The Art of Pivoting

Alexandre Dulaunoy

2025-12-18

Contents

List of Figures

1 The Art of Pivoting - Techniques for Intelligence Analysts to Discover New Relationships in a Complex World

This book explores how intelligence and cyber-security analysts can uncover hidden links between threat actor infrastructure and ongoing investigations by pivoting on both classic and unconventional indicators — many of which are often overlooked. The material is grounded in empirical, field-tested strategies used in cyber-security, digital forensics, cyber threat intelligence, and intelligence analysis more broadly.

Our goal is to provide analysts with a practical toolkit of analytical methods, supported by real-world examples, to enhance investigative workflows without locking them into a single mindset, strict model, or overly rigid technical strategy. Instead, the book encourages creative exploration, data-driven reasoning, and the use of diverse data points — from traditional IOCs to subtle metadata traces — as part of a flexible and repeatable analytical process.

The approach presented throughout this book is intentionally built upon open-source tooling, most notably the MISP threat intelligence platform and the AIL Project. By relying on transparent and widely adopted tools, every technique described here can be reproduced, validated, and reused by analysts, researchers, educators, or incident response teams. This ensures that the methodology is not theoretical or proprietary, but openly verifiable, community-driven, and designed to evolve. The book itself follows the same philosophy: it is an open, living document, publicly versioned, and contributions are welcomed directly via Git. Readers are encouraged to experiment, improve, and extend the content, making the entire workflow repeatable, auditable, and collaborative within the wider defensive security community.

1.1 Definition and the So-Claimed "Theory"

Before diving into practical techniques and real-world investigations, it is useful to clarify the language and concepts that form the backbone of pivot-based analysis. In threat intelligence, many terms are borrowed from other disciplines — statistics, social networks, cryptography, digital forensics — and are sometimes used informally or inconsistently. This chapter provides supporting definitions tailored for operational analysts, not academic theorists.

The goal is not to impose a rigid theoretical model, but to build a common vocabulary that helps explain how we move from isolated observations to connected intelligence. Terms

such as *data points*, *correlation*, and *pivoting* are defined in a way that reflects how analysts actually use them in the field. We also introduce the "Six Degrees" perspective to illustrate why small signals can lead to surprising discoveries.

This section therefore acts as a lightweight theory chapter — just enough structure to explain the logic behind the techniques, without restricting the creativity or intuition that make pivoting effective in practice.

1.1.1 Data Points

In this book, we use the term **data points** to describe any discrete element that can support correlation or pivoting during threat intelligence analysis. A data point may be a traditional indicator such as an IP address, domain, or file hash, observables at large but it can also include less conventional digital artifacts like cookie names, QR codes, favicon hashes, HTTP header sequences (HHHash), DOM structure, or Marketing Analytics tracking codes. Treating all of these elements uniformly as data points is useful because it avoids assuming that only strong or traditional indicators are relevant. Even weak or unexpected data points can become valuable when combined through composite correlation. In practice, pivoting is the act of moving from one data point to another through inferred or observed relationships, while correlation is the process of identifying those relationships.

This terminology provides a neutral and flexible way to reason about diverse signals extracted from tooling (such as AIL and MISP) during infrastructure and threat actor analysis.

1.1.2 Correlation

Correlation is the analytical process of identifying relationships between two or more data points. In threat intelligence, correlation helps determine whether separate observations — such as domains, certificates, HTTP headers, file hashes, or behavioral artifacts — may belong to the same actor, campaign, or infrastructure cluster. A single correlation does not always imply attribution, but it provides evidence that two elements share a meaningful connection, such as technical similarity, shared metadata, deployment patterns, or operational reuse. Correlation is often iterative and multi-layered: weak signals can become valuable when combined, and composite correlations frequently reveal links that are not visible through traditional indicators alone.

Correlation can be *automatic*, *manual*, or a combination of both.

1.1.2.1 Automatic Correlation

Automatic correlation is performed by tools and platforms using predefined rules, heuristics, or algorithms. Systems such as MISP automatically correlate data points by comparing attributes across datasets, events, or objects. This may rely on:

- Exact matches (e.g. identical hashes, domains, IPs, certificates fingerprints)
- Similarity or fuzzy matching (e.g. SSDEEP, TLSH, DOM structure, favicon hashes)
- Group-based logic (e.g. shared CIDR ranges, ASNs, hosting providers)
- Behavioral or structural fingerprints (e.g. HTTP header hashes, cookie names)

Automatic correlation can occur at different moments in the lifecycle of data:

- At ingestion time, when new data points are first introduced into a system and immediately compared against existing knowledge.
- Continuously or periodically, when correlation engines are re-run as datasets grow, algorithms improve, or historical context changes.

This continuous nature is important: correlation is not a one-time action. A data point that appeared insignificant at ingestion may become meaningful weeks or months later when additional context emerges.

Automatic correlation excels at scale and speed. It ensures consistency, repeatability, and broad coverage across large volumes of data. However, it operates strictly within the boundaries of what is encoded in rules and algorithms and can therefore generate false positives or noisy correlations, especially when weak or high-entropy data points are involved. This analytical noise can become a burden for analysts, requiring manual validation, contextual filtering, and prioritization to separate meaningful relationships from coincidental ones.

1.1.2.2 Manual and Human-Driven Correlation

Not all correlation can—or should—be automated. **Manual correlation** is performed by analysts who interpret data points using experience, intuition, and contextual understanding that may not be formally captured in systems.

This includes:

- Analytical reasoning based on investigative context
- Pattern recognition across disparate sources
- Hypothesis-driven linking of weak or unconventional signals
- Correlation derived from **HUMINT**, open-source research, linguistic cues, or cultural knowledge
- Insights gained without reliance on external tooling, or by combining outputs from multiple tools mentally rather than mechanically

Manual correlation often explains *why* two data points are related, not just *that* they are related. It is also a key mechanism for improving the overall quality of correlation by validating, refuting, or refining relationships produced by automated systems. By applying contextual understanding, investigative judgment, and external knowledge, analysts can filter out false positives and analytical noise generated by large-scale automatic correlation. In doing so, manual correlation frequently introduces new and more precise pivot paths that automated systems cannot anticipate or model.

1.1.2.3 Correlation as an Iterative Process

Correlation is not binary and not definitive. A single correlation does not imply attribution, ownership, or intent. Instead, it represents **a hypothesis supported by evidence**, whose strength increases as additional, independent correlations accumulate.

In practice, effective pivoting relies on the interplay between:

- Automated correlation to surface candidate relationships at scale
- Human analysis to interpret, contextualize, and validate those relationships

Correlation therefore sits at the core of pivoting: it is both a computational process and a cognitive one. Without correlation, pivoting is blind exploration. With correlation—especially when automated and human-driven approaches reinforce each other—pivoting becomes a structured, explainable, and repeatable analytical method.

> 🧠 If you design a threat intelligence platform or system that supports correlation, it is essential to support both automatic and manual correlation. Equally important is the ability to clearly distinguish between these two types and to allow analysts to trace back how a correlation was established. Being able to identify whether a relationship was generated by an algorithm, a rule, or human analysis helps analysts assess confidence, understand potential bias or noise, and make informed decisions when pivoting. In MISP, this distinction is implemented explicitly: automatic correlation and analyst-driven data relationships are handled as separate concepts. Correlations are generated by the system based on matching or similarity algorithms, while data analyst relationships represent intentional, contextual links created and validated by humans.

1.1.3 Pivoting

Pivoting is the investigative action of moving from one known data point to another in order to expand knowledge about a threat actor, infrastructure, or campaign. After correlation reveals that two elements are linked, pivoting uses that link as a new starting

point, allowing analysts to continue exploring outward in successive steps. For example, an analyst may begin with a domain, pivot to an IP address that hosts it, then pivot again to other domains hosted on the same server, or even to additional services that share a specific favicon hash, certificate, or cookie name. Pivoting is therefore an exploratory process: it transforms isolated observations into a growing graph of related data points and can uncover infrastructure or behavior that would remain hidden if indicators were examined in isolation.

The term *pivoting* can be confusing because it is used in two different contexts. In cyber defence and threat intelligence, pivoting refers to the analyst's process of moving from one data point to another to uncover related infrastructure, identities, or activity. This is entirely separate from the attacker's use of "pivoting," which typically means lateral movement inside a compromised network. This book focuses exclusively on the defender's perspective: pivoting as an investigative method for expanding knowledge, not as an adversarial technique for gaining additional access. However, the techniques used by attackers to pivot inside a compromised network can themselves become valuable data points for defenders. Traces of lateral movement — such as reused credentials, tunneling tools, shared SOCKS proxies, or staging servers — often leave forensic or network artifacts that can be extracted, correlated, and used as data points.

1.1.4 Pivot

A pivot itself is **not** a data point. A data point is a discrete element — such as a domain, certificate, cookie name, or hash — that can be compared, correlated, or stored. A pivot, on the other hand, is the investigative action of moving from one data point to another after discovering a relationship. The pivot is the step in the process, not the object being analyzed.

In practice:

- The **domain** is a data point.
- The **shared TLS certificate** is another data point.
- Moving from the domain to the certificate and then to other domains using the same certificate is the **pivot**.

So, a pivot is the *movement across relationships*, whereas a data point is the *thing being related*. Without data points, pivoting is not possible — but a pivot cannot be seen as a data point itself.

1.1.5 Six Degrees of Separation (Kevin Bacon Concept)

The concept of **Six Degrees of Separation** suggests that any two individuals in the world are connected through a short chain of relationships — typically no more than six steps. Popularized in social network theory and later known through the “Kevin Bacon game,” it demonstrates how large, complex networks can still produce surprisingly short paths between seemingly unrelated actors. Similar concepts exists in academic contexts as the ”Erdos number,” which measures how many co-authorship links separate a researcher from the mathematician Paul Erdos.

In cyber threat intelligence, the same idea applies to infrastructure and data points. A single indicator — such as a domain, cookie name, or TLS certificate — may appear isolated. However, when connected through successive correlations, it may lead to a different server, a shared service provider, a reused attack toolkit, or a specific actor’s operational mistake. Although each step may seem small, multiple pivots can reveal links that are not obvious when indicators are viewed alone.

This is why pivoting matters: it transforms a flat list of IOCs into a graph of relationships. Once correlations accumulate, even distant nodes may become connected within only a few steps, showing that many threat infrastructures are not as independent or hidden as they appear. The Six Degrees concept provides a useful mental model for understanding how hidden relationships emerge in large-scale intelligence data.

1.2 The Practice of Pivoting

1.2.1 Evolution of Pivoting Practices

Early pivoting techniques in threat intelligence were straightforward and deterministic: if two artifacts shared an exact value — the same IP address, file hash, email, or domain — they were assumed to be related. This model worked well when threat actors made simple operational mistakes or reused infrastructure without modification. A perfect match was enough to connect two samples, two campaigns, or two servers.

As adversaries became more fragmented and distributed, perfect matches grew rarer. Infrastructure is now dynamic, rapidly deployed, and often disposable. Servers change IPs, phishing kits are rebranded, domains rotate and are shared among TA, and payloads are customized. In this environment, single indicators can lose value sometime, and analysts must rely on additional signals.

Modern pivoting therefore focuses on *composite correlation* — combining multiple weak or partial data points to reveal a relationship that no single indicator could confirm on its own. A favicon hash may mean little, but paired with a shared Google Analytics ID, a similar

DOM structure, or identical cookie names, it becomes meaningful. Clustering techniques push this even further by grouping infrastructure based on patterns, similarity scoring, or behavioral fingerprints, rather than exact matches.

This chapter explores the full spectrum of pivoting methods, from simple one-to-one links to complex, multi-layered correlations that uncover relationships even when adversaries attempt to hide or fragment their infrastructure.

1.2.2 Exact Match Correlation

The most fundamental form of pivoting is based on **exact match correlation**, where two data points are linked because they share the same value. In MISP, this happens automatically: if two attributes contain the same hash, IP address, domain name, email address, filename, or other indicator, the platform correlates them.

The strength of this method depends entirely on the **type of data point** being compared. Some exact matches are **strong** signals:

- IP address
- Cryptographic file hashes (SHA-256, SHA-1, MD5)
- Unique TLS certificate fingerprints
- PGP key fingerprints

If two events contain the same SHA-256 hash, it is almost certain they refer to the same binary. These cryptographic hash values have an extremely low false-positive rate.

Other exact matches are **weak** signals because the values are generic, widely reused, or trivial to change:

- Filenames such as `invoice.pdf` or `update.exe`
- Common User-Agent strings
- Generic email subjects

These can produce a high number of correlations that are technically correct, but not meaningful. In other words, **exact match correlation is not fragile — it is simply more prone to false positives when the underlying data point is weak**. The advantage is that these false positives are usually easy to spot, because the context around the match (campaign, infrastructure, malware family) often doesn't align.

Exact match correlation is therefore an essential first step: fast, deterministic, and ideal when strong indicators exist. But it cannot stand alone. In some cases, the **number of exact matches** becomes a clue in itself. An indicator that correlates with a very large volume of unrelated events often signals a potential false positive. For example, an IP

address used as a sinkhole may appear in hundreds of malware reports without indicating any shared operator, and multi-homed infrastructure observed through passive DNS may link to a wide set of unrelated hostnames. In these situations, the explosion of matches is less a sign of strong correlation and more a warning that the data point is too generic to support attribution.

1.2.3 Fuzzy Matching Correlation

Once exact matching has exhausted its value, analysts often turn to **fuzzy matching correlation**. Instead of requiring two data points to be identical, fuzzy matching looks for *similarity*. Tools such as **SSDEEP**, **TLSH**, or **sdhash**[1] generate similarity hashes that allow analysts to connect files or payloads that share overlapping content, embedded resources, compiler artifacts, or packer stubs.

Fuzzy correlation is especially useful when adversaries produce many variants of the same malware family, or when a phishing kit is redeployed with small modifications. Two PE files that differ in signatures, strings, or timestamp can still score highly in TLSH or SSDEEP, revealing a relationship that exact match correlation would miss.

However, fuzzy matching must be treated with caution. **Increasing sensitivity increases false positives.** Many Windows executables share the same icons, version resources, or packer signatures, which can inflate similarity scores without implying a common operator or campaign. It is common in malware analysis to see large clusters of binaries that look "similar" but are operationally unrelated. For this reason, fuzzy matching should guide investigation, not conclude it: it generates hypotheses and leads, not attribution on its own.

1.2.4 Group-Based Correlation

A related technique is **group-based correlation**, where data points are clustered based on shared infrastructure or logical grouping rather than identical values. The most common example in MISP is **CIDR correlation**: if multiple malicious domains or IPs fall within the same network range, they may belong to the same operator, hosting provider, or automated deployment.

Group correlations help surface weak signals that would otherwise remain hidden. Even if each IP address appears only once, their presence in a suspicious subnet can indicate

[1] https://github.com/sdhash/sdhash Evaluating Similariy Digests: A Study of TLSH, ssdeep, and sdhash Against Common File Modifications shows the diversity of similary digests/fuzzing hashing and the difficulty to find the perfect one even for a single task such as classifying malware binaries.

shared provisioning, bulletproof hosting, or coordinated activity. Time-based grouping, certificate reuse within a range, or shared ASN ownership can reinforce the signal further.

But as with fuzzy matching, **group correlation can easily become too broad**. Large providers or cloud platforms host thousands of unrelated customers; passive DNS may show hundreds of benign domains sharing the same /24. When correlation returns *everything*, it stops being useful. Analysts can end up trapped in wide clusters that look meaningful but offer little investigative value.

Fuzzy and group-based correlations therefore extend the pivoting process beyond perfect equality. They are most effective when combined with context or layered with other data points — for example, when a CIDR match is reinforced by shared TLS certificates, similar DOM structure, or common malware resources. Just like exact matching, these techniques are valuable tools, but they must be interpreted carefully: the more inclusive the rule, the greater the analyst's responsibility to validate the result rather than assume it.

1.2.5 Analytical Strategies of Pivoting

Pivot Category	Pivot Type	Description / Examples
Data Points (selector, attribute, IoC)	Current Pivoting	Analysis based on live or near-real-time observations
Current Pivoting	Live Infrastructure Mapping	DNS queries, active scanning
Current Pivoting	Communication Behavior	Querying services, active monitoring
Current Pivoting	Operational Patterns	Key materials, similar malicious files
Data Points (selector, attribute, IoC)	Historical Pivoting	Analysis based on past or archived data
Historical Pivoting	Past Infrastructure Reuse	Similar network infrastructure, default setups
Historical Pivoting	Historical Public Records	WHOIS records, TLS key materials or parameters
Historical Pivoting	Historical Records	Leaked credentials, identifier or alias reuse

Pivot Category	Pivot Type	Description / Examples
Data Points (selector, attribute, IoC)	Predictive Pivoting	Analysis aiming to anticipate future activity
Predictive Pivoting	Recurring TTPs	Similar patterns or techniques reused over time
Predictive Pivoting	Predictive Patterns	Domain generation algorithms, reused naming patterns
Predictive Pivoting	Forecasting Tools / Techniques	Tools, techniques, or practices for forecasting activity

Pivoting can also be thought of across different points in time: **current**, **historical**, and **predictive**.

DNS is a good example of how all three can apply.

- **Current pivoting** looks at live DNS records. When an analyst resolves a hostname, the returned IP address represents the state of the infrastructure at the exact moment of analysis.

- **Historical pivoting** examines how that hostname resolved in the past, using sources such as Passive DNS. This can reveal previous hosting providers, older campaigns, sinkholes, or infrastructure that has since been abandoned.

- **Predictive pivoting** appears in more advanced cases. Domain Generation Algorithms (DGAs) can compute future domains based on time or seed values, meaning some malicious domains exist mathematically before they ever appear in DNS. By generating or monitoring these future domains, analysts can pivot forward in time and anticipate infrastructure before it becomes active.

This time-based perspective highlights that pivoting is not limited to a single snapshot. Infrastructure evolves, and useful intelligence emerges when current, historical, and potential future states are analysed together.

1.3 Rethinking Pivoting: From Strong Indicators to Unintended Traces

Pivoting is often described as an investigative art, but the goal is to shape it into a more reproducible and practical discipline, one that any analyst can apply methodically rather

than intuitively. However, the way we think about pivoting is sometimes constrained by rigid models or inherited practices that no longer reflect how modern adversaries operate. Frameworks like the Pyramid of Pain are useful, but they deserve to be re-examined: how hard is it really for threat actors to alter an indicator, and which indicators do they ignore entirely?

The reality is that attackers do not always understand the traces they leave behind, nor do they reliably anticipate the intelligence value of those traces. A well-known example is the *Anna-Senpai* case behind the Mirai botnet. The malware itself contained no strong identifying indicators that linked directly to the author, and much of its infrastructure was transient and intentionally disposable. Yet a small, seemingly irrelevant artifact—the reuse of the *Anna-Senpai* alias across unrelated online posts—became a pivot point that investigators followed across forums, leaked credentials, and historical activity. That weak signal, when correlated with other data points, eventually contributed to unmasking the individuals involved.

This illustrates a core principle of modern pivoting: even minor or disconnected artifacts can become powerful intelligence when correlated across time, platforms, and context. The evolution of pivoting lies not in abandoning classical models, but in refining how we interpret traces—especially the ones that attackers assume are meaningless.

1.3.1 The Danger of Single-Model Reasoning

Models like the Pyramid of Pain are valuable for teaching, but they can become restrictive if treated as absolute truth. The assumption that "high-value indicators are hard to change" encourages analysts to prioritize TTPs, certificates, cryptographic hashes, or infrastructure identifiers, while disregarding weaker signals. In reality, modern adversaries routinely automate or outsource the rotation of high-value indicators: cloud instances are redeployed in seconds, TLS certificates are regenerated for free, and containerized command-and-control servers can be destroyed and rebuilt faster than defenders can react.

At the same time, low-entropy or unconventional indicators often become the most revealing. Reused cookie names, favicons, Google Analytics identifiers, forum nicknames, vanity onion prefixes, or repeated API paths are frequently left untouched. These elements do not appear at the top of any model, yet they provide persistent and highly effective pivot points—because attackers do not consider them "indicators" worth hiding.

In short, **strict adherence to a single model introduces blind spots**. The real world rewards flexible analysis, composite correlation, and attention to the traces adversaries ignore or disregard.

1.3.2 Re-evaluating Our Indicator Collection and Pivoting Practices

Within the AIL project, we collect data from a wide range of sources: social networks, Tor hidden services, criminal forums, paste sites, and web infrastructure commonly used by threat actors. To make sense of this diversity, we built a dynamic correlation engine that allows new object types to be introduced easily. Instead of restricting analysis to a fixed list of traditional indicators, the system can correlate emerging artifacts such as QR codes, cookie names, HTTP header hashes, or vanity onion prefixes.

This required a deliberate change in mindset. Rather than treating established indicators as the primary source of truth, we began to focus on outliers and overlooked data points—elements attackers rarely consider significant, and therefore rarely bother to obfuscate. In the process, some of our older assumptions had to be challenged or discarded. What was once considered noise often turned into strong investigative leads when combined with other weak signals. By shifting from a rigid interpretation of indicators to a more exploratory, data-driven approach, we uncovered relationships that would have remained invisible using traditional practices.

This shift also changed how analysts think: instead of asking which indicators matter most, we now ask which indicators adversaries fail to hide. This perspective becomes essential as we move into composite correlation and clustering techniques, where weak signals converge into strong intelligence.

1.3.3 Looking at "Broken" Indicators and Still Using Them

Some indicators are known to be imperfect, yet remain surprisingly effective in real investigations. MurmurHash3, for example, is still widely used for favicon correlation. A single MMH3 hash can quickly reveal Tor hidden services that are also exposed on the clear web, allowing analysts to pivot across seemingly unrelated infrastructure with minimal effort.

If MurmurHash3 is known to be flawed, why continue using it? Because even with weaknesses and potential collisions, it works—and more importantly, attackers rarely consider favicons to be meaningful intelligence artifacts. Many copy the same web assets across panels, icons on webshell, custom css, phishing kits, and darknet storefronts without modification, leaving behind reliable pivot points.

There is an additional twist: when threat actors deliberately attempt to manipulate or collide favicon hashes, those collisions themselves become useful signals. Correlating clusters of colliding favicons can reveal common tooling, shared deployment scripts, or copied infrastructure. In other words, a "broken" indicator can still produce strong intelligence, either because adversaries ignore it, or because their attempts to evade it create new

patterns worth pivoting on. Stopping the calculation of such hashes would simply remove a cheap and surprisingly effective investigative tool.

1.3.3.1 Combining Weak and Strong Data Points for Pivoting

The continued use of seemingly weak data points, like MurmurHash3 (MMH3) for favicons or even simple MD5[2] file hashes, is justified when they are understood as correlation point rather than definitive attribution artifacts. The key is to integrate them strategically with strong data points within a structured pivoting workflow.

Weak data points are often the initial starting point for an investigation (e.g., A single alone MD5 hash from a security alert). They excel at quickly casting a wide net to find clusters of potentially related infrastructure or activity. Their low interest in threat intelligence practice and tendency for adversaries to ignore them make them ideal for the initial discovery phase.

> 🧠 If you design a Threat Intelligence database or storage model, be sure to include a schema that accommodates a diverse array of data points, including those categorized as weak data points (e.g., MMH3, MD5, cookie names). A robust model must treat all data points as correlatable objects, ensuring that initial investigations can effectively use high-volume, low-cost signals for breadth pivoting before confirming connections with high-fidelity, strong data points.

1.4 The Value of Uncommon Data Points

Uncommon data points, which you define as data points that fall outside the traditional set (IPs, domains, file hashes), are extremely interesting and valuable in the Cyber Threat Intelligence (CTI) process because they exploit **two key operational weaknesses of threat actors: neglect and reuse**.

1.4.1 Mindset Shift: From Perfection to Pragmatism

The traditional CTI process often focuses on strong data points (e.g., high-entropy unique hashes) that promise low false positives. However, this perspective is sometimes clouded by rigid models or legacy practices.

- Challenging Assumptions: We must reconsider our reliance on models like the Pyramid of Pain and critically assess how difficult it truly is for adversaries to alter high-

[2] Fast Collision Attack on MD5 presents an improved attack algorithm to find two-block collisions of the hash function MD5.

value indicators. Do threat actors always realize which traces they leave behind, and can they accurately gauge the intelligence value of what they expose?
- The Power of Outliers: A successful pivot often comes from focusing on outliers and overlooked data points. These artifacts are often simple, easy to extract, and offer a low-cost way to establish correlation.
- Imperfect is Not Useless: Even outdated or colliding indicators can still provide valuable correlations. The flaw of the indicator is irrelevant if the adversary neglects to change it.

1.4.1.1 Why Uncommon Data Points Are Effective

Uncommon data points are effective precisely because threat actors do not consider them meaningful intelligence artifacts. This neglect turns them into silent, high-utility fingerprints.

- Neglected Fingerprints: When threat actors copy the same web assets across phishing kits, darknet storefronts, or operational panels, they rarely think to modify innocuous files like favicons. A simple MurmurHash3 on a favicon enables quick hash-based pivoting to uncover related infrastructure, such as Tor hidden services exposed on the clear web.
- Operational Reuse: Simplicity and operational efficiency often lead to the reuse of metadata or infrastructure-specific artifacts.
  - HTTP Header Hashing (HHHash)[3]: HTTP (version 1) response headers can act as subtle fingerprints for linking threat infrastructure.
  - Cookie Names: Custom or reused cookie names can serve as low-noise data points for linking attacker-controlled web infrastructure.
  - Embedded Analytics: Despite the assumption that threat actors avoid including labels that could link their infrastructure , services like Google Analytics tracking codes are often reused across multiple malicious sites, revealing unexpected and meaningful correlations.
- Visual and Structural Clues: Non-traditional image and structure analysis introduces entirely new pivot points:
  - Structural Hashing: The `dom-hash` algorithm clusters Tor Hidden Services based on simple HTML structure, yielding excellent results in structural similarity detection.
  - Extracted Data: QR codes are increasingly seen on social networks and ransomware negotiation pages, providing embedded data points. Barcodes have proven to be valuable correlation points in social media and data leaks.

[3] HTTP Headers Hashing (HHHash) or improving correlation of crawled content which facilitates the hashing of similar returned HTTP headers.

  - Vision Models: Vision-language models can automatically generate descriptions of collected images (e.g., from social networks or screenshots), allowing investigators to correlate on terms, keywords, or objects identified in the images.

1.4.1.2 The Power of Composite Correlation The real strength of uncommon data points lies in composite correlation, their ability to connect elements when strong indicators fail.

- A single weak data point may be noisy, but when combined with others, it creates a unique, high-confidence cluster. For example, a Google Analytics ID alone may not prove attribution, but when combined with a specific favicon hash or dom-hash, they help cluster infrastructure belonging to the same threat actor.
- This approach is key to pivoting, which is the investigative action of moving from one known data point to another to expand knowledge about a threat. It transforms isolated observations into a growing graph of related data points.

By expanding our collection to include these unconventional data points, we can achieve deeper insights and better threat discovery.

1.5 An Inventory of Uncommon Data Points

1.5.1 DOM Hash (`dom-hash`): Structural Fingerprinting of HTML Content

HTML content is an abundant but often underutilized source of correlation in threat intelligence. While much prior work has focused on visual similarity, content hashing, or full document classification, many operational use cases do not require that level of complexity. During discussions with CERT-PL and empirical testing against large-scale web capture datasets (notably LookyLoo), we observed that a remarkably simple strategy yields excellent results for infrastructure pivoting: hashing the structure of the HTML document rather than its content.

This observation led to the development of the **dom-hash** algorithm.

1.5.1.1 Core Idea The central idea behind dom-hash is to treat the **Document Object Model (DOM) structure** of an HTML page as a fingerprint. Instead of hashing text, scripts, or resources—which are frequently modified or localized—dom-hash focuses exclusively on the **sequence of HTML tag names** present in the document.

By ignoring content and attributes and retaining only structural information, dom-hash captures layout and template reuse while remaining resilient to superficial changes.

In practice, many phishing kits, scam pages, ransomware leak sites, and underground storefronts reuse the same HTML templates across multiple deployments. Even when branding, language, or embedded resources change, the underlying DOM structure often remains untouched.

1.5.1.2 Algorithm Description

At a high level, the dom-hash algorithm follows these steps:

1. Parse the HTML document into a DOM tree.
2. Extract the ordered list of all HTML tag names.
3. Concatenate the tag names into a single string, preserving order.
4. Compute a cryptographic hash over that string.
5. Truncate the result to a fixed length for storage and correlation.

A reference implementation in Python is shown below using the BeautifulSoup library:

```
def _compute_dom_hash(html_content):
    soup = BeautifulSoup(html_content, "lxml")
    to_hash = "|".join(t.name for t in soup.findAll()).encode()
    return sha256(to_hash).hexdigest()[:32]
```

This simplicity is intentional. The algorithm is fast, deterministic, and easy to reproduce across tooling and datasets.

1.5.1.3 Why `dom-hash` Works in Practice

The strength of dom-hash lies not in theoretical uniqueness, but in **operational neglect**. Threat actors frequently copy and redeploy HTML templates without altering their structural composition. While they may change text, images, JavaScript, or styling, the DOM skeleton remains identical across campaigns or infrastructures.

Because dom-hash ignores volatile elements, it is resistant to:

- Language changes
- Branding modifications
- Variable content such as dates or victim identifiers
- Minor cosmetic edits

At the same time, it is sensitive enough to distinguish between fundamentally different templates.

1.5.1.4 Use Cases for Pivoting

dom-hash is particularly effective for:

- Linking phishing pages derived from the same kit as implemented in LookyLoo
- Correlating ransomware leak sites with similar layouts
- Identifying cloned underground marketplaces or scam portals
- Pivoting between Tor hidden services and clear-web infrastructure using shared templates

As with other uncommon data points, dom-hash is rarely sufficient on its own for attribution. Its value emerges when used as a **pivot primitive** and combined with other indicators such as favicon hashes, TLS certificates, cookie names, or hosting patterns.

1.5.1.5 Limitations and Noise Considerations

dom-hash is intentionally coarse and therefore has limitations:

- Very common frameworks or boilerplate templates may produce high-cardinality correlations.
- Minor structural changes (e.g. adding or removing wrapper elements) will alter the hash.
- Dynamic, heavily JavaScript-driven applications may yield unstable DOMs depending on rendering context.

For these reasons, dom-hash should be interpreted like other structural or weak indicators: as a way to surface candidate relationships that require analyst validation rather than as definitive proof of common ownership.

1.5.1.6 Positioning dom-hash Among Uncommon Data Points

dom-hash exemplifies a broader category of uncommon data points: indicators that are easy to compute, inexpensive to store, and often ignored by adversaries. Its effectiveness does not come from cryptographic strength or semantic understanding, but from exploiting operational reuse and oversight.

Used correctly, dom-hash enables analysts to pivot across infrastructure that would otherwise appear unrelated, reinforcing the central theme of this book: meaningful intelligence often emerges not from perfect indicators, but from simple signals applied creatively and in combination.

1.5.2 Favicons: Visual Artifacts as Infrastructure Fingerprints

Favicons are small icons associated with websites, typically served as part of a site's static assets. They are designed purely for user experience and branding, not for security or

identification. Precisely because of this, favicons have become a surprisingly effective and durable data point for infrastructure correlation in threat intelligence.

In many malicious ecosystems—phishing kits, scam portals, ransomware leak sites, underground marketplaces—favicons are copied verbatim across deployments. While operators frequently rotate domains, IP addresses, certificates, and even page content, favicons are often reused without modification, making them a reliable pivot point across otherwise fragmented infrastructure.

1.5.2.1 Why Favicons Matter

Favicons occupy a unique position among web artifacts:

- They are rarely customized per deployment.
- They are usually embedded early in development and forgotten.
- They persist across clear-web and Tor or I2P deployments.
- They are commonly reused across campaigns and time.

Threat actors tend to focus their evasion efforts on elements they perceive as indicators—domains, IPs, certificates, or payloads—while treating favicons as benign design assets. This operational blind spot turns favicons into durable fingerprints.

1.5.2.2 Favicon Hashing as a Correlation Primitive

The most common way to operationalize favicons is by hashing their binary content and using that hash as a correlation key. In practice, non-cryptographic hashes such as MurmurHash3 (MMH3) are frequently used because they are fast, widely implemented, and sufficient for clustering purposes.

While MMH3 is not collision-resistant and is unsuitable for security guarantees, this is largely irrelevant in the context of pivoting. The goal is not uniqueness, but **repeatability**: the same favicon reused across infrastructures will reliably produce the same hash.

This makes favicon hashes particularly effective for:

- Linking phishing sites derived from the same kit
- Correlating Tor hidden services with clear-web infrastructure
- Identifying cloned scam or fraud portals
- Discovering reused administrative panels or staging servers

1.5.2.3 Composite Correlation with Favicons

Favicons are rarely used in isolation. Their true value emerges when combined with other data points:

- A shared favicon hash and identical DOM structure
- A favicon hash combined with similar HTTP headers

- A reused favicon across domains sharing analytics identifiers
- A favicon hash linking infrastructure across different hosting providers

In composite correlation, favicons often act as the initial pivot that reveals a broader cluster, which can then be refined using stronger or more contextual indicators.

1.5.2.4 Noise, Collisions, and Interpretation Like all uncommon data points, favicon correlation is not immune to noise.

- Popular frameworks or widely used templates may share default favicons.
- Collision-prone hashing algorithms can group unrelated icons.
- High-cardinality favicon hashes may indicate generic assets rather than operator reuse.

However, these limitations are manageable. High-volume correlations are themselves a signal that a favicon is likely generic and should be deprioritized. Conversely, low-volume or thematically consistent reuse significantly increases investigative value.

Importantly, even deliberate attempts to evade favicon-based correlation—such as modifying or randomizing icons—can introduce new patterns worth tracking, especially when changes are inconsistent across deployments.

1.5.2.5 Beyond Favicons: Related Visual and Static Assets Favicons illustrate a broader class of **static visual and resource-based data points** that are often overlooked:

- Logos and embedded images
- CSS files, layout assets and properties
- JavaScript bundles reused across deployments
- Icon fonts and UI components

These artifacts share similar properties: they are operationally neglected, inexpensive to reuse, and rarely treated as indicators by adversaries. When extracted, hashed, and correlated, they provide additional pivot opportunities that complement favicons and DOM-based techniques.

1.5.2.6 Positioning Favicons Among Uncommon Data Points Favicons are a textbook example of how low-entropy, non-obvious artifacts can outperform traditional indicators in real investigations. Their strength lies not in cryptographic assurance, but in adversarial oversight and operational reuse.

When used as part of a broader pivoting strategy—combined with structural, temporal, and contextual data—favicons remain one of the most effective and accessible uncommon

data points available to analysts. They reinforce a recurring lesson of pivot-based analysis: intelligence value often comes from what attackers ignore, not from what they try to protect.

1.5.3 Graphically Encoded Information: QR Codes, Barcodes, and Visual Payloads

Graphically encoded information—such as QR codes, barcodes, and similar visual symbologies—is increasingly present in threat actor ecosystems. Once limited to logistics and retail, these encodings now appear regularly across social networks, Tor hidden services, phishing pages, scam portals, and even ransomware negotiation sites. Their growing adoption makes them an important and often overlooked source of correlation.

Unlike traditional indicators, these data points are not expressed as text or network artifacts. They are embedded visually within images, screenshots, banners, or scanned documents. As a result, they are frequently ignored by automated collection pipelines and underestimated by analysts—precisely the conditions that make them valuable for pivoting.

1.5.3.1 Why Graphically Encoded Data Matters

QR codes and barcodes serve a practical function for threat actors:

- Redirecting victims to payment pages or wallets
- Linking to messaging platforms or support channels
- Encoding identifiers, order numbers, or tracking references
- Facilitating cash-out, delivery, or coordination processes

Because these codes are meant to be scanned, not read, threat actors often reuse them verbatim across posts, platforms, and campaigns. This reuse creates durable correlation points that survive changes in text, language, or surrounding context.

1.5.3.2 QR Codes as Correlation Primitives

QR codes are especially prevalent in modern threat activity. They are observed in:

- Social media scams and fraud campaigns
- Tor hidden services and underground forums
- Ransomware leak sites and negotiation portals
- Phishing pages designed for mobile users

From an analytical perspective, a QR code is not just an image—it is a container for structured data. Once decoded, it often reveals:

- URLs or deep links
- Cryptocurrency addresses
- Messaging identifiers
- Embedded metadata or parameters

When the same decoded payload appears across multiple images or platforms, it becomes a strong pivot point. Even when QR codes are visually modified (resized, recolored, embedded in different layouts), their encoded content often remains identical.

1.5.3.3 Barcode Extraction Beyond Retail Contexts

Barcodes—such as Code 128, Code 39, Code 93, and similar formats—have also proven valuable in threat intelligence contexts. Initially implemented in response to law enforcement requests, barcode extraction quickly demonstrated broader utility.

Barcodes appear in:

- Large-scale data leaks and document dumps
- Screenshots of financial transactions or receipts
- Social media posts related to cash-out operations
- Internal references shared between threat actors

In many cases, barcodes encode identifiers that are reused across communications or platforms. When extracted and normalized, these identifiers can link seemingly unrelated datasets or interactions.

1.5.3.4 Operational Extraction and Correlation

The primary challenge with graphically encoded data points is not correlation, but **extraction**. Unlike text-based indicators, they require image processing and decoding before they become usable.

The AIL project implemented a multi-layered detection strategy for QR code extraction to handle cases where multiple QR codes are present, partially visible, or embedded within a single image, illustrating how challenging efficient and reliable extraction can be.

Once extracted, however, they behave like any other data point:

- They can be stored as normalized values
- They can be correlated across events and datasets
- They can serve as pivot points for further investigation

In practice, QR codes and barcodes often complement other uncommon indicators. A single QR code may connect a phishing page to a Telegram channel, a ransomware negotiation page, and a social media account—three environments that rarely share traditional indicators.

1.5.3.5 Noise, Reuse, and Analyst Validation As with all uncommon data points, context matters.

- Widely reused payment QR codes may indicate shared services rather than shared operators.
- Some barcodes encode generic or transactional data with limited intelligence value.
- High-volume reuse may signal infrastructural or service-level commonality rather than campaign-level linkage.

Analyst validation is therefore essential. The investigative value of a graphically encoded data point increases significantly when combined with temporal patterns, platform overlap, or supporting indicators such as DOM structure, favicon reuse, or hosting characteristics.

1.5.3.6 Beyond QR Codes and Barcodes QR codes and barcodes are representative of a broader class of **visual payloads**:

- Embedded links in images
- Steganographic markers
- Watermarks or visual identifiers
- Machine-readable symbols not intended for human interpretation

As vision and image-processing capabilities improve, these artifacts become increasingly accessible for large-scale analysis. Their intelligence value does not come from complexity, but from neglect: they exist outside the traditional mental model of "indicators," and are therefore rarely protected or rotated by adversaries.

1.5.3.7 Positioning Graphically Encoded Data Points in Pivoting Graphically encoded information reinforces a recurring theme in pivot-based analysis: meaningful correlations often originate from places analysts are not trained to look. QR codes and barcodes bridge digital and physical workflows, automate interaction, and silently carry identifiers across platforms.

When treated as first-class data points—extracted, normalized, and correlated—they enable pivots that bypass many of the defensive measures designed to evade traditional indicator-based analysis. Used in combination with other uncommon data points, they significantly expand the analyst's investigative surface without increasing reliance on brittle or easily rotated indicators.

1.5.4 HTTP Header Structure Hashing (HHHash): Fingerprinting Server Behavior

HTTP response headers are one of the most consistently overlooked sources of correlation in infrastructure analysis. While analysts often focus on domains, IP addresses, certificates, or web content, the structure and composition of HTTP headers—particularly in HTTP version 1—can act as subtle yet stable fingerprints of server behavior.

This observation led to the development of **HTTP Header Hashing (HHHash)**[4]: a lightweight technique that captures and hashes the *structure* of HTTP response headers rather than their dynamic values. The approach is intentionally simple and designed for pivoting, clustering, and exploratory analysis rather than precise identification or attribution.

> 🧠 When searching for new types of data points, focusing exclusively on values can distract from equally valuable structural characteristics. In many cases, analyzing the structure of a data point—rather than its dynamic or volatile values—yields more stable and actionable correlation opportunities, especially when those values are difficult to collect, store, or correlate reliably.

1.5.4.1 Core Idea The core idea behind HHHash is that many servers expose a characteristic pattern in how they construct HTTP response headers:

- Which headers are present
- The order in which they appear
- The casing and formatting conventions
- The combination of default and custom headers

These characteristics often remain stable across deployments because they are tied to:

- Specific web server software or versions
- Reverse proxies and load balancers
- Framework defaults
- Custom middleware or deployment templates

By hashing the normalized structure of HTTP/1 response headers, HHHash produces a compact representation that can be used as a correlation key.

1.5.4.2 Why HTTP/1 Headers Are Still Relevant HHHash explicitly targets **HTTP version 1**, where headers are transmitted in clear text and retain ordering and formatting information.

[4] HTTP Headers Hashing (HHHash) or improving correlation of crawled content which facilitates the hashing of similar returned HTTP headers.

Later protocol versions (HTTP/2 and HTTP/3) change header semantics significantly through compression and binary framing, making this approach unsuitable or far less informative.

Despite the age of HTTP/1, it remains widely deployed across:

- Phishing and scam infrastructure
- Command-and-control panels
- Underground services
- Legacy or misconfigured servers
- Transitional infrastructure exposed briefly during deployment

As a result, HTTP/1 header fingerprints continue to offer real-world investigative value.

1.5.4.3 HHHash as a Correlation Primitive

HHHash is particularly effective when used to:

- Cluster infrastructure deployed using the same server templates
- Link short-lived or rotating hosts operated by the same actor
- Identify reused reverse-proxy or backend configurations
- Pivot between domains or IPs that otherwise share no obvious indicators

Because HHHash abstracts away volatile values (timestamps, cookies, dynamic identifiers), it focuses on structural similarity rather than content equality.

In practice, HHHash often reveals relationships between servers that appear unrelated at the network or domain level but share the same operational setup.

1.5.4.4 Composite Correlation with HTTP Headers

Like other uncommon data points, HHHash is most effective when combined with additional signals:

- Shared HHHash and identical DOM structure
- HHHash combined with favicon reuse
- HHHash reinforced by hosting patterns or deployment timing
- HHHash aligned with similar TLS configurations

In composite correlation, HTTP header fingerprints often act as a *secondary* confirmation signal, strengthening hypotheses formed through other pivots.

1.5.4.5 Noise and Interpretation Challenges

HTTP header correlation must be interpreted carefully.

- Common web servers and default configurations can produce high-cardinality hashes.

- Content delivery networks and managed hosting providers may normalize headers across many unrelated customers.
- Security appliances can inject or reorder headers, obscuring origin behavior.

High-volume HHHash correlations often indicate generic infrastructure and should be deprioritized. Conversely, low-volume or thematically consistent reuse—especially across short timeframes or unusual hosting environments—can be highly informative.

As with other weak or structural indicators, HHHash should guide investigation, not conclude it.

1.5.4.6 Positioning HHHash Among Uncommon Data Points

HHHash belongs to a broader class of **behavioral and structural fingerprints**: indicators that do not identify *what* a server hosts, but *how* it behaves. Its value lies in exploiting operational reuse and default configurations that adversaries rarely consider intelligence-relevant.

Because HTTP headers sit at the intersection of application logic, middleware, and infrastructure, they often encode subtle but persistent characteristics of deployment practices. When extracted and correlated systematically, these characteristics enable pivots that bypass many of the evasion strategies aimed at traditional indicators.

Used alongside dom-hash, favicon hashing, and other uncommon data points, HHHash reinforces a central theme of pivot-based analysis: meaningful relationships are often hidden in plain sight, embedded in technical details that attackers assume are insignificant.

1.5.5 Cookie Names: Application-Level Fingerprints Beyond Volatile Values

HTTP cookies are traditionally viewed as transient state containers whose values change frequently and are therefore difficult to use for correlation. However, when the focus shifts from **cookie values** to **cookie names**, cookies become a surprisingly stable and low-noise data point for pivoting and infrastructure analysis.

Custom or reused cookie names often reflect application logic, framework defaults, or developer choices. Unlike values, which are typically randomized, time-bound, or user-specific, cookie names are rarely rotated and are commonly reused across deployments. This makes them well suited for correlation.

1.5.5.1 Why Cookie Names Matter

From an operational perspective, cookie names have several properties that make them valuable:

- They are usually hard-coded in application logic.

- They persist across redeployments and campaigns.
- They are rarely considered intelligence-relevant by threat actors.
- They often survive changes in hosting, domains, or certificates.

Threat actors may rotate infrastructure aggressively, but they rarely refactor application-level identifiers such as cookie names unless forced to do so. This operational inertia turns cookie names into durable fingerprints.

1.5.5.2 Cookie Names as Correlation Primitives

Cookie name correlation works by extracting the names of cookies set by a server (e.g. via `Set-Cookie` headers) and treating those names as data points. Correlating on names rather than values avoids most of the noise associated with session identifiers, tokens, or user-specific data. Cookie names can be extracted when crawling websites using standard browser capabilities (as performed in AIL using the Lacus framework), and they correspond to the same cookie names that a user would normally see during a regular browsing session.

Cookie names are particularly effective for:

- Linking phishing sites using the same kit or backend
- Correlating administrative panels or dashboards or similar software stack
- Identifying reused web shells or lightweight control interfaces
- Pivoting between staging, testing, and production deployments

Because cookie names operate at the application layer, they often reveal reuse that is invisible at the network or infrastructure level.

1.5.5.3 Cookie Values: Fragile but Occasionally Useful

While cookie values are generally too volatile for direct correlation, they should not be dismissed entirely. In some cases, values may encode:

- Static identifiers or flags
- Poorly randomized tokens
- Environment or campaign markers
- Developer mistakes or debugging artifacts

However, correlating cookie values safely usually requires additional normalization, decoding, or contextual validation. Without such processing, direct value-based correlation is prone to false positives and analytical noise. For this reason, cookie names should be treated as the primary indicator, with values used selectively and cautiously.

1.5.5.4 Composite Correlation with Cookie Names Cookie names rarely act alone. Their true value emerges in combination with other uncommon data points:

- Cookie name reuse combined with identical DOM structure
- Cookie names reinforcing favicon or HHHash correlations
- Shared cookie naming patterns across Tor and clear-web services
- Cookie names aligned with similar response headers or paths

In composite correlation, cookie names often function as a reinforcing signal—strengthening hypotheses generated through other pivots.

1.5.5.5 Noise, Defaults, and Interpretation Not all cookie names are meaningful in the context of an investigation, although some may still be useful for network reconnaissance or for identifying similar software and frameworks in use.

- Common framework defaults (e.g. `PHPSESSID`, `JSESSIONID`) generate high-volume, low-value correlations.
- Third-party services and analytics may introduce generic cookie names.
- Security appliances or proxies may inject their own cookies.

High-cardinality cookie name correlations usually indicate generic or framework-level behavior and should be deprioritized. In contrast, **unusual, custom, or semantically specific cookie names**—especially when observed across multiple deployments—are strong candidates for investigation.

1.5.5.6 Positioning Cookie Names Among Uncommon Data Points Cookie names exemplify a broader class of **application-layer structural indicators**: data points that reflect developer decisions rather than runtime state. Their effectiveness stems from being overlooked, stable, and inexpensive to extract.

When used thoughtfully, cookie name correlation provides a low-effort, high-return pivot that complements other structural techniques such as DOM hashing and HTTP header fingerprinting. They reinforce a central lesson of pivot-based analysis: durable intelligence often resides not in high-entropy values, but in the quiet, persistent details that attackers rarely bother to change.

1.5.6 Filenames: Weak Indicators with Contextual Intelligence Value

In threat intelligence, filenames are often dismissed as unreliable or noisy indicators. They are easy to change, frequently generic, and prone to false positives. In many technical

contexts—such as malware delivery or file hosting—filenames alone rarely provide strong evidence of shared ownership or coordinated activity.

However, dismissing filenames entirely overlooks an important reality: in certain environments, filenames carry **contextual and semantic meaning** that can reveal key aspects of a threat actor's activity. When treated as contextual data points rather than definitive indicators, filenames can become valuable tools for pivoting and investigative enrichment.

1.5.6.1 Why Filenames Are Usually Considered Weak

Filenames are generally weak indicators because:

- Common names (`invoice.pdf`, `update.exe`, `document.zip`) are reused widely
- They are trivial to modify or randomize
- Automated tooling often generates generic or meaningless names
- High-cardinality reuse produces substantial analytical noise

In isolation, filename correlation rarely supports attribution or infrastructure linkage. High-volume matches typically indicate coincidence rather than coordination.

1.5.6.2 Where Filenames Become Interesting

The investigative value of filenames increases significantly in **human-facing contexts**, where filenames are chosen deliberately rather than generated automatically. These include:

- Social networks and messaging platforms
- Data leak dumps and archive releases
- Shared folders, paste sites, or cloud storage links
- Screenshots, photos, or documents shared publicly

In these environments, filenames may encode:

- Language preferences or regional spelling
- Operational themes or campaign narratives
- Internal project names or tooling references
- Timestamps, versioning schemes, or sequencing
- Humor, aliases, or stylistic quirks unique to an actor

Such details often persist across platforms and time, making filenames useful for correlation when combined with other signals.

1.5.6.3 Filenames as Pivot and Enrichment Data Points

Rather than serving as primary indicators, filenames are best used as **pivot aids and enrichment signals**. They can help analysts:

- Group related documents within a leak
- Identify recurring naming conventions across releases
- Pivot between social media posts and leaked material
- Correlate content shared by the same actor under different identities

In practice, filenames often function as a *narrative glue*, connecting technical artifacts to human behavior and communication patterns.

1.5.6.4 Composite Correlation with Filenames Filenames gain strength when used in composite correlation:

- Similar filenames appearing alongside shared metadata
- Filename patterns aligned with specific platforms or communities
- Filenames reinforcing language or cultural indicators
- Filenames recurring in conjunction with specific file formats or content types

When combined with timestamps, platform context, or author identifiers, filenames can help reconstruct activity timelines or reveal operational consistency.

1.5.6.5 Noise, Ambiguity, and Analyst Judgment Filenames are inherently ambiguous and must be interpreted cautiously.

- Generic filenames should be deprioritized quickly
- Automated renaming by platforms can obscure original intent
- Translations or reuploads may alter filenames

Analyst judgment is essential. The question is not whether a filename is unique, but whether it is **informative in context**. In many cases, the filename's value lies not in correlation volume, but in the insight it provides into how and why content was shared.

1.5.6.6 Positioning Filenames Among Uncommon Data Points Filenames occupy a unique position among uncommon data points: they sit at the intersection of technical artifacts and human expression. While weak from a purely technical standpoint, they can reveal intent, habits, and operational patterns that stronger indicators cannot.

Used thoughtfully, filenames complement structural and behavioral indicators by adding semantic and contextual depth. They reinforce a key principle of pivot-based analysis: even fragile data points can yield meaningful intelligence when interpreted within the right context and combined with other signals.

1.6 Validating Correlation: Signal or Noise?

Not every correlation is equally useful. While correlation is essential for pivoting, it can also generate large volumes of relationships that are technically correct but analytically unhelpful. One of the core skills of an intelligence analyst is therefore the ability to assess whether a correlation represents a meaningful investigative lead or merely background noise.

Validation is not about disproving correlation, but about evaluating its **investigative value**. A correlation can be valid from a data perspective and still be irrelevant, misleading, or too generic to support further analysis. This section explores common correlation patterns that frequently produce noise, along with counter-examples where the same correlations become highly valuable.

1.6.1 Correlation Cardinality as a First Heuristic

A simple but powerful heuristic is **cardinality**: how many other data points are linked through the same correlation.

- **Low cardinality** correlations often indicate specificity and intentional reuse.
- **High cardinality** correlations may indicate shared infrastructure, generic services, or artifacts that are not actor-controlled.

Cardinality alone is not sufficient to accept or reject a correlation, but it provides an immediate signal for prioritization and deeper validation.

1.6.2 IP Address to Hostname Correlation

Correlating IP addresses to hostnames is a foundational technique in infrastructure analysis.

- **Low correlation count** (e.g. one IP hosting one or two related domains) can indicate dedicated infrastructure, staging servers, or early deployment phases of an operation.
- **High correlation count** (e.g. dozens or hundreds of hostnames resolving to the same IP) often indicates shared hosting, CDN nodes, sinkholes, or multi-tenant cloud services.

However, high-volume IP-to-hostname correlations are not inherently useless.

> 🔀 **Counter-example:** If a high-cardinality IP is associated with a narrow thematic set of domains (similar naming patterns, shared TLS certificates, common web content), it

> may represent a shared malicious hosting provider or a coordinated campaign using pooled infrastructure. In such cases, the *pattern* of the domains matters more than the number.

1.6.3 Domain to IP (Reverse) Correlation

The reverse correlation, pivoting from a domain to all IPs it has resolved to historically. can also generate noise.

- A domain resolving to many IPs over time is often explained by CDNs, load balancing, or defensive hosting practices.
- This makes individual IP correlations weak in isolation.

> 🔀 **Counter-example:** If historical resolution shows brief appearances in small, obscure network ranges before moving to mainstream hosting, those short-lived IPs may reveal early testing infrastructure or transitional deployment stages worth investigating.

1.6.4 File Hash Correlation (Especially MD5)

Exact hash correlation is usually considered strong, but volume matters.

- A hash that appears across a very large number of unrelated events may indicate a common library, installer stub, or benign component embedded in many binaries. Services like **hashlookup.io**[5] allow analysts to quickly determine whether a file hash corresponds to a known, widely distributed software component observed across multiple ecosystems. This helps avoid over-interpreting correlations that are technically correct but operationally irrelevant.
- MD5 hashes in particular may collide or be reused unintentionally.

> 🔀 **Counter-example:** If a hash appears frequently *but only within a constrained ecosystem* (same malware family, same delivery vector, same campaign timeframe), the repetition strengthens rather than weakens the correlation.

[5] **hashlookup.io** is an open service that aggregates cryptographic hashes observed across a wide range of public software distributions, package repositories, and datasets. It enables analysts to quickly identify hashes that are commonly associated with benign files or shared components, helping to reduce false positives when validating large-scale hash correlations.

1.6.5 TLS Certificate Correlation

TLS certificate reuse is often a high-confidence indicator.

- A certificate correlated across a very large number of domains may simply be a default certificate generated by an automated hosting panel or cloud provider.
- Wildcard or auto-issued certificates can dramatically inflate correlation volume.

> 🔀 **Counter-example:** A certificate reused across low-volume, short-lived domains, especially when paired with other shared artifacts (HTTP headers, favicons, cookie names), can indicate operator-level reuse and poor operational hygiene.

1.6.6 Filename and Path Correlation

Filenames and URL paths are classic weak indicators.

- Common names like `login.php`, `update.exe`, or `/admin/` correlate extremely widely and usually produce noise.
- High correlation volume here is almost always uninteresting.

> 🔀 **Counter-example:** Highly specific or unusual paths, parameter names, or directory structures—especially when reused across multiple infrastructures—can become strong pivots. Uncommon spelling, language-specific terms, or hard-coded developer paths often reveal shared tooling.

1.6.7 ASN and Hosting Provider Correlation

Correlating on ASN or hosting provider can be misleading.

- Large providers host millions of unrelated customers.
- High correlation volume usually reflects provider popularity, not coordination.

> 🔀 **Counter-example:** Repeated use of small, niche providers, short-lived ASNs, or providers known for abuse tolerance can significantly increase the value of this correlation, especially when combined with time-based analysis.

1.6.8 Time as a Validation Dimension

Time is often underused in correlation validation.

- High-volume correlations spread over years often indicate background noise.
- Correlations clustered tightly in time may indicate coordinated deployment or campaign activity.

A correlation that is weak spatially but strong temporally can still be highly valuable.

1.6.9 Analyst Validation: Accept, Refine, or Reject

For each correlation, an analyst should consider three outcomes:

- **Accept**: the correlation is meaningful and supports further pivoting.
- **Refine**: the correlation is too broad but becomes useful when constrained (by time, context, or additional data points).
- **Reject**: the correlation is technically correct but analytically irrelevant.

Rejecting a correlation is not failure, it is an essential part of maintaining analytical clarity and avoiding cognitive overload.

Ultimately, effective pivoting depends not on the quantity of correlations, but on the analyst's ability to evaluate their relevance. Correlation produces possibilities; validation turns them into intelligence.

1.7 Origin of the Book

This book began life as a presentation at the 2025 FIRST Cyber Threat Intelligence Conference held in Berlin (April 21-23, 2025). From the plenary stage, the theme "The Art of Pivoting" resonated with analysts, threat-intelligence practitioners, and incident-response teams alike: how to discover more from adversaries using existing information. Inspired by the lively exchange of ideas and the collaborative spirit of the conference, the authors decided to expand the talk into a full-length guide.

While the presentation's narrative was time-limited, the ambition of this book is broader. It preserves the pragmatic, analyst-oriented tone of the original session yet adds depth, case studies, workflows, and open-source tooling (such as MISP and the AIL Project) to support repeatable investigation. Above all, it remains an open, living document—rooted in the same community-driven ethos that brought the Berlin event together.

1.8 License