If you cite this paper, please use the TNSM reference: R. Hiesgen, M. Nawrocki, T. C. Schmidt, and M. Wählisch. 2024. The Log4j Incident: A Comprehensive Measurement Study of a Critical Vulnerability. *In IEEE Transactions on Network and Service Management (TNSM).* IEEE, Piscataway, NJ, USA, 14 pages. https://doi.org/10.1109/TNSM.2024.3440188

The Log4j Incident: A Comprehensive Measurement Study of a Critical Vulnerability

Raphael Hiesgen, Marcin Nawrocki, Thomas C. Schmidt, and Matthias Wählisch

***Abstract*—On December 10, 2021, Log4Shell was disclosed to the public and was quickly recognized as a most severe vulnerability. It exploits a bug in the wide-spread Log4j library that allows for critical remote-code-execution (RCE). Any service that uses this library and exposes an interface to the Internet is potentially vulnerable.**

In this paper, we report about a measurement study starting with the day of disclosure. We follow the rush of scanners during the first two months after the disclosure and observe the development of the Log4Shell scans in the subsequent year. Based on traffic data collected at several vantage points we analyze the payloads sent by researchers and attackers. We find that the initial rush of scanners ebbed quickly, but continued in waves throughout 2022. Benign scanners showed interest only in the first days after the disclosure, whereas malicious scanners continue to target the vulnerability. During both periods, a single entity appears responsible for the majority of the malicious activities.

***Index Terms*—Log4Shell, RCE, Scanning, Security, Network Telescope**

I. Introduction

Our digitizing world increasingly depends on distributed software systems and thereby faces a continuously growing threat landscape. New vulnerabilities in the core systems, such as Rowhammer (2015) [1], and Spectre (2018) [2]–[4], easily escalate to a global scale. Faulty protocol feature updates as with Heartbleed (2014) [5], [6], or threatening deployments of new protocols, such as with QUIC (2022) [7] severely threaten the Web ecosystem. Over the past decades, the Internet community has developed several measures to protect its infrastructure [8], [9] but threats from vulnerable software remain.

A most severe, recent vulnerability in this line is Log4Shell (2021). It enables remote code execution through vulnerable applications by injecting a prepared string into the omnipresent Log4j library. As an integral part of Java applications, Log4jShell can be exploited in a variety of application-dependent ways [10].

For hours, days, or weeks after the disclosure of such a vulnerability a race starts between multiple parties. Attackers want to exploit systems before they get patched. Increasingly professionalized cybercriminals are quickly aware of new exploits that become available for anyone willing to pay [11].

R. Hiesgen and T. C. Schmidt are with the Department Informatik at HAW Hamburg, Germany

M. Nawrocki is with NETSCOUT

M. Wählisch is with the Institute of Systems Architecture in the Faculty of Computer Science at TU Dresden, Germany, and with Barkhausen Institut, Germany

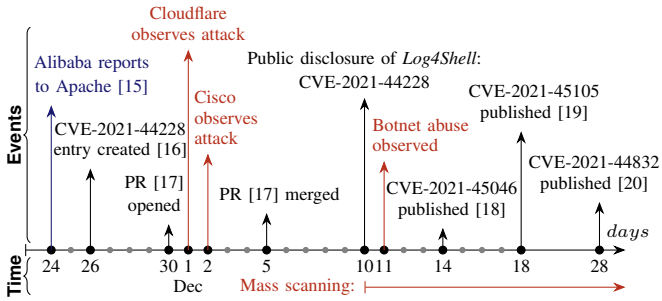


Fig. 1: The unfolding of the Log4Shell vulnerability from reporting to the consequences in 2021.

On the opposite side, operators and hosters are remarkably incautious in implementing standard security measures carefully and consistently [12], [13].

In case of an emergent threat, operators, hosters, and application service providers need to quickly update their systems before they get compromised. Security firms monitor malicious activities and researchers aim to understand details and asses the scale of the vulnerability. Releasing a patch and protective measures alongside a public disclosure is a necessary step to enable admins to secure their systems, but no guarantee exists for a timely implementation. In an initial study [14], we monitored this race during a two-month period starting from the day of its disclosure.

In the present work, we extend our previous research with an additional year-long view on the development of scans that hunt for the Log4Shell vulnerability presented in Section V. We collect and analyze data about attack traffic, its intrusive approaches, and its underlying infrastructure using reactive network telescopes at four vantage points on two continents. Our analyses focus on the behavior and the ecosystem of the malicious actors.

After recapitulating the events close to the Log4Shell disclosure in Section II and introducing our data set in Section III, we continue in two parts. First, we trace the scanners during the disclosure period in Section IV. We find that scanning started on the day of the disclosure and spiked about a week later with a focus on HTTP related ports. In this context, we present collected payloads and identify the URLs central to this exploit. Second, in Section V we examine how scanning continued through 2022 and reveal a large cluster of addresses with shared non-volatile infrastructure. Finally, Section VI discusses and concludes our findings.

II. Background and Related Work

The Log4Shell vulnerability (CVE-2021-44228 [16]) in the popular Log4j library was publicly disclosed on Dec 10, 2021 by Apache alongside a fix in Log4j library version 2.15.0. The bug allows for remote code execution (RCE) by injecting prepared strings into the logging library. NIST [21] classified this threat as a critical vulnerability with the highest severity rating. An impact assessment by Google [22] estimates that 4% (or 17,000 packages) on the Maven Central were affected, either directly or via dependencies. This is twice the impact of an average packet (mean 2%, median 0.1%), which highlights the popularity of Log4j.

From a high-level perspective, the exploit works by injecting a formatted message into the logging component, which then interprets and executes the message. Specifically, the Log4j library supports format messages that are evaluated by the library and can be used to add additional information to log messages, such as the Java version. One of the services that can be used for runtime evaluations is JNDI [23], the Java Naming and Directory Interface, which in turn can query a variety of lookup services. Log4Shell focuses on the services LDAP and RMI for injecting code and infecting the local machine.

A. A Brief History of the Log4Shell Incident

Figure 1 shows the timeline of observations around the Log4Shell incident. The vulnerability was originally reported to Apache on Nov 24, 2021, by the Alibaba Cloud Security Team [15]. The Chinese-based company quickly faced consequences of reporting the vulnerability directly to Apache [24], [25] instead of contacting national authorities first. A CVE record was created on Nov 26 [16] but not published until public disclosure on Dec 10. In the meantime, a pull request (PR) to address the vulnerability was opened on Nov 30 [17] and merged five days later. Cloudflare notes that they observed the first exploit one day after the PR on Dec 1st [26]. Cisco observed an exploitation attempt a day later on Dec 2nd, as published in their Talos Blog [27]. Both companies report that widespread scanning started on Dec 10.

Within a day of the public release, the Mirai and Muhstik botnets, crypto miners, and other malware were observed to use the exploit for propagation [28]–[30]. Microsoft further reports that nation-state attackers experiment with the exploit and integrate it into their activities [31].

The first fix was insufficient and further exploits followed (CVE-2021-45046 [18], CVE-2021-45105 [19], CVE-2021-44832 [20])—none of them as critical, though, as the initial. These vulnerabilities are discussed in more detail by Everson et al. [32] alongside mitigation strategies.

B. The Log4Shell Attack: How it Works

The Log4Shell exploit builds upon a JNDI injection vulnerability that was presented at BlackHat in 2016 [33]. JNDI enables queries of lookup services such as LDAP, the RMI Registry, or the DNS and can load Java objects returned by a service at runtime. Since the query argument is a URL, the lookup can be performed on local or remote services. Via this functionality, an attacker who controls the query can thus load arbitrary code from a location under his control—and this is where Log4j comes in.

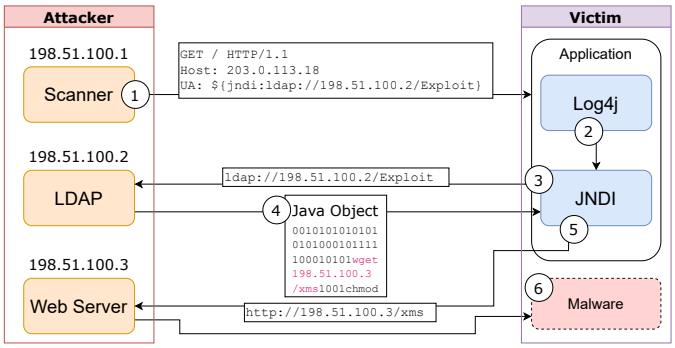


Fig. 2: The Log4Shell exploit tricks the application into loading and executing code from a remote server. It is initially instrumented by an executable log string (here "`UA:`"). Steps marked by circled numbers are explained in Sec. II-B.

Log4j comprises capabilities to interpret strings for enriching logged messages with additional information. Examples are lookups of the Java version or the hostname. These interpreted strings are escaped by wrapping them: `${prefix:query}`. In addition to harmless operations, Log4j accepts a prefix that triggers JNDI to perform the lookup, in which case the query includes its own scheme to signify the lookup services used for the query. This extends the known JNDI vulnerability by opening an attack vector via logged messages. Applications that log web requests, usernames, or generally user-controlled input are easy targets as a result—provided they fail to sanitize their inputs.

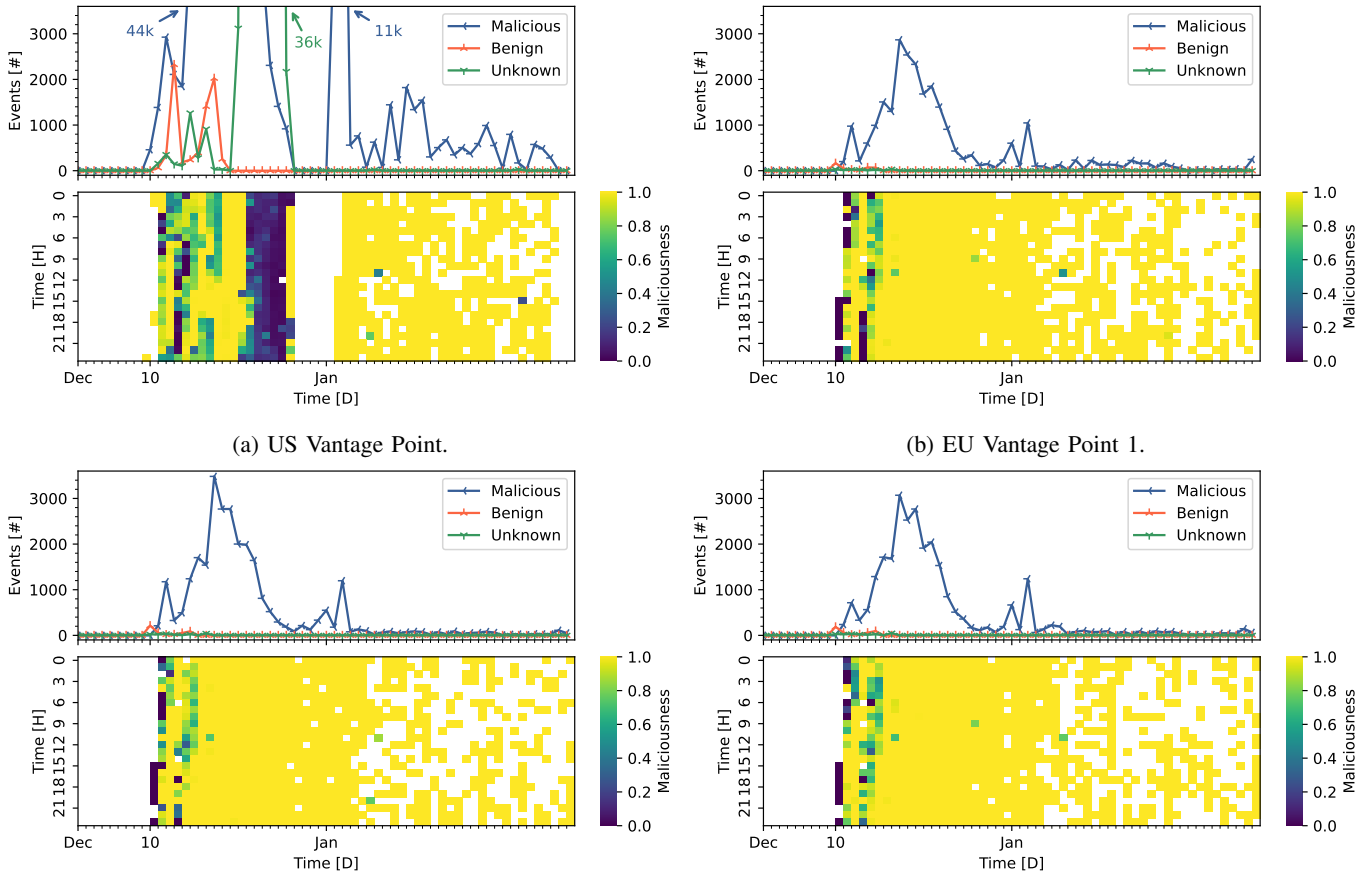
Figure 2 shows a possible exploit scenario initiated by an attacker. The attacker starts with a scan that sends HTTP requests to web servers ①. Here, the exploit string is placed in the user agent field. An information that operators might log to understand what browsers and operating systems they should support to provide good user experience. In our example, the exploit targets an LDAP lookup via JNDI on an LDAP server controlled by the attacker.

The victim application logs the input string using Log4j ②. Log4j in turn finds the escaped string and uses JNDI to perform the LDAP lookup from the remote address ③. The vulnerable Log4j implementation downloads the Java object prepared by the attacker ④ and loads it locally. This Java object contains a way to run shell commands on the local machine, which download the actual malware from a remote server ⑤ and execute it locally ⑥.

Following these steps, an injected payload leads to the execution of arbitrary code prepared by the attacker. The interpretation of strings by Log4j can further be used to obfuscate the payloads [31]. Instead of including the string "ldap" or an address, individual characters are escaped with an operation that leaves them unchanged.

III. Observing Log4j Scans: Methods and Data Set

Multiple groups of people hunt for exploitable services: attackers attempting to exploit vulnerable services, researchers



(a) US Vantage Point.

(b) EU Vantage Point 1.

(c) EU Vantage Point 2.

(d) EU Vantage Point 3.

Fig. 3: The intensity and maliciousness of scanners targeting the Log4Shell vulnerability during Dec'21 and Jan'22.

who want to examine and analyze, and the security industry, which wants to discover and close vulnerable services before they can be exploited. All of them scan the IP space. Scanning behavior, though, does not only depend on the originators but also on the instant of time. In case of a newly arriving vulnerability, the time directly after the release of the exploit is the most critical observation period, since more and more services are getting patched or taken offline over time.

As for the Log4j vulnerability, methods for monitoring and exploiting may appear similar. Scanners that want to identify vulnerable services without exploiting them can simply use LDAP servers which do not perform a valid lookup but only log accessing addresses instead. This makes it especially hard to attribute malice based on scans alone.

We observe scan attempts targeting TCP on four /24 IPv4 prefixes. Our vantage points neither host services nor are they part of a larger active network. We deploy Spoki [34], a reactive telescope that interacts with incoming packets in real-time to establish TCP connections and collect payloads for a few seconds before closing connections. Based on the C++ Actor Framework [35] Spoki is highly scalable and can handle millions of addresses in parallel. It exploits otherwise unused address space just like silent telescopes [36] but due to its responsiveness we expect its sensitivity to be more similar to honeypot platforms [37], [38].

Spoki is deployed at all four vantage points, which include one in the US (part of the UCSD Network Telescope [39]) and three in the EU. Two of the vantage points in the EU host neighboring networks (EU VP 2 & VP 3), which are separately announced. We separate the data set by /24 prefixes in our analysis to accommodate for topological differences. We lost data at the US vantage point from Dec 28 '21 to Jan 1 '22, and at the EU vantage points from June to mid-July in 2022.

Payloads targeting the Log4Shell vulnerability contain the escaped JNDI format string. As such, they are detectable by parsing the payloads if a few obfuscation techniques are taken into account. We use an open-source detector [40] to filter the collected payloads. In our analysis, we use the term event for a payload including a JNDI string. Hence, the number of payloads we collected equals the number of events. Note that we cannot observe all application-specific payloads as they may be encoded in an application-specific format. Nevertheless, binary protocols that contain a JNDI string in ASCII remain detectable.

We use MaxMind [41] for geolocating IPs. While geolocation is not highly accurate, mapping to countries is still reasonable [42]. PeeringDB [43] classifies the type of observed networks, for which we summarize *Not Disclosed* and networks without an entry under *Unknown*. GreyNoise [44] provides additional threat intelligence information.

IV. The Log4Shell Disclosure: A Race to the Vulnerable

The time around the Log4Shell disclosure was a turbulent time. Engineers rushed to understand the vulnerability and analyze whether their own infrastructure was threatened. This section focuses on the behavior of scanners during this period in three parts: a timeline of the events, an analysis of their payloads, and a closer look at the exploits observed in the wild.

A. Scanners

First, we examine the scanners using Log4Shell payloads. These are the sources in step ① of Figure 2.

1) Overview: We collect all events, *i.e.,* payloads that include a JNDI exploit string (see Section III), and classify the sources using GreyNoise [44] into the categories *malicious*, *benign*, and *unknown*. We cannot easily sort *unknown* sources into the other two categories because Log4j payloads sent during step ① do not compromise the system. Still, two payloads that strongly correlate with malicious behavior were used to further classify sources as malicious (*cf.* Section IV-C).

Figure 3 displays the scan activity over the months Dec'21 and Jan'22. The upper graphs show the time series per source type and day while the heat maps below visualize malicious intensity, calculated from the share of malicious events among all events per hour. The event count in the US graph grows to roughly 44k during the peak of the *malicious* events while the *unknown* events in the subsequent peak hit roughly 36k.

The first scans start on the evening of Dec 9 (UTC+0) at the US vantage point. The EU only sees the first packets nearly a day later (3PM, Dec 10) than the US (11PM, Dec 9). Moreover, the scans in the EU start with scans from BinaryEdge (https://www.binaryedge.io), a benign threat-intelligence provider—here from DigitalOcean hosts based in the US. At the US vantage point, the first scans originate from one source classified as malicious, in a UK AS.

Mid-December is the period with the highest activity. Noticeably, the only benign events are registered in the first weeks after the disclosure. Towards January the events per day drop to around 100 in the EU, and 2000 or less in the US. The interest in scanning for the vulnerability spiked about a week after the public disclosure. During this time all vantage points observe substantial malicious activity. The US vantage point observes a large share of unknown events as well.

The heat maps reveal a period of mixed activity before malicious sources take over–except for the unknown sources in the US around mid-December. While the benign actors are fast in the EU, they quickly lose interest. Here, researchers and threat intelligence providers have room to grow: start fast and perform continuous measurements.

Scanners are more interested in the US region during this time. From the observed 2023 sources, 1516 were exclusively seen in the US, and 123 only in the three EU telescopes. The EU vantage points provide a relatively similar view on the sources. Each EU VP observed at most 27 addresses exclusively.

2) Event Peaks: Shortly after the disclosure all vantage points observe benign scanners. At the US VP we observe two spikes up to 2000 packets on Dec 13 & 18. 90% of these packets are from a single AS, Alpha Strike (https://www.alphastrike.io). We observe scans focusing on HTTP-related ports such as 8080, 8081, as well as port 8983.

A very high spike in malicious events was observed in the US on Dec 19 & 20. During these two days, the vantage point records more than 40k packets per day. 80% of them originate from a single IP endpoint in Russia. This is one order of magnitude more than observed from any other source during this time. All these events target port 8080. Two days later, a second spike occurs, this time from unknown sources. During these 5 days the US VP observed more than 30k packets per day. As a result, the maliciousness decreases, which is visually striking in the heat map as a dark stripe from Dec 22 to 26.

Once again, we observe a large share of events (76%) originating from a single, unclassified IP address of the same Russian hoster as the previous, malicious address that caused the spike days earlier. In contrast to the previous scan these events are split between TCP ports 8080 (18%) and 5480 (75%). A closer inspection of these two events reveals that the first source scans more aggressively but for a shorter time. The hourly packet rate is up to twice as high. In contrast, the unknown source sends at a lower rate for a longer time, adding up to more than twice as many packets in total. Although their payloads have similarities–they exfiltrate the domain, computer name, OS, and java version–they do not show direct intent to perform RCE via Java objects. We also find no specific overlap in the payload strings and do not see sufficient evidence to classify the source as malicious.

A malicious spike in events can be observed at all three EU VPs on Dec 18. These spikes reach about 3k events per day. When grouping by AS, two ASes take more than 10% share: an American hoster and Chinese ISP. Both scan a range of HTTP-related ports in addition to ports 5480 and 7547. These ASes show up at the US VP with a slightly higher event count. Their share is comparatively small due to the high traffic volume from the aforementioned two Russian addresses.

Smaller event spikes from malicious sources are seen at the European VPs on Jan 1 & 3. These originate from a single hoster IP address. These probes focus on HTTP-related ports. The much larger spike in the US is caused by the same address. The long tail of smaller spikes in the US originates from a variety of ASes that focus on HTTP-related ports. More than 60% of the traffic originates from 4 hosters.

We expect a rise in Log4Shell events as scanners start looking for vulnerable hosts. Our observations reveal aggressive scanning behavior in few hosts. These trade scanning speed for increased noise. Since the vulnerability was not yet well understood, scanners explored different payloads or ports, thus creating even more noise. Given the wide coverage of Log4Shell they likely attempt to find as many victims as possible before machines get patched, are taken offline, or networks start blocking their addresses.

3) Who is Scanning?: Geolocation (see Section III) based on the source addresses shows that scans originate to a large extend from three countries. China and the US make up more

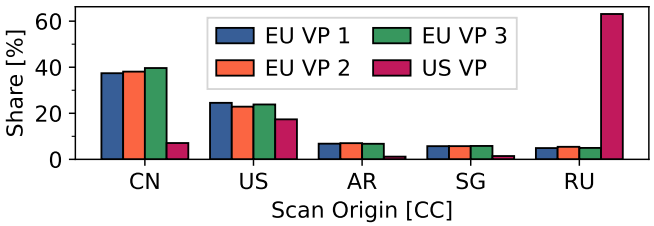


Fig. 4: The top 5 countries with at least 5% event share. Russian scanners focus on the US.

TABLE I: Overview over the network types from PeeringDB. Hosting providers originate the largest identifiable share.

	EU			US
	VP 1	VP2	VP3	VP 1
Hosting	34%	34%	35%	79%
Transit/Access	22%	23%	22%	4%
Education	7%	7%	7%	1%
Enterprise	<1%			<1%
Other	<1%	<1%	<1%	<1%
Unknown	36%	35%	36%	15%

than 60% of the events at each vantage point in the EU. In line with the traffic spikes caused by Russian sources in the US, Russia originates more than 60% of the events observed in the US and around 5% at the EU vantage points. Figure 4 summarizes the shares of the top five contributing countries. Table I shows the distributions of types for networks that originate scans (*cf.* Section III).

At all vantage points the largest share falls into the category *Content*, which includes hosting services. Scanners (temporarily) rent VMs here to perform scans and host related infrastructure. Three hosting ASes stand out: Two from the US and one from Russia, which predominantly scans the US VP.

The second-largest share goes to transit and access networks. While all network types might host infected machines, it is more likely that machines in these networks are compromised and part of a botnet [45], [46].

Education and business might actively seek to learn about the new vulnerability. The small share of education networks in the US is likely a lack of data in PeeringDB. Although it could further hint that US institutions prefer VMs in the cloud for measurements and infrastructure. 15% to 40% of networks could not be labeled.

4) What are Scanners Targeting?: The majority of scans aim for HTTP-related ports, such as 80, 8080, 8000. One of the highly active Russian scanners also focuses on 5480 (unofficially used for VMware VAMI). The three most popular ports account for more than 50%, likewise the top ten ports account for more than 85% of the events at each VP. Overall, we observe between 36 and 48 different ports as targets at each VP. While such focused scans are typical for botnets [45] they simply show an expectation of the attacker where to find vulnerabilities.

A port that stands out among the top ten in the EU is 7547. This port is associated with the TR-069 vulnerability related to home routers. While this port is frequently scanned (about 4%), Java is generally not the best fit for home routers and thus Log4j is not likely to be used in such an environment. Presumably, scanners are just testing the port in passing.

B. Payloads of the Scanners

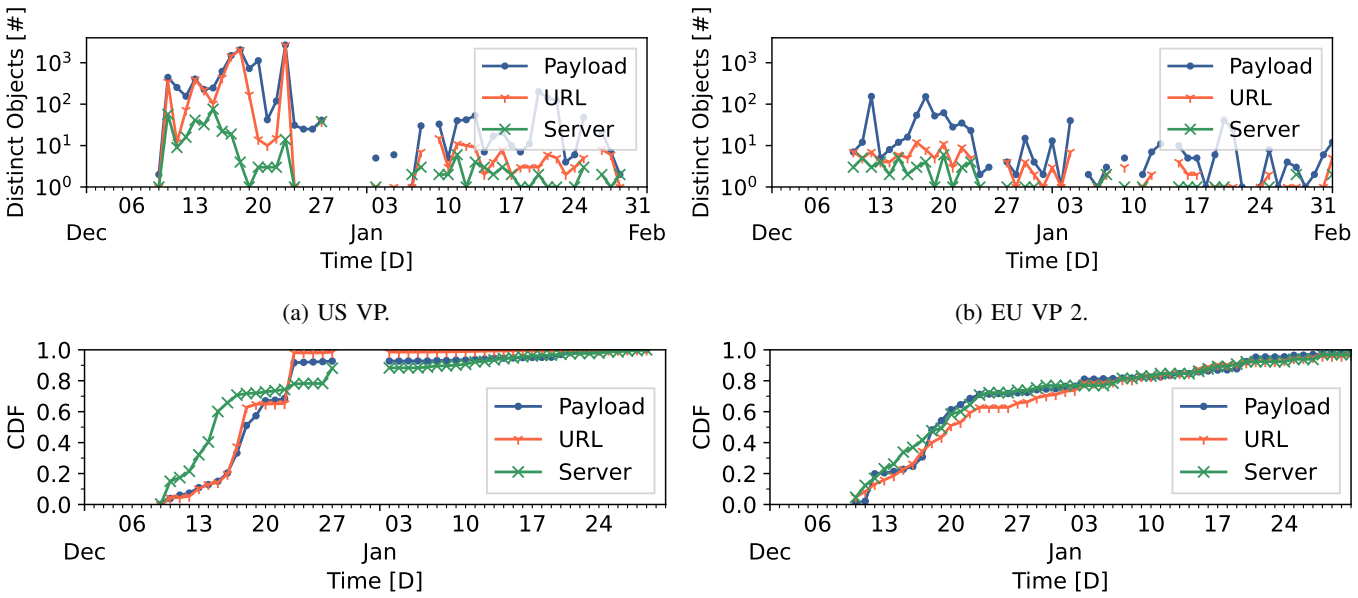
We now analyze in detail the collected payloads, *i.e.,* the data scanners send in step ①, to learn about the impact, how payloads change over time, which protocols scanners use, and the JNDI URL placement.

1) Initial Development: Our previous analysis revealed how scans reached a quick peak before they rapidly decline and settle in a low volume. We cannot tell from the event counts alone whether scanners keep their setups and continue to run unaltered, *i.e.,* reuse payloads, or whether they purposefully change the exploitation methods. Analyzing the variation in payloads over time can help to answer this question.

Figure 5 depicts the evolution of distinct payloads, URLs, and hosting servers during the disclosure time. Subfigures 5a and 5b show the distinct counts per day. In contrast, Subfigures 5c and 5d show the CDF of distinct objects over the complete two months. The payloads are the data we receive directly during the TCP handshake, URLs are the escaped JNDI URLs in the payloads, and servers are solely the addresses in those URLs, *i.e.,* the host and port information of LDAP or RMI server controlled by the attacker. Because payloads often include the address of the victim, *e.g.,* in the host parameter of the HTTP header, we replace the IP addresses from our subnets with a static string. This way, payloads that only differ by their destination will coincide.

The event spikes in Figures 5a and 5b correlate with the spikes about a week after the disclosure, see Figure 3. Note that the graph has a logarithmic y-axis. At both vantage points the distinct payloads count is much higher than the server count. Scanners use different payloads for the same injection string—likely to test the behavior of the scanned service as it might differ by application which data is logged. Similarly, the number of distinct URLs is higher than the number of distinct server addresses. In those cases, attackers re-use the same server, but with different paths. In practice this means that individual servers are used to distribute multiple types of downloads or malware. On some days the number of payloads equals the URL count. Here, scanners encoded additional data in the URL, such as a HEX-encoded JSON payload that includes the destination and other information.

We now analyze the evolution of the attack ecosystem over the complete time frame, *i.e.,* the new payloads, URLs, and servers. Figures 5c and 5d plot the corresponding CDFs. Noticeably, the behavior of distinct payloads and infrastructure largely differs between the US and the EU vantage points: In the US graph payloads and URLs have more extreme steps, one from Dec 17 to 20 and one from Dec 22 to Dec 23, 2021. These jumps lead up to the high spikes in malicious and later in unknown events. In contrast, the server line rises much earlier. The rise in payloads and URLs does coincide. This indicates varying attacks in the US backed by the same infrastructure—supposedly, attackers adapted their payloads and URLs to explore the attack surface.



(a) US VP.

(b) EU VP 2.

(c) US VP.

(d) EU VP 2.

Fig. 5: The relationship between payloads, URLs, and servers over time. (a) and (b) count the distinct objects per day. (c) and (d) aggregate distinct objects over the two-month period in a CDF. EU VPs are represented by EU VP 2 (Dec'21 to Jan'22).

At the EU VP, all three measures occur in line. There are two steeper increases in payloads shortly after mass scanning started on the Dec 12 and again on Dec 18. Still, we do not see a divergence as in the US. Payloads, URLs, and servers behave similarly, rising on similar days. Scanners vary their payloads. Most frequently we observe classic horizontal scans and scanners sending different payloads to a single endpoint before moving on.

2) Exploit String Placement: While focusing on HTTP related ports we observe that scan payloads either carry GET (between 91% and 98%) or PUT requests (remaining). Exploit strings in payloads become visible at various locations, an observation shared by [30]. This reflects attackers who still search for the best attack method, as well as attackers that try to alter their payloads continuously to evade detection. Table II summarizes the most popular placements in HTTP headers.

Regularly logged locations such as the `User-Agent` (UA) and `Authentication` (Auth) header fields are popular targets. The UA, which can be customized for specific purposes [47], identifies the client. Websites depend on it to deliver matching content. As a result, it is often logged to keep statistics on users. Authentication information is similarly important for debugging and access control. Since HTTP header manipulation is a known issue [48], logging any header fields should be done with care.

The lower shares in the US stem from the high variety in header configurations and fields sent by the Russian scanners. While we saw around 45 different header fields in the EU, the US VP observed 157 distinct fields. During January the field count shrank to 13, favoring `User-Agent` and `X-Api-Version`.

The two Russian scanners follow different scanning styles. The one tagged as malicious appears to partly randomize the headers. All payloads are GET requests that include the user agent `Go-http-client`, a `Host` field, the "randomized" field with the JNDI string, and end with `Connection: close`. Exception are payloads that contain the JNDI string as the user agent, in which case only the `Host` field follows. Not only do they change their fields but choose from several simple obfuscated URLs.

In contrast, the other Russian scanner rotates payloads systematically. It sends the same payload to several addresses before changing a single characteristic, such as the header field that contains the JNDI string or the obfuscation method. It looks like a nested loop that rotates the obfuscation in the inner loop and the header field in the outer one. The user agent is only included when it contains the JNDI string.

The obfuscation approaches we observe are built on the substitution used for the vulnerability itself. There are a few easy obfuscations that hide the strings "jndi", "ldap" or other parts of the URL, hiding keywords that can otherwise be found by simple pattern matching, thus making the payloads harder to detect. As an example, the string `${lower:j}` will substitute to "j". As the obfuscation is very easy to achieve and the general JNDI exploit is not new, obfuscated payloads

TABLE II: The most common header locations to store the JNDI URL. (UA: *User-Agent*, Auth: *Authorization*, X-Api-Ver: *X-Api-Version*)

	EU						US	
	VP 1		VP2		VP3		VP 1	
1.	UA	23%	UA	22%	Auth.	23%	UA	11%
2.	Auth.	20%	Auth.	21%	UA	22%	Path	9%
3.	Path	13%	Path	14%	Path	14%	Cookie	6%
4.	Cookie	10%	Cookie	11%	Cookie	11%	Auth.	6%
5.	X-Api-Ver.	10%	X-Api-Ver.	8%	X-Api-Ver.	9%	Referer	6%

quickly appeared in our data. At the EU vantage points, the first obfuscated payloads showed up shortly after midnight on Dec 11. In the US it took a day longer.

C. Examining the JNDI/LDAP Exploitation

We now investigate the final steps ③ to ⑤ of the attack, namely the malware requests triggered via JNDI. We also acquire and inspect the malware distributed via Log4Shell.

1) Analyzing the URLs: The URLs used to query via JNDI have four parts: a scheme, a host, a port, and a path.

Schemes. Tricking the victim into downloading Java objects via JNDI is central to the exploit, see step ③. JNDI supports a range of services, but only a few are used for Log4Shell.

Figure 6 shows the event count for each scheme we observed (note the logarithmic y-axis). The EU vantage points see LDAP almost exclusively. RMI occurs a handful of times at both vantage points. Aside from the events on Jan 16, which originate from the same ASes, there is no correlation between the VPs. As a third scheme, EU VP 3 observed a single HTTP request. The website behind the address gives security advice on how to patch Log4j (Feb'22). The name in the reverse DNS record of the scanning node matches BinaryEdge. In the US we observe two spikes of DNS schemes in January. 95% of these originate from the Alpha Strike AS, matching the benign spikes in Figure 1.

Aside from LDAP, attacks using RMI received attention in the media [49]. We only observed three (US) requests with the URL mentioned in the article. These attacks may have been focused elsewhere and did not hit our VPs in bulk.

Hosts. PeeringDB lacks information about nearly half of the hosting networks we observe. We notice that corresponding servers are mostly located in two ASes, both from Estonia (one provider shows up with two ASes, one of which is Ukrainian). Manual search identified both as hosters. Relabeling them accordingly, the servers we observe are mostly located in what PeeringDB labels as *Content* ASes, *i.e.,* hosters (EU: 70%, US: 80%). The large share of hosters fits the distribution model for malware. These servers need to be reachable from everywhere with a high uptime to allow compromised machines to download malware. Transit and access networks together make up roughly 20% at the EU VPs and 5% at the US VP.

The most popular locations for servers are displayed in Figure 7. While Estonia runs top in the EU, Russia takes the biggest chunk for the US by far. Notably, none of the Russian servers were observed in our EU vantage points, thus they exhibit a geographical preference [34], [50] in their scanning behavior. This is unexpected as Log4j is ubiquitous and not known to be more present in a specific geographic region.

Ports. Less than two percent of the LDAP servers used for the attacks are bound to the default LDAP port (389). Instead, the most used port is 1389. The share ranges from 93% to 96% at all vantage points. Other ports in use are 2420 and 80 in the EU, each between 1% and 2%. In the US we observe 12344 at around 2% as well. Such a high share could hint at common tools or tutorials used by the attackers.

Paths. Except for a single path, observed paths do not conform to the RFC that defines them [51] as they do not include a valid distinguished name. Two paths stand out among the LDAP URLs. First is `/Exploit`, which makes up the largest share at all vantage points with 70% to 80% in the EU and 20% in the US. A second group of paths shares the segments `Command/Base64` followed by a Base64-encoded segment. This group takes the second-largest share. These paths begin in a variety of ways, potentially hinting at their purpose, such as `TomcatBypass` or `GroovyBypass`. Decoding the Base64 segment reveals script code (mostly bash) that downloads an executable via HTTP to run locally.

At first glance, encoding commands in a URL path is an odd choice. However, the second piece of the puzzle is an LDAP server implementation, which dynamically builds Java objects that will run the Base64-encoded command. These servers can be found on GitHub in several repositories, although the original[1] is no longer available. This server—aptly named `JNDIExploit`—binds local ports for LDAP and HTTP. It responds to a variety of queries that include the `Base64` fragment and match the paths we observed. The most common variant we see is the `TomcatBypass`.

Having such tooling at hand makes it easy for attackers to set up the attack and run it. We confirmed that servers used in attacks exhibit this dynamic behavior by sending a custom Base64-encoded command. We received the Java object in response to our custom code. The `JNDIExploit` repositories use port 1389 as the default for LDAP, which matches our observation of common LDAP ports in the URLs.

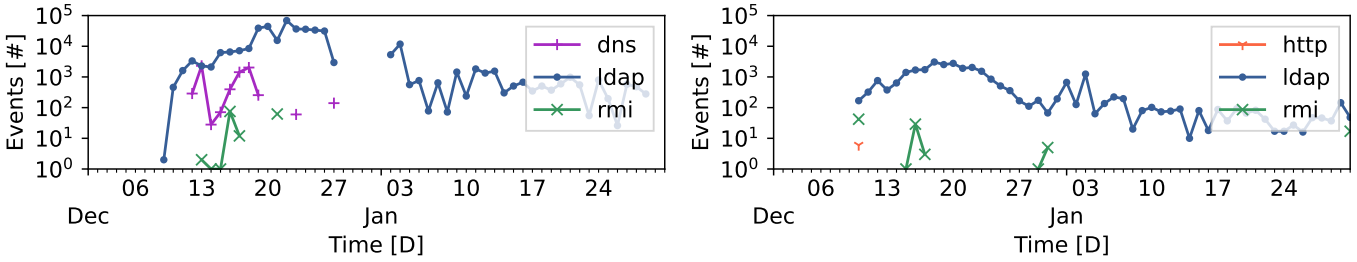
2) Downloading Malware: The Java objects are just an intermediate step to the goal of the attack: infecting the host with malware. We follow this path to find out which malware is distributed this way. When running the downloader in Feb'22, most servers were no longer available. We successfully download nine distinct Java objects, compare step ④.

The LDAP answers contain two important keys: `javaClassName` and `javaSerializedData`. In all cases the class name is set to `java.lang.String`. The objects we collected match the objects built by the JNDIExploit LDAP server, see Section IV-C1. The serialized objects look like a Java `StringRefAddr` object.

There are two versions among the objects. The Tomcat bypass instantiates a script engine to run JavaScript code while the Groovy bypass builds on Groovy itself. The script code is encoded in the serialized objects as ASCII. One of the Tomcat samples executes PowerShell code, which likely targets Windows. While most of the other scripts include a mechanism to determine the local OS, e.g., by checking the direction of slashes in a file path, they execute bash commands either way, and are thus unlikely to run on Windows.

We extract the download commands in our small samples set by hand and download three different binaries, compare step ⑤. (Four connections failed and three of the objects contain the same scripts, although they differ slightly.) The hashes of all downloads are registered on VirusTotal [52], where they were first submitted between the mid and end of Jan'22, i.e., while Log4Shell attacks were taking place. Two of the samples are scripts while one is a 32-Bit ELF binary.

[1] `https://github.com/feihong-cs/JNDIExploit`



(a) US VP.

(b) EU VP 3.

Fig. 6: Schemes in JNDI URLs over time. LDAP is used nearly exclusively (Dec'21 to Jan'22).

The malware acquired via the PowerShell code is itself a PowerShell script and downloads a binary with the name of a known crypto miner. Although the other shell script looks more sophisticated—it stops local programs, downloads an uninstaller to remove software, adds new cron tab entries, and tries to make its way into connected hosts via ssh—its goal is similar: installing a crypto miner.

Understanding the Base64-encoded commands opens another way to acquire malware. Instead of taking the round-trip via the LDAP server, which embeds them into a Java object, the commands can be decoded directly. Via this method we can acquire an additional binary: a 64-bit executable ELF file first registered on VirusTotal at the beginning of Dec'21. The low yield matches the churn in URLs we observe in Section IV-B.

3) Locating Malicious LDAP Servers: We finally explore the malicious server infrastructures by active scans. To this end, we utilize the fact that the Log4Shell exploit requires publicly reachable servers that return Java objects, compare step ③ and ④ in Figure 2.

Malicious servers predominantly listen on port 1389 coupled with a non-standard LDAP behavior. This is why we use ZMap [53] to scan TCP/1389 for open ports. We then identify unsecured LDAP servers by performing LDAP bind operations, which should fail on servers enforcing authentication. In a next step we query for the two most common paths observed: `/Exploit` and a path with `Base64` string.

We find 5.1M servers responding to SYNs, but only 1,110 allow for an unauthorized LDAP-binding. 81 servers return answers for `/Exploit`, and 179 for `Base64`. These sets overlap, which leads to 183 malicious LDAP servers in total (16%). Comparing to Figure 5, we infer that the number of servers in daily use and dormant servers differ by an order of magnitude.

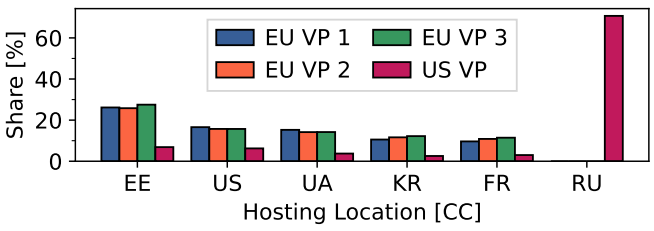


Fig. 7: The top 5 countries that host Log4Shell servers. Russia only takes a share for attacks targeting addresses in the US.

We collect six Java objects via the `/Exploit` path and 97 objects via the Base64 path. Their general structure matches the objects from Section IV-C2. Since Base64 requests are expected to encode our command, they should not return any malicious objects. Six objects are an exception. Here, servers return the same payload for both paths, likely a static response. We identify one downloader for a 64-Bit ELF binary, an object that runs PowerShell code, and one broken object. The remaining three payloads do not include script code.

These scans could eventually be used to identify and take down malicious LDAP servers, even though they are quickly moved and setup anew. Changing servers would also require adopting the payload included in scan campaigns.

D. Summary of Findings

Malicious and benign actors exhibited different scanning behavior. Both groups started scanning within 24 hours of the disclosure. However, benign scans ceased within the first week, whereas malicious scans peaked thereafter and continued with moderate intensity.

Available open-source tooling developed earlier for a JNDI exploit spread among attackers and lead to widely similar characteristics in their exploit strings. This enabled the detection of the malicious LDAP servers via scans.

V. The Continuous Development of Log4Shell Attacks

Log4j patches were released jointly with the disclosure and significant awareness was raised globally. Nevertheless, fixing of installations remained largely incomplete. Due to the nature of this software bug, every service instance stays vulnerable until patched [54]. Throughout 2022, Log4Shell continued to be a virulent threat to Internet infrastructure [55], [56].

In this section, we follow the activities of the Log4Shell scanners during 2022, analyze their activities and their infrastructure, and try to take a deeper look behind the scenes.

A. Overview

The trend of declining Log4Shell events foreshadowed in January 2022 did not sustain. Log4Shell scans continued in multiple waves throughout 2022. Figure 8 displays the temporal development of Log4Shell events recorded every week

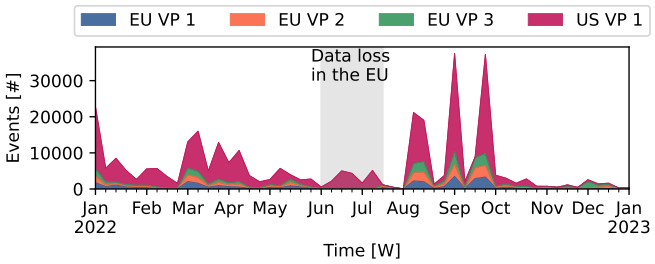


Fig. 8: The intensity of Log4Shell events through 2022. Events from the vantage points are stacked and aggregated per week.

at our four vantage points. Beginning in March we observe a small series of higher activity. August and September are the periods of the highest intensity. Thereafter, scanning events reduced in November and December.

1) Maliciousness: We continuously monitor the source addresses throughout 2022, 80% of which GreyNoise classifies as malicious. We further compare against a list of acknowledged scanners [57] and find no overlap. This is in line with our observations towards the end of the disclosure period, where benign scanners quickly stopped while malicious scanners remained active, *cf.* Section IV-A.

2) Source ASes: The monthly distribution of top contributing ASes is shown in Figure 9. All top ASes offer content hosting services to the public. We anonymized these hosters, of which some are prominent. In the EU, AS A is the top contributor while it is AS D in the US. Although it sends roughly the same number of packets to all VPs, AS E only stands out in the EU due to the difference in total packets received at each VP. AS B and AS C take places two and three at both VPs. They are mostly active in August (US) and September (US, EU).

3) Target ports: HTTP-like ports continue to be the most frequent targets with about 90%. The top three ports (80, 8080, and 8081) see most traffic, accumulating to around 60%, while the top 10 ports add up to roughly 80% of the traffic.

Starting in March 2022, scans *again* target port 7547, which is used by TR-069 (*cf.* IV-A4). This HTTP-based management protocol has been abused in a variety of exploits targeting router devices [34], [58], [59] and was also abused by Mirai variants [46], [60]. The payloads we observe are plain HTTP messages that do not contain indicators of the SOAP messages defined by TR-069. As such, it is likely that scanners simply test known HTTP endpoints for the Log4Shell vulnerability. Fritz!Box routers, which use port 8089 for TR-069 configuration, were targets alongside scans to port 7547. Note that these routers are not vulnerable to Log4Shell[2].

4) Schemas and Paths: The choice in JNDI schemas continues to favor LDAP. A negligible number of RMI and DNS events can be observed in March and May. `Base64` encoding dominates the paths with a share of more than 80% at each vantage point.

[2]https://avm.de/aktuelles/kurz-notiert/2021/schwachstelle-im-java-projekt-log4j-avm-produkte-nicht-betroffen/

B. The Context of the Event Peaks

Our activity logs in Figure 8 show high peaks in particular for the US vantage point. We now take a closer look to narrow down the related activities and their origins. We first analyze the geographic origins of the scanning sources. Next, we relate the top sources of each country via shared LDAP servers identified from the JNDI URLs. We find small groups as displayed in Figure 10. Throughout the following discussion, ASes remain identically enumerated as in Figure 9.

1) Geolocation: The contributions of source ASes (*cf.* Figure 9) does not directly map on countries, since several networks span different nations. AS E, for example, appears in GB, IN, BR, FR, and the US while AS D appears in US and BR.

No single country stands out during 2022. The ranking of source countries is more stable across VPs, although there are small differences between the EU and US vantage points. US prefixes originate between 35% and 45% of all events at each vantage point, followed by Poland (about 15%). The following top five include Panama and Brazil as well as China in the EU and the Netherlands in the US. The top three countries alone add up to at least 65% at each VP.

US. The most active group consists of two scan sources (in AS D and AS F) and one LDAP host (LDAP Server 2), contributing nearly 35% of the US events during the spikes. Noticeable, the same IP from AS F can be observed in a low-volume scan in May with URLs that use a different LDAP server (LDAP Server 1). The next five groups contribute between 17% and 10%. These clusters are small, they usually use a single LDAP server (sometimes two) and a maximum of eight sources for scanning (median: 1).

Brazil. The scans from *Brazil* in March and April originate from AS D, as well, which hosts both LDAP servers observed in the US scans. It shares an LDAP server with the low-volume scan from AS F in May.

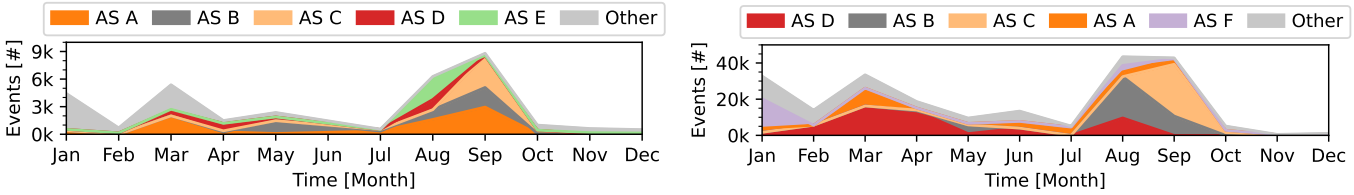
Poland. The scans from *Poland* in May originate from AS B, which is owned by a Polish hoster. They also use LDAP Server 1 in AS D. Subsequent scans in August and September from the same hoster use a different source address and LDAP server (LDAP Server 2).

Panama. In September, scans from AS C start using LDAP Server 2. While the addresses geolocate to *Panama* they apparently belong to a Swiss-based AS with an office registered in Panama.

2) Clustering: The most active addresses during the peaks can be interrelated via its use of shared infrastructure, *i.e.,* two LDAP servers and a single IP address, which uses both of these servers (Figure 10). Following these observations, we build clusters from all *sources* and LDAP *servers* that we observed.

It is noteworthy, though, that events originating from the same AS cannot simply be clustered based on their infrastructure. Events from AS E, for example, can be grouped in two clusters, one from the US, BR, and GB, and a second geolocating to FR and IN.

A cluster contains all sources that have at least one LDAP server in common. Our analysis reveals 233 clusters of various sizes as shown in Figure 12. While the most active cluster



(a) EU VP.

(b) US VP.

Fig. 9: Top five source ASes in 2022 per month ranked in the legends by total event share. All remaining ASes are aggregated as "Other". Names are anonymized. A hand full of ASes originate most events.

originates nearly 60% of all events, the second active cluster only contributes 5%. Overall, the top 10 clusters add up to 85% of the scanning activity. Ranking the number of sources, the most active cluster is 3rd with 25 source addresses and three LDAP servers. The largest cluster contains 55 source addresses and three LDAP servers. The second-largest cluster comprises 30 source addresses and two servers.

All addresses that stood out during the examination of the event peaks (Figure 10) belong to the most active cluster. A timeline of its activity is depicted in Figure 13, using a different color per source address. Each peak originates from a different address. During the first half of 2022 scans are more widely distributed and involve more addresses. In contrast, the peaks in Aug, Sep, and Oct are caused by a single address each.

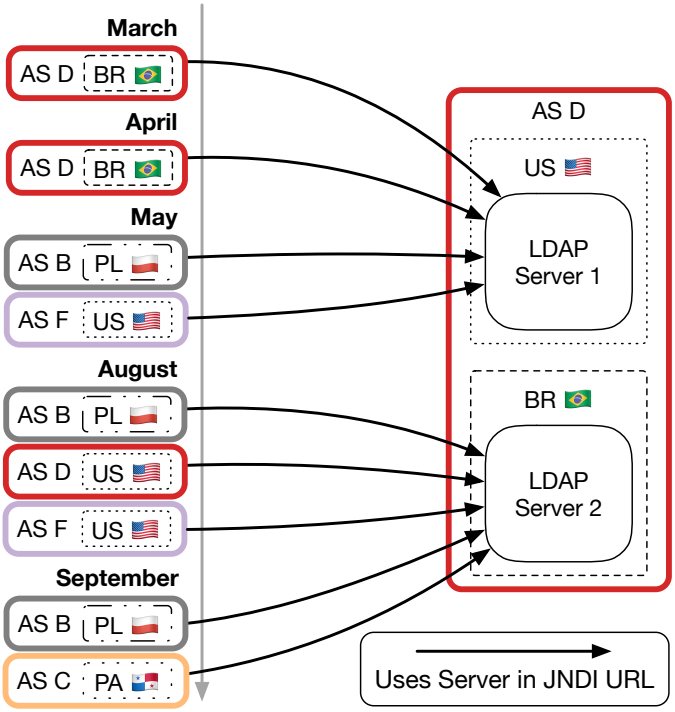


Fig. 10: The most active source ASes during each peak can be related via the LDAP servers placed in their JNDI URIs. Each box bundles one or several addresses. Geolocated country tags refer to IP address locations. AS B in May and August originates the same, single address.

We noted that several prefixes of provider C have been reported for blocking by fail2ban. Blocking can be considered a likely motivation for the constant address changes, since an attacker loses efficiency due to block lists and potentially risks even a takedown of the host. The LDAP servers, on the other hand, seem unperturbed and continue as persistent attack infrastructure.

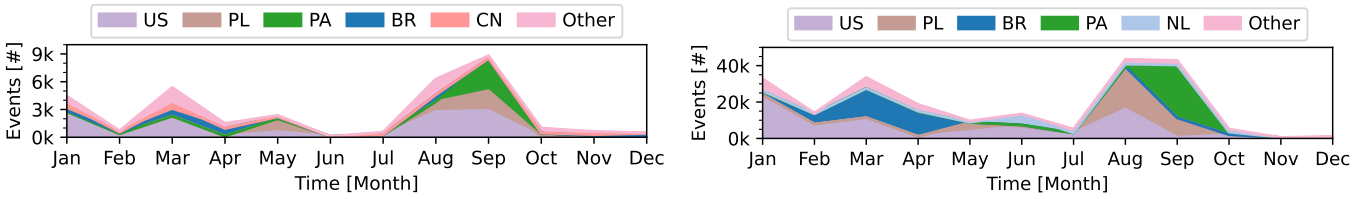
3) Downloaders: A large share of events in 2022 uses URLs with Base64 segments, *cf.* Section IV-C1. Given the address churn in the active cluster, we now check whether the downloaders transmitted as Base64-encoded segments are equally short-lived. Figure 14 contrasts the lifetimes of the sources (14a) with the lifetimes of downloaders (14b). Sources have much shorter lifetimes, *i.e.,* scanners use varying sources to propagate the same downloader. While some downloaders were only observed during a few days, the median window is about a month. The end of June marks a strong cut, with only one continuing downloader observed.

Downloaders offer malware as web servers. As such they are less likely to get blacklisted. In addition, their addresses are distributed during attack campaigns as part of downloader scripts, which makes longer lifetimes attractive for the attackers and easy to reach.

4) Discussion Questions: Our observations of the Log4Shell scanners through the lens of a reactive network telescope opens up consecutive questions. Our reactive measurements record what scanners deliver but do not actively request further information from the source hosts that interact with Spoki. Our available information comes from the established connections and their payloads as well as publicly available meta-information about the addresses.

Do 'scanners' rent or hijack hosts? A large share of the scans we observe originate from hosting providers. Nodes in these networks could be genuinely rented by the scanning entities or be compromised machines from third parties.

We analyzed the open ports of 20 hosts in the most active cluster and compared with historic data from Shodan [61], which shows open ports, identified services, and potentially an organization name. The organization name matches the hosting provider. The most common service is OpenSSH, which is a common way to remotely access VMs. Some hosts run additional services, such as nginx, RDP, or Apache httpd. Since we do not know about software versions, we cannot check for potential vulnerabilities. However, there are no fingerprints that collectively apply to these hosts, neither



(a) EU VP.

(b) US VP.

Fig. 11: Top five source countries in 2022 per month. All remaining countries are aggregated as "Other". A hand full of countries originate most events. Note the different scaling at y-axes between the EU and the US VP.

for hijacking through some vulnerable service nor by common administration patterns.

While our view is limited, these insights suggest that the VMs are specifically instantiated for Log4Shell scanning.

Should clustering extend to the web server addresses? Our clustering (s. Section V-B2) is based on the source of the scan and the LDAP server the attack uses. The Log4Shell exploit further uses a web server to download malware or another downloader, *i.e.,* a script to configure the system and install the malware binary (⑤ in Figure 2). Only the Log4Shell exploit string and the request/response from the JNDI services are specific to the Log4Shell exploit. Using an exploit to enter the system and run shell code to download the malware or a more complex downloader script is a common strategy that can be observed in other attack scenarios, such as two-phase scans [34]. While all clusters show at least one overlap between a scanning address and an LDAP address, less than 8% have an overlap between the web server and a scanning address or an LDAP address.

As such, we want to carefully avoid mixing the attack and the exploit steps. Attackers might use different attack methodologies for the same attack or pay others to run the attack campaign in the first place [11].

Cluster disappearance: What happened in October 2022? After long-lasting activity, the cluster which was responsible for a dominant share of events suddenly stopped scanning, *cf.* Figure 13. We could not find any news about take-downs or similar that clearly linked to this entity. Hence, we can only speculate why this stopped.

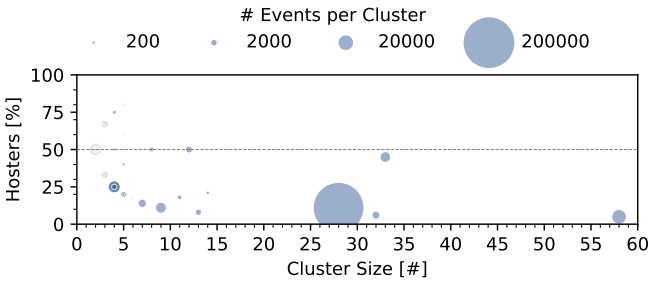


Fig. 12: Scanning clusters by count of source addresses and the share of hosters. Clusters above the middle line contain more hosters than scanners. Darker colors indicate overlapping circles.

Running a longer-lasting campaign from different scanning nodes likely required resources of the attackers, *i.e.,* work, planning, and online machines. As such, they may have shifted focus to networks that we cannot observe (*e.g.,* specific regions or ASes), a lack of success may have made the effort no longer valuable, or unreported law enforcement may have intervened.

C. The JNDI/LDAP Exploits

We finally take a look at the evolution of the actual exploits and the corresponding malware distribution.

1) Exploit placement: The user agent remains the most favored exploit string, although with a reduced share of 14%. The remaining top five placements remain relatively stable across vantage points but changed compared to our initial measurements: (*i*) `User-Agent` (14%), (*ii*) `X-API-Version` (13%), (*iii*) `Referer` (10%-12%), (*iv*) `Cookie` (9%-12%), (*v*) `X-Forwarded-For` (5%).

Apparently, attackers did not converge on a placement but positioned the exploit string wherever possible. It remains unclear whether they could not identify an optimal approach, or whether the variety was used to target different deployments.

2) Malware: At the time of writing, none of the malware persisted online. During our measurements, we download scripts that match the batch and shell script of a Monero crypto miner available on GitHub [62]. An issue from June 2022 reports the involvement of the script in Log4Shell attacks – without response. We also found an ELF binary that VirusTotal labeled as a Bitcoin miner.

3) Active LDAP scans: Due to the prevalence of the Base64 payloads, we repeated our scans for assisting LDAP servers in 2023. We found limited change in the number of available servers. 1,061 LDAP servers listening on port 1389 allowed for unauthorized binding – 39 less than at the beginning of 2022. Around 270 of these servers appeared in both scans.

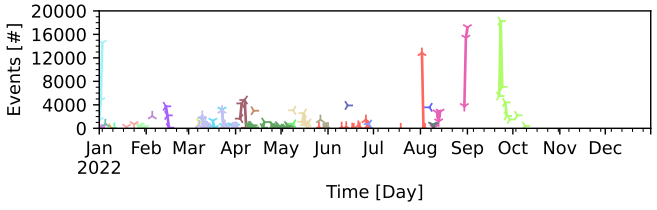
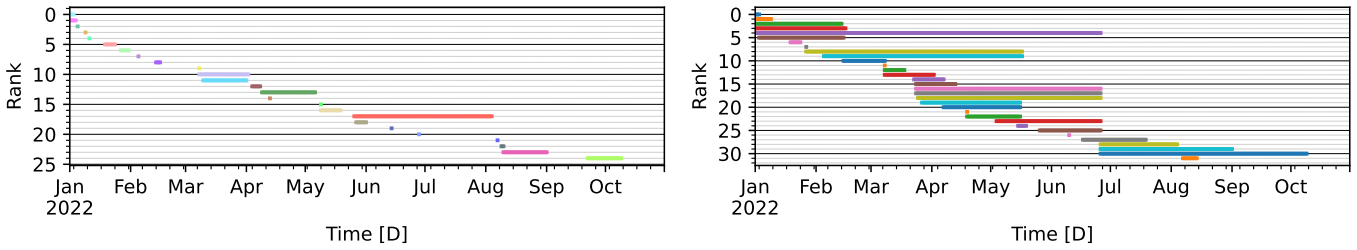


Fig. 13: Activity of the most active cluster. Each color represents a different source address.



(a) Source addresses lifetimes. (Colors match Figure 13.)

(b) Downloader lifetimes.

Fig. 14: The observation windows of source addresses and downloaders distributed by the most active cluster, ranked by time.

D. Summary of Findings

Attackers move their highly visible scan infrastructure regularly between hosters and countries, presumably to avoid protective counter actions, such as blocking. In contrast, they continuously rely on the identical infrastructure of LDAP servers and downloaders. Collecting information about these backend infrastructures from the exploits can be used to track and group malicious actors. Our measurements indicate that the majority of attacks can be attributed to a few groups of actors, the largest of which was accountable for about 60% of the attacks and completely vanished on Oct 9, 2022.

VI. Discussion and Conclusion

The Log4Shell exploit appeared as a disruptive incident on the Internet by allowing for remote code execution at a global server landscape with threatening ease. The list of affected software contains many popular applications [63]. YouTube videos explain details and give guidance on how to apply attack tools. The exploit is built on a conceptual lack of input sanitization—a common challenge that has plagued the industry in the form of SQL injections for years. The vulnerability saw wide coverage online. Blog posts, lists of vulnerable applications, and detection tools [64] were quickly published. At the same time, official organizations distributed security reports and issued warnings.

In this work, we observed Log4Shell scanning through reactive network telescopes from the time of its disclosure—and for a year thereafter. Among the most notable observations were large peaks of malicious events throughout our measurement period. These hit all vantage points but particularly targeted the US, giving the scans a geographical focus. This trend continued throughout 2022, in which we saw scanning waves during March, August and September. A large share of this traffic could be attributed to a single scanning entity.

Our analysis revealed common characteristics among scanners, such as the path used in LDAP scans. While both `Exploit` and Base64-encoded commands started out as popular paths, the latter dominated throughout 2022. It can be linked to an open-source project that makes it easy to set up an attack infrastructure.

From scanning, we could not infer the success rate of attackers, but observing scanning behavior can be an expressive indicator for the liveliness of the scene. The press reported on a few hacks that abused the Log4Shell vulnerability [65]–[67].

Log4Shell has been around for several years now, its long-term effects yet remain unclear. Many applications quickly saw patches, but rollout will eventually slow down. In contrast to the persistent Mirai, which is a family of Trojans, Log4Shell is bound to a specific vulnerability. Mirai and its descendants can incorporate new exploits into their toolkits and continue to spread. In early 2022 some analysts already reported about Mirai using Log4Shell to infect systems [68]. Attackers will likely continue to exploit Log4Shell as long as (unpatched) victims persist. However, as of December 7, 2023, the Infosecurity Magazine reported that only 125,000 servers still hosted software potentially vulnerable to Log4Shell [69].

Our measurement method Spoki [34], the reactive telescope, has enabled a continuous tracing of the malicious activities by following the scans of the various actors. In future work, we expect to quickly detect and monitor emerging attack campaigns that rely on explorative scanning for victims. Part of this future work on Internet-wide security monitoring will be the early discovery of malware from zero-day exploits.

Acknowledgments

We thank CAIDA for providing access to the UCSD Network Telescope and the network operators who support us in deploying our telescopes in Europe. We appreciated fruitful discussions with Johannes Klick and Stephan Lau. This work was partly supported by the German Federal Ministry of Education and Research (BMBF) within the project *PRIMEnet*.

References

[1] The MITRE Corporation, "CVE-2015-0565," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-0565.

[2] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre Attacks: Exploiting Speculative Execution," *Commun. ACM*, vol. 63, no. 7, pp. 93–101, Jun 2020.

[3] The MITRE Corporation, "CVE-2017-5753," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2017-5753.

[4] ——, "CVE-2017-5715," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2017-5715.

[5] Z. Durumeric, F. Li, J. Kasten, J. Amann, J. Beekman, M. Payer, N. Weaver, D. Adrian, V. Paxson, M. Bailey, and J. A. Halderman, "The Matter of Heartbleed," in *Proc. of ACM IMC*. New York, NY, USA: ACM, 2014, pp. 475–488.

[6] The MITRE Corporation, "CVE-2014-0160," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2014-0160.

[7] M. Nawrocki, P. F. Tehrani, R. Hiesgen, J. Mücke, T. C. Schmidt, and M. Wählisch, "On the Interplay between TLS Certificates and QUIC Performance," in *Proc. of 18th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*. New York, NY, USA: ACM, 2022, pp. 204–213. [Online]. Available: https://dl.acm.org/doi/10.1145/3555050.3569123

[8] E. Osterweil, P. F. Tehrani, T. C. Schmidt, and M. Wählisch, "From the Beginning: Key Transitions in the First 15 Years of DNSSEC," *Transactions on Network and Service Management (TNSM)*, vol. 19, no. 4, pp. 5265–5283, December 2022. [Online]. Available: https://doi.org/10.1109/TNSM.2022.3195406

[9] N. Rodday, Í. Cunha, R. Bush, E. Katz-Bassett, G. D. Rodosek, T. C. Schmidt, and M. Wählisch, "The Resource Public Key Infrastructure (RPKI): A Survey on Measurements and Future Prospects," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 21, no. 2, pp. 2353–2373, 2024. [Online]. Available: https://doi.org/10.1109/TNSM.2023.3327455

[10] Stack Overflow, "Here's how Stack Overflow users responded to Log4Shell, the Log4j vulnerability affecting almost everyone," Feb 2022, https://stackoverflow.blog/2022/01/19/heres-how-stack-overflow-users-responded-to-log4shell-the-log4j-vulnerability-affecting-almost-everyone/.

[11] K. Huang, M. Siegel, and S. Madnick, "Systematically Understanding the Cyber Attack Business: A Survey," *ACM Comput. Surv.*, vol. 51, no. 4, jul 2018. [Online]. Available: https://doi.org/10.1145/3199674

[12] P. F. Tehrani, E. Osterweil, J. Schiller, T. C. Schmidt, and M. Wählisch, "Security of Alerting Authorities in the WWW: Measuring Namespaces, DNSSEC, and Web PKI," in *30th The Web Conference (WWW'21)*. New York, USA: ACM, April 2021, pp. 2709–2720. [Online]. Available: https://doi.org/10.1145/3442381.3450033

[13] P. F. Tehrani, R. Hiesgen, T. Lübeck, T. C. Schmidt, and M. Wählisch, "Do CAA, CT, and DANE Interlink in Certificate Deployments? A Web PKI Measurement Study," in *Proc. of Network Traffic Measurement and Analysis Conference (TMA)*. Piscataway, NJ, USA: IEEE Press, May 2024, pp. 1–11. [Online]. Available: https://doi.org/10.23919/TMA62044.2024.10559089

[14] R. Hiesgen, M. Nawrocki, T. C. Schmidt, and M. Wählisch, "The Race to the Vulnerable: Measuring the Log4j Shell Incident," in *Proc. of Network Traffic Measurement and Analysis Conference (TMA)*. Laxenburg, MD, Austria: IFIP, June 2022, pp. 1–9. [Online]. Available: https://tma.ifip.org/2022/wp-content/uploads/sites/11/2022/06/tma2022-paper40.pdf

[15] Apache, "Apache Log4j Security Vulnerabilities," Feb 2022, https://logging.apache.org/log4j/2.x/security.html.

[16] The MITRE Corporation, "CVE-2021-44228," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228.

[17] R. Goers, "Restrict LDAP access via JNDI," Feb 2022, https://github.com/apache/logging-log4j2/pull/608.

[18] The MITRE Corporation, "CVE-2021-45046," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-45046.

[19] ——, "CVE-2021-45105," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-45105.

[20] ——, "CVE-2021-44832," Feb 2022, https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44832.

[21] NIST, "CVE-2021-44228 Detail," Feb 2022, https://nvd.nist.gov/vuln/detail/CVE-2021-44228.

[22] Google, "Understanding the Impact of Apache Log4j Vulnerability," Feb 2022, https://security.googleblog.com/2021/12/understanding-impact-of-apache-log4j.html.

[23] Apache, "JNDI Lookup plugin support," Feb 2022, https://issues.apache.org/jira/browse/LOG4J2-313.

[24] South China Morning Post, "Apache Log4j bug: China's industry ministry pulls support from Alibaba Cloud for not reporting flaw to government first," Feb 2022, https://www.scmp.com/tech/big-tech/article/3160670/apache-log4j-bug-chinas-industry-ministry-pulls-support-alibaba-cloud.

[25] ZDNet, "Chinese regulators suspend Alibaba Cloud over failure to report Log4j vulnerability," Feb 2022, https://www.zdnet.com/article/log4j-chinese-regulators-suspend-alibaba-partnership-over-failure-to-report-vulnerability/.

[26] Matthew Prince (Cloudflare), Feb 2022, https://twitter.com/eastdakota/status/1469800951351427073.

[27] Cisco Talos, "Threat Advisory: Critical Apache Log4j vulnerability being exploited in the wild," Feb 2022, https://blog.talosintelligence.com/2021/12/apache-log4j-rce-vulnerability.html.

[28] 360 Netlab, "Threat Alert: Log4j Vulnerability Has Been adopted by two Linux Botnets," Feb 2022, https://blog.netlab.360.com/threat-alert-log4j-vulnerability-has-been-adopted-by-two-linux-botnets/.

[29] Juniper, "Log4j Attack Payloads In The Wild," Feb 2022, https://blogs.juniper.net/en-us/security/in-the-wild-log4j-attack-payloads.

[30] J. Khoury, M. Safaei Pour, and E. Bou-Harb, "A Near Real-Time Scheme for Collecting and Analyzing IoT Malware Artifacts at Scale," in *Proc. of the ARES*, ser. ARES '22. New York, NY, USA: ACM, 2022. [Online]. Available: https://doi.org/10.1145/3538969.3539009

[31] Microsoft, "Guidance for preventing, detecting, and hunting for exploitation of the Log4j 2 vulnerability," Feb 2022, https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-exploitation/.

[32] D. Everson, L. Cheng, and Z. Zhang, "Log4shell: Redefining the Web Attack Surface," in *Proc. of the MADWeb*, 01 2022, pp. 1–8.

[33] A. Muñoz and O. Mirosh, "A Journey From JNDI/LDAP Manipulation to Remote Code Execution Dream Land," Feb 2022, https://www.blackhat.com/docs/us-16/materials/us-16-Munoz-A-Journey-From-JNDI-LDAP-Manipulation-To-RCE-wp.pdf.

[34] R. Hiesgen, M. Nawrocki, A. King, A. Dainotti, T. C. Schmidt, and M. Wählisch, "Spoki: Unveiling a New Wave of Scanners through a Reactive Network Telescope," in *Proc. of 31st USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, August 2022, pp. 431–448. [Online]. Available: https://www.usenix.org/system/files/sec22-hiesgen.pdf

[35] D. Charousset, R. Hiesgen, and T. C. Schmidt, "CAF - The C++ Actor Framework for Scalable and Resource-efficient Applications," in *Proc. of the 5th ACM SIGPLAN Conf. on Systems, Programming, and Applications (SPLASH '14), Workshop AGERE!* New York, NY, USA: ACM, Oct. 2014, pp. 15–28.

[36] J. Kepner *et al.*, "Spatial Temporal Analysis of 40,000,000,000,000 Internet Darkspace Packets," in *2021 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, 2021, pp. 1–8.

[37] M. Nawrocki, M. Wählisch, T. C. Schmidt, C. Keil, and J. Schönfelder, "A Survey on Honeypot Software and Data Analysis," Open Archive: arXiv.org, Technical Report arXiv:1608.06249, 2016. [Online]. Available: http://arxiv.org/abs/1608.06249

[38] M. Nawrocki, J. Kristoff, C. Kanich, R. Hiesgen, T. C. Schmidt, and M. Wählisch, "SoK: A Data-driven View on Methods to Detect Reflective Amplification DDoS Attacks Using Honeypots," in *Proc. of IEEE Euro Security & Privacy*. IEEE, July 2023, pp. 576–591. [Online]. Available: https://doi.org/10.1109/EuroSP57164.2023.00041

[39] CAIDA, "The UCSD Network Telescope," Website, 2012, last Access: May 2022. [Online]. Available: https://www.caida.org/projects/network_telescope/

[40] F. Roth, "log4shell-detector," Dec 2021, https://github.com/Neo23x0/log4shell-detector.

[41] "MaxMind – GeoLite Country." [Online]. Available: http://www.maxmind.com/app/geoip_country

[42] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye, "IP Geolocation Databases: Unreliable?" *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 2, pp. 53–56, Apr 2011.

[43] PeeringDB, "The Interconnection Database," website, 2019. [Online]. Available: https://www.peeringdb.com/

[44] GreyNoise, "IP Lookup API," https://greynoise.io.

[45] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescapè, "Analysis of a "/0" Stealth Scan from a Botnet," *IEEE/ACM Transactions on Networking*, vol. 23, no. 2, pp. 341–354, Apr 2015.

[46] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai Botnet," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1093–1110.

[47] Google, "Googe-Crawler (User-Agents)," May 2022, https://developers.google.com/search/docs/advanced/crawling/overview-google-crawlers.

[48] G. Tyson, S. Huang, F. Cuadrado, I. Castro, V. C. Perta, A. Sathiaseelan, and S. Uhlig, "Exploring HTTP Header Manipulation In-The-Wild," in *Proc. of the WWW*. Republic and Canton of Geneva, CHE: ACM, 2017, pp. 451–458.

[49] Juniper, "Log4j Vulnerability: Attackers Shift Focus From LDAP to RMI," Feb 2022, https://blogs.juniper.net/en-us/threat-research/log4j-vulnerability-attackers-shift-focus-from-ldap-to-rmi.

[50] P. Richter and A. Berger, "Scanning the Scanners: Sensing the Internet from a Massively Distributed Network Telescope," in *Proceedings of the*

Internet Measurement Conference, ser. IMC '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 144–157.
[51] M. Smith and T. Howes, "Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator," IETF, RFC 4516, June 2006. [Online]. Available: https://doi.org/10.17487/RFC4516
[52] "VirusTotal," https://www.virustotal.com/.
[53] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide scanning and its security applications," in *Proc. of the 22nd USENIX Security Symposium*. Berkeley, CA, USA: USENIX Assoc., Aug. 2013, pp. 605–620.
[54] E. Gent, "Log4Shell Still Has Sting in the Tail," Website, 12 2022. [Online]. Available: https://spectrum.ieee.org/log4shell-log4j-still-stings
[55] CISA, "Malicious Cyber Actors Continue to Exploit Log4Shell in VMware Horizon Systems," Website, 06 2022. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa22-174a
[56] L. Grustniy, "Log4Shell a year on – A year after discovery, the Log4Shell vulnerability is still making itself felt." Website, 12 2022. [Online]. Available: https://usa.kaspersky.com/blog/log4shell-still-active-2022/27531/
[57] M. Collins, "Acknowledged Scanners," https://gitlab.com/mcollins_at_isi/acknowledged_scanners, 2022, commit: c48b8840.
[58] S. Tal, "I hunt TR-069 admins," DEF CON 22 talk, 2017. [Online]. Available: https://defcon.org/images/defcon-22/dc-22-presentations/Tal/DEFCON-22-Shahar-TaI-I-hunt-TR-069-admins-UPDATED.pdf
[59] M. Hils and R. Böhme, "Watching the Weak Link into Your Home: An Inspection and Monitoring Toolkit for TR-069," in *Proc. of the ACNS*, ser. LNCS, M. Conti, J. Zhou, E. Casalicchio, and A. Spognardi, Eds., vol. 12147. Cham: Springer International Publishing, Oct 2020, pp. 233–253. [Online]. Available: https://doi.org/10.1007/978-3-030-57878-7_12
[60] B. Krebs, "New Mirai Worm Knocks 900K Germans Offline," Krebs on Security Blog, 2016. [Online]. Available: https://krebsonsecurity.com/2016/11/new-mirai-worm-knocks-900k-germans-offline/
[61] Shodan, "Shodan - Search Engine for the Internet of Everything," Website, 2014. [Online]. Available: https://www.shodan.io/
[62] C3Pool, "xmrig_setup," Website, 07 2023. [Online]. Available: https://github.com/C3Pool/xmrig_setup
[63] National Cyber Security Centrum, "Log4shell vulnerabilities," Feb 2022, https://github.com/NCSC-NL/log4shell.
[64] D. Everson, A. Bastola, R. Mittal, S. Munde, and L. Cheng, "A Comparative Study of Log4Shell Test Tools," in *Proc. of the SecDev*, 2022, pp. 16–22.
[65] Microsoft Threat Intelligence, "MERCURY leveraging Log4j 2 vulnerabilities in unpatched systems to target Israeli organizations," Website, 08 2022. [Online]. Available: https://www.microsoft.com/en-us/security/blog/2022/08/25/mercury-leveraging-log4j-2-vulnerabilities-in-unpatched-systems-to-target-israeli-organizations/
[66] C. Page, "North Korea's Lazarus hackers are exploiting Log4j flaw to hack US energy companies," Website, 09 2022. [Online]. Available: https://techcrunch.com/2022/09/08/north-korea-lazarus-united-states-energy/
[67] CISA, "Iranian Government-Sponsored APT Actors Compromise Federal Network, Deploy Crypto Miner, Credential Harvester," Website, 11 2022. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa22-320a
[68] B. Toulas, "Log4shell exploits now used mostly for DDoS botnets, cryptominers," Website, 03 2022. [Online]. Available: https://www.bleepingcomputer.com/news/security/log4shell-exploits-now-used-mostly-for-ddos-botnets-cryptominers/
[69] P. Muncaster, "Impact of Log4Shell Bug Was Overblown, Say Researchers," Infosecurity Magazine, 2023. [Online]. Available: https://www.infosecurity-magazine.com/news/impact-log4shell-overblown/

VII. Biography Section

Raphael Hiesgen is currently pursuing a Ph.D. at HAW Hamburg under the guidance of Prof. T. C. Schmidt and Prof. Wählisch. His research centers on Internet measurements, specifically in the context of security applications. Proficient in C++ and adept at crafting scalable systems, he developed Spoki, a reactive network telescope tailored for long-term measurements. His skill set spans from designing and implementing measurement systems to delving into the analysis of collected data. Spoki was initially presented at USENIX Security in 2022 and is undergoing continuous improvements to broaden its visibility into the scanning ecosystem.

Marcin Nawrocki is a Senior Research Analyst at NETSCOUT, contributing to the ATLAS Security Engineering and Response Team (ASERT). He transitioned to ASERT following his doctoral program in Internet intelligence at Freie Universität Berlin (FU Berlin), where he was mentored by Matthias Wählisch and Thomas C. Schmidt. Bringing over a decade of experience in advancing Internet research, Marcin specializes in improving both present and future Internet infrastructure, with a keen focus on attack mitigation and prevention. His contributions were internationally recognized, including the Best Paper Award and Community Award at CoNEXT'22, Best Presentation at CoNEXT'21, and a Silver Medal at the Student Research Competition (SRC) during SIGCOMM'18.

Thomas C. Schmidt (Member, IEEE) received the Ph.D. degree in mathematical physics from FU Berlin. He is a Professor of Computer Networks and Internet Technologies with the Hamburg University of Applied Sciences (HAW), where he heads the Internet Technologies Research Group. Prior to moving to Hamburg, he was a Director of a scientific computer centre in Berlin. Since then, he has continuously conducted numerous national and international research projects. He was the principal investigator in a number of EU, nationally funded and industrial projects as well as a Visiting Professor with the University of Reading, U.K. His continued interests lie in the development, measurement, and analysis of large-scale distributed systems like the Internet. He serves as co-editor and a technical expert in many occasions and he is actively involved in the work of IETF and IRTF. Together with his group, he pioneered work on an information-centric Industrial IoT and the emerging data-centric Web of Things. He is a co-founder of several large open source projects and a Coordinator of the community developing the RIOT operating system—the friendly OS for the Internet of Things.

Matthias Wählisch (Member, IEEE) received the Ph.D. degree (Highest Hons.) in computer science from Freie Universität Berlin. He is a full professor and holds the Chair of Distributed and Networked Systems at the Faculty of Computer Science at TU Dresden. He is also a Research Fellow at the Barkhausen Institut. His efforts are driven by improving Internet communication based on sound research. His research and teaching focus on scalable, reliable, and secure Internet communication. This includes the design and evaluation of networking protocols and architectures, as well as Internet measurements and analysis. He is the PI of several national and international projects, supported by overall 6.8M EUR grant money. Since 2005, he has been active within IETF/IRTF. He published more than 150 peer-reviewed papers. He co-founded some successful open source projects, such as RIOT and RTRlib, where he is still responsible for the strategic development.