Network Security Monitoring Opportunities and Best Practices for Log4j Defense

blog.gigamon.com/2021/12/14/network-security-monitoring-opportunities-and-best-practices-for-log4j-defense/

December 15, 2021

Threat Research / December 14, 2021

Joe Slowik

In early December 2021, Apache released a patch to address CVE-2021-44228 in the Log4j logging framework library in Java. The vulnerability enables remote code execution (RCE) when Log4j parses a specially crafted string to retrieve and load follow-on code. After release, multiple proof of concept (POC) implementations of exploit code became publicly available through various public repositories and social media platforms, resulting in rapid implementation of widespread scanning for this vulnerability and weaponization.

As of this writing, multiple overviews exist from government entities, security vendors, and other parties who have published overviews of this vulnerability and its functionality. While analysis is still ongoing, the consensus opinion among multiple researchers and analysts is that this item, given scope and relative ease of exploitation, is of serious concern and will require ongoing attention and mitigation efforts.

Implications

Log4j is a logging library built into Java-based frameworks. While not necessarily universal, its use and application are widespread for the many applications and services leveraging Java resources for functionality. As a result, the potential attack surface for this vulnerability is vast – so large that as of this writing its precise implications and reach are yet to be determined.

Specifically, concerns center not around Log4j as a service being directly reachable for potential attackers, but rather the chain of dependencies from front-line, internet-facing applications or APIs to back-end processing using Log4j. For example, an adversary may trigger this vulnerability by placing the embedded exploit string in seemingly harmless contexts – a chat message on an application or a resource request that will fail – but because these items will be logged and ultimately processed by Log4j behind the scenes, the exploit condition will trigger.

Log4j dependencies are often not immediately evident, and in many cases may be difficult to discern from an end-user perspective. The result is that from an organization's perspective, resolving the problem is not just an issue of patching the Log4j library, but ensuring that multiple other services depending on vulnerable versions of this library are patched as well. Until third-party vendors and application providers also patch their software to include the latest updates in Log4j eliminating this vulnerability, organizations will remain vulnerable even if their own internal use of this framework is updated.

Given the sequence of dependencies, Gigamon ATR anticipates that this vulnerability will remain relevant for months, if not years, to come. As software supply chains gradually identify and resolve dependencies, the problem will resolve itself, but will not be completely solved until some point likely far in the future. While concerning, as of this writing exploitation of Log4j appears limited to opportunistic or research-oriented scanning, cryptocurrency miner installation, and variations of botnets such as Mirai-like installations. Gigamon ATR expects that this will inevitably change, if it has not done so already, to include initial access for information theft, ransomware deployment, and targeted intrusions. Due to the sheer volume of scanning and related activity, such attempts may have already occurred but are "hiding" within the overall noise. Nonetheless, network defenders and IT asset owners must work diligently to begin addressing this threat before it grows completely out of control.

Potential Pitfalls

The mechanisms through which CVE-2021-44228 can be exploited remain vast. Since publication, researchers have posted multiple variations of POC code to various social media and other platforms to demonstrate adaptability and defense evasion. Given the scope, severity, and likely duration of this vulnerability, defenders will require a variety of approaches to securing networks before patching can ultimately resolve the issue – but a number of these approaches demonstrate significant issues and if taken in isolation can lead to a false sense of security.

Avoiding an Indicator-Focused Approach

Many organizations, since initial publication of Log4j exploit activity, seek to bolster defense by publishing lists of network infrastructure identified in attempted exploitation. Unfortunately, this leads to a variety of issues:

- Ease of adversary creation or utilization of new infrastructure for multiple phases of operations allows for evasion of block or alerting lists.
- Defender inability to differentiate between infrastructure used purely for research-focused scanning and actual malicious operations to focus response and resources.
- Historical nature of identified infrastructure, meaning that such items could at best be used for analysis of prior exploit attempts but will likely be insufficient for future-oriented defense.

While indicator ingestion may be valuable for forensic purposes to determine if a known entity has probed an environment previously, this approach will be ill-suited to establish ongoing defense as exploitation continues. Given the ease with which adversaries can create new infrastructure, the sheer volume of potential malicious endpoints, and the availability of multiple services (e.g., dynamic DNS, cloud providers, proxy services, and other capabilities) that can obscure the true source of activity, an indicator-based approach may provide some short-term security but ultimately will prove to be a losing game. Furthermore, the sheer volume of items identified in both originating traffic and referenced infrastructure hosting exploit payloads mean any block or alert list will rapidly become exceedingly long, potentially to the point of causing resource issues, with significant difficulty in curating and updating the accuracy of such lists.

Defenders can still utilize reports of malicious infrastructure to determine exposure to or extent of past activity but must recognize the limitations of this approach if attempting to secure environments moving forward from the point of reporting. Focusing instead on *how* adversaries are using this vulnerability as opposed to *from where* may therefore represent a more valuable position.

Understanding Adversary Adaptability

Focusing on attacker implementations of Log4j exploits has its own problems, however. Since publication, multiple offensive security researchers identified various ways to “fuzz” or alter delivery of the Log4j exploit string to evade content-focused inspection and detections. As a result, static signatures based around the specific content of network traffic can be easily evaded and give rise to innumerable possibilities to encode or obfuscate the exploit payload.

While opportunities still exist to capture “common” implementations of CVE-2021-44228 exploitation, such activity will likely only encompass low-effort, lower-risk scanning, enumeration, and potentially commodity exploitation activity. Although valuable to an extent, such an approach will likely fail against determined, targeted intrusions willing to invest the time required to modify or obfuscate exploitation activity to evade defensive controls. As these are the most worrying vectors, applying “common” detection logic can lead to a mistaken sense of coverage for this threat leaving the most threatening and impactful adversaries with inadequate defensive coverage.

As such, while implementing and monitoring various mechanisms of Log4j exploitation can be useful at point of initial exploitation, defenders must also recognize the limitations of this approach. Simply implementing network security monitoring (NSM) signatures for this activity represents a good step in gaining visibility, but will almost certainly not be sufficient to catch all intrusions, especially the most focused or worrisome actors.

Mapping Attack Surface and Exposure

Finally, attack surface identification and reduction are critical for running a security program and managing defense irrespective of threat. Defenders must also, however, understand the scope and extent of such efforts. In the case of libraries such as Log4j, specific implementations and presence of this artifact are obscured as the item is frequently embedded in other commercial products, and its use is not immediately identifiable.

Furthermore, while mapping attack surface is indisputably valuable, on its own it does not represent a security control as much as it builds security awareness. Actual implementation requires either applying a patch for a given service (if it is even available) or limiting access to such a service. For the latter, the unique accessibility of Log4j as a logging framework frequently accepting external input for awareness purposes makes limiting attack surface extremely difficult, if not outright impossible in many instances. The complexity of its use and embedding within software products and services make the former daunting and mean mitigating patches may not be available for weeks, if not months.

Defenders should certainly expend efforts to understand what services their organization exposes to external entities and how these might be accessed, but in cases like the Log4j issue this represents an intermediate step and not a solution to the problem. Attack surface identification can be used to vector defensive resources, but until effective patches become available for multiple services, ranging from end-user applications to server-side software to various IOT and ICS products, reducing this surface will be extremely difficult for the foreseeable future.

Detection and Defensive Opportunities

Multiple, seemingly obvious defensive approaches to CVE-2021-44228 feature various pitfalls and limitations that will leave organizations vulnerable to its exploitation until all vulnerable services are fully patched. While no “complete” solution exists at this time, multiple strategies can be employed to reduce the effectiveness of exploitation activity, or to identify its use and application after success. This defense in depth approach, while not completely removing the threat, can be usefully implemented to improve the security posture of organizations until more complete solutions (patches) become available.

Limiting Exposure and Outbound Communication Channels

First, and linked to attack surface mapping and reduction, organizations do retain some control over what is allowed into networks, and what communication can leave them. While complete management may remain impossible without essentially isolating a network from the broader internet, defenders and their organizations retain freedom of movement in limiting the scope and degree of what inbound and outbound communications are possible.

By first performing or having adequate attack surface management information, organizations can then move to limit certain likely attack vectors. For example, many implementations of CVE-2021-44228 activity require retrieving an object via LDAP. While

redirections or other mechanisms of retrieving Java Naming and Directory Interface (JNDI) objects exist (e.g., DNS and Remote Method Invocation mechanisms have also been identified), doing so via LDAP remains a common and direct mechanism for doing so. Limiting or monitoring LDAP communications leaving the monitored network can identify suspicious instances of this protocol's use, which through analytic approaches can be linked to attempts to communicate with Log4j services should sufficient visibility and processing capabilities exist.

Furthermore, sanitization and examination of inputs to external facing processes and functions, while seemingly simple advice, can go a long way to reduce fuzzing and evasion opportunities associated with this activity. Although almost certainly not capable of completely removing possibilities of exploitation, such techniques can reduce such instances and when applied in conjunction with other controls produce a more manageable defensive posture. Unfortunately, such approaches are seldom achievable as an after-action security measure, but rather require focused attempts at the application development stage to implement before content is parsed or decoded for ingestion and follow-on execution or use.

Identifying Post-Exploitation Activities

Once defenders manage attack surface and limit it as best as possible, subsequent efforts can focus on identifying adversary post-exploitation behaviors. Simply exploiting CVE-2021-44228 is insufficient to achieve adversary goals – rather, additional functionality is required: installing a cryptominer, deploying a remote access tool (RAT), or exfiltrating information. At this stage, NSM becomes a vital component in monitoring for and identifying such behaviors.

Furthermore, such approaches are not uniquely relevant for Log4j subversion, but apply to a multitude of initial access vectors such as other exploits or code execution vectors. By implementing robust defenses and detections around post-exploitation behaviors, such as lateral movement and command and control (C2) activity, security personnel achieve defense in depth vis a vis multiple, potential threats.

As part of this process, organizations must ensure they possess and are actively monitoring detections and alerts for post-exploitation activity such as C2 behaviors or indications of lateral movement. While in these instances the adversary has already gained access to the defended environment, defenders can ensure more rapid and focused response to mitigate and remove the intruder more rapidly, limiting the extent of the incident.

Profiling Environments to Identify Alterations

Finally, simply understanding one's environment and knowing what "normal" looks like can help determine when matters go awry. While Log4j is widespread in application and use, its primary vectors of concern for enterprise environments are server-side applications running this process and making it available indirectly via logging. Understanding *which* servers and

applications are involved are an aspect of earlier discussions around attack surface mapping. Determining *how* these assets function and communicate represents a further step in profiling the operating environment.

By having a means to profile and understand network traffic behaviors over time in monitored environments, defenders become armed with ways to detect deviations. For example, seeing anomalous external network connections from an application or logging server can be a telltale sign of potentially malicious activity, such as active C2 or a reverse shell, and trigger further investigation.

Questions like “what is normal” are difficult to answer in the middle of a security incident or event of concern without prior knowledge, though. Defenders must therefore work to build this level of insight, understanding, and recognition in advance and overtime to enable future identification of anomalous or suspicious modifications to asset behavior.

Conclusion

The Log4j vulnerability, under active and pervasive exploitation, will likely remain a significant issue for security teams for months if not years to come as software dependencies are unearthed and patches slowly released. While patching the vulnerability remains the most effective and assured way of eliminating this as a potential attack vector, defenders are not helpless in the interim. Multiple approaches exist to address exploitation and follow-on adversary activity to ensure continued visibility into and follow-on response to intrusions. However, network defenders must do so adequately understanding the capabilities and limitations afforded by such mechanisms, as described above, and apply multiple approaches to achieve a true defense in depth posture. In such fashion, network security personnel can ensure not just robust defense against items such as CVE-2021-44228, but against similar events emerging in the future.

Recommendations for Gigamon Customers

Gigamon continues to scope potential impact scenarios from CVE-2021-44228 in internal products and services. Gigamon customers can refer to community portal documentation for the latest news and updates on Log4j impacts and patching information. Gigamon ThreatINSIGHT customers can find suggested queries and detection strategies to identify vulnerable software through the notification alert in the web portal.

CONTINUE THE DISCUSSION

People are talking about this in the Gigamon Community's Security group.

Share your thoughts today

RELATED CONTENT

REPORT

2022 Ransomware Defense Report

GET YOUR COPY >

WEBINAR

ThreatINSIGHT: Eliminating Adversaries' Dwell Time Advantage

WATCH ON DEMAND >

WEBINAR

Deep Dive INSIGHTS: Fighting Ransomware and Shifting Security Priorities

WATCH ON DEMAND >

REPORT

Gigamon ThreatINSIGHT Guided-SaaS Network Detection and Response

GET YOUR COPY >

OLDER ARTICLE
'Tap'ping the Myths of Cloud Migration
NEWER ARTICLE
The Control Plane's Evolving 5G Role in Enabling Visibility of the Future

TOP