel linking those appearance times to swarm dynamics and peer composition

(Seidenberger et al. 2026b). The modeling gains come from aligning observations with an upstream clock that lets downstream diffusion be summarized as functions of Δt rather than calendar time. This alignment also creates a way to talk about data censoring, in that if a title never leaks within the observation window, it is right-censored, not "safe forever."

Getting good baselines. Because protocol ecosystems are observational and nonstationary, many claims are predicated on establishing credible baselines. The dissertation therefore uses comparative experimental designs that are realistic under deployment constraints.

In blockchain infrastructure studies, the most direct baseline is not the "Internet in general" but a *matched control host* that is similar in region, provider, and exposure profile. This is how we can estimate the incremental "visibility premium" of being a node that is discussed in Chapter 4: the extra scanning, exploitation attempts, or resource pressure that arrives because the host is discoverable in the overlay and not merely because it is an Internet-facing machine (Seidenberger and Maiti 2025c,a).

The same logic appears in the email inbox audit work, but the baseline is constructed differently. Here the goal is not to compare two machines, but rather a user's inbound email against counterfactual expectations. The study shows that a large share of observed email volume can be attributed to a small set of intermediaries, which is the empirical basis for treating those entities as de facto stakeholders in privacy outcomes (Seidenberger et al. 2025a).

In studying the illicit distribution of media via torrents on the BitTorrent protocol, the baseline problem is framed as heterogeneity in titles and media release strategies by studios. The macro panel includes covariates that encode legitimate availability and release context so that comparisons are made between titles that are similar in observable ways, reducing the risk that apparent effects are merely differences between content types or distribution strategies (Seidenberger et al. 2026b).

Mapping dependencies. A large share of the dissertation's empirical contribution is in measuring and mapping infrastructure dependencies. Decentralized networks can appear decentralized in some aspects while relying on centralized intermediaries. A few cloud providers, a small set of ASNs, or a small set of physical corridors can be enough to create correlated failure domains and misaligned incentives. These hidden centralization patterns are treated as part of the threat model. In practice, this means explicitly mapping all the relationships between nodes and their interconnected dependencies. For example, an IP address maps to an ASN, which maps to a provider, which maps to a region, which maps to a physical location. This mapping is not always straightforward.

3.3.1 Summary

Part I has argued that many of the Internet's most consequential systems are not platforms with a single operator, but protocol networks, which are sociotechnical decentralized systems whose security and resilience emerge from a set of heterogeneous actors following a standard. This framing shifts the central question from whether a system is "decentralized" in the abstract to where decentralization is actually realized and where centralization persists in practice. Chapters 1 and 2 therefore treated the *observability gap* as a first-order security bottleneck and positioned longitudinal, multimodal measurement as prerequisite to make credible security claims. The result was a measurement-first methodology: build telescopes that can operate under partial observability, drift, heterogeneous deployments, and adversarial adaptation, while explicitly managing ethical and governance risks and treating dataset construction and stewardship as core scholarship. This chapter then introduced the modeling language used to convert these incomplete observations into decision-relevant signals, recurring empirical motifs, and cross-layer linkage that uncovers hidden dependencies by connecting overlay behavior to the underlying infrastructure that actually enables it.

With this foundation in place, Part II makes *mechanism claims* about systemic processes that repeatedly shape the privacy, security, and resilience of decentralized protocol networks.

The chapters that follow use the telescopes and motifs developed in Part I to shed light on real networks at scale. Part II shows that the openness of these networks creates an attack surface, protocol incentives can drive centralization, physical and infrastructural chokepoints create hidden centralization, and temporal dynamics can create significant cascading risk and tail outcomes. In other words, Part II operationalizes the concepts presented in Part I to reveal underlying mechanisms that, once understood, can inform how we improve these systems in Part III.

Part II

Cross-Cutting Security Mechanisms in Decentralized Protocol Networks

Chapter 4

Visibility as Attack Surface

Open and permissionless protocol networks are built to be discoverable and reachable. In the systems studied in this dissertation, nodes must be reachable by others for the network to form and function. A BitTorrent peer cannot exchange pieces of a file without inbound connectivity. A DNS resolver cannot answer queries if it cannot be reached. An Ethereum node cannot serve blocks, propagate attestations, or maintain a healthy peer set if it is not discoverable and connectable. In Chapter 1, this dissertation argued that these systems constitute a form of *hidden critical infrastructure*. They enable planet-scale coordination among mutually distrusting actors, but they are structurally under-observed and therefore difficult to secure.

This chapter makes the claim that in open protocol networks, *visibility is an attack surface*. The same properties that enable permissionless participation, such as enumerability, reachability, and stable identification, also create a resource that adversaries can allocate against. In cyber kill-chain terms, visibility mainly reduces friction in the early phases. It lowers the cost of reconnaissance and target development by making participants easy to enumerate, fingerprint, and revisit (Wolchok and Halderman 2010; Kim et al. 2018). When hosts are discoverable and stably identifiable, adversaries can select and prioritize targets more reliably and can scale initial access attempts, rather than spending effort to locate viable targets.

The central empirical question is therefore not merely whether these networks are attacked (they are), but whether the *incremental* exposure associated with protocol participation is measurable and distinguishable from the baseline cyber activity on the public Internet. This claim is supported by a case study of the Ethereum mainnet, the world's largest public proof-of-stake (PoS) blockchain network. A paired deployment of control hosts and live Ethereum nodes as the experimental group was used to estimate this "visibility premium." Over a two-month measurement window, with multiple vantage points across five Amazon Web Services (AWS) regions, 130.9 million reconnaissance/exploitation attempts from 12.5 million unique source IPs allowed for controlled comparisons between Ethereum node and non-node hosts in the same regions. The resulting evidence shows (i) a statistically strong and operationally meaningful increase in hostile activity attributable to running a node, (ii) specific targeting behavior against Ethereum-specific characteristics, and (iii) broad exposure of such vulnerabilities from a nontrivial fraction of publicly reachable Ethereum nodes exposing auxiliary services beyond their required P2P ports.

The evidence here is scoped to the paired cloud deployment and scan design used in the Ethereum study, and the email section is presented as a mechanism vignette to show portability across protocol families rather than as a causal estimate of harm to a specific user's privacy.

4.1 What is Visibility?

In cybersecurity terms, a network is "visible" if hosts can be discovered (do I know they exist?) and reached (do I have a path to communicate with them?). It also extends to *enumerability*, which is the ability to produce a list of targets. In protocol networks, the ability to enumerate hosts on a network often comes directly from the discovery and rendezvous mechanisms of the protocol. If a protocol uses a distributed hash table (DHT), a peer discovery table, or a stable directory of identifiers, then an adversary can often enumerate participants

at scale without interacting with each one. Enumerability can be direct or indirect. Direct enumerability means that the protocol itself provides a list of endpoints or node identifiers. Indirect enumerability means that some observable artifact allows enumeration, such as a publicly shared index, network explorer, or third-party measurement feed. Ethereum relies on a Kademlia-style DHT for peering, which allows for direct enumeration of hosts on the network (Maymounkov and Mazieres 2002). However, not all hosts that are discovered are *reachable*.

Reachability is the ability to actually deliver traffic to a target and receive responses. It is the interface between the overlay network protocol, which is a logical entity, and the underlying Internet, which is a physical entity. For example, a node may join the network and advertise via the DHT that its inbound IP address is `100.x.y.z`. IP addresses starting with 100 (specifically 100.64.0.0/10) are Carrier-Grade NAT (CGNAT) addresses. These are private, non-routable addresses on the public Internet. This would make the node inaccessible for an inbound-initiated connection outside of that CGNAT. NAT traversal and NAT hole-punching are important topics in P2P networking and still present challenges for protocol networks. A target that is enumerable (I know it exists), but not reachable (I cannot talk to it), is a poor candidate for remote exploitation. Conversely, a target that is reachable but not easily enumerable may still be attacked, but is less likely to receive concentrated attention. Reachability is also not all or nothing. It can mean reachability to a port used by the protocol, or reachability to an auxiliary service on the same host.

Another key component of visibility is *identifiability*, the ability to recognize the same target over time and across observations. Stable IP addresses, long-lived hostnames, and persistent public key identifiers make targets easier to track. If a target is easier to track, then one-off opportunities can turn into sustained campaigns. If an adversary can reliably refind the same host, then it can iterate tactics, adapt to defenses, and invest in more expensive attacks. Identifiability can be significantly moderated by the operational deployment of the host. For instance, nodes hosted on the cloud may be both highly reachable and relatively

stable in their addressing compared to nodes run on consumer devices within residential networks.

4.1.1 Visibility in Protocol Networks

At the lowest layer, visibility first occurs on the Internet as routable IP addresses, open ports, and network activity. This layer of visibility is independent of the overlay protocol and is most applicable for general Internet scanning. A host can be discovered just by being online. However, protocol networks typically include a discovery mechanism that helps participants find one another. Ethereum and BitTorrent use a combination of a DHT for purely decentralized peer discovery but also have means of centrally facilitated rendezvous via bootstrap servers and trackers. Stable identifiers provide needed continuity in the network. These include node IDs, public keys, and content identifiers; even email addresses are stable identifiers. Identification stability at the protocol level can improve system usability and coordination, but can also then be leveraged by a would-be attacker.

A recurring observation in protocol networks is that "the protocol" is rarely the only thing exposed by the host. Operators of the host server often expose management interfaces, monitoring dashboards, RPC endpoints, and other remote administration tools on the same or adjacent hosts. These auxiliary services are frequently outside the protocol's threat model, but they can enable real-world exploitability and create pivot opportunities from "generic host compromise" to "protocol participant compromise." Visibility is an intersection of surfaces that can be separately measured and risk mitigated.

It is important to emphasize that openness and visibility are explicit design goals of many protocol networks. Permissionless participation and public verifiability are core commitments for these systems, and broad reachability is part of how they reduce dependence on any one jurisdiction, provider, or intermediary. The argument here is that visibility carries a measurable security cost which should be understood empirically and reduced where possible. If reachability is a prerequisite for system liveness, adversaries will allocate effort toward

reachable participants, creating a "visibility premium" for hosts participating in protocol networks. That added host-level risk should be treated as part of the ecosystem's security posture.

Understanding visibility in this way matters because adversarial effort is not uniformly distributed. Every stage of an adversary's kill chain comes at a cost, and attackers shape their strategies based on the expected return of an attack (minimize cost, maximize benefit). Protocol networks present value to attackers because they expose long-lived hosts that sit on economically meaningful systems (depending on the specific network). A useful adversary-centric framing is to treat visibility as allocating *attention*. The Internet is already a hostile environment where reconnaissance traffic reaches most publicly routable hosts (Barford et al. 2010; Fachkha and Debbabi 2015). The key question for protocol networks is whether participation changes the distribution of adversary effort. If running a node changes both the volume of traffic received and the content of that traffic, then that is an important signal of adversarial resource allocation. The visibility premium is therefore the incremental hostile activity associated with being a protocol-network participant, estimated via comparisons to matched non-participant baselines (control hosts) in comparable environments.

4.1.2 Security and Privacy Costs

When a node is easy to find and reliably reachable, it attracts attention, and that attention has real costs. On the security and operations side, most protocol nodes are ordinary Internet-facing servers. If they expose anything beyond the protocol port, such as SSH, a dashboard, or an API, attackers will scan and probe those services just as they would on any other host. If they gain access, the harms are familiar: stolen data, service disruption, or use of the host to attack other systems. Those harms can also affect the protocol itself by changing client settings or degrading how the node participates. Even without a classic compromise, attackers can still cause trouble by driving enough traffic at the node to slow it down or make it unreliable. In a protocol network, that pressure can show up as fewer

peers, slower propagation, delayed processing, and other performance failures that matter even if the protocol does not lose safety. The broader point is that stable visibility lets attackers come back repeatedly, learn what works, and escalate over time. That matters because these networks depend on long-running, stable participants. In Ethereum, we can measure patterns consistent with that story.

On the privacy side, the same "easy to reach" property creates extra metadata leakage. Protocols often require stable identifiers so participants can find each other, but those identifiers also give third parties a way to interact with and learn about participants' identity. In Ethereum, that usually means IP addresses and the ability to link infrastructure to on-chain identities or track participation over time. In email, it is even simpler as the email address is the identifier, and reachability is the whole service, so once an address is disclosed, anyone can send to it and observe the metadata that comes with that interaction.

4.2 Case Study: Targeting of Ethereum Nodes

This case study quantifies the visibility premium for Ethereum mainnet nodes. Beacon nodes must be discoverable and reachable to participate in consensus layer gossip and discovery. At the same time, they run on ordinary Internet-connected hosts, which makes them subject to the full spectrum of scanning and opportunistic exploitation attempts. The analysis in this section is based on the Ethereum node targeting study (Seidenberger and Maiti 2025c) and the underlying multimodal EtherBee observatory deployment (Seidenberger and Maiti 2025a).

4.2.1 Measurement design and data

Deployment setup. To estimate the incremental targeting associated with node participation, the study uses a paired experimental/control design deployed across five geographic regions for a two-month period. In each region, two matched cloud hosts were provisioned:

one *experimental* host running a mainnet Ethereum beacon node and one otherwise similar *control* host running the same security sensors but *no* Ethereum node. This yields ten vantage points total: five experimental and five control. Within-region pairing is central to causal identification because both hosts in a pair share geography and cloud deployment context.

Each host was instrumented to observe hostile traffic at the host boundary while also capturing the Ethereum node's participation on the network. Specifically, the deployment combines (i) a honeypot platform (T-Pot) configured for broad service coverage, (ii) full packet capture and session-level indexing (Arkime Project 2024), and (iii) node-level client metrics exported by the Ethereum clients. This multimodal design makes it possible to pair incoming hostile traffic with the node's own network performance. Honeypot logs provide application-layer evidence of probing and exploitation attempts. Network session metadata provides evidence of who connects, when, and at what volume. Client metrics provide node-local outcomes such as peer counts and health signals. Together, these sources let us distinguish signals from generic "Internet noise." All ten hosts ran the T-Pot platform, which orchestrates multiple containerized honeypots at varying levels of interactivity (T-Pot Project 2024). The firewall was configured so that ports 1–63999 were exposed to the honeypot suite while ports 64000+ were reserved for management and the P2P ports used by the Ethereum clients. This separation prevents accidental overlap between legitimate protocol participation and honeypot surfaces, simplifying attribution during analysis.

The study distinguishes between two categories of hostile interaction. The first is *Reconnaissance*, which refers to sustained scans and single-probe connections that do not exhibit exploit behavior. The second is *Attacks* (attempted intrusions and exploits), interactions that go beyond basic scanning, such as repeated SSH brute-force attempts or HTTP requests for sensitive URIs (e.g., environment and credential files). This pragmatic distinction separates baseline scanning activity (Pauley et al. 2023) and Internet background radiation (Pang et al. 2004) from higher-effort behaviors that reflect the expected adversary behavior.

Ethereum node configuration. Each experimental host ran a mainnet beacon node consisting of a consensus layer client and execution layer client. The deployment used Lighthouse (consensus) and Nethermind (execution) with default settings except for the advertised P2P ports, which were set to 64000 (consensus) and 64333 (execution) to avoid conflicts with honeypot ports and simplify filtering in packet captures. To avoid staking real funds while still observing validator performance signals, the deployment enabled the Lighthouse Attestation Simulator, which produces simulated attestations each epoch and exports metrics that capture whether the node would have performed validator duties on time given its connectivity and state. This simulator provides a meaningful proxy for node health and performance without turning the measurement into a real validator operation.

Dataset scale. Over the two-month window, the deployment recorded 130.9 million reconnaissance and exploitation attempts originating from 12.5 million unique source IPs across the ten vantage points. In parallel, the five beacon nodes recorded 45.4 million P2P TCP sessions on their beacon-node ports, enabling peer-tenure analysis. In addition, to assess how auxiliary service exposure appears in the broader ecosystem, the study performed a one-shot scan of 6,185 active beacon nodes gathered from a public node explorer and probed for common service ports and default Ethereum RPC/P2P ports.

Scope limits. Two limitations matter for interpreting the results. First, cloud-hosted vantage points can differ from residential, enterprise, or privately peered environments. The findings should therefore be interpreted as *incremental risk conditional on these deployment conditions*. Second, the measurements are still partial in that the honeypots capture only interactions that reach the sensors, and protocol encryption limits payload visibility. The chapter's claims therefore remain tightly scoped to differences in observed hostile traffic volumes and targeting patterns, and node-local performance changes aligned to observed hostile traffic.

Table 4.1: Generalized linear mixed model estimating the visibility premium of running an Ethereum beacon node.

	Coeff.	SE	z	P> |z|
Fixed effects				
Intercept	2.804	0.100	28.117	<0.001
Group (Experimental)	1.111	0.008	147.289	<0.001
Random effects				
Group variance (region)	0.050	0.030		
Model fit				
Obs.	100,000			
Regions	5			

4.2.2 Result 1: The Visibility Premium

The first empirical question is whether node participation is associated with a systematic increase in observed hostile activity after accounting for region effects. The node targeting study answers this with a generalized linear mixed model (GLMM) that predicts daily attack counts with a fixed effect for group (experimental versus control) and a random effect for region. The key coefficient is the group effect.

The model yields a statistically strong and practically large visibility premium. The group coefficient is 1.111 on the log expected daily attack count with $p < 0.001$. Interpreted multiplicatively, this corresponds to roughly a threefold increase in expected attack volume for experimental hosts relative to matched controls ($\exp(1.111) \approx 3.0$), conditional on region. In other words, after controlling for geography and cloud service provider, running an Ethereum beacon node is associated with substantially more hostile attention than running the same host without node participation.

This result shows that on the Ethereum network, visible nodes receive substantially more attacks than matched controls. The remainder of the case study asks *how* that attention

manifests, which surfaces are targeted, how attacker infrastructure differs from node infrastructure, and whether this attention can translate into measurable operational harm.

4.2.3 Result 2: Targeting focuses on exposed ports and services

The visibility premium is not only a matter of volume. A more diagnostic question is whether the additional hostile activity is *semantically aligned* with the surfaces that distinguish node hosts from generic servers. The case study shows that it is. Elevated targeting concentrates on (i) the ports and processes that make Ethereum nodes usable and (ii) the auxiliary services that operators commonly expose.

Targeting by port. The main takeaway from the port-layer data is that *protocol-specific* ports are being targeted. The standard Ethereum P2P ports receive both sustained reconnaissance and exploitation attempts at higher levels on experimental nodes than on matched controls. Figure 4.1 compares attack counts by port across the two groups and shows this elevation for both the P2P and RPC ports. Adversaries are therefore directing effort at the interfaces that identify and enable the node as a protocol participant, not only at generic administrative surfaces.

Web probing and sensitive URIs. A complementary signal comes from application-layer probing of web services. The study uses Tanner, a web-application honeypot, to record HTTP requests and classify them by whether they target known sensitive resources such as environment files, credential endpoints, or other exploitable paths (Foundation 2023). Figure 4.2 shows that several categories of sensitive URIs are requested substantially more often on experimental hosts than on controls.

This behavior is consistent with exploit-oriented probing rather than purely opportunistic scanning. Requests for environment files (e.g., `.env`), application configuration files, and cloud metadata endpoints are a common pathway for credential discovery in real-world exploitation chains. The distributional differences between experimental and control hosts

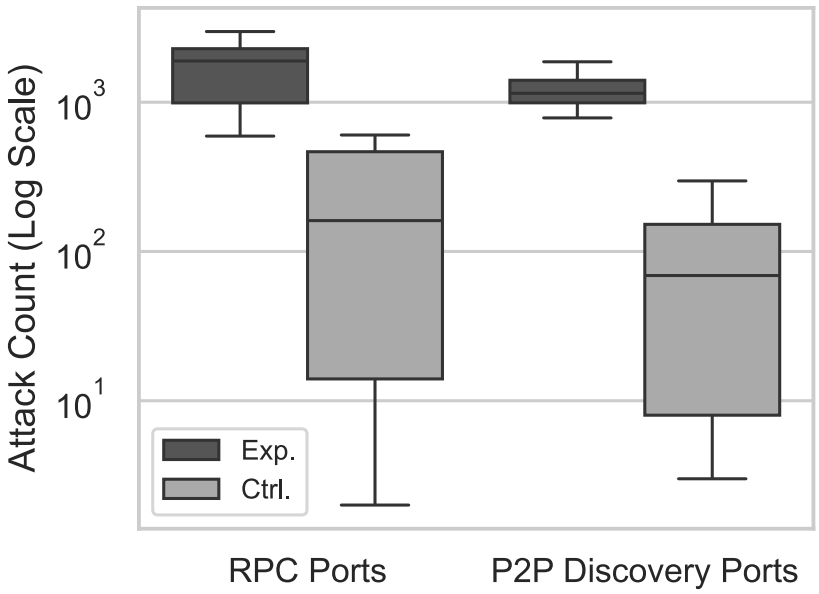

Figure 4.1: Attack counts on experimental hosts and matched control hosts, by port.

are statistically significant in the study, reinforcing that the additional attention is not random noise but structured probing aligned with the expected value of the host.

SSH credential guessing. Finally, the study examines SSH credential guessing behavior using Cowrie's captured login attempts (Cowrie 2023). Rather than relying only on mean attempt counts, it summarizes the inequality of credential attempts via Lorenz curves and Gini coefficients (metrics introduced in Chapter 3). Figure 4.3 shows a distinct difference in credential concentration patterns between experimental and control hosts.

Specifically, the average Gini coefficient for credential attempts is 0.25 for the control hosts and 0.43 for the experimental hosts. The control group's lower value indicates a relatively even spread of attempts across many username/password combinations (generic scanning). The experimental group's higher value indicates that attempts against node hosts are more concentrated on fewer combinations—suggesting more focused or targeted credential-guessing strategies against Ethereum nodes rather than broad, opportunistic probing. The key point for this chapter is not to over-interpret attacker intent, but to emphasize that the additional pressure is *structured* and differs in distributional shape, not only in volume.

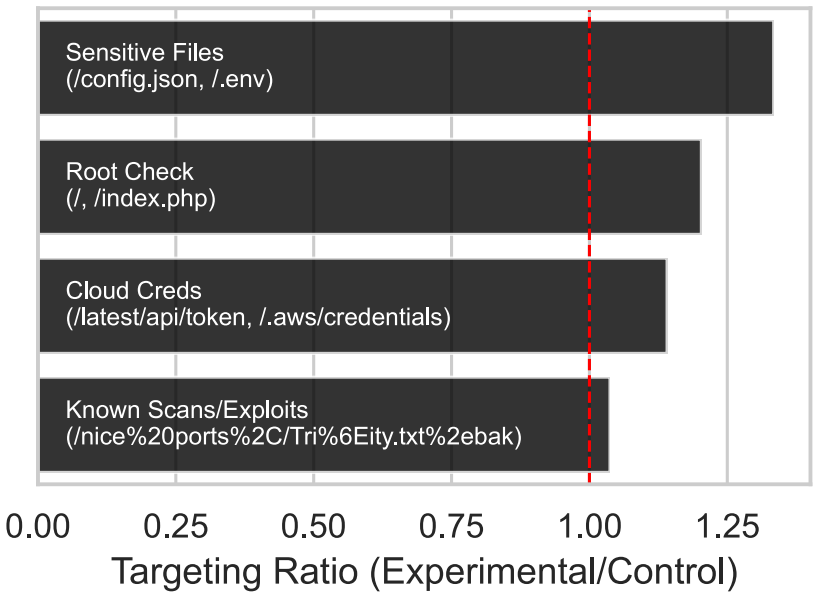

Figure 4.2: Ratios of sensitive URI requests observed on experimental hosts relative to controls.

Taken together, the port-, URI-, and SSH results support a consistent picture that the visibility premium is not just "more packets." The data reveal that Ethereum nodes receive higher-intensity hostile behavior focused on RPC and discovery ports, credential-harvesting endpoints such as `.env` and `/.aws/credentials`, and common administrative services such as SSH and HTTP(S). This aligns with the hypothesis that always-on, publicly advertised nodes have heightened targeting by adversaries.

4.2.4 Result 3: Attackers come from different networks

If visibility attracts targeting, a further adversary-centric question is whether attackers tend to operate from the same infrastructure regions that host nodes, or from distinct reconnaissance infrastructure. The node targeting study addresses this by comparing the distribution of node IP density to the distribution of hostile-source IP density across IPv4 address space.

Figure 4.4 presents a dual-density comparison as a heatmap for the distribution of node IPs and a corresponding heatmap for the distribution of attack-source IPs. The overlap is

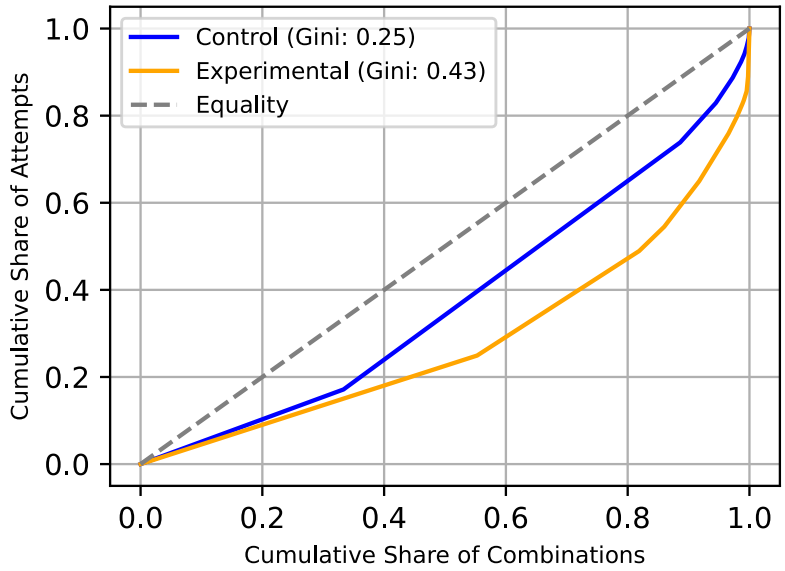

Figure 4.3: Lorenz curves for SSH credential attempts on experimental and control hosts, illustrating differences in concentration.

low. The Jaccard index between the two distributions is 0.274, and divergence measures (KL divergence and Earth Mover's Distance) further indicate substantial mismatch. This mismatch is important because it suggests that the observed hostile activity is not "neighbors in the same hosting regions." Instead, it is consistent with the use of distinct scanning and exploitation infrastructure that targets node-hosting regions without being co-located within them.

For the dissertation's broader argument, this result reinforces why protocol-network security is constrained by observability. The infrastructure that hosts nodes and the infrastructure that targets nodes need not coincide. Defenders operating only from within their own hosting environment therefore see only a local projection of a broader adversarial ecosystem. Multi-vantage telemetry is required to separate "where nodes are" from "where attacks originate". However, later we can see how we can use this observation to inform better security practices for node operators.

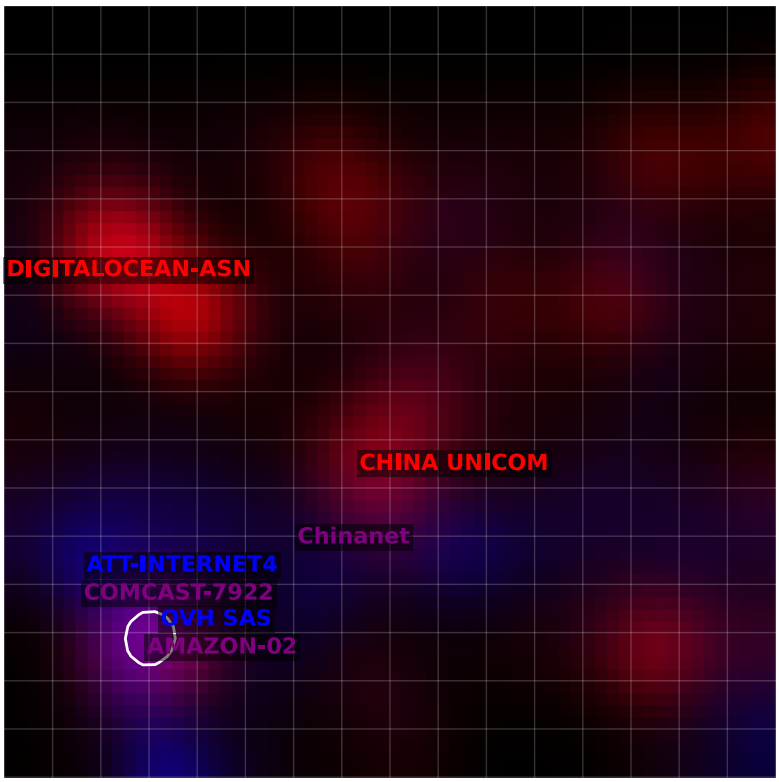

Figure 4.4: Dual-density comparison of beacon node IP density and hostile source IP density across IPv4 space.

4.2.5 Result 4: Auxiliary service exposure expands attack surface

The experiment showed that being an Ethereum beacon node attracts extra hostile attention compared to matched controls, but we also need to know how common the relevant risk surface is among ordinary nodes that are not intentionally instrumented to expose wide attack surfaces such as our experimental hosts. To answer that, we mimic a typical attacker workflow by pulling a list of currently active beacon node endpoints from a public node explorer and running a limited TCP port scan for common service ports and the default Ethereum ports.

The findings show that nodes in the wild commonly run auxiliary services on the same host as their Ethereum nodes. Among 6,184 recently active beacon nodes with complete scan results, 25.68% (1,588/6,184) had at least one open port beyond their P2P ports. About

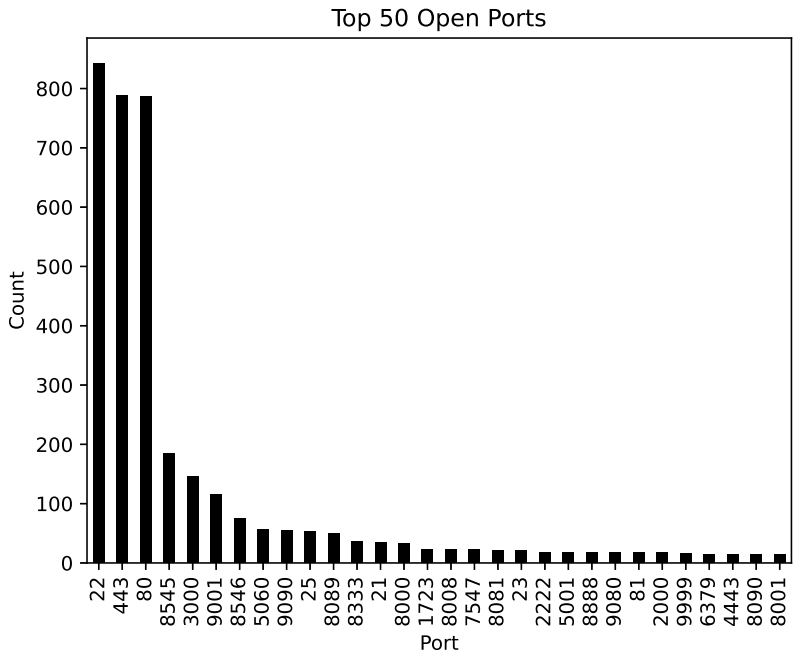

Figure 4.5: Distribution of open auxiliary services observed in a scan of active beacon nodes.

7.45% exposed exactly one additional port and 18.22% exposed multiple open ports. The most prevalent open ports were SSH (22) at 13.63% of nodes, HTTPS (443) and HTTP (80) at approximately 12.7% each, followed by the Ethereum execution layer RPC port (8545) at 2.99% and Grafana's default port (3000) at 2.36%.

Open port combinations show further evidence that the operators of these nodes run common administrative stacks and identifiable port patterns (that map to common deployment and server administration patterns). The most common combination was HTTP/HTTPS (80, 443) at 4.66%, followed by HTTP/HTTPS with SSH access (22, 80, 443) at 1.57%. Of particular concern from an exploitability perspective, 1.12% of nodes exposed both SSH and RPC (22, 8545), enabling direct remote administration on a host that also exposes an API-capable interface. The study also reports that 213 nodes (3.44% of the sample) had default RPC ports (8545 or 8546) exposed.

As a sanity check, the scan results were reviewed to make sure they reflected real beacon nodes rather than honeypots or other measurement infrastructure. The hosts do not show

the large, contiguous open-port ranges typical of telescopes, and the scanned IPs also appear in the P2P session data as active peers. Taken together, that evidence makes it likely that the open ports reflect ordinary operator configurations rather than artifacts of measurement infrastructure.

The scan also shows that this attack surface is common in practice. Many nodes expose services that are not required for P2P participation. If a node can be discovered and reached, any extra exposed service becomes another place an attacker can probe, and it creates realistic paths to compromise or disruption that do not depend on breaking the protocol itself.

4.2.6 Result 5: Reachability degrades local performance

The prior results demonstrate that visibility changes who gets targeted and how. The final question in this case study is whether that targeting can translate into measurable operational harm for the node itself, even without a successful compromise.

The study presents a contained incident vignette where a honeypot sensor on an experimental host observed a sharp spike in inbound hostile traffic consistent with a reflection-style attack or aggressive scanning burst. During this interval, host CPU utilization increased sharply as the kernel processed the additional packet load. Concurrently, the consensus client's peer count dropped by roughly 20%, indicating churn and dropped connections, and the Lighthouse Attestation Simulator reported degraded performance (unable to perform simulated validator duties on time). The incident did not involve a protocol-level exploit; it relied on the node's visibility and available bandwidth to impose a transient local denial-of-service condition.

This vignette does not claim a specific exploit risk that would affect Ethereum-wide liveness, but rather serves as a demonstration of a mechanism pathway:

$$\text{Visibility} \rightarrow \text{Increased inbound traffic} \rightarrow \text{Peering instability} \rightarrow \text{Performance degradation.}$$

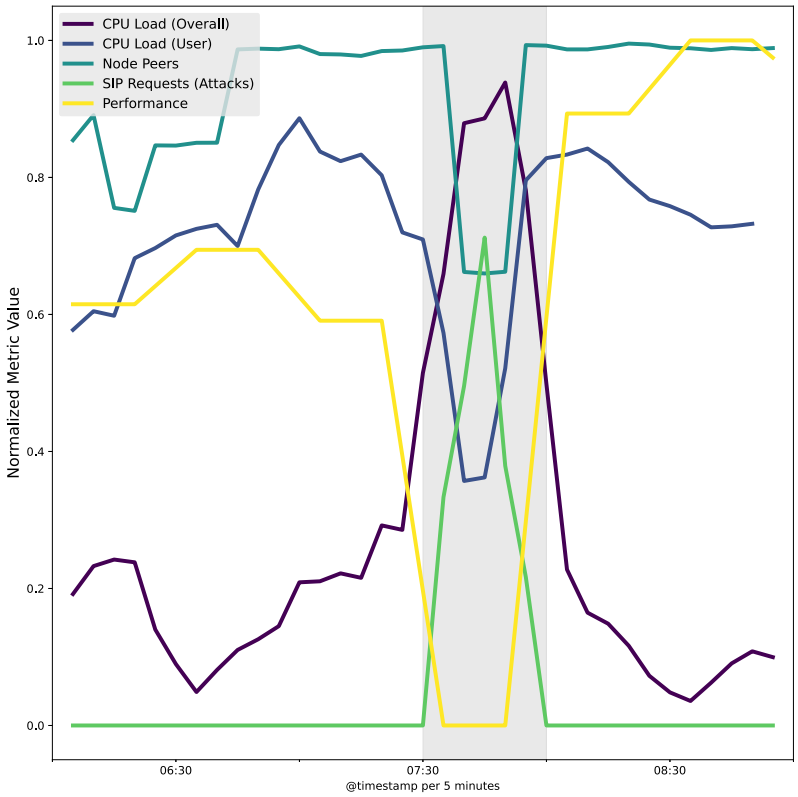

Figure 4.6: Temporally aligned traces showing an attack burst, host resource pressure, peer-count degradation, and attestation-simulator impact on an experimental host.

The key point is that an attacker does not need to defeat cryptography or find a client vulnerability to cause harm through this pathway. Because the node must stay reachable to do its job, it can be pressured through ordinary network traffic that consumes CPU and disrupts peering. This makes the cost of visibility an operational concern and highlights why host-level hardening and service minimization matter as first-order efforts that are distinct from, and underlie, protocol-level safety and liveness assumptions.

Summary of the Ethereum case study. Across both the experiment and the ecosystem-wide scan, the evidence supports this chapter's central claim that visibility comes with a security price. Running an Ethereum beacon node produces a measurable visibility premium: experimental hosts receive substantially more exploit-like interactions than matched controls. That added inbound pressure is also targeted in ways that match how nodes are operated in

practice, concentrating on protocol-adjacent and operator-adjacent interfaces such as specific ports, sensitive HTTP paths, and SSH authentication attempts. The sources of this traffic are distributed differently from where legitimate nodes are hosted, which is consistent with the broader observation that adversarial traffic often comes from a distinct infrastructure ecosystem (Durumeric et al. 2014a). The public-node scan further shows that a nontrivial fraction of real nodes expose auxiliary services beyond what P2P participation requires, expanding the set of reachable surfaces an adversary can probe.

A vignette shows how this visibility can translate into measurable harm to a node's ability to participate in consensus without a protocol-level exploit. A burst of inbound hostile traffic coincides with host resource pressure, peer loss, and degraded simulated validator duties, illustrating a straightforward pathway from reachability to operational degradation. Session metadata also suggests that this exposure can be persistent rather than fleeting. Among the top 1,000 peers by volume in each region, the mean peer tenure is 26.46 days over the two-month window, indicating that a substantial share of high-activity connectivity is recurrent. From an adversary's perspective, that persistence makes targeting "sticky" by providing stable, reachable infrastructure that can be revisited repeatedly over time. The next section extends the same visibility lens to a different protocol family where reachability is mediated by stable identifiers rather than routable P2P endpoints.

4.3 Vignette: Who Can Reach the Inbox? Delegated Sending in Email

In the Ethereum case study, "visibility" means that a node is reachable on the public Internet at specific addresses and ports (routable endpoints). A node has an IP address and open ports, and adversaries can direct their effort against that surface. This vignette applies the same lens to a protocol family where reachability is mediated primarily by a *stable identifier* rather than a stable IP address—email. In email, an address is itself the rendezvous object.

Once an address is disclosed, it becomes a durable, globally reachable interface that any sender can attempt to contact. As with open P2P networks, the central question is not whether an email inbox receives hostile or unwanted traffic in general, but how *protocol participation* (i.e., giving an address to a service) expands the practical contact surface and changes who can reach the participant. This section draws from the year-long inbox audit in *Why You've Got Mail* (Seidenberger et al. 2025a), which evaluates inbox privacy implications of email marketing practices in online apps and services.

Measurement design. The study created accounts across 150 popular services and apps using unique email addresses (one address per service) that all forward into a single catch-all inbox. Registration was performed in a way that minimizes user-driven causes of additional mail (e.g., opting out of optional communications where possible and otherwise not engaging with the app/service post-signup), so that subsequent messages primarily reflect service-initiated contact and the surrounding sending ecosystem. Emails were collected passively over a 361-day window (March 2023–February 2024) and stored as full message files, enabling analysis of sender identity, authentication, and infrastructure provenance.

Who can reach the inbox. A key operational feature of email is that the sender identity visible to the recipient (e.g., `From:`) need not correspond to the infrastructure that actually transmits the message. The study therefore classifies messages by the *effective sending relationship* using email authentication and header evidence, distinguishing *internal senders* (the service itself), *authorized third-party senders* (delegated email marketing/CRM infrastructure acting on behalf of the service), and *unknown third-party senders* (senders that cannot be attributed to the service or an authorized delegate). The important point is that once a user discloses an address to a service, that service may delegate delivery to third parties that send emails on its behalf. In practice, this expands who can reach the inbox beyond the brand the user believes they are interacting with.

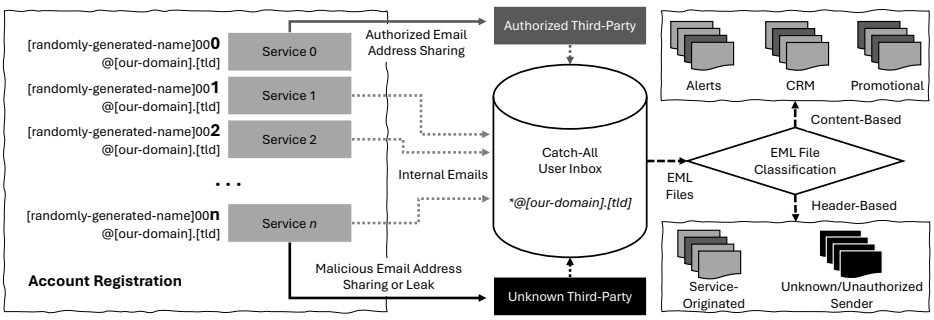

Figure 4.7: Framework for email collection, processing, and classification.

Evidence of expanded reachability. Over the collection window, the study received 4,847 emails from 109 of the 150 services and apps. Service-initiated email marketing was pervasive: 92.9% of the services sent marketing emails, and most of those services used *authorized third parties* as the primary sending channel. At the infrastructure layer, messages were overwhelmingly authenticated, with 99.96% passing SPF checks and 81.64% passing DKIM checks, indicating that the observed third-party senders were operating with explicit authorization rather than spoofing. Importantly, the study reports that no unknown third-party spam emails were detected during this year-long audit, suggesting that in this setting the dominant mechanism is not an obvious "leak $\rightarrow$ external spammer" pathway, but rather that more parties can reach the inbox because services routinely use authorized third-party senders.

Where the mail actually comes from. The study traces where emails originate in terms of network infrastructure (ASNs) and sending domains. This provides a way to understand the provenance of inbound email to an inbox. Once an address is enrolled with many services, the inbox becomes reachable via a broader and more heterogeneous set of third parties that operate content distribution networks. The fact that email transits these intermediaries is often invisible to users, and it creates a privacy risk when addresses are shared across more entities than users expect from what appears to be a single brand relationship. A small set of large infrastructures accounts for most observed volume (e.g., major email/marketing

providers and hyperscale clouds), which means that the "who can reach me" question is mediated by a relatively narrow but powerful sending ecosystem.

A more complete exploration of this intermediated sending ecosystem is covered in Chapter 5. That chapter shows how this sending ecosystem is concentrated and how those dependencies create cross-service tracking and shared failure domains.

What this means for visibility. This vignette shows, complementing the Ethereum case study, that visibility on open protocols comes with not just a security premium, but also a privacy cost. In Ethereum, reachability is an always-on host at a routable address that adversaries can enumerate and pressure. In the Simple Mail Transfer Protocol (SMTP), or email, an email address serves as a stable identifier that, once disclosed, becomes a lasting way to receive inbound contact (Klensin 2008). The takeaway is not that delegated email marketing is itself an "attack," but that reachability expands how many entities can reliably reach a participant. That expansion has security and privacy costs. It creates more channels for unwanted contact, the potential for abuse, and it complicates attribution when harm occurs because the visible sender and the infrastructure that actually sent the message often differ. Permissionless reachability (here, of an identifier), while useful, is also a resource others can use. Visibility therefore scales the potential attack surface for a participant of an open protocol such as SMTP.

4.4 Note: Cheap to find, cheap to target

The prior case studies focus on *participants* as targets (reachable nodes and reachable email inboxes). But one layer "up," many open protocol networks expose discovery planes that make it cheap to enumerate both *participants* and *resources* at scale. This reduces the cost of building target lists and monitoring new targets as they appear, even without deep interaction with each one. In a decentralized content distribution network such as BitTorrent, for example, the targets may be illicit torrents rather than the nodes hosting them.

Originally, BitTorrent relied only on the concept of trackers to find swarms (peers), but that was not enough to scale. The DHT (distributed hash table) was introduced to allow for decentralized discovery of swarms without a central directory. This same property enables large-scale crawling. In *MagnetDB* (Seidenberger et al. 2025b), an open-source DHT crawler (`magnetico`) was deployed from a self-hosted vantage point and configured to track and index torrents from up to 10,000 peers simultaneously. Over a 300-week collection period (December 2018–September 2024), this crawler gathered 28.6 million torrents consisting of more than 950.6 million individual files. Operating at this scale required substantial bandwidth (on the order of tens of terabytes per month), but the key lesson for this chapter is a qualitative one. The DHT makes large-scale enumeration feasible with straightforward engineering and a long enough time horizon to gather a large dataset.

The BitTorrent DHT serves as an example of a dual-use technology, as the same openness that makes the network resilient and permissionless also provides criminals trafficking in illicit media content a low-friction way to discover what exists, maintain watchlists, and generally be more effective. This is why *MagnetDB* intentionally withholds direct magnet links from its public release. The torrent *infohashes* and *magnet links* are stable identifiers that can be used to discover torrents and track their availability over time. The broader implication for open protocols is that discovery mechanisms are both key enablers of the network and reconnaissance channels whose observability properties shape who can target the ecosystem and at what cost.

4.5 Chapter Synthesis and Bridge

This chapter established a scoped claim that in the measured deployment settings, visibility is not a neutral protocol property but a measurable exposure multiplier. The Ethereum case study showed an incremental hostile-traffic premium for participating nodes, targeting focused on operationally exposed services, and a plausible pathway from traffic pressure to

validator-relevant degradation without protocol compromise. The email vignette showed the same mechanism in identifier-centric form where the stable reachability and delegated delivery of email expand contact surfaces through intermediated infrastructure.

What remains unresolved is the ecosystem-level consequence of these exposures. Visibility explains how participants become easy to find and pressure, but it does not by itself explain why risk and control concentrate into a relatively small set of providers, networks, and custodial intermediaries. Chapter 5 addresses that next step by shifting from exposure at the participant edge to dependence in the infrastructure core, showing how local optimization and delegation produce de facto centralization and hidden trust surfaces that shape privacy, security, and resilience outcomes.

Chapter 5

De Facto Centralization

Chapter 4 argued that in open protocol networks, *visibility is an attack surface*. Permissionless participation requires discoverability and reachability, but those same properties lower the cost of reconnaissance and enable sustained targeting. This chapter advances a complementary mechanism claim that even when a protocol is architected to be decentralized, *de facto centralization* often emerges because participants optimize locally under constraints. In practice, protocol networks become centralized not only through explicit governance decisions, but through the steady accumulation of *delegations*. Managed services, cloud hosting, outsourced compliance, and default routing choices incentivize convergence on a small set of intermediaries.

The thesis of this chapter is that **decentralized design is routinely undermined by local incentives**. When participants face operational constraints, they delegate to intermediaries, making centralization the equilibrium outcome. The relevant question for privacy, security, and resilience is not *what the specification allows*, but *who participants must trust to function under real constraints*.

This chapter proceeds as follows. Section 5.1 defines de facto centralization and the mechanisms that drive trust delegation. Section 5.2 presents the primary case study: email marketing ecosystems as a measurable instance of delegated custody, drawing on the *Why You've Got Mail* inbox analysis (Seidenberger et al. 2025a). Section 5.3 presents a secondary

case study in censorship-resistant DNS-over-HTTPS (DoH), showing how hyperscalers simultaneously shield users from blocking and concentrate critical dependencies (Seidenberger et al. 2026a). Section 5.4.2 provides the bridge to Chapter 6 by showing how delegated dependencies translate into physical chokepoints. Section 5.4.3 summarizes implications for evaluating decentralization claims.

5.1 Local Optimization and Delegation

Decentralization is often described as a logical rather than a physical protocol property: Do many authorities control the system? Is trust distributed? Can participants join without permission? That lens is necessary, but it is not sufficient on its own. In deployed protocol networks, security and resilience also depend on the *dependencies* a participant physically relies on to function—to send messages, resolve names, propagate blocks, or reach peers. Those dependencies are often created outside the protocol specification. A node operator chooses a cloud provider; an application chooses a marketing platform; a browser chooses a default DoH resolver; a censorship-circumvention tool chooses a rendezvous strategy that meets real performance constraints.

5.1.1 De Facto Centralization as Effective Dependence

De facto centralization is defined here:

> *A protocol network is de facto centralized to the extent that successful participation depends on a small, concentrated set of third parties regardless of whether the protocol specification is logically decentralized. These third parties can include infrastructure providers, service intermediaries, or even reliance on a specific legal jurisdiction.*

This definition is a sociotechnical one, focusing on *effective trust and dependence* and what we actually observe, not on design objectives or logical definitions alone. De facto centralization asks: *who do participants delegate trust to*, and *how concentrated is that reliance*?

The dependency structure can be represented as a bipartite graph. Let V denote protocol participants (nodes, senders, resolvers, content producers), and let D_ℓ denote dependencies at layer ℓ (ASNs, cloud providers, email service providers, CDNs). An edge (v, d) exists if participant v depends on dependency d for some operation. Concentration in D_ℓ under the induced edge distribution—measured via top-k shares, Gini coefficients, or the Herfindahl–Hirschman Index introduced in Chapter 3—quantifies how much effective dependence concentrates in a small set of dependencies.

This framing connects to the threat model philosophy (Chapter 1). *Structural adversaries* are actors who pursue convenience, profit, and performance—they are optimizers. They are not attacking the protocol, but their local optimization reduces diversity and creates correlated failure domains.

5.1.2 Mechanisms That Make Delegation the Equilibrium

Across the protocol families studied in this dissertation, participants seek to minimize operational burden and maximize performance under uncertainty. The result is a shift from "run it yourself" to "delegate it to specialists." Four sub-mechanisms are especially consistent.

Convenience. Operating infrastructure is dominated by fixed costs and economies of scale. When those costs exceed the perceived marginal benefit of self-operation, managed services dominate. This is the general mechanism behind hosted RPC endpoints in blockchains, outsourced email delivery in SMTP ecosystems (Klensin 2008), and third-party DoH resolvers. Convenience reduces friction for adoption and makes participation accessible to non-experts, but it concentrates control in those who can amortize operational costs across many customers.

Latency heuristics. In decentralized networks, performance is often measured by metrics such as latency, fault-tolerance, and throughput. Heuristics that "prefer fast peers" or "use the nearest resolver" are individually rational, but collectively pull structure toward well-connected hubs, regions, and providers. Over time, such heuristics create "corridor pull," where network flows concentrate along a few high-performance paths and then reinforce those paths through feedback loops (Chapter 3). The results from the EtherBee study show that Ethereum's byte-weighted peer focal point drifts toward North America–Europe submarine cable corridors, driven by client-side peer pruning based on latency (Seidenberger and Maiti 2025a).

Reputation coupling. Reputation often attaches not only to individual actors but also to the infrastructure they share. Reputation management in email is a canonical example. Email depends on the reputation of IP ranges, sending domains, and providers. If an IP address has been associated with spam, messages from that IP get blocked. This creates an externality where senders who share infrastructure inherit each other's reputational consequences, even without coordination. Managing reputation requires scale and expertise, so the ecosystem converges on a few large intermediaries that can handle abuse monitoring. The result is that many unrelated brands become connected through the providers they have in common. Section 5.2 shows how a concentrated email marketing infrastructure creates a de facto "reputation commons," coupling many brands through shared sending providers and IP space.

Operational outsourcing. As protocol networks grow in economic significance, they attract regulatory attention, and operators face compliance costs they may not be equipped to handle. In response, operators delegate key functions to vendors with compliance certifications, legal teams, and dedicated security resources. Even absent formal regulation, ecosystems converge on specialized intermediaries for security monitoring, spam filtering, CDN caching, and related services (Feulner et al. 2025). Once a certain trust delegation becomes

commonplace in an ecosystem, it becomes positively reinforced through social mechanisms. Path dependence makes reversal costly, and small providers or hobbyist deployments become less viable even when the protocol treats all participants as equals (Pierson 2000).

5.1.3 Measuring Hidden Centralization

The challenge is that de facto centralization is often invisible in protocol specifications. Measurement must target dependencies that the spec omits. The dissertation's approach treats dependencies as empirical objects: if a dependency influences correctness, availability, or privacy in deployment, it belongs in the model regardless of whether it appears in the protocol specification.

Practically, this means constructing dependency graphs from linkage keys exposed by telescopes (Chapter 2):

- **Network provenance:** IP $\rightarrow$ ASN $\rightarrow$ provider/region. This mapping converts a protocol participant's advertised endpoint into an infrastructure dependency.
- **Application provenance:** domain names, DKIM signing domains, SPF-authorized infrastructure, and header-level routing traces (for email).
- **Rendezvous identifiers:** resolvers, DoH endpoints, IPNS names, bootstrap servers (for encrypted DNS and related systems).

Concentration should be reported at multiple layers—provider, ASN, and service—and in both entity space (how many participants use a dependency) and volume space (how much activity flows through it). Volume-weighted concentration often reveals more severe centralization because large participants account for disproportionate traffic.

With this framework established, we turn to two empirical settings where delegation is especially measurable: email marketing ecosystems (Section 5.2) and censorship-resistant DoH (Section 5.3).

5.2 Case Study: Who Is Actually Sending You Email?

Many brands, few pipes.

Email is an archetypal protocol network because anyone can run a mail server, the specifications are public, and no central authority controls participation. Yet in reality, email as we know it today is dominated by a few large platforms and is deeply intermediated. It has become much less of a protocol network and more of a common transport standard. Modern services rarely send mail directly from infrastructure they operate. Instead, they delegate sending and reputation management to specialized providers—email service providers, CRMs, and marketing platforms. This makes email a useful setting for studying trust delegation because (i) message headers expose rich provenance signals, (ii) incentives create strong reputational coupling, and (iii) email addresses (inboxes) provide a passive sensor for longitudinal measurement.

This section draws on the inbox sensor telescope described in Chapter 2 and the *Why You've Got Mail* analysis (Seidenberger et al. 2025a). The core finding is that **email involves more parties than users realize**. When a user receives a message from a brand, they naturally assume the interaction is bilateral—"this company is emailing me." In practice, most brands delegate email delivery to third-party infrastructure providers. The user's email address, metadata, and interaction history flow through intermediaries that the user never explicitly consented to and may not know exist. This information asymmetry matters for privacy: the trust relationship the user believes they have with a brand is actually mediated by a concentrated set of infrastructure providers operating behind the scenes.

5.2.1 What the Inbox Reveals

The study design is described in Chapter 2 and illustrated in Figure 4.7. Over the year-long collection period, the study received 4,847 emails from 109 of the 150 registered services. A first important finding is what *did not* happen: **no unknown third-party spam was**

detected. Every email received could be traced to a service the user had registered with or an authorized third party acting on that service's behalf. Of the emails received, 99.96% passed SPF (Sender Policy Framework) checks and 81.64% passed DKIM (DomainKeys Identified Mail) checks—both authentication mechanisms that verify the sender is authorized by the domain owner. This suggests that opting out during registration is effective at preventing emails from truly unknown sources, at least among popular services operating under post-GDPR and post-CCPA regulatory environments.

However, the emails that *did* arrive tell a more complex story about who handles user data. The study distinguishes three source types:

1. **Internal senders:** the service sends email directly from its own infrastructure.
2. **Authorized third parties:** the service delegates sending to a provider who transmits email on the service's behalf, authenticated via SPF and DKIM.
3. **Unknown third parties:** emails from entities the user has no relationship with (none detected in this study).

The central empirical observation is that delegated sending is the default. Most brands do not operate their own email servers. They delegate the mechanics of delivery to third-party providers who run the infrastructure, manage IP reputation, and handle abuse complaints. This means a small number of intermediaries end up handling most of the email traffic in the ecosystem.

5.2.2 Infrastructure Concentration

How concentrated is the email sending infrastructure? The study measures concentration at two layers: sending domains (the organizational identity) and autonomous systems (the network infrastructure).

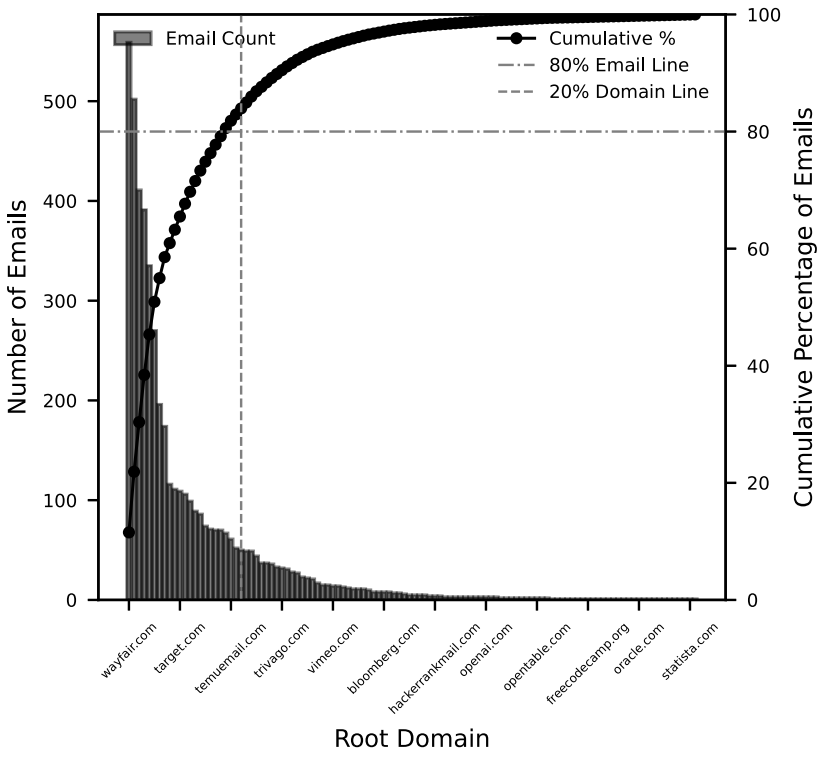

Figure 5.1: Pareto diagram of email volume by root domain. The distribution follows the 80-20 principle: a few large senders dominate total volume, with the top 10 root domains accounting for 63.23% of all emails received.

Domain concentration. Email identity is mediated by domains, but observed sending domains are heavily concentrated. The **top ten root domains account for 63.23% of all emails received**. The distribution has high skewness (3.55) and kurtosis (12.92), indicating a heavy-tailed pattern where many brands exist, but a small number of senders dominate volume. Figure 5.1 shows this Pareto distribution. Even when a user's inbox appears diverse—"many different companies email me"—the underlying sending ecosystem is concentrated.

Network infrastructure concentration. The concentration becomes sharper when tracing emails to their network origins. By mapping each email's sending IP address to its registered Autonomous System Number (ASN), the study reveals who actually operates the sending infrastructure. **The top eight ASNs account for 89.35% of all emails**. The

dominant intermediaries are specialized email service providers—Salesforce, Mailgun, SendGrid, SparkPost—along with cloud platforms such as Amazon Web Services and Google Cloud Platform.

Figure 5.2 visualizes these flows. The left side shows ASNs—the actual networks that the mail is sent from—while the right side shows the brands those emails represent. The pattern shows many independent brands, but their emails route through a small set of shared intermediaries. Some companies do send from their own registered ASN (Wish.com, LinkedIn, Adobe), but most delegate to specialized third parties.

This concentration has direct implications. Many unrelated companies and their services have shared risk through the infrastructure they have in common. An outage or misconfiguration at a major email provider can affect many brands simultaneously. From a privacy perspective, these intermediaries can observe metadata across many services—even if they do not analyze message content, they see which brands email which users and when. This gives them significant information power across markets.

5.2.3 Reputation and IP Hopping

Why does email concentrate on a few intermediaries? Convenience and economies of scale explain part of it, but email also concentrates because **deliverability depends on reputation**. Spam filters evaluate not just message content but the reputation of the sending IP address and domain. If an IP has been associated with spam, messages from that IP may be blocked or sent to spam folders. This creates an externality in that one sender's bad behavior can affect others who share the same infrastructure.

Managing reputation requires expertise and scale. Operators must track when recipients mark messages as spam or when mail bounces, respond to abuse complaints, and sometimes rotate IP addresses to avoid accumulated reputational damage. These capabilities favor large intermediaries who can amortize costs across many customers. Small senders often cannot justify the investment, so they delegate to specialists.

The study provides evidence of active reputation management. By cross-referencing each sending IP against a third-party reputation database, the analysis found 612 spam reports associated with 84 unique IP addresses across 29 companies. Figure 5.3 visualizes this relationship hierarchically. The ASN boxes contain the companies that send through them, whose area is proportional to the number of spam reports. The pattern reveals that even well-known brands may rely on infrastructure that carries varying levels of reported spam activity—not because those brands are spamming, but because they inherit the reputation characteristics of their shared infrastructure.

There is also evidence of "IP hopping"—distributing email across many IP addresses to dilute the impact of any single IP being blacklisted. The study found a strong positive correlation between the number of unique IPs a company uses and the total spam reports associated with those IPs (Pearson's $r = 0.71$, $p < .0001$; Spearman's $r_S = 0.55$, $p = .002$). Companies with higher spam report volumes tend to spread their sending across more addresses, consistent with reputation management strategies.

These findings illustrate how shared infrastructure creates shared fate. If many brands use the same intermediaries, their deliverability and reputation events become correlated—not through coordination, but through common dependence. This is both an availability concern (an intermediary outage affects many brands) and a privacy concern (the intermediary observes activity across many services).

5.2.4 What This Means for Users

The consequences of hidden de facto centralization in an open protocol network such as email generalize beyond SMTP. From a privacy perspective, users perceive email as a bilateral relationship—"this company is emailing me"—but the actual custody chain is multilateral. Third-party intermediaries can observe metadata such as which users receive mail from which brands, and when across many services. Even without analyzing message content, these

intermediaries gain cross-context visibility into user behavior. This is a privacy exposure that users did not consent to and likely do not realize exists.

The security picture is similarly double-edged. When 89% of email flows through 8 ASNs, compromises at those intermediaries have outsized impact—a single breach can expose data or disrupt communications for many brands at once. However, the same concentration creates defensive leverage as security improvements deployed at major intermediaries benefit the entire ecosystem. System resilience follows the same logic. Outages or policy changes at major email providers can affect many brands simultaneously. This is a common-cause failure mode induced not by protocol design but by operational delegation. The protocol specification says nothing about Salesforce or SendGrid, yet those providers' availability determines the health of a large subset of the ecosystem.

Finally, concentration creates regulatory opportunity. Policymakers seeking to improve opt-out compliance could focus attention on the small number of third-party marketing services that handle most traffic, rather than attempting to regulate thousands of individual brands. The infrastructure chokepoint is also a policy chokepoint. In short, email provides a clear empirical demonstration of this chapter's central claim: decentralization at the protocol level ("anyone can run a mail server") coexists with centralization in practice ("a handful of intermediaries deliver most mail"). Although email is an open and decentralized protocol network in theory, it is de facto centralized in practice.

5.3 Case Study: Censorship-Resistant DNS

Hide in the crowd they cannot afford to disperse.

The email case study examined a decentralized protocol network where de facto centralization emerges from convenience economics and reputation management. This section examines DNS, which is *not* inherently a decentralized protocol network but a hierarchical distributed system with centralized governance (root servers, registrars, TLD operators).

DNS is relevant to this dissertation because the *infrastructure layer* beneath DNS exhibits the same concentration patterns, and that concentration can be strategically exploited.

Censors weaponize DNS's hierarchical structure. Because DNS queries traverse a predictable path from stub resolver to recursive resolver to authoritative servers, censors can intercept or poison queries at choke points they control. These include ISP-operated resolvers, national DNS infrastructure, or network-level firewalls. DNS-based censorship is low-cost and effective precisely because the system is centralized enough to offer convenient interception points. Encrypted DNS protocols such as DNS-over-HTTPS (DoH) respond by hiding queries within ordinary HTTPS traffic, but this only shifts the problem: *where should the encrypted resolver be hosted?* Public resolvers (Cloudflare, Google, Quad9) are easily enumerated and blocklisted (Pearce et al. 2017). Private resolvers require infrastructure that may itself be blocked.

Here is where this chapter's central mechanistic claim becomes relevant. The same infrastructure concentration that creates de facto centralization in decentralized protocol networks—the convergence on a small number of cloud providers, ISPs, and CDNs—also creates *collateral damage constraints* that censors must respect. Hyperscalers host essential services that censors cannot afford to block. Because of this, the concentration of internet infrastructure can be leveraged to creatively increase the privacy and security of DNS as a protocol.

This is the design space that NinjaDoH (Seidenberger et al. 2026a) explores. The system uses capabilities enabled by hyperscaler infrastructure to create censorship-resistant properties, accepting dependence on AWS or similar providers in exchange for the protection that their economic importance provides. The trade-off is explicitly that operational independence is sacrificed for practical viability. This mirrors the broader pattern of this chapter that participants optimize under constraints, and the resulting equilibrium concentrates dependence on a small set of intermediaries. Here, that concentration is leveraged as part of

a *deliberate design choice* that exploits the same infrastructure consolidation that creates vulnerabilities elsewhere.

The constraints that drive this choice are threefold:

1. **Evasion:** The resolver must be difficult to block. Traditional approaches—fixed IP addresses, stable domain names—are trivially enumerated and added to blocklists.

2. **Performance:** The resolver must be fast enough for everyday use. Solutions that route DNS through more decentralized anonymization networks (e.g., Tor) incur latency overhead that degrades user experience and limits adoption.

3. **Deployability:** The resolver must be operable with realistic effort and cost. Exotic or expensive infrastructure requirements limit who can provide the service.

Under these constraints, NinjaDoH implements a moving-target DoH service hosted entirely within hyperscaler address space. The system's design and evaluation illuminate both the benefits and the costs of this approach.

5.3.1 Hyperscalers as a Shield

The first part of the trade is straightforward in that hyperscaler infrastructure is difficult to block because it hosts economically essential services. AWS alone hosts hundreds of thousands of websites and applications. Blocking the entire AWS IP address space to prevent access to a DoH resolver would also block e-commerce, banking, media, and enterprise services that depend on the same infrastructure. Most censors, whether state actors, ISPs, or enterprise administrators, are unwilling to accept this collateral damage.

NinjaDoH explicitly leverages this constraint. The system operates entirely within hyperscaler address space, using cloud-allocated IP addresses that rotate frequently:

> "Entirely blocking outbound HTTPS traffic to hyperscalers such as AWS, GCP, and Azure is impractical for most censors" because these providers host "numerous essential web services" (Seidenberger et al. 2026a)

This is another example of the sociotechnical nature of these planet-scale distributed systems. The fact that a censor would be unwilling to block an entire hyperscaler address space is not inherent censorship resistance but rather *political-economic* censorship resistance. The censor's optimization problem must account for the cost of blocking legitimate services that constituents depend on. The very property that makes it a de facto centralization concern can become a shield for certain services on the public Internet.

The mechanism generalizes to domain fronting, CDN-based circumvention, and related techniques that exploit the same constraint. Censors cannot easily block infrastructure that carries traffic they need to permit. The recurring pattern is that *concentration creates collateral damage constraints*, and those constraints can be exploited by privacy and censorship-resistance tools. But exploiting this shield requires accepting dependence on the hyperscaler's address space, compute resources, and policies (Fifield et al. 2015).

5.3.2 Hyperscalers as Dependency

The second part of the trade is the cost. Using hyperscaler infrastructure as a shield creates dependence on that infrastructure. The system's liveness becomes contingent on components that the censorship-resistant system designer does not control.

NinjaDoH illustrates these dependencies by designing and implementing a moving-target DoH resolver that rotates public IP addresses on a configurable interval (1–3 minutes in evaluation) and publishes updates via IPFS/IPNS (a decentralized content delivery network) so clients can discover the current address. Figure 5.4 shows the architecture. The server orchestration loop allocates new IP addresses from the hyperscaler's pool, generates fresh TLS certificates for the new addresses, and publishes routing information to IPNS. The system maintains a "ladder" of overlapping IP addresses so clients can update without service interruption.

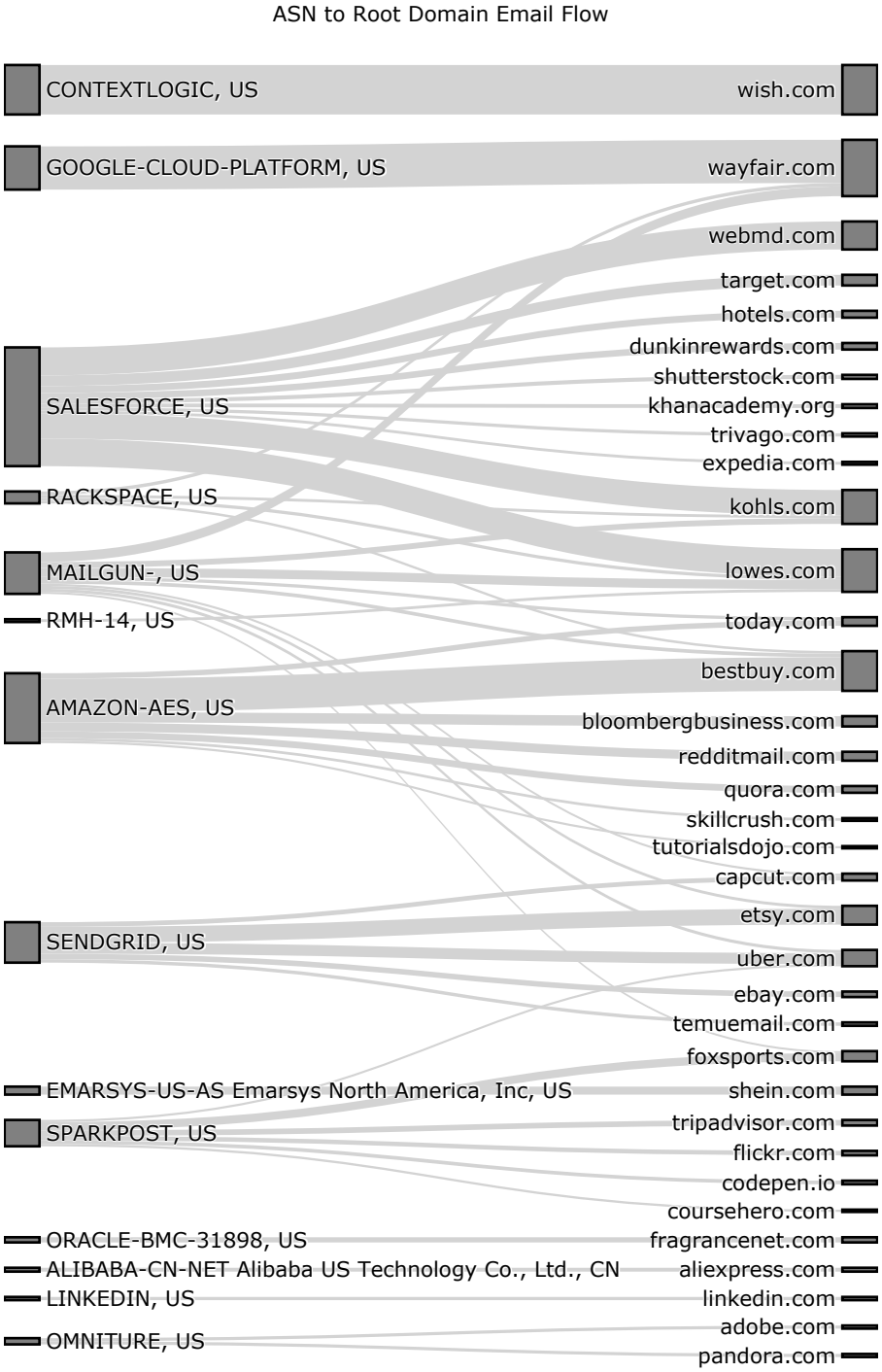

Figure 5.2: Flow diagram of the top 50 email sender relationships. The left column shows ASNs (the actual sending infrastructure), and the right column shows the brands those emails represent. Most marketing email flows through a small set of intermediaries, regardless of the nominal brand identity.

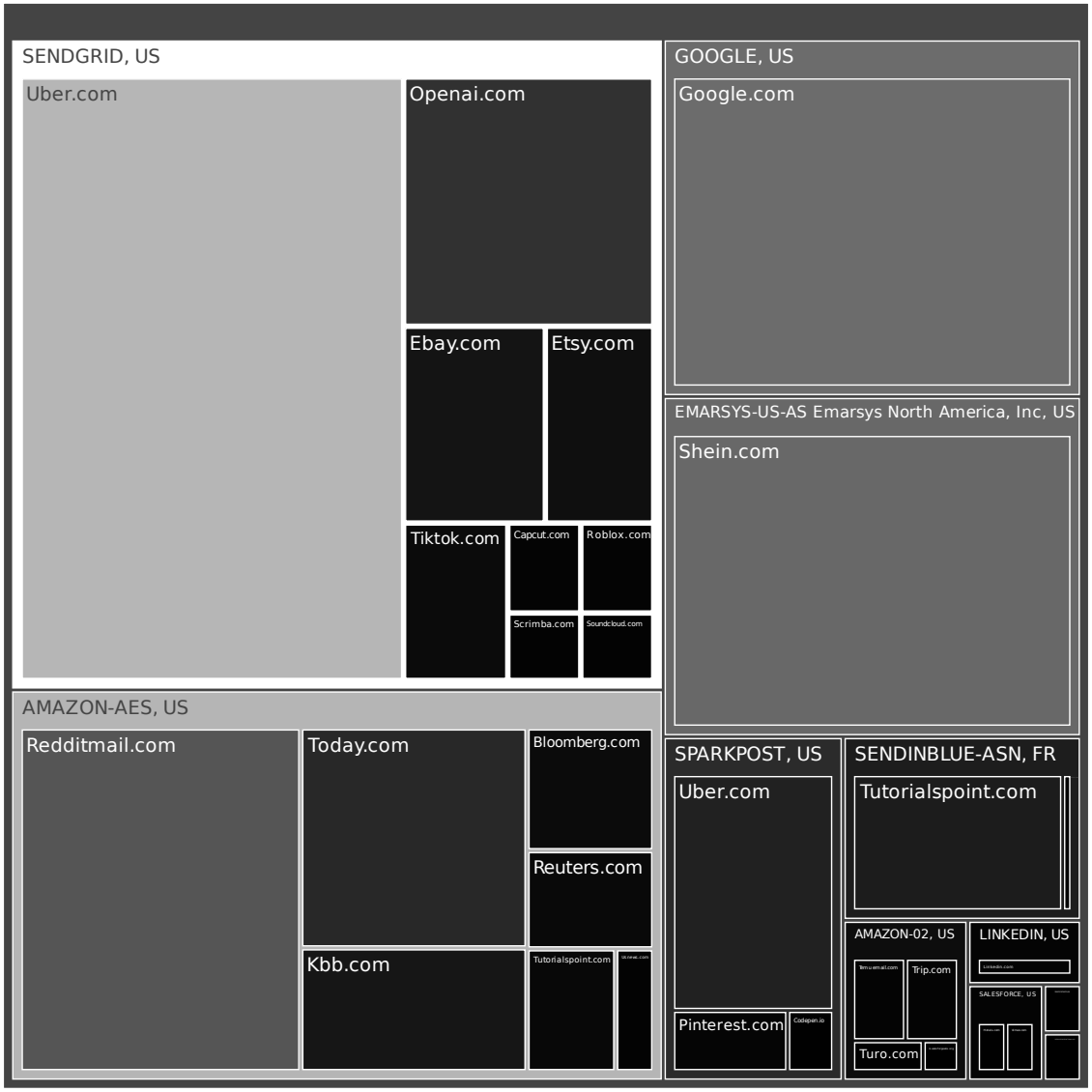

Figure 5.3: Hierarchical treemap of community-reported spam associated with sender IPs, organized by ASN and company. Lighter shades indicate higher spam report counts. Even well-known brands may rely on infrastructure with varying reputation histories.

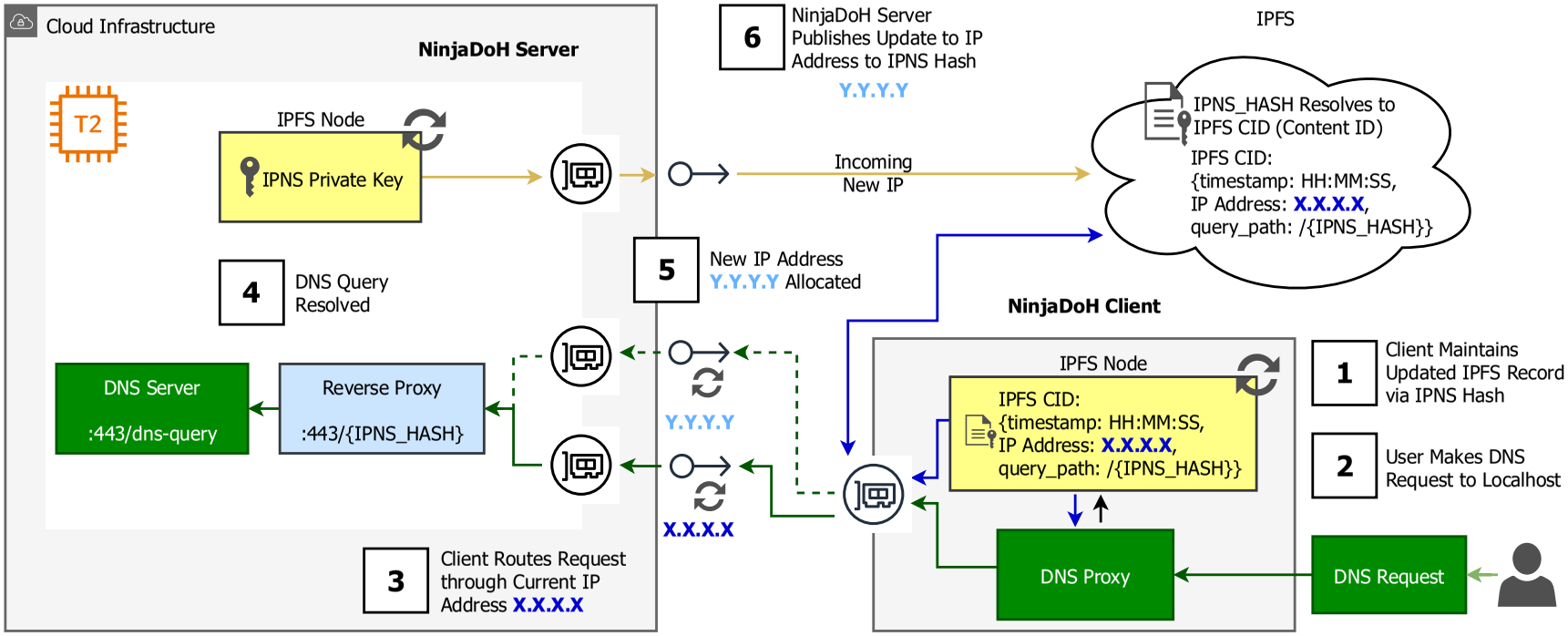

Figure 5.4: NinjaDoH architecture. The client retrieves the server's rotating IP address via IPNS, while the server cycles through the hyperscaler's IP allocation pool. Censorship resistance is achieved through collateral damage constraints, but the system depends on hyperscaler infrastructure and IPFS/IPNS reachability.

This architecture exemplifies de facto centralization:

- **IP allocation:** The system requires the hyperscaler to permit rapid IP allocation and release. AWS's IPv4 address pool (estimated at over 100 million addresses at time of writing) enables this, but the capability depends on AWS's continued policies. A change in allocation limits, abuse detection, or pricing would directly affect the system's viability.
- **Compute and network availability:** The DoH resolver runs on hyperscaler compute (a `t2.small` instance in the prototype). Outages, region-specific failures, or account-level actions affect service availability.
- **IPFS/IPNS reachability:** Clients discover the current IP address via IPNS, a decentralized naming layer built on IPFS. While IPFS is itself distributed, clients often access it through centralized gateways. The connection between the client and the NinjaDoH server depends on this layer remaining reachable.

None of these dependencies appear in the DoH protocol specification. A user reading the RFC would see a standard for encrypting DNS queries, not a requirement to trust AWS's IP allocation policies or IPFS gateway availability. Yet in this deployed system, those dependencies determine whether the user can resolve names. The point here is that de facto centralization is not always an absolute negative. There are real tradeoffs along the spectrum between decentralized and centralized designs, and those tradeoffs can be used for either beneficial or harmful ends. NinjaDoH illustrates why centralization can still be attractive in practice as it is significantly more performant and deployable than more decentralized alternatives such as Tor.

5.3.3 Why Outsourcing Infrastructure Wins

NinjaDoH provides quantitative evidence for why hosting on hyperscaler infrastructure rather than operating independent infrastructure can become an equilibrium design choice. The

data show that the alternatives, either sacrificing performance (Tor-based approaches) or sacrificing evasion (static infrastructure), are unacceptable under the threat model that real users face.

Performance: the latency gap that drives adoption. Censorship-resistance tools must be fast enough for everyday use; otherwise, users abandon them for less secure alternatives. The evaluation measured DNS resolution latency across multiple configurations. NinjaDoH achieves a mean ping-adjusted resolution time of **12.68ms**, compared to 7.77ms for public DoH resolvers (Cloudflare, Google) and 7.85ms for a control server on identical infrastructure without the moving-target overhead. The 4–5ms overhead from the client proxy and IP rotation is negligible for practical use. The comparison that matters for censorship circumvention is against DNS over Tor, the most common alternative for hiding DNS queries. DNS over Tor achieves a mean resolution time of **601.16ms**, roughly 50 times slower than NinjaDoH. Users will not tolerate half-second delays on every DNS lookup when browsing the web. The hyperscaler-based design achieves censorship resistance at a latency cost users can accept.

Evasion: what the moving target buys. NinjaDoH is systematically tested against prevalent DoH blocking techniques. The system successfully evades **4 of 5 common blocking methods** employed by commercial firewalls. Table 5.1 summarizes the results. The only method that remains effective is strict IP allowlisting, where all non-approved destinations are blocked by default. This approach is expensive to maintain and disrupts legitimate traffic, making it impractical for most network environments.

Against machine learning-based detection, the results are similarly favorable. State-of-the-art ML models trained to detect DoH traffic achieved an average recall of only 0.506 against NinjaDoH, which is essentially random guessing (MontazeriShatoori et al. 2020). Even against an adaptive adversary who specifically trains models on captured NinjaDoH traffic, the best model achieved a precision of 0.764, recall of 0.635, and F1-score of 0.578.

Table 5.1: NinjaDoH evasion of common DoH blocking methods. Adapted from Seidenberger et al. (2026a).

Blocking Method	**Evaded?**	**How NinjaDoH Responds**
Domain blocking	✓	Does not use domain names; clients connect directly to IP addresses discovered via IPNS.
IP blocking	✓	Moving-target strategy rotates IP addresses faster than blocklists can update.
Application identification	✓	Uses randomized query path (not the standard `/dns-query`), defeating active probing.
SNI blocking	✓	TLS handshake uses IP-based certificates; SNI field is empty or omitted.
IP allowlisting	×	Strict allowlisting blocks all unknown IPs; dynamic rotation is ineffective.

These metrics are insufficient for reliable censorship because at this low level of precision, blocking detected traffic would produce unacceptable false positive rates.

Cost: outsourcing can be cheaper. Hosting on hyperscaler infrastructure can also be economically rational. The prototype NinjaDoH server costs approximately $23.55 USD per month on AWS, which is less than two premium VPN subscriptions and affordable for small groups or privacy-conscious individuals. Operating equivalent infrastructure independently, outside a hyperscaler's address pool, would require acquiring IP address space, managing BGP announcements, handling abuse complaints, and maintaining compute resources. The capital required to independently operate the infrastructure at scale for this system exceeds what most users or small operators would be able to deploy.

These results reinforce the chapter's central argument. **Outsourcing infrastructure to hyperscalers is a rational response to constraints**. Users need performance, operators need flexibility, and the cloud provides capabilities that have led to novel system architectures.

Hosting on AWS satisfies all three at acceptable cost and results in de facto centralization. A tool that is architecturally independent of any particular provider becomes operationally dependent on whichever provider can satisfy the constraint set. The protocol specification says nothing about AWS, but the deployed system cannot function without it. In fact, much of the modern cloud computing stack is so reliant on certain cloud capabilities at scale that business models are dependent on them (Seidenberger et al. 2026a).

5.4 Synthesis

5.4.1 Parallel Structure, Different Pressures

The two case studies reveal the same structural pattern driven by different pressures. In email, brands outsource sending and reputation management to specialized intermediaries who can amortize fixed costs and maintain deliverability at scale. In censorship-resistant DNS, tool designers outsource compute and IP provisioning to hyperscalers because it is the only option that satisfies evasion, latency, and cost requirements simultaneously. The outcomes are structurally parallel:

- **Shielding via concentration:** Hyperscalers provide protection precisely because they are concentrated. The collateral damage from blocking AWS protects services hosted within AWS—analogous to how large email intermediaries' reputation protects smaller senders who route mail through them.
- **Leverage via concentration:** The same concentration creates leverage points. A hyperscaler policy change, account suspension, or regional outage can affect many circumvention deployments simultaneously. If AWS were to prohibit moving-target IP allocation or if IPFS gateways were blocked in a censored region, NinjaDoH users in that region would lose service.

- **Shared failure domains:** Multiple independent circumvention tools that adopt the hyperscaler strategy become fate-shared through their common infrastructure dependence, just as multiple email brands become connected through common sending infrastructure.

What distinguishes the DNS case study is the *explicit* nature of the trade. NinjaDoH's design documentation acknowledges the hyperscaler dependence and articulates the threat model assumptions that make it acceptable: neutral cloud provider, no endpoint control by the adversary, DNS-based (not IP-based) censorship (Seidenberger et al. 2026a). This transparency enables operators and users to evaluate whether the dependence fits their specific threat model. De facto centralization is often invisible; making it visible is the first step toward managing it.

5.4.2 From Institutional to Physical Concentration

Outsourcing operations to concentrated intermediaries can centralize *infrastructure geography and routes*. When many services converge on the same cloud providers, ASNs, and regions, they inherit the physical constraints of those infrastructures: undersea cables, metro-area fiber corridors, shared IXPs, and regional power and cooling dependencies.

$$\text{Outsourcing} + \text{scale} \rightarrow \text{Infrastructure concentration} \rightarrow \text{Correlated failure domains} \rightarrow \text{Cascades under shock.}$$

The EtherBee measurements show that Ethereum's byte-weighted peer focal point drifts toward North America–Europe submarine cable corridors, driven by client-side peer pruning based on latency (Seidenberger and Maiti 2025a). This "corridor pull" occurs even though the participant set remains globally distributed. Over time, such concentration produces the corridor dependence patterns that make physical disruptions disproportionately consequential. The email case study shows a parallel pattern at the institutional layer where many brands share a small set of ASNs and providers handling most traffic. The DoH case study

shows it at the infrastructure layer where censorship resistance is achieved by depending on centralized hyperscalers.

Chapter 6 treats physical chokepoints as an *endogenous consequence* of the outsourcing patterns studied here, not as a separate phenomenon. The resilience modeling later in the dissertation emphasizes that concentration and drift can shift the system between regimes: from "many independent failures" to "common-cause shocks that trigger cascades."

5.4.3 Chapter Takeaways

This chapter has advanced a mechanism claim that complements Chapter 4's visibility argument.

1. **Centralization is often a behavioral equilibrium.** Protocol networks can be logically decentralized while becoming operationally centralized through incentives, defaults, and governance constraints.

2. **De facto centralization is best defined as effective dependence.** The relevant question is "who you must trust to function," not "what the specification claims."

3. **Outsourcing operations creates hidden trust surfaces.** Brands outsource email sending to marketing platforms; circumvention tools outsource hosting to hyperscalers. In both cases, custody of user data and availability of service flow through intermediaries that users may not know exist.

4. **Measurement must target dependencies that the spec omits.** Dependency graphs, multi-layer concentration indices, and provenance analysis are the appropriate toolkit for evaluating decentralization claims in practice.

5. **Outsourcing couples security, privacy, and resilience outcomes.** When many participants route traffic or host services through the same intermediaries, those intermediaries gain cross-context visibility, become attractive attack targets, and create correlated failure domains.

The empirical anchors of this chapter illustrate the dissertation's methodological take that protocol networks should be evaluated by *measured dependence*, not by architecture diagrams. In email, a diverse surface of brands masks a concentrated custody layer—89.35% of volume flowing through 8 ASNs, with Salesforce, Oracle, Intuit, and Amazon handling the bulk of delivery. In censorship-resistant DNS, a moving-target design reduces censorship leverage at the DNS layer but induces dependence on hyperscaler infrastructure.

With these mechanisms established, Chapter 6 shows how de facto centralization manifests as physical chokepoints, how they can be measured and discovered, and how those chokepoints amplify correlated shocks into systemic failures.

Chapter 6

Overlay Dependence and Physical Chokepoints

Decentralized systems fail at the speed of their smallest cut-set. You can spread a protocol across many machines, but it still runs on the same Internet. Traffic must pass through real routes, real networks, and real physical links, and there are only so many of them. When many operators make the same sensible choices (the fastest path, the cheapest provider, the default service), their "independent" nodes end up sharing the same weak points. Then a small fault becomes a shared failure that hits everyone at once.

Chapter 5 showed that delegating trust to intermediaries is a predictable outcome that concentrates control in otherwise decentralized networks. This chapter pushes that claim down a layer to the physical bottlenecks that global decentralized networks inherit from the underlying Internet. Client software choices and operational convenience can funnel traffic into a small set of routes between regions, creating shared failure points that remain invisible if one only looks at where nodes are located on a map. Systems that are logically decentralized can become physically centralized along a handful of critical paths, so that an ordinary disruption becomes a correlated shock affecting many participants at once. This chapter supports that claim with measurements from Ethereum's P2P network using the EtherBee dataset (Seidenberger and Maiti 2025a) and the analysis in Seidenberger et al.

(2026c). It then connects the same pattern to BitTorrent, where federated file-sharing swarms depend heavily on a few key participants (Seidenberger et al. 2026b).

6.1 Chokepoints in Ethereum P2P

Ethereum's security and liveness depend on fast propagation of blocks, attestations, and related gossip-layer messages (Buterin et al. 2020). Yet the delivery of such messages relies on the public Internet. Even if nodes are nominally spread across many countries and jurisdictions, the paths that actually carry traffic may be far more concentrated than it would seem. Because the EtherBee dataset contains records of millions of P2P sessions over months, it reveals which peers exchange the most data and which routes between regions carry the bulk of traffic. Even when nodes are geographically distributed, the *effective topology* contracts toward core regions and a small set of intercontinental linkages.

6.1.1 Why Chokepoints Emerge Even in Open Networks

Chapter 4 treated the node as the primary unit that is discoverable, reachable, and therefore attractive to adversaries. That framing is part of a larger picture. The topology of the network is itself a target and a determinant of network resilience. In graph-theoretic terms, two networks with identical node counts can differ sharply in fragility because they differ in their minimum cut-set, which is the smallest set of edges whose removal partitions the graph. A network that funnels traffic through a few paths has a small cut-set across that bottleneck even if its node count is large. Full disconnection is the extreme case; even removing or degrading edges that do not partition the graph can force traffic onto longer paths, increasing latency. In a consensus protocol where timeliness matters, longer paths mean slower propagation, and slower propagation degrades performance well before any partition occurs.

This extends the thesis from Part I (Chapters 1–3) that risk depends not only on what is deployed but also on how it is connected. Under stress, message propagation depends on the highest-volume connections and the stability of the peers that carry that traffic. If those connections all rely on the same physical routes, then disruption along those routes degrades the entire protocol without requiring many nodes to go offline.

The mechanism is straightforward. Ethereum clients keep a limited set of peers and use response time (RTT, round-trip timing/latency) as one signal of peer quality. Clients implement peer-scoring algorithms that penalize and drop slow peers while keeping faster ones. Over time, this favors low-latency connections and sheds high-latency ones. Because latency between two hosts on the Internet is highly correlated with geographic distance (Cortes-Goicoechea et al. 2023; Seidenberger and Maiti 2025a), this process pulls the effective peer set toward nearby regions and concentrates long-distance connectivity into whichever routes happen to be fastest. Especially across intercontinental distances, the well-provisioned submarine cables linking major data-center hubs dominate (Durairajan et al. 2015). Over time, this produces two effects: the average distance to active peers shrinks, and the few remaining long-distance connections all use the same intercontinental links.

This pattern is not unique to Ethereum. It is an instance of the positive feedback loop described in Chapter 3, where local optimization reshapes the network, and the reshaped network changes future local optimization. Prior work on Ethereum has shown this effect, where validators located in non-core geographies earn fewer rewards because their attestations are, on average, slightly behind the head of the chain (Cortes-Goicoechea et al. 2023; Seidenberger and Maiti 2025a).

6.1.2 How EtherBee Measures Route Concentration

Whether traffic concentrates into a few physical routes requires observing actual connections over time and weighting peers by how much data they exchange. EtherBee was designed for this kind of cross-layer measurement. It combines session metadata—who connected to

whom, when, and how much data they exchanged—with node context and infrastructure information, so that daily peer sets can be reconstructed and mapped into physical space. Chapter 2 introduced EtherBee as a multi-source node observatory, and here it serves as a way to infer where effective connectivity accumulates *in practice*, not just where nodes exist "on paper."

Weighting peers by traffic volume. For each day and vantage point we take the set of *distinct* peers that exchanged P2P traffic with that host and weight each peer by the total bytes exchanged that day. Heavier-traffic peers then exert more influence on the metrics, so that the "effective" peer set reflects where data actually flows, not just how many peers appear. Formally: let t index days and let $\mathcal{P}_t$ be the set of peers that exchanged traffic with the vantage point on day t. For each peer $i \in \mathcal{P}_t$, let $b_{i,t}$ be the total bytes exchanged with that peer on day t. The normalized traffic weight is

$$w_{i,t} = \frac{b_{i,t}}{\sum_{j \in \mathcal{P}_t} b_{j,t}}. \tag{6.1}$$

So $\sum_{i \in \mathcal{P}_t} w_{i,t} = 1$; the $w_{i,t}$ are used in the weighted average distance and in the focal-point (center-of-mass) calculation below.

Each peer's IP address is mapped to geographic coordinates $(\varphi_{i,t}, \lambda_{i,t})$ using MaxMind GeoLite2. IP geolocation is imperfect because provider points of presence, carrier-grade NAT, and VPNs all introduce noise. However, this practice is accepted as a standard proxy for physical location (Wang and Kangasharju 2013; Niaki et al. 2020). The analysis emphasizes trends over time rather than absolute point estimates. The analysis uses two complementary metrics for route concentration. Both are computed per vantage point per day and weighted by traffic volume so that they reflect actual connectivity rather than simple peer counts.

Metric 1: Traffic-weighted average distance. Let (φ_v, λ_v) denote the vantage point's coordinates. Let $d((\varphi_v, \lambda_v), (\varphi_{i,t}, \lambda_{i,t}))$ denote the great-circle distance between vantage and peer i on day t. The traffic-weighted average distance is:

$$\overline{D}_t = \sum_{i \in \mathcal{P}_t} w_{i,t} \cdot d\Big((\varphi_v, \lambda_v), (\varphi_{i,t}, \lambda_{i,t})\Big) . \tag{6.2}$$

A decreasing $\overline{D}_t$ over time signals contraction where the traffic-weighted peer set is becoming geographically closer to the vantage point, consistent with peer selection that prefers low-latency neighbors.

Metric 2: Traffic-weighted geographic center. Distance contraction alone does not reveal *where* connectivity accumulates. A daily byte-weighted "center of mass" shows this. For each peer i, convert latitude and longitude to a unit-sphere vector:

$$\begin{aligned} x_{i,t} &= \cos(\varphi_{i,t}) \cos(\lambda_{i,t}), \\ y_{i,t} &= \cos(\varphi_{i,t}) \sin(\lambda_{i,t}), \\ z_{i,t} &= \sin(\varphi_{i,t}). \end{aligned} \tag{6.3}$$

Compute the byte-weighted mean vector:

$$\overline{x}_t = \sum_{i \in \mathcal{P}_t} w_{i,t} x_{i,t}, \quad \overline{y}_t = \sum_{i \in \mathcal{P}_t} w_{i,t} y_{i,t}, \quad \overline{z}_t = \sum_{i \in \mathcal{P}_t} w_{i,t} z_{i,t}. \tag{6.4}$$

Convert back to latitude and longitude:

$$\begin{aligned} \overline{\varphi}_t &= \operatorname{atan2}\left(\overline{z}_t, \sqrt{\overline{x}_t^2 + \overline{y}_t^2}\right), \\ \overline{\lambda}_t &= \operatorname{atan2}(\overline{y}_t, \overline{x}_t) . \end{aligned} \tag{6.5}$$

The point $(\overline{\varphi}_t, \overline{\lambda}_t)$ answers: where on Earth does this vantage point's traffic center itself today? Drift and clustering in the focal point reveal where effective connectivity accumulates over time.

Why these metrics matter. The measurement design reflects a core claim of this dissertation that where nodes physically exist on a map is not the same as the effective topology

of the network. Protocol networks often look geographically diverse in node lists and maps, but a core component of their resilience and security depends on the distribution of their actual traffic across routes. Traffic-weighted metrics target the effective paths nodes take to talk to each other. This distinction also explains why chokepoints can remain hidden, as a protocol may appear decentralized by participant counts and unique IP addresses, while actual data exchange depends on a much smaller set of physical routes.

6.1.3 Contraction, Concentration, and Drift

This section examines evidence of (i) a decrease over time in the traffic-weighted average distance from each vantage point to its peers, (ii) focal-point convergence along inter-regional routes, and (iii) marginalization of peripheral regions that is invisible without traffic weighting.

Distance contraction in core regions. Over a 107-day observation window, four of five vantage points show statistically significant decreases in traffic-weighted average distance to peers: North America, Europe, Asia-Pacific, and the Middle East. This pattern is consistent with geographic clustering around low-latency regions and a shrinking effective peer set. The approximate linear trends over the 107-day window are:

Region	$\approx$ Slope (km/day)	p-value
North America	-6.17	0.002
Europe	-4.46	< 0.0001
Asia-Pacific	-12.55	< 0.0001
Middle East	-4.05	< 0.0001
South America	—	n.s. 0.797

South America is a notable outlier that underscores the role of physical infrastructure in shaping effective topology. From the South American vantage point's own coordinates, the distance trend is not statistically significant ($p = 0.797$), suggesting no local geographic

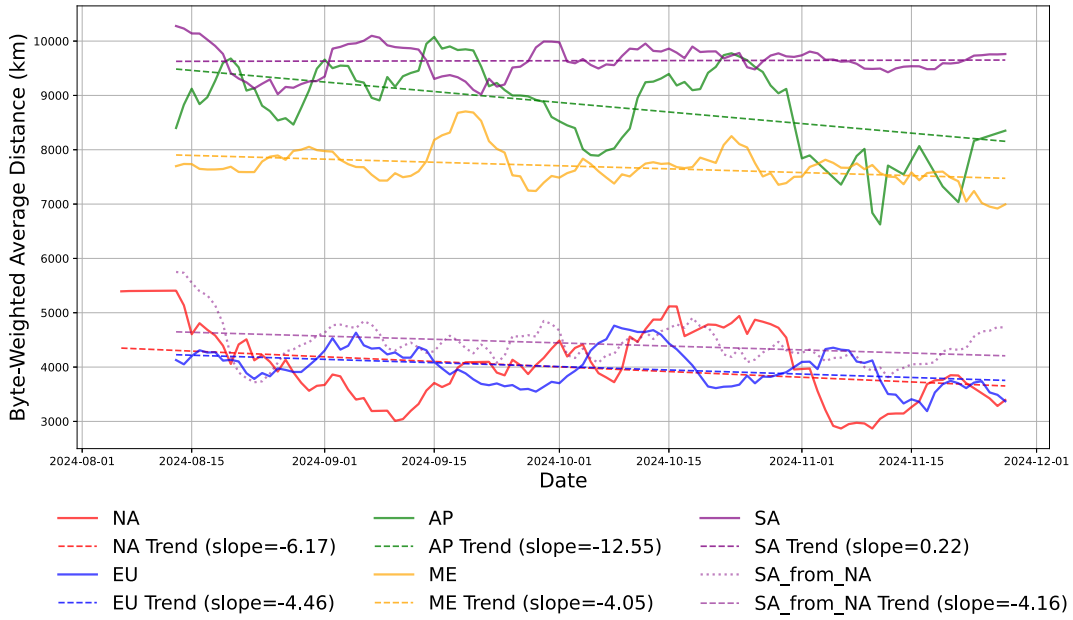

Figure 6.1: Daily traffic-weighted average distance between each vantage point and its effective peer set. Four of five regions exhibit significant distance contraction over time, consistent with latency-driven geographic clustering.

clustering. However, if we run a counterfactual, where the node's location is in North America, a significant contraction emerges: ≈ -4.16 km/day ($p < 0.001$). Rather than attracting geographically proximate peers, the South American node's effective peer base concentrated into North America. Pairing this finding with a qualitative assessment of undersea cable geography shows the primary routes out of São Paulo run northward to the U.S. East Coast (Pérez 2023). Since South America has a sparse local node population, latency-based peer scoring steers the node toward peering with nodes in North America rather than nodes in South America. The result is that South America operates as a peripheral appendage to the NA–EU core, connected through a narrow set of submarine links rather than through regionally diverse paths. This finding illustrates that overlay network peering inherits and amplifies the constraints of the physical layer beneath it.

Two points matter for understanding this pattern. First, it arises from ordinary client behavior (keep responsive peers, drop unresponsive ones) without any coordinated or centralized control. Network clustering of this kind is antithetical to Ethereum's protocol design, yet it emerges as a side effect of each node independently optimizing its own connections. Second, the effect bridges two layers. Peer selection is a software decision, but the distance it acts on is a physical property of the underlying network. Software choices are quietly reshaping which physical routes the network depends on.

Focal points converge along the North America–Europe corridor. The focal-point trajectories provide the most direct evidence of physical chokepoints. When daily traffic-weighted centers of mass are plotted on a world map, they drift and cluster along great-circle routes between North America and Europe, aligning with known transatlantic submarine cable paths (Seidenberger et al. 2026c). A substantial share of peer-to-peer data exchange concentrates into a narrow set of inter-regional links rather than spreading evenly across many independent routes.

This is the critical shift from "node geography" to "route dependence." A map of node locations does not show this effect. The focal-point view does, because it is traffic-weighted and therefore sensitive to where effective connectivity accumulates.

6.1.4 Route Concentration Creates Systemic Risk

Route concentration matters because it changes which disruptions are systemically significant and how adversarial pressure can amplify. It is self-evident that a cable cut breaks things. The larger danger is that when specific cables concentrate connectivity, the resulting chokepoints can amplify attacks, especially when combined with targeting, censorship, or correlated faults.

Chapter 4 showed that visible nodes receive measurable additional pressure and that this can translate into peer churn and performance degradation even without protocol exploits.

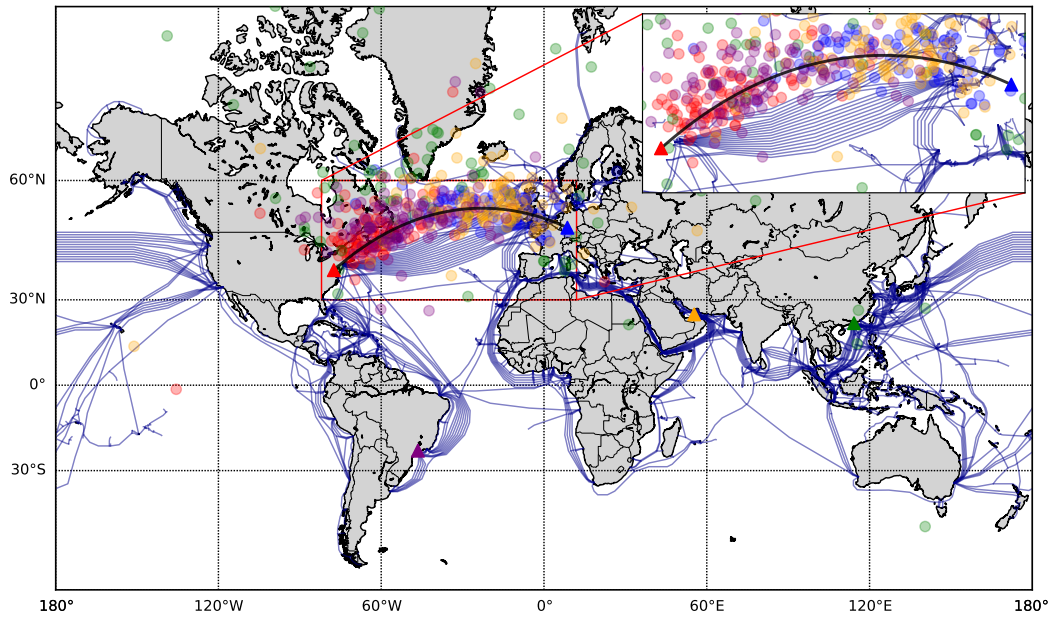

Figure 6.2: Daily traffic-weighted geographic focal points for Ethereum peer-to-peer data exchange, by region. Focal points converge toward the North America–Europe corridor, suggesting dependence on a narrow set of inter-regional physical routes, notably transatlantic submarine cables.

Route concentration adds a structural multiplier: if the overlay already concentrates effective connectivity into a small set of regions and routes, then adversarial pressure (or disruption) in those regions can have outsized impact.

Two pathways are especially relevant:

- **Propagation degrades without mass validator loss.** Disruption along a route can increase latency and packet loss, causing peers to score each other poorly and drop connections. The overlay may remain "up" in the sense that many nodes are online, but propagation becomes slower and less reliable. In proof-of-stake systems, this can degrade attestation timeliness and consensus effectiveness without large changes in the number of validators nominally online.

- **Correlated failures amplify.** A route is a shared dependency in the sense of Chapter 3. When many high-volume connections rely on the same physical path, the probability of correlated degradation increases. Under stress, recovery traffic and reconnection attempts can further load the affected route, creating a feedback loop that resembles a cascade rather than independent failures.

The key point is that chokepoints are structural vulnerabilities that can intersect with adversarial targeting. If the same routes and regions are both structurally central for propagation and heavily targeted for denial, enumeration, or censorship, then the system becomes vulnerable to disproportionate disruption from selective pressure on a small part of the underlying network.

6.2 Cross-protocol parallels

Corridor dependence is one form of chokepoint, but it is not the only one. Across protocol families, chokepoints arise as (i) corridors and routes, (ii) intermediaries and custodians, and (iii) micro-network hubs that dominate short windows. This section connects Ethereum's corridor dependence to three structurally parallel cases from earlier chapters and papers underpinning the dissertation.

6.2.1 BitTorrent swarms as federated micro-networks organized around seed capacity

BitTorrent is often described as a large peer-to-peer system, but the more decision-relevant unit of analysis for chokepoints is not "BitTorrent as a whole." It is the **title-specific swarm**. Swarms spin up, peak, and collapse, and they are linked only loosely through discovery layers (Chapter 2) and shared client software. This means BitTorrent behaves less like a single monolith and more like a **federation of ad hoc, title-centric distribution micro-networks** (Seidenberger et al. 2026b).

No stable global chokepoints, but strong local hubs. Because the swarm is the unit of coordination, BitTorrent has no single, durable global chokepoint analogous to a stable domain, CDN, or control plane. Enforcement approaches that target stable global points (a domain, a resolver, a platform operator) do not transfer cleanly. However, micro-network chokepoints emerge as well-provisioned early seeders and early best-seeded releases dominate the high-leverage window when diffusion is most sensitive to supply conditions.

Diffusion is supply-driven in the early window. The empirical modeling in Seidenberger et al. (2026b) shows that swarm diffusion is explained largely by supply-side shocks in the early phase. Logistic diffusion fits most hashes, while Bass imitation terms are near zero for almost all titles, suggesting limited evidence for social imitation in early adoption dynamics. Instead, early availability and seeding bandwidth shape diffusion trajectories. This aligns with the interpretation that the initial "leader" in a swarm is often the best-seeded release, and that seed capacity is a high-leverage structural input.

First-mover advantage is sharp and short-lived. Chokepoints can be temporal as well as structural. The same study finds that the leader's active-peer share remains meaningfully positive for roughly one week and largely fades within about two weeks (Seidenberger et al. 2026b). This is a chokepoint window where early seed capacity governs a short period when small differences in provisioned bandwidth can determine downstream diffusion outcomes.

Swarm participation is heavy-tailed across jurisdictions and ISPs. Even though BitTorrent lacks stable global chokepoints, participation concentrates strongly in the infrastructure and jurisdictional layers. The study reports heavy concentration by country and ISP, with the top ten countries accounting for roughly two-thirds of unique peers and the top ten ISPs accounting for roughly one-quarter. Inequality is high (country-level and ISP-level Gini coefficients both near 0.9) (Seidenberger et al. 2026b). Interpreted through this chapter's lens, BitTorrent's chokepoints are not stable global control points, but rather (i)

micro-network hubs in early windows and (ii) high-participation jurisdiction and network clusters that bound where risk and enforcement pressure concentrate.

6.2.2 Email: many brands, few pipes

Email provides a structurally analogous chokepoint example even though its topology is not an overlay in the same sense as P2P gossip. The user-facing surface is diverse because many brands send messages to the same user email. Yet the infrastructure layer is sharply concentrated, creating chokepoints for availability, surveillance, and metadata custody.

The inbox audit results from Chapter 5 establish two concentration facts (Seidenberger et al. 2025a):

- **Domain-layer concentration.** The top ten root domains account for 63.23% of observed email volume.
- **ASN-layer concentration.** The top eight ASNs account for 89.35% of observed volume.

This is "many brands, few pipes," where an open protocol surface masks a concentrated set of infrastructural chokepoints. In Chapter 5, this was framed as delegated custody and de facto centralization. In Chapter 6, it also functions as a chokepoint example because those ASNs are physical and organizational bottlenecks where outages, policy changes, or surveillance capabilities can have ecosystem-wide effects even though SMTP is nominally an open and permissionless protocol.

6.2.3 Censorship-resistant DoH: shield and chokepoint

NinjaDoH provides an explicit case where infrastructure concentration is not only inherited but deliberately exploited (Seidenberger et al. 2026a). Its feasibility premise is that blocking outbound HTTPS to major hyperscalers such as AWS, GCP, or Azure is impractical for many

censors because of collateral damage against essential hosted services. NinjaDoH's design uses a moving-target strategy—short-interval IP rotation and private update distribution—to stay ahead of blocklists while remaining performant.

Three empirical anchors are salient for this chapter:

- **Dependence as design choice.** NinjaDoH requires hyperscaler infrastructure with a large IP allocation pool and supports short-interval IP rotation as an explicit moving-target mechanism.
- **Practicality.** The prototype deployment cost is on the order of $23.55/month, illustrating why outsourcing wins under deployability constraints.
- **Chokepoint duality.** Hyperscalers function simultaneously as *shield* (hard to block without collateral damage) and *chokepoint class* (policy actions, account enforcement, and regional outages create correlated constraints).

Interpreted through the mechanism lens from Chapter 5, local optimization under constraint (low latency and low cost) rationally induces dependence on hyperscalers. Interpreted through this chapter's lens, the hyperscaler becomes a physical chokepoint class that concentrates both protective cover and correlated failure risk.

6.3 Chapter Synthesis and Bridge

This chapter established that protocol overlays can be logically decentralized while becoming physically concentrated in operation. In Ethereum, traffic-weighted measurements showed distance contraction, focal-point convergence on the NA–EU corridor, and peripheral-region tethering. The cross-protocol parallels showed the same structural principle in different forms: micro-network hubs in BitTorrent, infrastructure concentration in email, and deliberate hyperscaler dependence in censorship-resistant DNS.

The unresolved question is temporal severity. This chapter identified where correlated dependencies accumulate, but it did not quantify how quickly disruption propagates, how long degradation persists, or how large the upper-tail outcomes become under bursty stress. Chapter 7 addresses this gap by shifting from spatial concentration to temporal dynamics, linking bursty stressors and stable peering substrates to cascade formation and tail-risk outcomes, then uses simulation to characterize disruption distributions beyond what measurement windows alone can observe.

Chapter 7

Temporal Dynamics, Cascades, and Tail Risk

This dissertation's earlier chapters establish that protocol networks exhibit strong concentration and shared dependencies across infrastructure layers. This chapter shifts the lens from looking at *where* dependencies accumulate to *when* risk materializes. The core claim is that **time is a first-class security variable**. For planet-scale decentralized systems, security and resilience evaluation should prioritize bursts and churn that lead to correlated failures and tail risk. Average-case snapshots do not capture the time-correlated structure of failure modes.

Time matters here in two ways. First, many protocol-network phenomena are most naturally expressed as time series because the systems themselves evolve and because adversarial actions tend to be bursty rather than stationary. Second, the most consequential harms often occur in short windows whose probability mass is small but whose impact dominates expected loss to whatever performance metric is being measured. To study these short windows, the measurement strategy must treat time itself as a primary design constraint.

This chapter develops the argument in four stages. Section 7.1 grounds the discussion in BitTorrent, where the appearance of illicit media content is a temporal problem at global scale. That section uses the Dataset Value Taxonomy (DVT) to formalize why long-running telescopes are needed to make credible claims about temporal structure and tail risk in protocol networks. Section 7.2 shows that marketing email exhibits weak global seasonality but strong timing regularities and traffic bursts, with the main lesson that temporal structure

becomes visible only after long collection windows. Section 7.3 then turns to Ethereum, where large-scale measurement reveals both attack waves and temporal structure in the peer topology itself. Those findings matter for understanding how cascades form. Finally, Section 7.4 uses simulation results to measure baseline tail risk in Ethereum, which then serves as the benchmark for evaluating the interventions in Chapter 8.

7.1 Temporal Investment in Datasets

A decentralized system's security model that includes time must distinguish among three related temporal patterns: **bursts**, **correlated failures**, and **cascades**. Bursts are short-lived deviations from baseline behavior. Correlated failures and cascades, by contrast, describe failure structure. Correlated failures can occur simultaneously, while cascades are sequential. Moreover, correlated shocks can trigger cascades by changing connectivity or load in ways that propagate onward through a protocol's application layer and down into its infrastructure. This distinction matters because "resilience" depends not only on whether a system degrades, but also on how quickly it recovers and whether it crosses qualitative thresholds during recovery (e.g., a short propagation impairment versus prolonged loss of effectiveness).

Illicit media distribution via BitTorrent provides an empirical example of why time matters. A piracy ecosystem distributes media content such as TV and movies as soon as possible after their release (Kryczka et al. 2011). The interesting question is *how quickly* a pirated title appears after its legitimate release, and whether the content and security of those early torrent releases differ from what appears on the network later. The Ethereum studies show the dangers of cascades, where bursts of adversarial traffic targeting Ethereum nodes can induce resource stress that then propagates into application-layer performance degradation.

Time is also a constraint on the *epistemic value* of a dataset dealing with protocol networks. In adversarial ecosystems, what can be measured, and therefore what can be claimed, depends on temporal investment. As explained in the DVT, the marginal cost of downstream

analysis is falling as AI-enabled scientific workflows become more common, so the bottleneck shifts upstream to sustained data collection with telescopes that have the right vantages and instrumentation. Across the studies in this dissertation, *temporal span* determines whether drift, seasonality, bursts, and time-to-event claims can be identified and tested.

The DVT operationalizes **Temporal Investment** as a measurable construct composed of four dimensions: elapsed duration, sampling frequency, external cycle dependence, and latency to insight. Elapsed duration captures how long the data collection must run; sampling frequency captures the granularity at which events are recorded; external cycle dependence captures whether valid inference requires spanning exogenous cycles (weekly, seasonal, release cycles, etc.); and latency to insight captures the lag between collection and usable analysis. Tables 7.1 and 7.2 map each major dataset used in this dissertation to these four components. The goal is not to claim that one dataset is always "better," but to make explicit which datasets can support minute-scale inferences versus which require *years* to reliably expose early windows such as the first hours after a media release (Section 7.1), burn-in transients that fade only after weeks of collection (Section 7.2), or tail-risk episodes like the attack waves measured on Ethereum (Section 7.3).

Table 7.1: Temporal investment dimensions (part 1 of 2): short- and medium-horizon datasets. Combined with Table 7.2, this covers all major datasets used in the dissertation.

Dataset (chapter role)	**Elapsed duration**	**Sampling frequency**	**External cycle dependence**	**Latency to insight**
NinjaDoH (short-horizon illustration)	15-minute adaptive-adversary run; 10 IP rotations; >1 GB PCAP. Separate 3-minute ML baseline capture.	Packet-level capture; IPNS poll every 5 s; pseudo-random IP rotation every 1–3 min.	None; dynamics are internal to the rotation and flow timescale.	Near-real-time: one-window evaluation. Mean rotated-DoH resolution 12.68 ms vs. 7.85 ms direct; zero dropped queries across three rotations.
Email inbox audit (long-horizon timing regularities)	361 days (Mar 2023–Feb 2024); 4,847 emails from 109 services.	Daily series with time-of-week and time-of-day features; accounts normalized by days since first email.	Weekly cycle (6.96 days) with annual, semi-annual, and quarterly harmonics.	Full-year horizon required: seasonality explains only 1.17% of variance.
Ethereum / EtherBee (medium horizon, high volume)	Three-month collection (Aug–Nov 2024); 133.8M attack events from 12.5M source IPs.	Continuous high-rate streams across five beacon nodes; 45.4M P2P TCP sessions, 20.97 TB indexed metadata, 94,659 P2P IPs.	Burst-driven attack waves tied to adversary campaigns and infrastructure events, not calendar-time seasonality.	Operational bottleneck dominates: 6-node Elasticsearch cluster (227 TB), 30.1 TB primary storage, and >33.7B documents after 3 months.

Table 7.2: Temporal investment dimensions (part 2 of 2): long-horizon BitTorrent datasets used for burn-in and event-time inference.

Dataset (chapter role)	**Elapsed duration**	**Sampling frequency**	**External cycle dependence**	**Latency to insight**
BitTorrent / MagnetDB (long-horizon telescope)	300-week continuous collection, 2018-12-30 to 2024-09-29; 93.7% uptime.	Continuous DHT crawling tracking up to 10,000 peers simultaneously; 28.6M torrents and 950.6M files indexed.	Strong event-time anchoring: new torrents appear on the day of release. Burn-in indexes pre-existing content before steady-state discovery.	Burn-in stabilizes after ∼2 years; only after stabilization can the telescope reliably timestamp new torrent appearances.
BitTorrent / MagnetDB (time-to-event inference)	1-month micro panel, Jul–Aug 2025.	Day-scale survival outcomes via Cox model (n = 1,469 titles). Micro panel: 28 titles, 244 unique torrents, 167,607 distinct IPs across 220 countries.	Event-time aligned to release date: t=0 at earliest public release. Never-leaked titles are right-censored at Dec 31, 2024.	Median time-to-leak ∼120 days for exclusive titles and ∼275 days for multi-homed titles; each additional provider reduces instantaneous leak hazard by ∼4.2%.

Two immediate conclusions follow from Tables 7.1 and 7.2. First, short-horizon studies can support credible claims about mechanisms that operate at seconds-to-minutes timescales, but they cannot answer questions about long-run phenomena. Second, the BitTorrent and Ethereum telescopes illustrate that long horizons are expensive—not only in elapsed duration but also in operational infrastructure—yet without those long collection windows, certain research questions remain out of reach.

7.1.1 What "Enough Time" Looks Like

MagnetDB offers direct empirical evidence that dataset viability is a function of measurement duration. The telescope runs a continuous DHT crawler and, over a 300-week span (December 2018 to September 2024), yields a dataset of 28.6 million torrents and 950.6 million files. Scale alone is not the point; what matters is that coverage does not become "complete enough" immediately, even under continuous measurement.

Two operational realities constrain the kinds of temporal claims one can make early in collection. First, the crawler has a burn-in phase during which it must discover all previously released content on the network before it can reliably timestamp the appearance of *new* releases. This discovery process stabilizes only after roughly two years, at which point backfilling ends and the dataset enters a steady-state regime. Before that point, discovery timestamps cannot be trusted because the crawler is still indexing pre-existing content. Second, over the full collection window, expected uptime is 93.7%, with the remaining downtime introducing sporadic gaps. Together, these facts mean that a short-run snapshot of the DHT will systematically undercount or mischaracterize long-tail content and will conflate true absence with backfill incompleteness.

A deeper methodological lesson is that dataset viability is a question about whether the dataset has matured enough temporally to support stable inference. In BitTorrent, the measurement target is large and adversarially shaped—publishers, trackers, and release

groups all influence what content appears, when, and how—so the dataset must persist long enough to separate persistent structure from transient artifacts.

Streaming fragmentation as a natural experiment. The supporting study *Front Running the Red Carpet* (Seidenberger et al. 2026b) adds to the power of MagnetDB via additional longitudinal measurement to make testable claims about when piracy begins. The study is motivated by a structural feature of the modern entertainment market. Over the past decade, media rights have splintered across a growing number of subscription platforms—Netflix, Disney+, Hulu, Apple TV, and many others—each holding exclusive titles as a competitive strategy. This streaming service fragmentation creates what the study terms *media access friction*: the cumulative cost to households of searching for, subscribing to, and coordinating across platforms to watch what they want. When that friction is high, as when a viewer cannot easily or affordably watch a title through legitimate channels, substitution toward lower-friction alternatives, including illicit peer-to-peer distribution, becomes more likely (Danaher et al. 2020). The study tests whether this substitution is empirically observable by measuring whether the number of legitimate providers carrying a title affects *when* a pirated copy first appears on the BitTorrent network.

Exclusivity accelerates piracy onset. Using MagnetDB as the discovery telescope and a macro panel of n=1,469 film and TV titles tracked from January 2018 through December 2024, the study fits a Cox proportional hazards model in which the event is a title's first torrent appearance on the DHT. The number of legitimate streaming providers is the primary covariate, with controls for audience engagement (log vote count) and production budget. The results show that each additional provider reduces the instantaneous hazard of a first pirated copy by approximately 4.2% (hazard ratio 0.959, $p < 0.0001$). In practical terms, a title exclusive to a single service reaches a 50% probability of having leaked within about 120 days of release, whereas a title carried by eight services does not reach that threshold until roughly 275 days (Figure 7.1). The takeaway is that **exclusivity reshapes the early**

window where exclusive titles leak sooner, and the difference is measured in months. In a security model where the most consequential harms concentrate near release, shifting onset by months materially changes the threat surface.

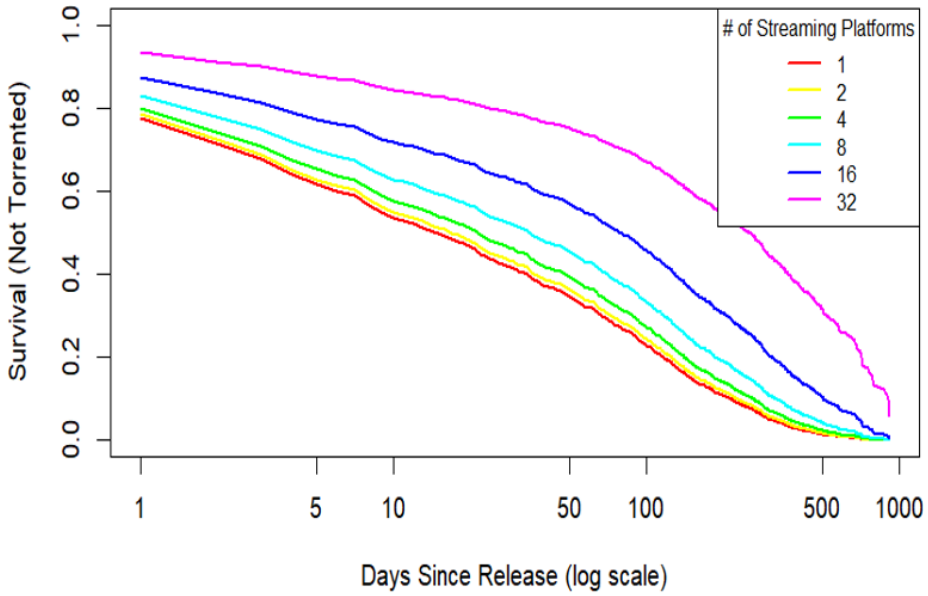

Figure 7.1: Predicted survival curves for time to first torrent as a function of the number of streaming-service providers. Covariates are fixed at sample means. Titles exclusive to fewer platforms leak substantially earlier.

Popularity accelerates onset independently. Audience engagement, measured by log-transformed vote count, is also a significant predictor of leak timing. Each unit increase in log votes raises the instantaneous hazard by about 19% (hazard ratio 1.192, $p < 0.0001$), meaning that more popular titles leak sooner even after controlling for how many platforms carry them. Both market structure and audience demand independently shape the timing window. High-profile exclusive titles therefore face a compounded risk where fewer legitimate outlets *and* higher demand each pull the leak forward.

Timing and volume have different drivers. A negative-binomial model of total torrent volume per title reveals that provider count has no predictive power over how many pirated copies ultimately circulate (coefficient 0.00012, $p = 0.980$), while engagement metrics remain strong predictors of volume (log vote count: incidence rate ratio 1.292, $p < 0.0001$). Exclusivity determines *when* piracy begins; demand determines *how much* eventually circulates. A model that evaluates only average volume would entirely miss the early-window effect, because exclusivity shifts onset without changing total volume. This distinction is precisely why time must be treated as a first-class variable and the security-relevant question is often not "how much" but "how soon."

Early windows carry elevated security risk. Early release windows for these media torrents have qualitatively different security properties with real risks to end users. In a malware analysis of torrents appearing before a title's official release, approximately one-third of pre-release torrents contain suspicious executable files, compared to only about 2% of post-release torrents—a fourteen-fold increase (Seidenberger et al. 2026b). Sandbox analysis of representative samples reveals behavior consistent with staged trojans: environment reconnaissance via registry queries, zero-padded file inflation to mimic legitimate media file sizes, and dormant follow-on processes awaiting further triggers. The pre-release window is short relative to the full lifecycle of piracy for any given title, but the per-torrent severity is far higher. In a time-aware threat model, "average malware risk across all torrents" understates the danger faced by users who download during the narrow period when risk peaks.

Short observation windows complement long ones. A 28-title micro panel, tracked in July–August 2025, complements the multi-year macro analysis by illustrating what shorter collection windows reveal and miss. The micro study captured 244 unique torrents involving 167,607 distinct peer IPs across 220 countries and 7,277 ISPs. Two findings are relevant. First, early swarms are highly global with participants from many countries and networks, consistent with fast diffusion through a decentralized system. Second, there is a measurable

first-mover advantage—the earliest well-seeded torrent for a given title captures a sharp lead in active peer share that persists for roughly one week before fading (Figure 7.2). Infrastructure also matters at the geographic level because a 50 Mbps increase in country-median download bandwidth corresponds to about a 14% shorter time-to-first-appearance, indicating that distribution capacity shapes when a given jurisdiction first encounters a pirated copy.

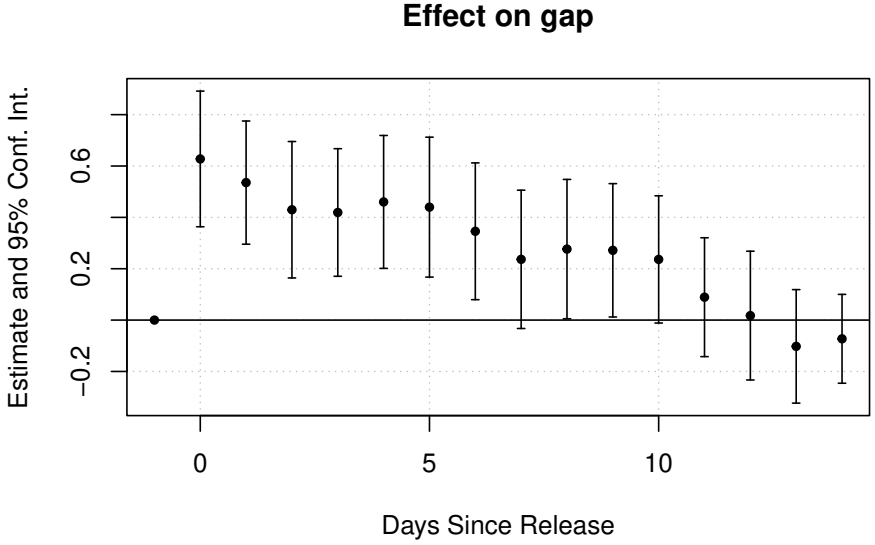

Figure 7.2: First-mover advantage in competing torrents. Points plot the estimated difference in daily active-peer share between the earliest well-seeded torrent and later entrants for the same title, aligned at the leader's release (day 0). The leader's share is significantly higher for roughly the first week before attenuating toward zero.

At the same time, the micro panel underscores the lesson from Tables 7.1 and 7.2: short windows can characterize burst dynamics and diffusion mechanics, but they cannot substitute for multi-year investment when the question is whether observed behavior is typical, whether the measurement has stabilized, or how onset shifts across hundreds of titles and providers.

Synthesis. BitTorrent provides this section's canonical example because it combines a long-horizon measurement, MagnetDB, with results that isolate early-window timing effects (Seidenberger et al. 2026b). Three observations are tightly aligned with the broader argument that time is a primary security variable. First, long horizons are required to build a measurement instrument whose discovery stabilizes—MagnetDB requires roughly two years of continuous collection before steady-state discovery is reliable. Second, early-window onset is empirically measurable and has distinct drivers where market structure (provider count) shifts time-to-first-appearance by months, and audience demand accelerates onset, even though neither predicts total volume. Third, tail-window harms are concentrated, as suspicious executables are more prevalent before release than after, and the elevation is temporally localized rather than uniformly distributed.

These observations set up the remainder of the chapter. The email study shows that even where global seasonality is weak, timing regularities and bursts are still present and strategically exploited by senders. The Ethereum study reveals bursty attack waves and persistent peers—empirical anchors that motivate the study of cascades. Finally, simulation is needed because the most consequential correlated disruptions are rare enough that even multi-month measurement may not observe them directly.

7.2 Timing Regularities in Email

BitTorrent's early-window results are event-driven because piracy onset is anchored to a title's release date, with the security-relevant question being how quickly that onset occurs. With email there is no single triggering event; instead, the timing structure of email emerges from the strategies of many independent senders operating on weekly, daily, and campaign-level rhythms. The inbox audit described in the *Why You've Got Mail* study provides a clear case where sustained temporal investment reveals structure that would be invisible in

a short snapshot—and where the most important finding is that the obvious signal (weekly seasonality) explains almost none of the variance.

Chapter 5 examined this dataset through the lens of infrastructure concentration and delegated trust: who actually sends email on behalf of brands, and how concentrated are those intermediaries? This section revisits the same dataset through a temporal lens: *when* do emails arrive, what rhythms and bursts characterize the ecosystem, and how do those timing patterns differentiate sender strategies?

Dataset and temporal normalization. The email dataset spans 361 days (March 2023 through February 2024) and contains 4,847 emails from 109 of the 150 registered services. Because account creation dates vary across services, the study normalizes each account's time series by days since first email rather than by calendar date. This avoids confounding temporal patterns with account-age artifacts and ensures that comparisons across senders reflect genuine timing strategies rather than registration order. In DVT terms (Table 7.1), this dataset represents a mid-horizon temporal investment that was long enough to observe weekly cycles and long-run trends, but still at a human-scale span of one year rather than multiple years.

Weak global seasonality. A Fourier transform of the aggregate email time series identifies a weekly component with frequency 0.1436 and period 6.96 days—the expected result, since many marketing and operational processes run on weekly schedules. However, the more important finding is how little this periodicity explains. An additive time-series decomposition with 7-day seasonality (Figure 7.3) shows that the seasonal component accounts for only 1.17% of total variance. The dominant feature is a decreasing linear trend over the year, not a rhythmic cycle. This result is a caution against equating "weak seasonality" with "no timing structure." Global periodicity is not the only—and often not the dominant—source of temporal risk.

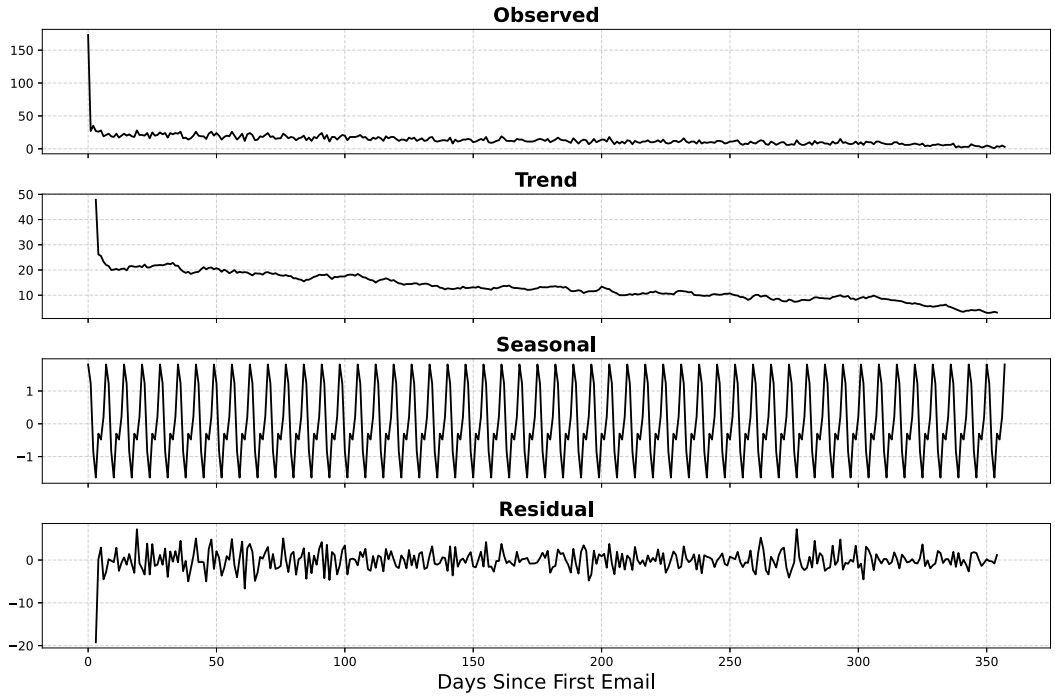

Figure 7.3: Additive time-series decomposition of aggregate emails received over 361 days. The seasonal component (7-day period) explains only 1.17% of variance; the dominant signal is a decreasing trend.

Burst windows persist beneath weak seasonality. Even though aggregate seasonality is weak, time-of-week and time-of-day structure is pronounced. Polar heat maps of email arrival times (Figure 7.4) show that most emails arrive mid-week (Tuesday through Thursday), with send-time peaks concentrated around midnight and during the early afternoon (13:00–15:00). These patterns are consistent with operational rhythms—batch sends at midnight, campaign timing during business hours—but they represent *burst windows*: temporally localized periods of elevated load and user exposure that recur consistently. In the vocabulary introduced at the start of this chapter, these are bursts rather than seasonality. They produce consistent, predictable concentrations of email traffic that can be strategically exploited by senders and that matter for filtering, abuse detection, and infrastructure provisioning.

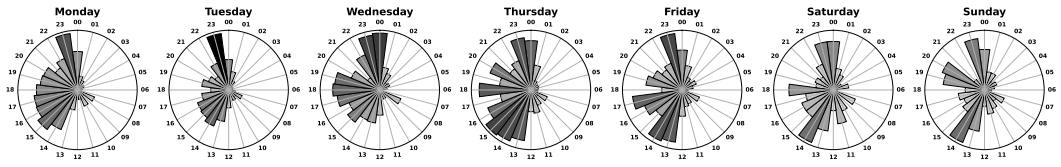

Figure 7.4: Email frequency by hour of day for each day of the week. Activity concentrates mid-week (Tuesday–Thursday) with peaks near midnight and during the early afternoon.

Temporal patterns differentiate sender strategies. Timing patterns are not only global properties of the ecosystem; they also discriminate between types of senders. The study constructs features from each account's time series—trend, seasonal, and residual components alongside structural counts such as email type distributions—and applies PCA followed by K-Means clustering. The best configuration separates senders into two clusters with a silhouette score of 0.831 (Figure 7.5). One cluster corresponds to low-activity generalist accounts that send infrequently and with little temporal regularity. The other corresponds to high-volume promotional senders that concentrate their output on weekdays and at specific hours. In other words, *when* a sender sends is enough to reveal *what kind* of sender it is—timing alone separates the ecosystem into distinct behavioral strata.

Infrastructure churn as a temporal signal. Temporal dynamics also appear at the infrastructure level. Chapter 5 documented that a small set of autonomous systems handles the vast majority of marketing email. The temporal complement is that the *rate of IP address rotation* co-varies with sender reputation. The study finds a strong positive correlation between the number of unique sending IPs a company uses and the total community-reported spam associated with those IPs (Pearson's $r = 0.71$, $p < .0001$). While correlation does not establish causation, the pattern is consistent with a strategy in which senders cycle through IP addresses to dilute the reputational impact of abuse reports on any single address. This "IP hopping" is a temporal infrastructure strategy where the churn rate of sending addresses is itself a signal of how aggressively a sender operates. The parallel to BitTorrent is direct—in both systems, temporal strategy is an attack surface. In BitTorrent, it manifests as

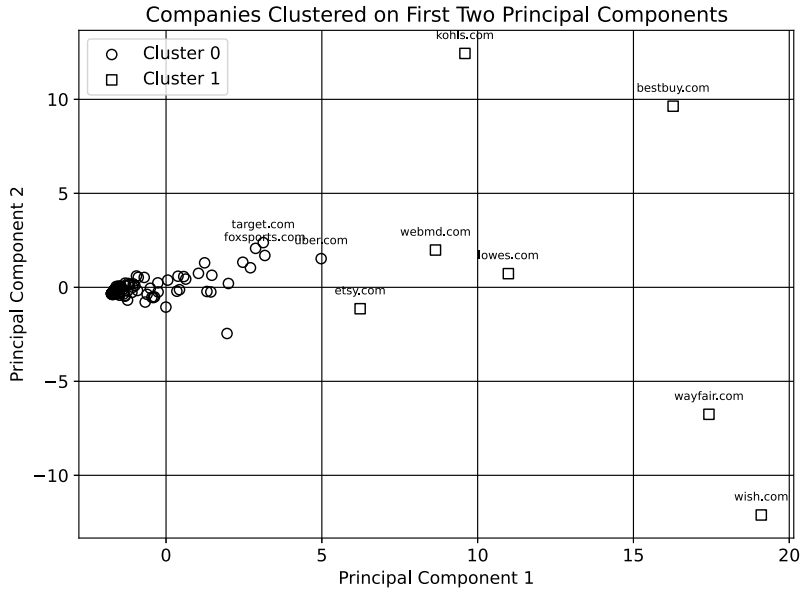

Figure 7.5: Sender clustering in the first two principal components. Two clusters emerge with a silhouette score of 0.831: low-activity generalists (left) and high-volume promotional senders (right).

pre-release publishing and early leak timing. In email, it manifests as burst scheduling and infrastructure rotation.

Why a year of collection matters. A one-year audit was designed as a sustained collection window that would make it possible to (i) detect the weekly component robustly via frequency methods, (ii) quantify how little variance that component actually explains, and (iii) separate persistent burst windows. A few weeks of data might reveal that emails cluster mid-week, but could not determine whether that pattern is stable across seasons, whether the decreasing trend is real or a transient artifact, or whether sender clusters are persistent groupings or ephemeral fluctuations. The dataset's temporal investment is therefore directly tied to the conclusion that global seasonality is weak while timing regularities are strong, concentrated at specific hours and days, and stable enough to differentiate sender strategies over a full year.

These results reinforce and extend the argument from the preceding section. Where BitTorrent demonstrates that early-window onset is measurable and consequential, email shows that even in a system without a single triggering event, temporal structure is rich, persistent, and strategically exploited. Both cases share a common lesson that long collection windows are prerequisites for claims about timing, and the most interesting temporal features—bursts, sender-specific rhythms, infrastructure churn—are often invisible in short snapshots and poorly captured by global averages.

7.3 Temporal Stressors in Ethereum

BitTorrent and email show that timing determines when harm concentrates. Ethereum adds a further dimension because it is a live protocol network with economic stakes and consensus constraints. Therefore, temporal stressors can trigger *cascades*, where an initiating shock propagates through the overlay into application-layer degradation. Chapter 6 examined Ethereum's overlay through the lens of physical chokepoints and route concentration. This section complements that analysis by focusing on the temporal dimension: how attack traffic arrives in waves, how the peer network provides a stable substrate for stress propagation, and why these empirical observations motivate simulation to study tail risk.

Temporal investment at ecosystem scale. The EtherBee dataset and the supporting node infrastructure study together provide the empirical basis for Ethereum's temporal dynamics. Over a three-month collection window (August through November 2024), honeypots deployed across ten geographically diverse vantage points logged 133.8 million attack events originating from 12.5 million unique source IPs. Simultaneously, five beacon nodes recorded 45.4 million P2P TCP sessions, yielding 20.97 TB of indexed session metadata covering 94,659 unique P2P IP addresses. The operational footprint illustrates the cost of sustained high-throughput measurement, as a six-node Elasticsearch cluster with 227 TB of total disk capacity accumulated 30.1 TB of primary storage and more than 33.7 billion documents over

the three-month window (Table 7.1). For Ethereum, temporal investment is inseparable from ingestion scale—measuring attack waves and session dynamics at ecosystem scope requires both elapsed time and sustained infrastructure capable of keeping pace with the data rate.

Attack waves are nonstationary. Ethereum's most direct temporal stressor is attack traffic itself. Distributed denial-of-service (DDoS) activity constitutes 58% of measured attacks in the EtherBee dataset, and the daily time series of attacks reveals pronounced wave structure: sharp spikes of activity separated by quieter intervals, driven by adversary campaigns and infrastructure events rather than by calendar-time periodicity (Seidenberger and Maiti 2025a). This nonstationarity matters because evaluating Ethereum's exposure via average attack rates would obscure the burst windows that actually stress nodes. During a spike, inbound traffic can saturate a node's available bandwidth, force the operating system to prioritize processing unsolicited packets over protocol messages, and trigger peer scoring penalties that cause the node to shed connections. The relevant security question is not "how many attacks per day on average" but "how intense are the worst bursts, and what happens to the network during those windows."

Peer stability provides a pathway for cascades. Attack waves alone do not produce cascades; cascades require a propagation pathway—a network of connections through which localized stress can spread. The node infrastructure study provides an empirical anchor for that substrate through peer tenure measurements. Among the top 1,000 peers ranked by traffic volume (n = 5,000 peer-day observations), mean tenure is 26.46 days. This implies that the overlay is not purely ephemeral at the scales that matter for traffic flow and, consequently, for stress propagation. If high-traffic peers were uniformly short-lived, bursty attacks might degrade local performance without creating multi-day persistence effects. A mean tenure on the order of weeks makes it plausible that degradation and recovery dynamics can span hours to days—a timescale over which attack waves, peer churn, and recovery load interact.

Chapter 6 showed that this peer substrate is also geographically concentrated as traffic-weighted focal points drift toward the North America–Europe corridor, and peripheral regions such as South America are effectively tethered to core hubs. The temporal complement is that stable peers are precisely the medium through which burst-driven stress can propagate beyond the initially affected nodes.

From stressor to cascade: a three-stage mechanism. Ethereum makes the distinction introduced at the start of this chapter—between bursts, correlated failures, and cascades—empirically traceable. Cascades unfold as a three-stage sequence: (1) an *initiating event* such as an attack spike, a client bug, or a physical corridor disruption; (2) an *overlay response* in which the peer network reorganizes—connections drop, latency increases, peers rescore each other—degrading the topology; and (3) an *application-layer outcome* such as reduced consensus effectiveness, missed attestations, or delayed block propagation.

The supporting measurement provides a direct example of this mechanism in action. During a representative attack burst on one of the experimental beacon nodes, the honeypot recorded a sharp spike in inbound traffic consistent with a reflection-style attack. Over that interval, CPU utilization on the host rose markedly as the kernel processed the additional packet load, the consensus client's peer count dropped by roughly 20% due to increased churn and dropped connections, and the attestation simulator reported that it was unable to perform its validator duties on time. No protocol-level vulnerability was exploited; the attacker simply took advantage of the node's visibility and available bandwidth, forcing the host to prioritize handling unsolicited traffic. The result was a transient, local denial-of-service that directly impaired validator-relevant functionality. This is the cascade mechanism operating at a single node: external pressure (stage 1) degrades peer connectivity (stage 2) and disrupts application-layer duties (stage 3).

Bridging measurement to simulation. The three-month measurement window captures attack waves, peer tenure distributions, and individual cascade episodes, but it cannot

directly observe the tail events that dominate systemic risk. Large correlated failures, sustained region-wide pressure, and physical corridor disruptions are rare enough that they may not occur during any single measurement campaign—yet these are precisely the events whose impact dominates expected loss. Rather than inferring tail behavior solely from observed time series, the next section uses the empirical measurements to calibrate a simulator that explores these tails systematically.

7.4 Simulating Baseline Tail Risk

The preceding sections establish that temporal stressors in protocol networks are bursty, non-stationary, and can trigger cascades through stable peers. The empirical measurements from EtherBee and the node infrastructure study anchor what those stressors look like in practice. This section uses those anchors to answer a question that measurement alone cannot: *what do the tails of the disruption distribution look like under plausible stress scenarios when no mitigation is applied?*

The simulator developed in the node infrastructure study *Node Infrastructure First* models an Ethereum-like P2P and proof-of-stake stack as a discrete-time, stochastic network. It generates a geographically grounded topology, maintains peer connections through latency-aware discovery and pruning, injects failures through composable event primitives, and tracks consensus-relevant outcomes at each step. The design is intentionally effects-based, focusing on capturing the mechanisms through which node-level stress propagates into application-layer degradation—bandwidth saturation, peer churn, propagation delays, and resynchronization load—without re-implementing the full Ethereum protocol. Each simulation step corresponds to approximately 6.4 minutes of real time (Seidenberger et al. 2026c), allowing recovery times reported in steps to be interpreted as hours-scale operational windows.

What the simulator measures. Two time series are the primary outputs. *PoS effectiveness* $E(t)$ captures the fraction of validator duties (block proposals and attestations) that

are performed on time at each step—it can degrade even when most nodes remain nominally online, because propagation impairment can delay messages past their inclusion deadline. *Offline fraction* $O(t)$ captures the share of validators that are completely unavailable. From these trajectories, three scalar metrics summarize each disruption episode: *degradation depth* D (the peak drop below baseline effectiveness), *recovery time* T_r (the number of steps to return to a reference threshold near baseline), and *loss* ℓ (the baseline-normalized area of the effectiveness deficit integrated over the disruption window). The simulator runs large ensembles and reports quantiles ($p90$, $p95$) rather than only averages, because for resilience the question that matters is how bad the worst cases are, not how the typical case performs.

Scenarios grounded in real incidents. The simulator evaluates four scenario channels, each grounded in a real Ethereum incident or a plausible stress test motivated by the measurements in the preceding sections (Table 7.3). S1 models a contagion-like overload inspired by the Fusaka mainnet Prysm bug (Prysm Team 2025), where expensive-to-verify messages exhaust node resources and spread degradation through peering relationships. S2 models a correlated client bug inspired by the Besu incident (Beiko 2019), where a software defect forces a large fraction of nodes to halt and perform full chain resynchronization. S3 models sustained denial-of-service pressure inspired by the 2016 DoS waves that forced emergency hard forks (Ethereum Foundation 2016). S4 models a transatlantic corridor disruption, a forward-looking stress test motivated by the geographic concentration documented in Chapter 6. Each scenario is run as 16,000 paired simulation runs across a systematic parameter sweep, with baseline (no intervention) and intervention variants sharing identical random seeds to isolate treatment effects.

Baseline tail risk. Table 7.4 reports the baseline results—the disruption landscape when no interventions are applied. Two properties stand out. First, tail losses differ materially across scenarios. The overload (S1) and resync (S2) scenarios cluster near a median loss of 0.09–0.10, representing moderate but bounded disruption. The sustained-pressure (S3) and

Table 7.3: Scenario suite for simulator evaluation. Each scenario is grounded in a real Ethereum incident or a measurement-motivated stress test, and exercises a distinct failure channel.

ID	Real-world anchor	Failure channel
S1	Prysm overload bug	Contagion-like overload: expensive messages spread resource exhaustion through peering
S2	Besu client bug	Correlated shutdown followed by mass resynchronization
S3	2016 DoS waves	Sustained external traffic pressure saturates node bandwidth
S4	NA–EU corridor disruption	Physical link impairment partitions core regions, followed by recovery wave

corridor-disruption (S4) scenarios are substantially worse, with median losses of 0.24–0.27, reflecting deeper and more prolonged degradation. Second, recovery-time tails can be long even when medians are moderate. S2 has a median recovery of about 3.4 hours, but its 90th-percentile recovery stretches to nearly 10 hours. S3 is worse still: a median of about 4.8 hours but a 90th-percentile tail approaching 11.6 hours. These upper quantiles are the operationally relevant numbers—they represent the disruption windows that drive worst-case planning, validator penalties, and downstream service degradation.

What the baselines reveal about tail risk. The separation between median and tail outcomes is the central result. Even when the typical disruption lasts a few hours, the upper tails can approach half-day recovery windows under plausible stress. This is the operational meaning of tail risk in protocol-network resilience: the distribution's upper quantiles—not its averages—drive worst-case planning. The corridor scenario (S4) illustrates this most sharply. Its median recovery is fast (about one hour), because many nodes regain connectivity quickly

Table 7.4: Baseline tail risk across scenarios. Loss ℓ is the baseline-normalized effectiveness deficit. Recovery time is reported in simulation steps (1 step $\approx$ 6.4 min) with approximate real-time equivalents in parentheses.

Scenario	**Med.** ℓ	$p90$ ℓ	$p95$ ℓ	**Med.** T_r	$p90$ T_r
S1 (Prysm overload)	0.093	0.160	0.182	40 ($\approx$4.3 h)	67 ($\approx$7.1 h)
S2 (Besu resync)	0.095	0.182	0.209	32 ($\approx$3.4 h)	93 ($\approx$9.9 h)
S3 (sustained ingress)	0.239	0.315	0.338	45 ($\approx$4.8 h)	109 ($\approx$11.6 h)
S4 (corridor disruption)	0.272	0.337	0.357	9 ($\approx$1.0 h)	47 ($\approx$5.0 h)

once the link is restored. But its 90th-percentile recovery is five hours, because a fraction of nodes must fully resynchronize after falling behind, and that resynchronization wave competes for bandwidth with normal protocol traffic. The gap between median and tail is where cascades live, because the initiating event is brief but the recovery dynamics that follow can stretch the disruption far beyond the event itself.

These baselines also connect back to the empirical anchors established earlier in the chapter. The nonstationary attack waves measured by EtherBee support the plausibility of the burst-driven inputs underlying S3. The peer tenure measurements (mean 26.46 days for top peers) support the plausibility of multi-hour propagation and recovery dynamics through stable peering relationships. And the geographic concentration documented in Chapter 6 directly motivates S4, where a corridor disruption between core regions produces outsized impact precisely because the overlay has concentrated its effective connectivity along those routes.

Transition to interventions. This chapter establishes the baseline temporal and tail-risk landscape across three protocol-network families. BitTorrent shows that early-window onset is measurable only with multi-year investment and that short, high-severity tails dominate security risk. Email shows that even when global seasonality is weak, burst windows and

temporal strategies differentiate infrastructure behavior. Ethereum shows bursty attack waves at scale and peer stability that supports cascades, and the simulator quantifies what those cascades look like when no mitigation is in place. Chapter 8 takes these same simulator outputs, metrics, and scenarios and evaluates whether proposed interventions—a private authenticated overlay mesh and an out-of-band snapshot recovery mechanism—can reduce the depth, duration, and tail severity of disruption episodes.

Part III

Interventions

Chapter 8

Measurement-Driven Interventions

Chapter 7 established a baseline disruption landscape using the node-infrastructure simulator. Those runs revealed moderate but nontrivial median losses under overload and resynchronization scenarios, substantially worse losses under sustained pressure and corridor disruption, and recovery-time tails that stretch well beyond their medians even in the less severe cases. The baseline numbers (Table 7.4) showed, for example, that S2's median recovery of roughly 3.4 hours is paired with a $p90$ near 10 hours, and that S3's $p90$ recovery approaches 11.6 hours. These tails—not the medians—drive worst-case operational planning.

This chapter translates those measured failure mechanisms into an **intervention evaluation**. Two operator-scale defenses drawn from the *Node Infrastructure First* study (Seidenberger et al. 2026c) are tested against the same scenario suite, using the same metrics, under the same tail-aware reporting framework. The design constraint requires that interventions be (i) motivated by measured exposure and cascade pathways, (ii) evaluable under the simulator's established outputs, and (iii) composable, so that operators can layer defenses without requiring protocol redesign.

The interventions are grounded in the empirical evidence established in Part II. Chapter 4 showed that running an Ethereum beacon node produces a roughly 3× visibility premium in hostile traffic (Table 4.1), that over a quarter of active beacon nodes expose auxiliary services beyond their P2P ports (Figure 4.5), and that traffic pressure alone can degrade a node's peer connectivity and validator performance without any protocol exploit (Figure 4.6).

Chapter 6 documented the corridor concentration that makes transatlantic connectivity a systemic chokepoint. Together, these measured pathways—from visibility to targeting, from exposure to load, from load to performance degradation—justify the two failure channels the interventions target, namely propagation impairment (the network is slow) and hard outages requiring state reconstruction (the node must rebuild).

The chapter proceeds as follows. Section 8.1 defines the evaluation framework and metrics. Section 8.2 specifies the two interventions. Section 8.3 describes the simulator architecture, experiment infrastructure, and a prototype overlay implementation. Sections 8.4 and 8.5 present and synthesize the results. Section 8.6 discusses deployment considerations, and Section 8.7 states the evaluation's limitations.

8.1 Evaluation Framework

8.1.1 Two Operator-Scale Defenses

This chapter evaluates two interventions, each targeting a distinct failure channel observed in the baseline simulations:

- **Intervention C (Connectivity):** an authenticated, membership-gated overlay mesh that provides a protected propagation channel for consensus traffic when public peering becomes unstable or overwhelmed.
- **Intervention R (Recovery):** out-of-band snapshot recovery—an operator-deployed mechanism that shortens time-to-recovery after hard outages by enabling snapshot-first state rebuilding rather than full peer-based reconstruction.

The two defenses are evaluated as a 2×2 factorial set—baseline, C only, R only, and C+R together—supporting both ablation analysis (isolating each defense's marginal contribution) and interaction analysis (whether the defenses are additive or synergistic when combined).

8.1.2 Metrics and Outputs

To keep results directly comparable to the baseline landscape established in Chapter 7, success is defined using the same simulator outputs and scalar summaries. The primary time series are **PoS effectiveness** $E(t)$, the fraction of validator duties performed on time at each simulation step, and **offline fraction** $O(t)$, the fraction of validators completely unavailable. Effectiveness can degrade under propagation impairment even when most nodes remain nominally online, making it a more sensitive indicator than offline fraction alone.

Three scalar summaries characterize each disruption episode:

- **Loss** (ℓ): the baseline-normalized area of the effectiveness deficit, integrated over the disruption window.
- **Recovery time** (T_r): the number of steps required to return to a reference effectiveness threshold near baseline.
- **Depth** (D): the peak drop below baseline effectiveness.

All metrics are reported as quantiles (median, $p90$, $p95$) rather than averages alone. As Chapter 7 demonstrated, the gap between median and upper-tail outcomes is where the most consequential disruption resides. Following the calibration used throughout this dissertation, one simulation step corresponds to approximately 6.4 minutes of real time (Seidenberger et al. 2026c), so a recovery time of 40 steps translates to roughly 4.3 hours.

The evaluation uses **paired comparisons**. For each random seed, all four variants are executed with identical stochastic draws—node placement, peering topology, bandwidth tiers, and event timing—so that intervention effects can be computed as per-seed paired differences (e.g., $\Delta\ell = \ell_{\text{variant}} - \ell_{\text{baseline}}$). This paired design isolates treatment effects from the substantial run-to-run variability that arises from stochastic network generation.

Because the outcome distributions are non-normal and heavy-tailed, significance summaries use nonparametric paired tests—specifically, one-sided Wilcoxon signed-rank tests

for median comparisons and McNemar exact tests for paired tail-exceedance rates, where exceedance is defined relative to a baseline-defined threshold ($\tau = q_{90}(\ell_{\text{baseline}})$). In addition to $\Delta\ell$, the analysis tracks whether interventions reduce outcome dispersion using ΔA, where A is the mean absolute deviation from the run median m:

$$A = \frac{1}{N} \sum_{i=1}^{N} |x_i - m|.$$

A negative ΔA indicates that the intervention compresses the distribution of outcomes, a desirable property for operational predictability.

8.2 Intervention Design

This section specifies each intervention's design and how it maps to the simulator. Section 8.3 provides full implementation details, including the simulator architecture and a working prototype of the overlay mesh.

8.2.1 Intervention C: Authenticated Overlay Mesh

Intervention C proposes an authenticated, membership-gated overlay that complements public peering. Its architecture separates two concerns:

- **Control plane.** An on-chain registry manages membership and discovery. It distributes the metadata needed to form overlay tunnels but does not route traffic itself.
- **Data plane.** Encrypted point-to-point tunnels (using WireGuard (Donenfeld 2017)) carry overlay traffic over authenticated links, decoupling overlay connectivity from public P2P discovery. Bootstrapping can optionally use privacy-preserving rendezvous before transitioning to direct tunnels, reducing the need for stable public endpoints as first-contact points.

Membership and diversity constraints. A stake-gated policy admits nodes above a stake threshold or within a top-K quantile by effective stake, focusing resources on consensus-critical participants. For correlated-risk management, the membership and peering policy can enforce diversity constraints, ensuring overlay peers are not concentrated in a single geographic region or network path group. This directly addresses the corridor dependencies documented in Chapter 6, where traffic-weighted focal points drift toward the North America–Europe corridor.

Simulator representation. Enabling C designates a subset of nodes as overlay participants (selected with stake-weighting to cover a target fraction of total stake) and constructs a fixed-degree overlay graph. Two operational controls are relevant:

1. Overlay participants may still use public peering, but their public peer cap can be reduced.

2. Under specified stress, participants can temporarily disable public peering and rely on the overlay alone ("fallback"), modeling a defensive retreat that limits further exposure.

The key independent variables are adoption fraction, overlay degree, and whether fallback is enabled.

8.2.2 Intervention R: Out-of-Band Snapshot Recovery

Intervention R targets hard-outage scenarios requiring substantial state reconstruction, such as hardware crashes, data corruption, or client bugs that necessitate database rebuilds. It provides a snapshot-first recovery path so that a node can move from a cold start to productive participation faster and with less load imposed on healthy peers.

Mechanism. Snapshots are published out-of-band with metadata enabling clients to select an appropriate snapshot and verify its authenticity and integrity. In the prototype design,

snapshots are stored in decentralized object storage and retrieved by downloading pieces in parallel from multiple storage nodes. This parallel-retrieval approach improves robustness but introduces performance variability, making tail behavior operationally relevant for recovery planning.

Security assumption. The simulator evaluation focuses on performance and recovery dynamics. It *assumes* that authenticity and integrity are ensured by the publication and verification pipeline (cryptographic verification and trusted publication or consensus anchoring). This assumption is stated explicitly to keep the evaluation's claims precise. The results measure recovery speed under snapshot assistance and do not address adversarial snapshot poisoning.

Simulation model. Enabling R designates a fraction of nodes as snapshot-capable. When such a node enters recovery, it attempts snapshot-first rebuilding if a recent snapshot is available. It downloads a snapshot of size S via a storage overlay (parallel retrieval from bounded storage peers under a download budget), then performs peer-based catch-up only for the remaining backlog accumulated since the snapshot was taken.

8.3 Implementation and Research Artifacts

The interventions described in the previous section are evaluated through two complementary artifacts. The first is a discrete-time network simulator that enables large-scale Monte Carlo experiments, and the second is a prototype implementation that demonstrates the overlay mesh end-to-end using real cryptographic tooling. The simulator is the primary evaluation artifact; the prototype is included to show that the overlay design can be implemented with existing components. This section describes both.

8.3.1 Simulator Architecture

The evaluation simulator is implemented in Python using NetworkX as its graph engine (Seidenberger et al. 2026c). It models an Ethereum-like proof-of-stake network as a discrete-time, stochastic system in which nodes maintain peer connections, propagate messages, and perform validator duties at each step.

Node model. Each node is parameterized by client type (Geth, Nethermind, or Besu), infrastructure provider (AWS, Azure, GCP, or on-premises), controlling entity type (solo operator, corporation, DAO, or university), geographic location (latitude/longitude sampled from region-specific distributions), and bandwidth capacity (download and upload, drawn from log-normal distributions). Nodes follow a state machine with five states: susceptible (S), infected (I), failed (F), syncing (SYNC), and recovered (R). The S→I and S→F transitions model contagion-based and external failure modes, respectively; the SYNC state models backlog-based chain resynchronization after an outage; and the R state marks nodes that have completed recovery. Each node maintains separate public and mesh peer sets with configurable capacity caps (default: 50 public, 8–12 mesh), and tracks a local chain-head slot that falls behind the global head during outages.

Network generation. The simulator generates networks at a default scale of 10,000 nodes. Geographic distribution concentrates nodes in North America (≈45%) and Europe (≈35%), with the remainder spread across Asia, South America, Africa, and Oceania—proportions chosen to approximate the measured Ethereum node distribution documented in Chapter 6. Peer discovery uses a DHT-like random-selection process with latency-aware pruning (Haversine distance, configurable maximum), and connection establishment incurs a 1–3 step delay. For overlay-enabled runs, mesh participants are selected by stake-weighting (top validators by effective stake) and connected via a ring-plus-random-edges topology that guarantees overlay connectivity while respecting a target mesh degree.

Step execution. Each simulation step executes six phases in sequence. (1) P2P maintenance (prune connections to offline peers, enforce capacity limits); (2) external failures (random hardware failures, client-specific bugs, DoS pressure on public peers, and correlated path/corridor outages); (3) infection spread through peer connections, with separate transmission rates for public (β_{public}) and mesh (β_{mesh}) edges—the lower mesh rate reflects the authenticated, traffic-controlled nature of overlay links; (4) recovery transitions, including backlog-based resynchronization where sync throughput depends on the recovering node's download bandwidth and its peers' available upload capacity; (5) peer discovery for online nodes; and (6) PoS block production and attestation, where proposers are selected via stake-weighted sampling and attestation timeliness is evaluated against a configurable deadline (default: 1,200 ms) using propagation-delay estimates derived from the geographic topology.

Recovery and resynchronization model. When a node recovers from an outage, it enters the SYNC state and must download a backlog proportional to the number of slots missed. Backlog size includes block data (≈1 MB/slot) plus, for large gaps, a state snapshot component. Sync throughput is bounded by the node's download capacity and the aggregate upload budget of its online peers. Snapshot-enabled nodes (Intervention R) bypass much of this backlog by downloading a pre-built snapshot from a storage overlay, then performing peer-based catch-up only for the remaining gap. The storage path is subject to its own bandwidth constraints and can be degraded by active corridor events, modeling real-world variability in decentralized storage retrieval.

Resilience metrics. The simulator computes resilience metrics from each run's effectiveness trajectory $E(t)$. The primary metric pipeline identifies the pre-shock baseline P_{pre} (median effectiveness in a configurable window before the disruption onset), the trough $P_{\min}$, the depth $D = P_{\text{pre}} - P_{\min}$, the recovery time T_r (steps to return to a target fraction of the baseline), and the normalized loss ℓ (deficit area divided by baseline $\times$ window length).

These metrics are computed per-run and then aggregated across the ensemble to produce the quantile summaries ($p50$, $p90$, $p95$) reported in this chapter.

8.3.2 Experiment Design and Execution

The 16,000 paired runs per scenario are generated using a **sliced Latin hypercube design** (SLHD) (Seidenberger et al. 2026c; Qian 2012) that provides both global stratification across the full parameter space and per-slice stratification for batch execution. The design sweeps over key independent variables—adoption fraction, overlay degree, infection rates, failure probabilities, and corridor-event parameters—while holding random seeds constant across the four variants (baseline, C, R, C+R) to enable paired comparisons.

The experiment framework is implemented as a FastAPI web application that serves as both the execution environment and the analysis interface. The webapp exposes a parameterized simulation API (`POST /api/run`) that accepts the full parameter vector, executes the simulation, and returns interactive Plotly visualizations of the results. For large-scale experiments, the framework supports batch execution with checkpoint persistence (Apache Parquet), allowing experiments to be paused, resumed, and merged across machines. The analysis module provides automated computation of paired treatment effects ($\Delta\ell$, ΔA), tail-risk quantile summaries, heatmap generation over two-dimensional parameter slices, Pareto frontier analysis, and global sensitivity estimation via surrogate models. The webapp and its source code are released as a research artifact alongside the supporting study to enable independent replication and extension of the results presented here.

8.3.3 Prototype Overlay: Registry and Rendezvous

To demonstrate that Intervention C is not merely a simulator abstraction, a prototype implementation realizes the full overlay lifecycle using production-grade cryptographic tooling. The prototype consists of three components: two Solidity smart contracts for on-chain

coordination, a Tor-based privacy-preserving rendezvous protocol for initial contact, and WireGuard tunnel establishment for the authenticated data plane.

Contracts. Two minimal smart contracts, deployed on an Ethereum-compatible chain, serve as the overlay's control plane:

- `SimpleCoordination`. Stores each participant's WireGuard public key (a 32-byte Curve25519 key, stored as `bytes32`) and network endpoint (IP:port). Any Ethereum address can call `registerNode()` to publish its overlay metadata; the contract maintains a global directory queryable via `getAllNodes()`.
- `OnionCoordination`. Stores ephemeral Tor onion addresses. A node that wishes to accept initial contact without exposing its public IP calls `registerOnion()` to publish a `.onion` address; peers retrieve it via `getOnion()`.

The contracts are intentionally minimal—no access-control lists or Sybil resistance beyond the cost of an Ethereum transaction—to isolate the coordination mechanism from policy decisions that would vary across deployments. In a production system, stake-gated membership (as described in Section 8.2.1) would be enforced at the contract level.

Privacy-preserving rendezvous via Tor. The prototype implements a rendezvous protocol that allows two nodes to establish contact without either revealing its public IP to the network at large. The workflow proceeds as follows:

1. **Publish.** The initiating node creates an ephemeral Tor hidden service (using the Stem library to interface with the local Tor daemon) and publishes the resulting `.onion` address to the `OnionCoordination` contract.
2. **Connect.** A peer retrieves the onion address from the contract and connects through the Tor network (SOCKS5 proxy on port 9050), reaching the hidden service without learning the initiator's IP.

3. **Exchange.** Once the Tor-mediated connection is established, both nodes exchange their real IP addresses and WireGuard public keys over the encrypted Tor channel.

4. **Transition.** With real endpoints in hand, both nodes tear down the ephemeral onion service and establish a direct WireGuard tunnel for ongoing overlay traffic.

This two-phase approach—Tor for bootstrapping, WireGuard for steady-state—avoids the latency and throughput limitations of Tor for bulk traffic while preserving the privacy benefit during the discovery phase. The ephemeral nature of the onion service means that the public record (the contract) contains only a transient address that becomes useless after the rendezvous completes.

WireGuard overlay establishment. Once real endpoints are exchanged, the prototype generates WireGuard configuration files for each participant and brings up authenticated tunnels on a private overlay subnet (e.g., `10.66.66.0/24`). Each tunnel uses Curve25519 key agreement with the public keys previously registered on-chain, providing mutual authentication without a certificate authority. Figure 8.1 shows the on-chain registry state after a four-node demo registration, Figure 8.2 shows the Tor-mediated endpoint exchange, and Figure 8.3 shows a successful overlay ping and WireGuard handshake between two nodes in isolated network namespaces.

```
Authenticated mesh overlay demo (local test chain)

SimpleCoordination deployed at: 0x5FbDB2315678afecb367f032d93F642f64180aa3

Registered nodes (on-chain):
 idx  address                                   wg_pubkey(bytes32)                                  endpoint              mesh_ip
 ---  ----------------------------------------  --------------------------------------------------  --------------------  -----------
 0    0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266  0x855c70e415dc256b84829b1f61aa4bc0833622c147709f98f6e7c3727edcc754  127.0.0.1:51820
 1    0x70997970C51812dc3A010C7d01b50e0d17dc79C8  0xf705417daf754cd8b8e4f99ca92a7bcda4ff36b565f3aefa53ca327cbfa7b427  127.0.0.1:51821
 2    0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC  0x0ab92d5fa2aedd07baf749e209f9f32fd74663a9b49790f6ecbddfe15a1f4622  127.0.0.1:51822
 3    0x15d34AAf54267DB7D7c367839AAf71A00a2C6A65  0x102fa0969483dfb532e12e2a8ee0cd14d8fef30ee86fa9aa1dd36ef248836778  127.0.0.1:51823

Generated WireGuard configs under: registry/demo_out
```

Figure 8.1: On-chain mesh registry after four nodes register their WireGuard public keys, network endpoints, and overlay IP addresses via the `SimpleCoordination` contract.

```
Onion rendezvous demo (real Tor runtime, local chain)

=== Server (registry/onion.py) ===

[*] Connecting to Tor control port...
[*] Authenticated to Tor.
[*] Created ephemeral onion: xrlpisl6665ojxgsayddnjqg4mvbs5glpctlp2p235miaj7cnlorwgad.onion
[*] Published onion address to contract. TX status: 1
[*] Listening for peer on 127.0.0.1:12345
[*] Waiting for a peer to connect over onion...
[*] Peer connected from: ('127.0.0.1', 55828)
[*] Received peer IP info: 203.0.113.11:51820
[*] Sent our real IP to peer.
[*] Peer IP is: 203.0.113.11:51820
[*] Both sides now have real endpoints; they can switch to direct authenticated tunnels.
[*] Tearing down ephemeral onion.

=== Client (registry/onionClient.py) ===

[*] Connecting to onion service at xrlpisl6665ojxgsayddnjqg4mvbs5glpctlp2p235miaj7cnlorwgad.onion:80 via Tor...
[*] Connected to the onion service.
[*] Sent our real IP: 203.0.113.11:51820
[*] Received peer real IP: 203.0.113.10:51820
[*] Connection closed.
```

Figure 8.2: Privacy-preserving rendezvous via Tor. The server creates an ephemeral onion service and publishes its address to the `OnionCoordination` contract; the client connects through Tor to exchange real IP addresses without either party exposing its endpoint publicly.

Takeaway. The prototype demonstrates that the overlay mechanism evaluated in simulation can be realized with existing, production-grade components: Ethereum smart contracts for decentralized coordination, Tor for privacy-preserving bootstrapping, and WireGuard for high-performance authenticated tunneling. The full end-to-end workflow—from contract deployment through registration, Tor rendezvous, and overlay tunnel establishment—is reproducible from the released source code and documented artifacts.

8.4 Scenario Results

This section reports the effects of C, R, and C+R across the four-scenario suite defined in Chapter 7 (Table 7.3): S1 (Prysm overload), S2 (Besu resync), S3 (2016 DoS pressure), and S4 (corridor disruption). Each scenario was evaluated with 16,000 paired simulation runs across a systematic parameter sweep, with all four variants sharing identical random seeds to isolate treatment effects. Baseline values reproduced here for convenience are drawn from Table 7.4; all intervention values are from Seidenberger et al. (2026c). Figure 8.4 provides a visual overview: each panel plots the median effectiveness trajectory $E(t)$ for the four variants on a shock-aligned time axis, with interquartile-range bands showing the run-to-run variability that the tables below summarize as scalar quantiles.

8.4.1 S1: Prysm Overload (Propagation Stress)

S1 models contagion-like overload inspired by a mainnet Prysm bug where expensive-to-verify messages exhaust node resources and spread degradation through peering, primarily stressing propagation rather than full resynchronization. The upper-left panel of Figure 8.4 shows the median effectiveness trajectory: the baseline dip and the progressively shallower troughs under C, R, and C+R are visible, with the C+R trajectory remaining closest to the pre-shock level throughout the event window.

```
WireGuard authenticated mesh overlay demo (2-node namespaces)

Namespaces:
 - wgdemons0 (192.168.100.1/24 on veth-wg0)
 - wgdemons1 (192.168.100.2/24 on veth-wg1)

WireGuard configs (restricted perms, stored in WSL):
 - /tmp/wg_local_demo/wgmesh0.conf
 - /tmp/wg_local_demo/wgmesh1.conf

Bringing up namespaces + veth...

Bringing up WireGuard interfaces...
[#] ip link add wgmesh0 type wireguard
[#] wg setconf wgmesh0 /dev/fd/63
[#] ip -4 address add 10.66.66.2/32 dev wgmesh0
[#] ip link set mtu 1420 up dev wgmesh0
[#] ip -4 route add 10.66.66.3/32 dev wgmesh0
[#] ip link add wgmesh1 type wireguard
[#] wg setconf wgmesh1 /dev/fd/63
[#] ip -4 address add 10.66.66.3/32 dev wgmesh1
[#] ip link set mtu 1420 up dev wgmesh1
[#] ip -4 route add 10.66.66.2/32 dev wgmesh1

Ping over overlay:
PING 10.66.66.3 (10.66.66.3) 56(84) bytes of data.
64 bytes from 10.66.66.3: icmp_seq=1 ttl=64 time=0.529 ms
64 bytes from 10.66.66.3: icmp_seq=2 ttl=64 time=0.252 ms
64 bytes from 10.66.66.3: icmp_seq=3 ttl=64 time=0.282 ms

--- 10.66.66.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2029ms
rtt min/avg/max/mdev = 0.252/0.354/0.529/0.124 ms

wg show (after ping) - wgdemons0:
interface: wgmesh0
  public key: hVxw5BXcJWuEgpsfYapLwIM2IsFHcJ+Y9ufDcn7cx1Q=
  private key: (hidden)
  listening port: 51820

peer: 9wVBfa91TNi45PmcqSp7zaT/NrVl8676U8oyfL+ntCc=
  endpoint: 192.168.100.2:51821
  allowed ips: 10.66.66.3/32
  latest handshake: 2 seconds ago
  transfer: 564 B received, 656 B sent
  persistent keepalive: every 25 seconds

wg show (after ping) - wgdemons1:
interface: wgmesh1
  public key: 9wVBfa91TNi45PmcqSp7zaT/NrVl8676U8oyfL+ntCc=
  private key: (hidden)
  listening port: 51821

peer: hVxw5BXcJWuEgpsfYapLwIM2IsFHcJ+Y9ufDcn7cx1Q=
  endpoint: 192.168.100.1:51820
  allowed ips: 10.66.66.2/32
  latest handshake: 2 seconds ago
  transfer: 508 B received, 564 B sent
  persistent keepalive: every 25 seconds
```

Figure 8.3: WireGuard overlay handshake. Two nodes in isolated network namespaces establish an authenticated tunnel using keys from the on-chain registry. The output shows a successful overlay ping and `wg show` confirming an active handshake with transfer counters.

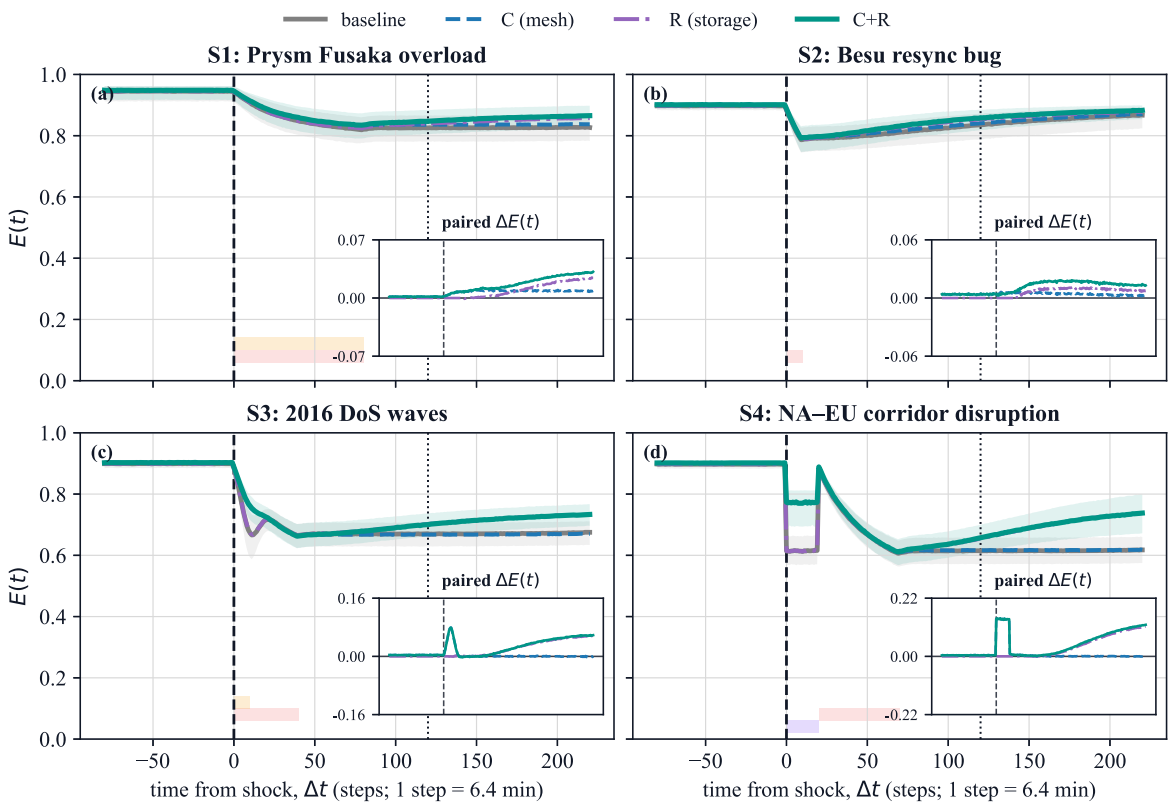

Figure 8.4: Effectiveness trajectories across all scenarios. Each panel shows median $E(t)$ for baseline, C, R, and C+R on a shock-aligned time axis ($\Delta t = t - t_0$). The dashed line marks the shock start and the dotted line marks the end of the scoring horizon H. Shaded bands show the interquartile range for baseline and C+R. Event windows are indicated by timeline bars (orange: infection/overload, red: failures/aftershock, purple: corridor impairment).

Table 8.1 reports the intervention effects. Connectivity hardening (C) produces the larger single-intervention improvement, consistent with propagation stress being the dominant failure channel in this scenario. Snapshot recovery (R) yields smaller but measurable gains, helping nodes that are pushed into harder failure states under the overload. The combined defense (C+R) achieves the best outcome on both median loss and recovery time.

In operational terms, median recovery drops from roughly 4.3 hours (baseline) to 1.1 hours under C+R—a fourfold acceleration. The tail results illustrate why intervention evaluation must be tail-aware. The $p90$ loss improves from 0.160 to 0.144 under C+R, a meaningful

Table 8.1: S1 (Prysm overload): intervention effects on loss and recovery time. Recovery time is in simulation steps (1 step ≈ 6.4 min) with approximate real-time equivalents.

Variant	**Med.** ℓ	$p90$ ℓ	$p95$ ℓ	**Med.** T_r **(approx.)**
Baseline	0.093	0.160	0.182	40 (≈4.3 h)
C	0.087	0.149	0.178	16 (≈1.7 h)
R	0.090	0.155	0.183	29 (≈3.1 h)
C+R	0.083	0.144	0.177	10 (≈1.1 h)

reduction, but $p95$ loss improves only modestly (0.182 to 0.177). The most extreme outcomes are harder to eliminate, reinforcing the observation from Chapter 7 that tail-aware reporting is essential for resilience decisions.

In overload-like propagation stress, **connectivity hardening (C) is the dominant lever**. The overlay provides a protected propagation substrate that maintains message timeliness even as public peering degrades. Adding snapshot recovery provides further improvement for nodes that fall into hard failure states, and the combined defense reduces both median disruption and tail-event likelihood.

8.4.2 S2: Besu Resync (Recovery Dominates)

S2 models a correlated client bug inspired by a Besu incident where a large fraction of nodes halt simultaneously and must perform full chain resynchronization. The dominant disruption channel is time spent in nonproductive sync states and the load that syncing imposes on remaining peers.

Table 8.2 shows that the intervention roles are reversed relative to S1. Snapshot recovery (R) drives the largest improvements by directly addressing the primary bottleneck: it replaces slow peer-based reconstruction with faster snapshot-first rebuilding. Connectivity

Table 8.2: S2 (Besu resync): intervention effects. T_r is reported in steps with approximate hours; the $p90$ recovery time highlights the persistent "long recovery" tail.

Variant	**Med.** ℓ	**Med.** T_r **(approx.)**	$p90$ T_r **(approx.)**
Baseline	0.095	32 (≈3.4 h)	93 (≈9.9 h)
C	0.092	18 (≈1.9 h)	92 (≈9.8 h)
R	0.083	10 (≈1.1 h)	85 (≈9.1 h)
C+R	0.083	6 (≈0.6 h)	83 (≈8.9 h)

hardening (C) yields smaller but measurable gains by improving propagation when public peering is stressed during resync waves.

The median recovery improvement is significant, going from 3.4 hours at baseline to 0.6 hours under C+R. But the $p90$ recovery times tell an equally important story: even with both interventions active, the $p90$ recovery remains near 8.9 hours (compared to 9.9 hours at baseline). This lingering tail reflects runs in which a large fraction of nodes must resynchronize simultaneously, overwhelming even the snapshot pipeline. The interventions compress the typical case dramatically while only moderately shortening the worst cases—a pattern that underscores the importance of reporting tail quantiles alongside medians.

In resync-dominated failure modes, **rapid recovery (R) is the primary lever**. Pairing it with connectivity hardening yields additional recovery-time reduction and improves robustness when concurrent propagation stress compounds the resync bottleneck.

8.4.3 S3: Sustained DoS Pressure

S3 models sustained denial-of-service pressure inspired by the 2016 DoS waves that forced emergency hard forks: external traffic saturates node bandwidth and stresses public ingress over a prolonged window.

Table 8.3: S3 (sustained DoS pressure): intervention effects on loss and recovery time.

Variant	**Med.** ℓ	$p90$ ℓ	**Med.** T_r **(approx.)**
Baseline	0.239	0.315	45 (≈4.8 h)
C	0.236	0.308	24 (≈2.6 h)
R	0.230	0.305	37 (≈3.9 h)
C+R	0.226	0.298	21 (≈2.2 h)

Baseline losses in S3 are substantially larger than in S1 or S2 (median $\ell \approx 0.24$ versus ≈0.09), reflecting sustained degradation rather than a short transient. Table 8.3 shows that both interventions contribute, but with a notable asymmetry in how they affect median versus tail outcomes.

Recovery (R) reduces both median loss and tail-event rates by shortening the duration and load of recovery for nodes forced into hard failure states under sustained pressure. Connectivity (C) improves median loss but is *regime-dependent*: the heatmap analysis (Figure 8.5) shows that C's benefit depends on DoS intensity and the timing of the defensive fallback window. Under low pressure, aggressive fallback—retreating to the overlay when the threat does not warrant it—can be neutral or slightly harmful, because it voluntarily reduces peer diversity when the public network is still functioning adequately. Under high pressure, the same fallback provides substantial protection. This regime dependence demonstrates that connectivity defenses under sustained ingress must be deployed with threshold-awareness.

C+R produces the largest median loss improvement and reduces tail-event rates most consistently across the adoption–intensity parameter space. Under sustained public-ingress pressure, **recovery acceleration (R) is critical for tail-risk reduction**, while **connectivity hardening (C) must be deployed with regime awareness**—fallback policies and intensity thresholds—to avoid counterproductive outcomes under low-pressure conditions.

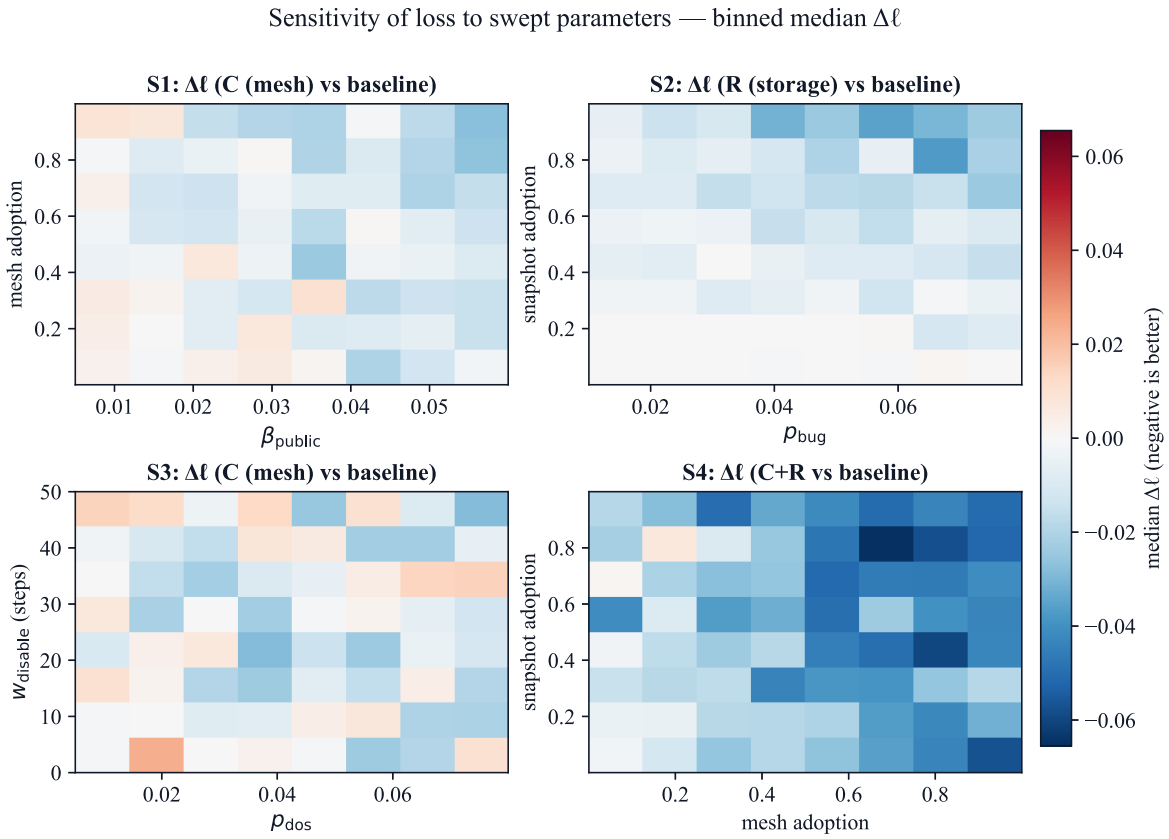

Figure 8.5: Sensitivity of loss improvements to swept parameters. Each panel shows the binned median paired effect $\Delta\ell$ for the scenario's primary intervention (C for S1/S3, R for S2, and C+R for S4) as a function of two key swept parameters (negative is better). The regime-dependent benefit of C in S3 is visible in the upper-right panel, where aggressive fallback under low pressure can be neutral or slightly harmful.

8.4.4 S4: Corridor Disruption

S4 models a North America–Europe corridor disruption motivated by the route concentration measured in Chapter 6. It is found that transatlantic connectivity emerges as a chokepoint in Ethereum's traffic-weighted focal points, so route impairment can create systemic propagation disruption even without widespread validator loss.

Baseline losses are the highest of any scenario (median $\ell \approx 0.27$), consistent with a chokepoint failure mode. Table 8.4 shows that connectivity hardening (C) produces the largest

Table 8.4: S4 (corridor disruption): intervention effects on loss and recovery time. A value of 0^* indicates that the treated run never incurred the baseline-defined recovery deficit.

Variant	**Med.** ℓ	**Med.** T_r **(approx.)**
Baseline	0.272	9 (≈1.0 h)
C	0.248	0^*
R	0.263	9 (≈1.0 h)
C+R	0.240	0^*

single-intervention improvement by directly augmenting reachability: overlay edges maintain cross-region propagation even when the public corridor is impaired. Snapshot recovery (R) provides smaller median gains but contributes to reducing severe tails by enabling faster recovery of nodes that fall into hard outages during and after the corridor event.

The zero-valued recovery times for C and C+R require interpretation. The evaluation uses a shared-target recovery definition anchored to the baseline run under the same seed. If the treated trajectory meets the recovery threshold before the baseline reaches its trough—or before the baseline-defined recovery target is crossed—the shared-target delay evaluates to zero. This is particularly plausible in corridor events where the overlay prevents the initial trough by maintaining cross-region connectivity, so the treated system never experiences the baseline's worst window. A T_r of zero does not imply literal instantaneous recovery; it means the treated variant avoided the baseline's recovery deficit entirely.

S4 exhibits the **largest median loss reductions** of any scenario. Connectivity augmentation reduces median loss from 0.272 to 0.248 (C alone) and to 0.240 (C+R), a paired improvement of -0.031 under C+R. When a hidden physical dependency becomes dominant, **connectivity augmentation (C) is the primary resilience lever**; pairing with R further reduces the probability of severe tail outcomes during recovery waves.

8.5 Cross-Scenario Synthesis

The scenario-by-scenario results establish that C and R act on distinct failure channels. This section makes that interaction explicit by reducing the results to a small set of operator-relevant effect sizes.

8.5.1 Paired Effects and Tail Rates

For each scenario and intervention, Table 8.5 reports the median paired loss difference ($\Delta\ell$, negative is better), the tail-event rate under a baseline-defined exceedance threshold ($\tau = q_{90}(\ell_{\text{baseline}})$, where 10% is the baseline rate by definition), and the median paired change in outcome dispersion (ΔA, negative indicates reduced variability).

Two observations follow directly from Table 8.5:

1. **C+R is consistently best on median loss** across all four scenarios, with paired improvements ranging from -0.009 (S1) to -0.031 (S4).

2. **Tail-event rates are reduced most strongly by C+R**, reaching their largest reduction in S4 (3.0% exceedance under C+R versus the 10% baseline rate). Recovery (R) often contributes a disproportionate reduction in tail-exceedance probability in recovery-dominated regimes: in S2, R alone reduces the tail rate to 6.2%, a larger reduction than C alone (9.8%).

The dispersion results (ΔA) are also operationally meaningful. R produces a notably large dispersion reduction in S2 (-0.0034), consistent with snapshot-first recovery compressing the distribution of outcomes by eliminating prolonged resync tails. By contrast, C's dispersion effect is scenario-dependent: it slightly *increases* dispersion in S3 ($+0.0009$), reflecting the regime-dependent nature of the fallback policy described in Section 8.4.3. Under some parameter combinations, the fallback increases outcome variability instead of compressing it.

Table 8.5: Cross-scenario synthesis of paired intervention effects. $\Delta\ell$: median paired loss difference vs. baseline (negative = improvement). Tail rate: fraction of runs exceeding the baseline-defined $p90$ loss threshold. ΔA: median paired change in dispersion (negative = less variable outcomes). Data from Seidenberger et al. (2026c).

Scenario	**Variant**	**Med. $\Delta\ell$**	**Tail rate (%)**	**Med. ΔA**
S1 (overload)	C	−0.006	7.0	−0.0008
	R	−0.003	8.3	−0.0010
	C+R	−0.009	5.8	−0.0016
S2 (resync)	C	−0.002	9.8	−0.0005
	R	−0.006	6.2	−0.0034
	C+R	−0.010	6.6	−0.0022
S3 (DoS)	C	−0.005	9.2	+0.0009
	R	−0.009	6.8	−0.0009
	C+R	−0.014	6.0	−0.0008
S4 (corridor)	C	−0.022	4.8	+0.0003
	R	−0.008	6.8	−0.0008
	C+R	−0.031	3.0	−0.0001

8.5.2 Mapping Interventions to Failure Channels

The scenario suite supports a direct mapping from intervention levers to the failure channels they address:

- **C helps most when propagation and hidden dependencies dominate.** Its strongest effect appears in S4 (corridor disruption, $\Delta\ell = -0.022$) and S1 (overload, $\Delta\ell = -0.006$), both scenarios in which connectivity impairment is the primary degradation mechanism.

- **R helps most when recovery dominates.** Its strongest effect appears in S2 (resync, $\Delta\ell = -0.006$), and it contributes meaningfully to tail reduction in S3 and S4 where hard outages and recovery waves occur under sustained pressure.

- **C+R is consistently best**, indicating that propagation protection and recovery acceleration complement each other. Across all four scenarios, the combined defense achieves both the lowest median loss and (with few exceptions) the lowest tail-event rate.

This mapping is the payoff of a measurement-first approach. The empirical mechanisms motivating each defense—visibility and targeting leading to propagation impairment (Chapter 4); hard outages requiring time-consuming resynchronization (Chapter 7)—reappear as separable simulator channels. The intervention effects are measurable in the same $E(t)$, ℓ, and T_r outputs used to characterize baseline risk, making the entire evaluation self-consistent because the same metrics that identified the problem are used to judge the response.

8.6 Deployment Considerations

The simulator results inform operator-facing guidance on where to invest and what coordination problems must be solved for benefits to materialize.

8.6.1 Network Effects and Coordination

Both interventions exhibit network effects because their benefits depend on covering a nontrivial fraction of the validator set, which requires coordination across operators rather than isolated action by a single node. This coordination burden is especially relevant in ecosystems such as Ethereum, where many validators are multiplexed onto fewer physical nodes and fleet operators, so that adoption by a relatively small number of stakeholders can disproportionately determine coverage and tail outcomes. The heatmap analysis (Figure 8.5)

shows that gains are modest at low adoption fractions and accelerate as coverage increases, a pattern consistent with network-effect dynamics.

8.6.2 Governance and Centralization Trade-offs

The interventions introduce governance questions directly connected to the dissertation's broader theme that interventions must reduce correlated risk without creating new chokepoints:

- **Overlay membership.** Stake-gated overlays concentrate effort on high-impact participants but can create perceived tiers of connectivity and raise fairness or centralization concerns. Diversity constraints are needed to avoid concentrating overlay peers in a single region or provider, which would replace one correlated dependency with another.
- **Snapshot services.** Snapshot-first recovery reduces recovery time but requires clear trust boundaries around authenticity, freshness, and publisher accountability. Multi-provider redundancy is preferable to reliance on a single snapshot source, which would create a new chokepoint in the recovery pipeline.

8.6.3 Operator Priorities

Keeping this chapter's scope narrow, the measured exposure landscape supports three immediate operator priorities:

1. **Reduce auxiliary-service exposure.** The scan results from Chapter 4 show that open auxiliary ports are common (25.68% of nodes expose at least one extra port), with a clear list of prevalent services and risky combinations. Closing or restricting these services directly reduces attacker leverage for workload amplification and compromise.
2. **Harden propagation-critical connectivity under stress.** Where operator coordination is possible, deploying an authenticated overlay with explicit diversity constraints

targets propagation impairment and route dependence. The results show that this lever dominates corridor-style disruptions (S4) and meaningfully improves overload regimes (S1).

3. **Treat recovery pipelines as first-class resilience infrastructure.** Snapshot-first recovery produces the largest gains in resync-dominated scenarios (S2) and reduces tail-event rates across multiple regimes. Operationally, this implies continuous testing of recovery workflows, bandwidth provisioning for recovery, and verified snapshot publication standards.

8.7 Limitations

The evaluation is intentionally effects-based and should be interpreted accordingly.

Simulator abstraction. The simulator models nodes, edges, geography, latency, bandwidth budgets, and state transitions to enable Monte Carlo ensembles and broad parameter sweeps, but it does not reproduce packet-level transport, detailed client networking behavior, or full Ethereum protocol execution. PoS effectiveness is a stake-weighted timeliness proxy rather than a full finality simulation. The metrics are therefore operational indicators and should not be read as direct statements about finality safety or liveness under real deployments.

Security assumptions for R. Snapshot-assisted recovery is evaluated primarily for performance and load shifting. It assumes that authenticity and integrity are handled by the publication and verification pipeline; adversarial snapshot poisoning and censorship are not simulated. Performance gains should not be conflated with security guarantees.

Adoption correlations. Adoption and deployment are parameterized via participation fractions and selection policies. Real-world deployment may be more correlated by provider

and fleet than the simulator's default sampling, which could change absolute outcomes. The simulator is therefore best viewed as a comparative instrument for estimating *relative treatment effects* and tail-risk reduction under controlled perturbations. It does not predict exact field outcomes.

These limitations sharpen rather than weaken the measurement-first claim. The chapter's contribution is an evaluable, tail-aware comparison of interventions under a controlled scenario suite—exactly what is needed to prioritize operator-scale defenses under uncertainty.

8.8 Chapter Summary

This chapter evaluated two operator-scale interventions—an authenticated overlay mesh (C) and out-of-band snapshot recovery (R)—using the node-infrastructure simulator and the resilience metrics established in Chapter 7. The key results, drawn from Tables 8.1–8.5, are:

- **C improves median loss most strongly in corridor disruption (S4):** baseline ℓ of 0.272 drops to 0.248 under C alone and to 0.240 under C+R, the largest single-scenario effect.
- **R drives the biggest recovery gains in resync regimes (S2):** median recovery time drops from 32 steps (≈3.4 h) to 10 steps (≈1.1 h) under R and to 6 steps (≈0.6 h) under C+R.
- **C+R reduces baseline-threshold tail-event rates across all scenarios** (Table 8.5), reaching its largest tail reduction in S4 (3.0% exceedance under C+R versus the 10% baseline rate).
- **Connectivity (C) is regime-dependent under sustained pressure (S3):** its benefit depends on DoS intensity and fallback-policy thresholds, and it can slightly increase outcome dispersion when applied in low-pressure regimes.

The evidence supports an intervention thesis that **operator-scale defenses at the node-infrastructure layer can be made measurable, evaluable, and composable**, and their success can be reported in the same tail-aware resilience metrics used to characterize baseline disruption risk. Chapter 9 carries these results, alongside the mechanism findings from Part II, into a concluding synthesis with practical takeaways, research directions, and policy implications.

Chapter 9

Synthesis and Conclusion

Chapter 8 showed that two interventions applied by operators can be effective in minimizing the effects of node and network disruptions. When compared against baselines, they can reduce disruption depth, recovery time, and tail-event rates. This chapter closes this work by translating the full body of evidence into an integrated synthesis of what was learned, what should be done in practice, what research should come next, and what policy implications follow. This chapter's role is to integrate the thesis contributions, reduce ambiguity about practical application, and set bounded next steps that match the evidence established in Chapters 1 through 8.

The central thesis of this dissertation remains measurement-first security. Protocol networks can appear robust when one looks only at average-case performance or evaluates them from a single vantage point in time or space. In practice, they can carry concentrated, time-varying, and correlated risk that only becomes visible through sustained observation. Across Ethereum, BitTorrent, email, and DNS, the combined evidence shows that credible claims depend on longitudinal, cross-layer, and multi-vantage observation, followed by intervention testing with the same metrics used to diagnose baseline risk.

This chapter is written for protocol designers, infrastructure operators, measurement researchers, and policy stakeholders. Section 9.1 answers the three research questions directly. Section 9.2 provides practical takeaways by audience. Section 9.3 outlines high-leverage

research directions. Section 9.4 states bounded policy implications. Section 9.5 concludes with the dissertation's final claims and limits.

Two caveats are important for this chapter. First, the practical guidance is grounded in results from earlier chapters rather than offered as stand-alone advice. Second, the recommendations are qualified where needed, because mechanisms and intervention effects vary by context and operating conditions.

9.1 Answering the Research Questions

9.1.1 RQ1 (Measurement)

Research question. How can credible, longitudinal, multi-vantage telescopes be built for protocol networks that enable reasoning about the system as a whole?

Answer. Credible protocol-network telescopes require a deliberate combination of longitudinal coverage, multi-vantage observation, cross-layer linkage, and explicit validity design from study inception.

Part I built this capacity through four reusable design patterns applied across three long-horizon empirical instruments and one adversarial evaluation artifact. These design patterns are multi-vantage observation, paired deployments and controls, cross-layer panels, and longitudinal collection. MagnetDB provided the multi-year discovery telemetry needed for event-time analysis in BitTorrent (Seidenberger et al. 2025b). EtherBee combined beacon-node and honeypot telemetry to quantify the visibility premium of running an Ethereum node, the route concentration of the physical layer, and large-scale reconnaissance activity against Ethereum (Seidenberger and Maiti 2025a). The email telescope enabled the study of both the temporal structure of corporate email marketing and risks to user privacy (Seidenberger et al. 2025a). NinjaDoH contributed a protocol design and implementation that was evaluated against realistic adversaries to create a censorship-resistant moving-target DoH service (Seidenberger et al. 2026a). Chapter 2 showed why these design

patterns are necessary for internal, construct, and external validity in this domain, and introduced the Dataset Value Taxonomy (DVT) as a framework for reasoning about when a dataset's temporal investment, scale, and accessibility are sufficient to support the claims a study intends to make (Seidenberger and Maiti 2025b).

Chapter 3 complemented telescope-building by introducing a modeling toolkit for inference under partial observability, including concentration and heavy-tail reporting, time-to-event models, churn and tenure measures, and tail-aware resilience metrics reused throughout the dissertation. The key methodological lesson is that the epistemic claims supported by large-scale Internet telescopes depend on how those telescopes and collections are designed. A study cannot recover later from missing multiple vantage points or insufficient temporal span once data collection ends. Designing for validity at the start, and evaluating dataset maturity through a framework such as the DVT, is therefore a methodological requirement. In practice, the answer to RQ1 is that measurement infrastructure is not background setup; it is one of the main determinants of which comparative and causal claims a protocol-network study can defend.

9.1.2 RQ2 (Mechanisms)

Research question. Which systemic mechanisms commonly shape privacy, security, and resilience across protocol families?

Answer. Four recurring mechanisms shape outcomes across the studied ecosystems: visibility as attack surface, de facto centralization, physical chokepoints, and temporal tail risk.

Across the four protocol families studied, these mechanisms recur in different forms. Visibility as attack surface was demonstrated in Ethereum, where beacon nodes receive roughly three times more hostile traffic than matched controls, and in email, where stable identifiers expand the contact surface of users' private information through a chain of intermediaries (Chapter 4). Those intermediaries are illustrative of the de facto centralization

measured in email infrastructure, where eight ASNs carry 89.35% of observed volume. In censorship-resistant DNS, NinjaDoH's feasibility depends on hyperscaler address space as a way to withstand blocking by already centralized censors (Chapter 5). Physical chokepoints were documented in Ethereum's traffic-weighted focal-point convergence toward the North America–Europe corridor, and structurally paralleled in BitTorrent's micro-network hubs that depend on a few well-provisioned initial seeders (Chapter 6). Temporal tail risk was established across BitTorrent, where pre-release windows carry fourteen-fold elevated malware risk, and Ethereum, where bursty attack waves propagate into cascading degradation (Chapter 7).

These mechanisms also map onto the ecological threat model introduced in Chapter 1. Visibility as attack surface is exploited by ordinary and strategic adversaries who allocate reconnaissance effort against discoverable targets. De facto centralization and physical chokepoints are produced largely by structural adversaries—rational optimizers whose local decisions create correlated dependencies without intentional harm. Temporal tail risk is amplified by all three adversary types, since bursts, cascades, and early-window harms arise from the interaction of intentional attacks, background abuse, and incentive-driven concentration.

The mechanisms are coupled, because logical centralization and physical chokepoints can magnify one another. Visibility can focus adversarial effort on the same limited infrastructure that concentration already stresses. Temporal dynamics determine when these interactions become most costly and hardest to recover from. Together, these four mechanisms form the organizing principles of Part II (Chapters 4 through 7). In practice, RQ2 means risk assessment should model the interactions among these mechanisms rather than focusing solely on one factor at a time.

9.1.3 RQ3 (Interventions)

Research question. Which interventions are practical, composable, and evaluable under real constraints?

Answer. Practical interventions target a specific measured failure channel, remain deployable at operator scale, compose with other controls, and are evaluated against a realistic baseline.

The most comprehensive evaluation of this question was in Ethereum, where the dissertation's measurement infrastructure was richest. Chapter 8 evaluated connectivity hardening (Intervention C), snapshot recovery (Intervention R), and their composition (C+R) across scenarios grounded in observed real-world incidents on the Ethereum mainnet. The broadest result is composability under scenario diversity: C+R improved both central tendency and upper-tail outcomes across scenarios, with the largest tail gains appearing in scenarios that included a major transatlantic cable disruption. At the same time, the chapter showed that intervention effectiveness depends on the failure mode in play. Connectivity hardening through a VPN overlay can be neutral or slightly counterproductive when hosts are not under significant pressure, because it adds both technical and administrative overhead.

In other words, the appropriate intervention must be mechanism-aligned. Hardening the P2P connectivity layer can help fight against the negative externalities of logical concentration and physical chokepoints (Chapter 6), while using a snapshot recovery tactic to get nodes back online more quickly after a failure is best at reducing tail risk (Chapter 7); their composition therefore functions as a portfolio of options rather than two unrelated controls that can simply be "set-and-forget".

Two complementary results from other protocol families reinforce the same conclusions. NinjaDoH (Chapter 2) demonstrated a build-measure-evaluate loop for censorship-resistant DNS, where the intervention was assessed against latency, cost, and evasion metrics under adversarial firewall and ML-based detection pressure (Seidenberger et al. 2026a). The piracy-onset study (Chapter 7) showed that market structure itself functions as a measurable lever for intervention (via content-distribution policy rather than protocol configuration): each additional streaming provider reduces the instantaneous piracy hazard by approximately 4.2% (Seidenberger et al. 2026b). The narrowing from four protocol families in RQ1 and

RQ2 to primarily Ethereum in Chapter 8 reflects the depth of measurement infrastructure required to study protocol networks comprehensively.

This combination of results matters because it replaces generic defense claims with parameterized deployment logic. The question is not "does the intervention work?" in the abstract. The question is "under which scenarios, thresholds, and operating constraints does the intervention improve the specific outcome we care about?" In practice, RQ3 means deployment should be conditional on scenario-specific triggers, explicit rollback criteria, and predeclared success metrics that include tail outcomes.

9.1.4 Cross-RQ Synthesis

Taken together, the answers to the three RQs describe a step-by-step process for comprehensively studying planet-scale decentralized networks. First, RQ1 establishes the need for high-quality, multi-vantage, and multi-modal telescopes that can support credible longitudinal inference. Second, RQ2 uses those telescopes to identify the mechanism structure that drives observed outcomes, including visibility as attack surface, de facto centralization, physical chokepoints, and temporal tail risk. Third, RQ3 evaluates interventions against those diagnosed mechanisms, so deployment decisions can be tied to measured baselines rather than operator intuition. The steps are sequential and interdependent such that without RQ1-quality measurement, mechanisms are underidentified; without RQ2 mechanism framing, interventions are chosen against symptoms rather than causes; and without RQ3-quality evaluation, deployment claims cannot be distinguished from engineering intuition unsupported by measured baselines.

This loop also clarifies why many protocol-network debates appear cyclical. Communities often change policy or software in response to incidents, but without stable baseline instrumentation and explicit counterfactuals. Later, incident patterns reappear and stakeholders disagree on whether the previous interventions worked. The dissertation's empirical arc suggests that this is, in significant part, a measurement-design failure—and one that

compounds coordination difficulties by depriving stakeholders of the shared evidence base needed to evaluate competing proposals. The remedy is to treat baseline measurement, mechanism diagnosis, and intervention evaluation as one continuous program rather than separate project phases.

9.2 Practical Takeaways

There are three main audiences for operational guidance based on findings from this body of work: protocol designers, infrastructure operators, and measurement researchers. The recommendations below integrate concrete actions, the chapters that motivate them, and the main tradeoffs that qualify them. The guidance presented here is not a warranty; it should be validated locally for the specific protocol and problem being addressed.

9.2.1 For Protocol Designers

Protocol designers should treat visibility as a budgeted resource by minimizing default discoverability beyond what liveness requires, hardening defaults for auxiliary services such as RPC and management endpoints, and separating operational interfaces from public protocol reachability wherever possible. Chapters 4 and 7 show that this matters because visibility functions as an attack surface and can amplify attacks that exploit a known open port. The tradeoff is that reducing visibility can also slow interoperability and create a network bootstrapping problem, so care should be taken up front when designing initial protocol implementations.

Designers should also optimize for diversity of failure domains rather than latency alone. The evidence in Chapters 6 and 8 supports adding diversity-aware terms to peer selection using region, ASN, and provider constraints, reserving connection slots for diversity-preserving peers, and monitoring concentration drift over software releases. That said, diversity controls can impose measurable latency and throughput costs; if the policy is too rigid, operators

may disable it, so the policy needs to remain bounded, measurable, and explainable in operator-facing tooling.

Recovery paths should be treated as first-class protocol components rather than emergency add-ons. Chapters 7 and 8 support building authenticated snapshot and state-transfer pathways into the architecture, defining integrity checks and provenance metadata, and publishing recovery service-level objectives (SLOs) before incidents occur. The main caveat is that recovery paths add operational complexity and attack surface, including poisoning, stale-state, and trust-bootstrap risks, so these subsystems need threat modeling and adversarial testing on par with core consensus or routing paths.

9.2.2 For Operators

Operators should instrument baseline risk before changing production behavior. Chapters 2, 7, and 8 support establishing baseline distributions for disruption depth, recovery time, effectiveness metrics, and tail exceedance rates before intervention rollout, then retaining comparable windows afterward so changes can be attributed credibly. This requires sustained monitoring budgets and can slow deployment, but skipping baseline instrumentation makes it difficult to defend claims that outcomes actually improved or regressed.

Interventions should also be rolled out in stages with clear thresholds for when to change modes. Chapter 8 supports defining pressure indicators, trigger points, and rollback rules that distinguish normal operation from periods of high stress, then rehearsing those transitions in staging before production rollout. The trade-off is more complicated runbooks and more mental load for the on-call team, and the thresholds themselves will drift as adversaries adapt, so recalibration and post-incident review need to become routine rather than exceptional work.

Operators should exercise correlated-failure drills rather than relying only on random-failure tests. Chapters 6 and 8 support including disruptions to key routes, provider-region outages, and dependency-cluster failures in tabletop and live exercises, and reporting both

median and tail recovery outcomes from those drills. These exercises are costly and may expose uncomfortable fragility, but that cost is precisely why they are valuable. Random-failure testing is unlikely to capture the kinds of interconnected systemic risks discussed here.

9.2.3 For Measurement Researchers

Measurement researchers should invest in longitudinal measurements. Chapters 2 and 7 support designing collection plans that account for the time required to understand a network's baseline behavior and capture rare events. This choice slows short-term publication velocity and increases maintenance burden, but it materially improves mechanism identification and research reproducibility.

Cross-layer linkage should be treated as a core design requirement rather than a late-stage enhancement. Chapters 5 and 6 support building schemas and pipelines that connect protocol entities to infrastructure and governance context while tracking mapping confidence and revision history over time. The main caution is that cross-layer linkage introduces attribution uncertainty and privacy concerns, so studies should publish confidence bounds and sensitivity analyses instead of presenting linkage as exact.

Researchers should also report tails and uncertainty as first-class outputs. Chapters 7 and 8 support publishing medians alongside upper-tail quantiles, exceedance rates, and scenario definitions, and reporting uncertainty intervals for comparative effects and intervention deltas. Tail-aware inference often demands larger samples and clearer assumptions, but average-only reporting routinely hides the operationally dominant failure regimes.

9.3 Research Directions

A first high-leverage direction is the development of cross-layer observatories that can maintain continuously updated dependency maps from protocol-level entities to infrastructure

and governance layers across multiple ecosystems. That would require persistent multi-source ingestion of protocol telemetry, routing data, and provider metadata, along with entity-resolution pipelines, provenance tracking, and periodic ground-truth audits to control mapping drift. The expected payoff is earlier detection of hidden concentration and correlated-failure risk, and stronger external validity for cross-ecosystem security claims. The challenge is organizational as much as technical, because the relevant data exist in fragments and change quickly, so a durable observatory would need versioned linkage methods, reproducible snapshots, and governance rules for sensitive metadata handling. This becomes more important as certain planet-scale decentralized networks are increasingly treated as public commons.

A second direction is standardized resilience reporting so that claims become comparable across papers, postmortems, and operator dashboards. Achieving that would require shared metric definitions, common scenario templates that include correlated failures, and reporting pipelines that publish medians and tail quantiles with uncertainty context. The payoff would be more comparable intervention evidence, less ambiguity in incident interpretation, and faster transfer of lessons between protocol families. The most pragmatic near-term step is a lightweight reporting profile rather than a heavy compliance framework, with at least four standardized elements: scenario description, baseline window, intervention condition, and an outcome summary that focuses on the tail risk.

A third direction is mechanism-interaction modeling that asks which pathways among visibility, concentration, chokepoints, and temporal dynamics most strongly predict severe disruption outcomes. That line of work would require joint models that fuse cross-layer telemetry with event-time annotations, plus controlled simulation studies that vary one mechanism while holding others fixed. The payoff would be intervention portfolios chosen for combined effect rather than isolated gains, which would reduce the risk of local improvements that worsen systemic fragility. This direction moves beyond additive reasoning, and the dissertation already starts down this path with the corridor-disruption scenario (S4 in

Chapter 8) that produced the worst baseline outcomes precisely because route concentration (Chapter 6) interacted with temporal tail dynamics (Chapter 7) to create cascading degradation that neither mechanism would have produced alone. Many real incidents are interaction-dominated in the same way. Capturing those regime transitions requires parsimonious interaction models that can reveal whether interventions should be sequenced, combined, or withheld under particular operating conditions.

A fourth direction is longitudinal and adversarial re-validation that asks which measured effects persist as protocols, software clients, and attacker strategies evolve. Pursuing that agenda would require scheduled replication campaigns, version-aware instrumentation, adversarial adaptation loops, and comparability protocols across measurement epochs. The payoff would be a cumulative, self-correcting evidence base in which claims remain current instead of turning into static historical artifacts. Re-validation should therefore be treated as scientific maintenance rather than optional follow-up work, because protocol upgrades, infrastructure shifts, and adversarial adaptation can all invalidate old priors. A disciplined replication cadence that pairs each major mechanism claim with a scheduled refresh interval and a predefined comparability protocol is essential if design guidance is to remain reliable over time.

Across all four directions, the common bottleneck is durable measurement capacity. The strongest claims in this dissertation came from settings with higher temporal investment, broader vantage diversity, and tighter alignment between baseline diagnosis and intervention testing (Seidenberger and Maiti 2025b).

9.4 Policy Implications

9.4.1 Infrastructure Dependency Transparency and Disclosure

Chapters 5 and 6 show that not knowing which interconnected dependencies exist is itself a systemic risk. Protocol ecosystems can look decentralized while still relying on a small

set of infrastructure providers, Internet routes, or political jurisdictions. A bounded policy implication is periodic aggregate dependency disclosure by major operators and intermediaries, including provider concentration, regional concentration, and known single points of dependence.

This recommendation does not require publication of sensitive operational details. The objective is risk visibility, not tactical transparency for adversaries. Disclosures should be aggregated, trend-oriented, and independently auditable so they support ecosystem-level situational awareness without exposing exploit-ready specifics.

A pragmatic near-term instrument is voluntary disclosure templates administered by neutral organizations, with guardrails that separate descriptive dependency reporting from normative judgments and that avoid revealing exploitable operational detail. If adopted widely, such templates can establish baseline expectations before stronger formal requirements are considered.

9.4.2 Public Investment in Redundant Infrastructure

Chapter 6 and the scenario results in Chapter 8 show that some of the largest risks are around physical chokepoints and routing concentration. In scenario (S4), the simulated disruption of the transatlantic cables connecting the US and EU produced the highest baseline losses of any scenario class evaluated, with $p90$ recovery times roughly five times the median. This supports a bounded policy implication that targeted public or consortium investment should be in infrastructure redundancy that benefits multiple protocol ecosystems. This would include diverse interconnection capacity, resilient regional hosting options, and improved route diversity for under-served regions.

This is not an argument that physical redundancy replaces protocol-level hardening. The evidence supports a layered view that software defenses and infrastructure diversity are complements. When critical protocol traffic shares narrow physical paths, **market incentives**

alone may under-provide resilience because costs are concentrated while benefits are diffuse.

Policy programs in this area should remain outcome-based, with investments tied to measurable resilience improvements such as reduced concentration, faster recovery under correlated-failure drills, and lower tail exceedance rates. Funding criteria should reward multi-ecosystem utility and open evaluation protocols rather than favoring single-vendor lock-in, since redundancy spending can otherwise reproduce the concentration patterns it seeks to reduce.

9.4.3 Public-Good Funding for Long-Horizon Measurement

The dissertation's strongest findings required sustained collection and maintenance over long horizons. That cost profile rarely aligns with incentives of any single commercial actor, especially when the primary beneficiaries are broad ecosystems and future researchers. A bounded policy implication is to fund long-horizon measurement infrastructure as a public-good research asset, with explicit support for maintenance, documentation, and independent replication. Not all datasets should be open, and central institutions should not control these protocols, but durable measurement capacity is a prerequisite for credible security science, and current funding models frequently under-provide that capacity.

Effective funding programs should require clear measurement protocols, explicit limitations, privacy and ethics safeguards, and mechanisms for third-party verification. These conditions preserve scientific rigor while improving long-term evidence quality. Evaluation criteria should include durability, not only novelty of a measurement program. Programs that continuously maintain high-quality baselines can deliver larger long-run benefits than short projects that produce one-time analyses without sustainable infrastructure.

9.5 Concluding Remarks

This dissertation began from a simple observation that protocol networks increasingly function as critical coordination infrastructure, yet their security is still often assessed through short-horizon, single-vantage, or average-case lenses that hide high-impact failure behavior.

The dissertation's main contributions follow from that observation. First, it builds and operationalizes reusable telescopes and evaluation artifacts: MagnetDB, EtherBee, the email inbox sensor, and NinjaDoH. It also formalizes dataset maturity and stewardship through the Dataset Value Taxonomy (DVT). Second, it identifies four recurring mechanisms—visibility as attack surface, de facto centralization, physical chokepoints, and temporal tail risk—that explain why similar failure patterns appear across different protocol families. Third, it evaluates operator-scale interventions, specifically connectivity hardening and snapshot recovery, in a scenario-grounded, tail-aware simulator, showing that composable controls can improve both typical and upper-tail outcomes when judged against baseline-consistent metrics.

Those contributions are still bounded by the dissertation's scope. The four protocol families studied here—Ethereum, BitTorrent, email, and DNS—do not exhaust the space of protocol networks, so whether the same mechanisms dominate in other families remains an open empirical question. The Ethereum measurement window spans roughly three months, which captures substantial dynamics but still reflects a particular phase of the network's evolution. The Chapter 8 interventions were evaluated in simulation rather than field deployment, so real-world coordination costs, adoption dynamics, and adversarial adaptation in production remain to be tested. Some results rely on effects-based abstractions rather than full protocol reimplementation, and intervention benefits remain scenario- and threshold-dependent. These limits do not negate the findings, but they do bound the claims and define the replication agenda.

A mature science of secure protocol networks will require durable shared measurement infrastructure, standardized resilience reporting, and repeated adversarial re-validation as

protocols evolve. The contribution of this dissertation is to provide a foundation for that trajectory. Moving toward that science means measuring hidden dynamics rigorously, identifying recurring mechanisms across ecosystems, and evaluating interventions with close attention to tail outcomes and systemic risk.

In the near term, success would mean protocol communities funding solutions that mitigate tail-risk events at the physical infrastructure layer, operators exercising correlated-failure scenarios as routine practice, and research groups maintaining longitudinal datasets that are refreshed rather than archived and forgotten. Each of those outcomes is measurable, each is directly connected to the evidence developed here, and together they would make future decisions in this field more comparable and more defensible.

References

Acquisti, A., L. Brandimarte, and G. Loewenstein, 2015: Privacy and human behavior in the age of information. *Science*, 347 (6221), 509–514.

Alata, E., V. Nicomette, M. Kaâniche, M. Dacier, and M. Herrb, 2006: Lessons learned from the deployment of a high-interaction honeypot. *Proc. EDCC '06*, IEEE, 39–46.

Alpern, B. and F. B. Schneider, 1985: Defining liveness. *Information Processing Letters*, 21 (4), 181–185, doi:10.1016/0020-0190(85)90056-0.

Anderson, R., 2001: Why information security is hard-an economic perspective. *Seventeenth annual computer security applications conference*, IEEE, 358–365.

Arkime Project, 2024: Arkime: Network analysis & packet capture. Arkime Project, `https://github.com/arkime/arkime`, Version 5.2.0.

Avizienis, A., J.-C. Laprie, B. Randell, and C. Landwehr, 2004: Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1 (1), 11–33.

Bailey, M., D. Dittrich, E. Kenneally, and D. Maughan, 2012: The Menlo report. *IEEE Security & Privacy*, 10 (2), 71–75.

Barford, P., Y. Chen, A. Goyal, Z. Li, V. Paxson, and V. Yegneswaran, 2010: Employing honeynets for network situational awareness. *Cyber Situat. Awareness (CSA)*, 71–102.

Bass, F. M., 1969: A new product growth for model consumer durables. *Management Science*, 15 (5), 215–227, doi:10.1287/mnsc.15.5.215.

Beiko, T., 2019: Mainnet consensus bug identified and resolved in Hyperledger Besu. Hyperledger Foundation, `https://wiki.hyperledger.org/display/BESU/Mainnet+Consensus+Bug+Identified+and+Resolved+in+Hyperledger+Besu`, accessed 2026-03-03.

Böttger, T., F. Cuadrado, G. Antichi, E. L. Fernandes, G. Tyson, I. Castro, and S. Uhlig, 2019: An empirical study of the cost of DNS-over-HTTPS. *Proceedings of the Internet Measurement Conference*, 15–21.

Boyd, D. and K. Crawford, 2012: Critical questions for big data. *Information, Communication & Society*, doi:10.1080/1369118x.2012.678878.

Brumley, D. and D. Boneh, 2005: Remote timing attacks are practical. *Computer Networks*, 48 (5), 701–716.

Bruneau, M., et al., 2003: A framework to quantitatively assess and enhance the seismic resilience of communities. *Earthquake Spectra*, 19 (4), 733–752.

Buldyrev, S. V., R. Parshani, G. Paul, H. E. Stanley, and S. Havlin, 2010: Catastrophic cascade of failures in interdependent networks. *Nature*, 464 (7291), 1025–1028, doi:10.1038/nature08932.

Buterin, V., et al., 2020: Combining GHOST and Casper. *arXiv preprint arXiv:2003.03052*.

Cavoukian, A. et al., 2009: Privacy by design: The 7 foundational principles. *Information and privacy commissioner of Ontario, Canada*, 5 (2009), 12.

Chandola, V., A. Banerjee, and V. Kumar, 2009: Anomaly detection: A survey. *ACM Computing Surveys*, 41 (3), 1–58, doi:10.1145/1541880.1541882.

Chin, K. and K. Omote, 2021: Analysis of attack activities for honeypots installation in Ethereum network. *Proc. IEEE Blockchain '21*, IEEE, 440–447.

Clauset, A., C. R. Shalizi, and M. E. J. Newman, 2009: Power-law distributions in empirical data. *SIAM Review*, 51 (4), 661–703, doi:10.1137/070710111.

Coleman, J. S., E. Katz, and H. Menzel, 1966: *Medical Innovation: A Diffusion Study*. Bobbs-Merrill Company, Indianapolis, IN.

Cortes-Goicoechea, M., T. Mohandas-Daryanani, J. L. Muñoz-Tapia, and L. Bautista-Gomez, 2023: Unveiling Ethereum's hidden centralization incentives: Does connectivity impact performance? *2023 Fifth International Conference on Blockchain Computing and Applications (BCCA)*, 89–96, doi:10.1109/BCCA58897.2023.10338892.

Cowrie, 2023: SSH/telnet honeypot. Cowrie Project, `https://github.com/cowrie/cowrie`.

Cox, D. R., 1972: Regression models and life-tables. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 34 (2), 187–202, doi:10.1111/j.2517-6161.1972.tb00899.x.

Crovella, M. E. and A. Bestavros, 1997: Self-similarity in World Wide Web traffic: evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5 (6), 835–846, doi:10.1109/90.650143.

Danaher, B., J. Hersh, M. D. Smith, and R. Telang, 2020: The effect of piracy website blocking on consumer behavior. *MIS Quarterly*, 44 (2), 631–660.

Donenfeld, J. A., 2017: WireGuard: next generation kernel network tunnel. *NDSS*, 1–12.

Durairajan, R., P. Barford, J. Sommers, and W. Willinger, 2015: InterTubes: A study of the US long-haul fiber-optic infrastructure. *Proceedings of the 2015 ACM conference on special interest group on data communication*, 565–578.

Durumeric, Z., M. Bailey, and J. A. Halderman, 2014a: An Internet-Wide view of Internet-Wide scanning. *23rd USENIX Security Symposium (USENIX Security 14)*, 65–78.

Durumeric, Z., et al., 2014b: The matter of Heartbleed. *Proceedings of the 2014 conference on internet measurement conference*, 475–488.

Englehardt, S., J. Han, and A. Narayanan, 2018: I never signed up for this! privacy implications of email tracking. *Proceedings on Privacy Enhancing Technologies*.

Ethereum Foundation, 2016: Hard fork no. 4: Spurious Dragon. Ethereum Foundation, `https://blog.ethereum.org/2016/11/18/hard-fork-no-4-spurious-dragon`, accessed 2026-03-03.

Fachkha, C. and M. Debbabi, 2015: Darknet as a source of cyber intelligence: Survey, taxonomy, and characterization. *IEEE Commun. Surv. Tutor.*, 18 (2), 1197–1227.

Feulner, S., T. Guggenberger, J.-C. Stoetzer, and N. Urbach, 2025: Beyond disintermediation: A multiple case study of emerging intermediary roles in blockchain applications. *Electronic Markets*, 35 (1), 98.

Fiebig, T., F. Lichtblau, F. Streibelt, T. Krueger, P. Lexis, R. Bush, and A. Feldmann, 2016: SoK: an analysis of protocol design: avoiding traps for implementation and deployment. *arXiv preprint arXiv:1610.05531*.

Fifield, D., C. Lan, R. Hynes, P. Wegmann, and V. Paxson, 2015: Blocking-resistant communication through domain fronting. *Proceedings on Privacy Enhancing Technologies*.

Florêncio, D. and C. Herley, 2012: Where do all the attacks go? *Economics of information security and privacy III*, Springer, 13–33.

Foundation, M., 2023: Tanner: Web application honeypot. MushOrg Foundation, `https://github.com/mushorg/tanner`, Tanner/SNARE project.

Gandomi, A. and M. Haider, 2015: Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35 (2), 137–144, doi:10.1016/j.ijinfomgt.2014.10.007.

Gencer, A. E., S. Basu, I. Eyal, R. Van Renesse, and E. G. Sirer, 2018: Decentralization in Bitcoin and Ethereum networks. *Financial Cryptography and Data Security: 22nd International Conference, FC 2018, Nieuwpoort, Curaçao, February 26–March 2, 2018, Revised Selected Papers*, Springer, 439–457.

Granovetter, M., 1978: Threshold models of collective behavior. *American Journal of Sociology*, 83 (6), 1420–1443, doi:10.1086/226707.

Halevy, A., P. Norvig, and F. Pereira, 2009: The unreasonable effectiveness of data. *IEEE intelligent systems*, 24 (2), 8–12.

Hazel, G. and A. Norberg, 2008: BEP 9: Extension for peers to send metadata files. BitTorrent Project, last modified 2017-03-26; accessed 2025-08-31, `https://www.bittorrent.org/beps/bep_0009.html`, last modified 2017-03-26; accessed 2025-08-31.

Hoang, N. P., et al., 2021: How great is the Great Firewall? measuring China's DNS censorship. *30th USENIX Security Symposium (USENIX Security 21)*, 3381–3398.

Hoffman, P. and P. McManus, 2018: DNS Queries over HTTPS (DoH). RFC 8484, Internet Engineering Task Force (IETF). URL `https://www.rfc-editor.org/rfc/rfc8484`.

Honda, M., Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda, 2011: Is it still possible to extend TCP? *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, 181–194.

Hoofnagle, C. J., J. M. Urban, and S. Li, 2012: Privacy and modern advertising: Most US Internet users want Do Not Track to stop collection of data about their online activities. *Amsterdam Privacy Conference*.

Hu, Z., L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, 2016: Specification for DNS over Transport Layer Security (TLS). RFC 7858, Internet Engineering Task Force (IETF). URL `https://www.rfc-editor.org/rfc/rfc7858`.

Imamura, M. and K. Omote, 2019: Network deployments of Bitcoin peers and malicious nodes based on darknet sensor. *Proc. WISA'18*, Springer, 117–128.

Kermack, W. O. and A. G. McKendrick, 1927: A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 115 (772), 700–721, doi:10.1098/rspa.1927.0118.

Kim, S. K., Z. Ma, S. Murali, J. Mason, A. Miller, and M. Bailey, 2018: Measuring Ethereum network peers. *Proceedings of the Internet Measurement Conference*, 91–104, doi:10.1145/3278532.3278542.

Klensin, J., 2008: Simple Mail Transfer Protocol. RFC 5321, Internet Engineering Task Force (IETF). URL `https://www.rfc-editor.org/rfc/rfc5321`.

Kryczka, M., R. Cuevas, C. Guerrero, A. Azcorra, and A. Cuevas, 2011: Measuring the BitTorrent ecosystem: Techniques, tips, and tricks. *IEEE Communications Magazine*, 49 (9), 144–152.

Laney, D., 2001: 3D data management: Controlling data volume, velocity, and variety. Research Note 949, META Group. Application Delivery Strategies.

Léna, P., F. Lebrun, and F. Mignard, 2013: *Observational astrophysics*. Springer Science & Business Media.

Leonelli, S., 2014: What difference does quantity make? on the epistemology of big data in biology. *Big data & society*, doi:10.1177/2053951714534395.

Little, R. J. and D. B. Rubin, 2019: *Statistical analysis with missing data*. John Wiley & Sons.

Loewenstern, A. and A. Norberg, 2008: BEP 5: DHT protocol. BitTorrent Project, last modified 2020-01-21; accessed 2025-08-31, `https://www.bittorrent.org/beps/bep_0005.html`, last modified 2020-01-21; accessed 2025-08-31.

Maslej, N., et al., 2025: Artificial intelligence index report 2025. *arXiv preprint arXiv:2504.07139*.

Master, A., 2023: Modeling and characterization of Internet censorship technologies. Ph.D. thesis, Purdue University.

Maymounkov, P. and D. Mazieres, 2002: Kademlia: A peer-to-peer information system based on the XOR metric. *International workshop on peer-to-peer systems*, Springer, 53–65.

MontazeriShatoori, M., L. Davidson, G. Kaur, and A. H. Lashkari, 2020: Detection of DoH tunnels using time-series classification of encrypted traffic. *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, IEEE, 63–70.

Newman, M., 2010: *Networks: An Introduction*. Oxford University Press, doi:10.1093/acprof:oso/9780199206650.001.0001.

Newman, M. E. J., 2002: Spread of epidemic disease on networks. *Physical Review E*, 66 (1), 016 128, doi:10.1103/PhysRevE.66.016128.

Niaki, A. A., S. Cho, Z. Weinberg, N. P. Hoang, A. Razaghpanah, N. Christin, and P. Gill, 2020: ICLab: A global, longitudinal Internet censorship measurement platform. *2020 IEEE Symposium on Security and Privacy (SP)*, IEEE, 135–151.

Ohm, P., 2009: Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA l. Rev.*, 57, 1701.

Pang, R., V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson, 2004: Characteristics of Internet background radiation. *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, 27–40.

Pastor-Satorras, R. and A. Vespignani, 2001: Epidemic spreading in scale-free networks. *Physical Review Letters*, 86 (14), 3200–3203, doi:10.1103/PhysRevLett.86.3200.

Pauley, E., P. Barford, and P. McDaniel, 2023: DScope: A cloud-native Internet telescope. *32nd USENIX Security Symposium*.

Paxson, V., 1997: End-to-end routing behavior in the Internet. *IEEE/ACM Transactions on Networking*, 5 (5), 601–615.

Pearce, P., B. Jones, F. Li, R. Ensafi, N. Feamster, N. Weaver, and V. Paxson, 2017: Global measurement of DNS manipulation. *26th USENIX Security Symposium (USENIX Security 17)*, 307–323.

Pérez, R. G., 2023: Submarine cables across the Atlantic: Geopolitics and security of a critical infrastructure. *Atlantic Center Report*, 3, 57–82.

Pierson, P., 2000: Increasing returns, path dependence, and the study of politics. *American political science review*, 94 (2), 251–267.

Plice, R. K., O. V. Pavlov, and N. P. Melville, 2008: Spam and beyond: An information-economic analysis of unwanted commercial messages. *Journal of Organizational Computing and Electronic Commerce*, 18 (4), 278–306.

Pouwelse, J., P. Garbacki, D. Epema, and H. Sips, 2005: The BitTorrent P2P file-sharing system: Measurements and analysis. *International Conference on Peer-to-Peer Systems (IPTPS)*, Springer, 205–216.

Prysm Team, 2025: Fusaka mainnet Prysm incident. Offchain Labs, `https://prysm.offchainlabs.com/docs/misc/mainnet-postmortems/`, section "Fusaka Mainnet Prysm Incident", accessed 2026-03-03.

Qian, P. Z., 2012: Sliced latin hypercube designs. *Journal of the American Statistical Association*, 107 (497), 393–399.

Rinaldi, S. M., J. P. Peerenboom, and T. K. Kelly, 2001: Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE control systems magazine*, 21 (6), 11–25.

Rogers, E. M., 2003: *Diffusion of Innovations*. 5th ed., Free Press, New York, NY.

Seidenberger, S., O. Ajisegiri, N. Pursell, F. Raja, and A. Maiti, 2025a: Why you've got mail: Evaluating inbox privacy implications of email marketing practices in online apps and services. *Proceedings of the Fifteenth ACM Conference on Data and Application Security and Privacy (CODASPY '25)*, doi:10.1145/3714393.3726516.

Seidenberger, S., M. Beret, R. Wijewickrama, M. Jadliwala, and A. Maiti, 2026a: NinjaDoH: A censorship-resistant moving target DoH server using hyperscalers and IPNS. *Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb)*.

Seidenberger, S. and A. Maiti, 2025a: EtherBee: A global dataset of Ethereum node performance measurements coupled with honeypot interactions and full network sessions. *2025 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 1–3, doi:10.1109/ICBC64466.2025.11114523.

Seidenberger, S. and A. Maiti, 2025b: From big data to valued data: A dataset value taxonomy for AI-native empirical research. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, Vol. 8, 2294–2305.

Seidenberger, S. and A. Maiti, 2025c: Initial evidence of pervasive reconnaissance targeting Ethereum node infrastructure. *Proceedings of the 7th ACM International Symposium on Blockchain and Secure Critical Infrastructure*, 1–8.

Seidenberger, S., N. Pursell, and A. Maiti, 2025b: MagnetDB: A longitudinal torrent discovery dataset with IMDb-matched movies and TV shows. *Proceedings of the International AAAI Conference on Web and Social Media*.

Seidenberger, S., N. Pursell, and A. Maiti, 2026b: Front-running the red carpet: A measurement study of media torrent leaks and user risks, under review at Journal of Quantitative Description: Digital Media.

Seidenberger, S., K. Trinh, and A. Maiti, 2026c: Node infrastructure first: Measuring attacks, modeling cascades, and rapid recovery tactics for public blockchains, under review at Distributed Ledger Technology (DLT).

Skålvik, A. M., C. Saetre, K.-E. Frøysa, R. N. Bjørk, and A. Tengberg, 2023: Challenges, limitations, and measurement strategies to ensure data quality in deep-sea sensors. *Frontiers in Marine Science*, 10, 1152 236.

Star, S. L. and K. Ruhleder, 1994: Steps towards an ecology of infrastructure: complex problems in design and access for large-scale collaborative systems. *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 253–264.

Sterbenz, J. P., D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, 2010: Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. *Computer Networks*, 54 (8), 1245–1265.

Stutzbach, D. and R. Rejaie, 2006: Understanding churn in peer-to-peer networks. *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, ACM, 189–202.

T-Pot Project, 2024: T-Pot: The all in one honeypot platform. T-Pot Project, `https://github.com/telekom-security/tpotce`, Version 24.04.0.

Troncoso, C., M. Isaakidis, G. Danezis, and H. Halpin, 2017: Systematizing decentralization and privacy: Lessons from 15 years of research and deployments. *Proceedings on Privacy Enhancing Technologies*, 2017 (4), 307–329, doi:10.1515/popets-2017-0052.

Van Dijck, J., T. Poell, and M. De Waal, 2018: *The platform society: Public values in a connective world*. Oxford university press.

Wang, L. and J. Kangasharju, 2013: Measuring large-scale distributed systems: case of BitTorrent mainline DHT. *IEEE P2P 2013 Proceedings*, IEEE, 1–10.

Wang, T., C. Zhao, Q. Yang, S. Zhang, and S. C. Liew, 2021: Ethna: Analyzing the underlying peer-to-peer network of Ethereum blockchain. *IEEE Transactions on Network Science and Engineering*, 8 (3), 2131–2146.

Watts, D. J., 2002: A simple model of global cascades on random networks. *Proceedings of the National Academy of Sciences*, 99 (9), 5766–5771, doi:10.1073/pnas.082090499.

Wolchok, S. and J. A. Halderman, 2010: Crawling BitTorrent DHTs for fun and profit. *4th USENIX Workshop on Offensive Technologies (WOOT 10)*.

Ye, J., X. D. C. De Carnavalet, L. Zhao, M. Zhang, L. Wu, and W. Zhang, 2024: Exposed by default: A security analysis of home router default settings. *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, 63–79.

Zhang, B., A. Iosup, J. Pouwelse, D. Epema, and H. Sips, 2010: Sampling bias in BitTorrent measurements. *European Conference on Parallel Processing*, Springer, 484–496.

Appendix A - List Of Symbols

α	Recovery-threshold fraction of baseline performance.
A	Mean absolute deviation from the run median, used for dispersion.
β	Coefficient vector in the Cox proportional hazards model.
β_{public}	Contagion/transmission rate on public peering edges.
β_{mesh}	Contagion/transmission rate on authenticated mesh edges.
$b_{i,t}$	Bytes exchanged with peer i on day/step t.
D_ℓ	Set of dependencies at protocol layer ℓ in the bipartite model.
d	Relative drawdown (depth of degradation) from baseline.
$d(\cdot,\cdot)$	Great-circle distance between a vantage point and peer i.
D	Peak drop below baseline effectiveness in simulator metrics.
$\overline{D}_t$	Traffic-weighted average peer distance at time/day t.
$E(t)$	PoS effectiveness trajectory over time.
G	Gini coefficient (inequality/concentration measure).
$h(t \mid X)$	Hazard function conditioned on covariates X.
$h_0(t)$	Baseline hazard in the Cox model.
H	Herfindahl–Hirschman concentration index, $H = \sum_i s_i^2$.
H	Scoring horizon length in simulator trajectory evaluations.
i, j	Generic entity/sample indices in summations and pairwise terms.
J_t	Jaccard similarity between consecutive observed sets.
K	Carrying capacity (logistic/Bass diffusion models).
k	Top-k rank parameter for concentration and churn summaries.

L	Deficit area (severity $\times$ duration) over a horizon.
$L(p)$	Lorenz curve at population fraction p.
ℓ	Context-dependent: baseline-normalized loss metric, or layer index in D_ℓ.
$\lambda_{i,t}, \lambda_v, \overline{\lambda}_t$	Peer longitude, vantage longitude, and weighted focal longitude.
m	Run-level median used in dispersion metric A.
μ	Mean activity in concentration formulas.
N	Sample size (e.g., number of runs/observations).
$N(t)$	Cumulative adopters/adoptions at time t.
n	Number of entities/observations in a sample or model.
$O(t)$	Offline-validator fraction trajectory over time.
p	Population fraction (Lorenz curve) or innovation-rate parameter (Bass model).
$p90, p95$	90th and 95th percentile quantiles of an outcome distribution.
$\mathcal{P}_t$	Set of peers active/observed at time/day t.
$P(t)$	System performance proxy trajectory over time.
P_{pre}	Pre-shock baseline performance level.
$P_{\min}$	Minimum observed performance during disruption.
q	Imitation-rate parameter in the Bass diffusion model.
$q_{90}(\cdot)$	90th-percentile quantile operator.
r	Logistic growth rate.
r_s	Spearman rank-correlation coefficient.
s_i	Share of activity/load attributed to group/entity i.
S	Snapshot size in out-of-band recovery model.
t	Time index (calendar, event, or simulation-step index).
t_0	Inflection/event anchor time (context-dependent).
T	Session-tenure random variable (lifetime/duration).
T_r	Recovery time (steps to baseline-relative target).

$T_{\mathrm{rec}}(\alpha)$	Time to recover above fraction α of baseline.
ΔA	Paired change in dispersion metric A.
$\Delta \ell$	Paired change in loss relative to baseline.
Δt	Event-time offset from an anchor event.
τ	Tail threshold (e.g., baseline-defined exceedance cutoff).
τ_t	Turnover at time t, defined as $1 - J_t$.
$\mathrm{Top}_k(t)$	Set of top-k entities at time window t.
$\mathrm{Turnover}_t^{(k)}$	Top-k turnover statistic between adjacent windows.
V	Set of protocol participants in the bipartite dependency model.
V_t	Set of entities observed in window/time t.
(v, d)	Edge linking participant v to dependency d in a bipartite graph.
$w_{i,t}$	Normalized traffic weight for peer i at time/day t.
x_i	Activity associated with entity i.
$x_{i,t}, y_{i,t}, z_{i,t}$	Unit-sphere coordinates mapped from peer geolocation.
$\overline{x}_t, \overline{y}_t, \overline{z}_t$	Traffic-weighted mean unit-sphere coordinates.
X	Covariate vector in hazard/survival models.
$\varphi_{i,t}, \varphi_v, \overline{\varphi}_t$	Peer latitude, vantage latitude, and weighted focal latitude.

Appendix B - List Of Acronyms and Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
AS	Autonomous System
ASN	Autonomous System Number
AWS	Amazon Web Services
BGP	Border Gateway Protocol
BM25	Best Matching 25 retrieval model
CCPA	California Consumer Privacy Act
CDN	Content Delivery Network
CGNAT	Carrier-Grade Network Address Translation
CPU	Central Processing Unit
CRM	Customer Relationship Management
CSV	Comma-Separated Values
DDoS	Distributed Denial-of-Service
DAO	Decentralized Autonomous Organization
DHT	Distributed Hash Table
DKIM	DomainKeys Identified Mail
DNS	Domain Name System
DoH	DNS-over-HTTPS
DoS	Denial-of-Service
DVT	Dataset Value Taxonomy

EMD	Earth Mover's Distance
EU	Europe / European Union (context-dependent)
F1	F1-score (harmonic mean of precision and recall)
GB	Gigabyte
GCP	Google Cloud Platform
GDPR	General Data Protection Regulation
GLMM	Generalized Linear Mixed Model
HHI	Herfindahl–Hirschman Index
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IMDb	Internet Movie Database
IP	Internet Protocol address
IPFS	InterPlanetary File System
IPNS	InterPlanetary Name System
ISP	Internet Service Provider
IXP	Internet Exchange Point
KL	Kullback–Leibler (divergence)
L2	Layer 2 (secondary protocol layer)
ML	Machine Learning
MB	Megabyte
NA	North America
NAT	Network Address Translation
P2P	Peer-to-Peer
PCAP	Packet Capture
PCA	Principal Component Analysis
PoS	Proof-of-Stake
RFC	Request for Comments

RIPE	Reseaux IP Europeens (RIPE NCC)
RPC	Remote Procedure Call
RQ	Research Question
RTT	Round-Trip Time
SA	South America
S1–S4	Scenario identifiers used in Chapters 7 and 8
SLHD	Sliced Latin Hypercube Design
SMTP	Simple Mail Transfer Protocol
SNI	Server Name Indication
SOCKS5	Socket Secure 5 proxy protocol
SPF	Sender Policy Framework
SSH	Secure Shell
SYNC	Node resynchronization state in the simulator
TB	Terabyte
TCP	Transmission Control Protocol
TLD	Top-Level Domain
TLS	Transport Layer Security
Tor	The Onion Router
TV	Television
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
USD	United States Dollar
VPN	Virtual Private Network
WHOIS	Who Is protocol / registration lookup service