Article

Not peer-reviewed version

Zero-Day Exploits in Cybersecurity: Case Studies and Countermeasure

Azheen Waheed , Bhavish Seegolam , Mohammad Faizaan Jowaheer , Chloe Lai Xin Sze , Ethan Teo Feng Hua , Siva Raja Sindiramutty *

Posted Date: 29 July 2024

doi: 10.20944/preprints202407.2338.v1

Keywords: Zero-day exploits; Cybersecurity; HAFNIUM; Log4j vulnerability; Malware defense

Article

Zero-Day Exploits in Cybersecurity: Case Studies and Countermeasure

Azheen Waheed, Bhavish Seegolam, Mohammad Faizaan Jowaheer, Chloe Lai Xin Sze, Ethan Teo Feng Hua and Siva Raja Sindiramutty

School of Computer Science Taylor's University Subang Jaya; azheen.waheed@gmail.com, bhavishseegolam23@gmail.com, jowaheerfaizaan@gmail.com, chloelxs03@gmail.com, ethanteo2004@gmail.com, siva.sindiramutty@taylors.edu.my

Abstract: Zero-day threats are a more severe and constantly developing menace to various participants including large companies, government offices, and educational establishments. These entities may contain valuable information and essential operations that attract cyber attackers. These exploits are especially devastating as they target weaknesses that an organization's vendors are not even aware of, making them have no protection against them. This paper focuses on the background and use of zero-day exploitation and the structure and technologies of these complex malware attacks. We examine two notable real-life cases: the case of 'HAFNIUM targeting Exchange Servers with zero-day exploits' that was investigated by Microsoft 365 Security and Microsoft Threat Intelligence, and the 'Log4j vulnerability' case that was reported by the National Cyber Security Centre. These cases show the critical effects of zero-day vulnerabilities and measures taken to combat them. Additionally, this paper outlines different strategies that can be used to prevent zero-day attacks with the help of modern technologies. These are fast patch release, effective IDS/IPS, and a security model that involves constant vigilance and use of behavioral analytics. Thus, by studying the lifecycle of zero-day vulnerabilities and the structure of the exploits, one can enhance the protection of the organization against the threats that are invisible to traditional security systems. This extensive survey is designed to be useful in understanding the characteristics of zero-day vulnerabilities, the technologies for their mitigation, and the constant threat and development in the field of cybersecurity. Thus, it is possible to strengthen the protection against these threats that are constant and develop with time by analyzing the previous events and predicting potential problems.

Keywords: zero-day exploits; cybersecurity; HAFNIUM; Log4j vulnerability; Malware defense

Introduction

A zero-day exploit refers to a type of cyberattack vector that leverages an undiscovered or unfixed security vulnerability in computer hardware, software, or firmware (Adepu et al., 2019; Gill et al., 2022). The term "zero-day" describes a software or device flaw wherein the vendor has no time to address it because malicious actors can use it to gain access to systems that are already vulnerable (Shandilya et al., 2024). This paper aims to provide a detailed understanding of zero-day exploits and their lifecycle, analyze real-life case studies to highlight the impact of zero-day vulnerabilities, discuss strategies and technologies for mitigating zero-day attacks and emphasize the importance of continuous vigilance and proactive security measures in cybersecurity.

Zero-day is frequently paired with the terms vulnerability, exploit, and attack. It is useful to know the distinctions between them (Kaspersky, 2018; Azam et al., 2023, Ali, I, 2022, Ashfaq. Et al.,): A zero-day vulnerability is a flaw in software that is found by attackers before the vendor is made aware of it. Attacks using zero-day vulnerabilities are likely to be successful because there is no patch available due to the vendors' ignorance. The technique hackers employ to target systems with an undiscovered vulnerability is known as a zero-day exploit. Using a zero-day exploit to harm or steal data from a system that has a vulnerability is known as a zero-day attack (Touré et al., 2024). Zero-day vulnerabilities usually affect big businesses, government agencies, hardware, firmware, Internet of Things, and users with access to sensitive corporate information, among other targets

 doi:10.20944/preprints202407.2338.v1

(GeeksforGeeks, 2020; Singh, Buyya and Kim, 2024). Zero-day vulnerabilities are free points of entry that cybercriminals can use to compromise any target they choose. These vulnerabilities can be found in widely used software such as Adobe Reader and Microsoft Office. As a result, a newly discovered unpatched vulnerability can fetch a market value of $5000- $250,000 (Bompos, 2020).

Example: A highly intelligent computer worm known as Stuxnet took advantage of four distinct zero-day software flaws in Microsoft Windows operating systems (Hurst and Shone, 2024; Gouda et al., 2022). Iran's nuclear facilities were the target of several attacks in 2010 using Stuxnet. After infiltrating the computer networks of a nuclear power facility, it sent malicious commands to the centrifuges that were being used to enrich uranium. These instructions spun the centrifuges so quickly that they malfunctioned, and all in all, Stuxnet damaged 1,000 centrifuges (www.ibm.com, n.d.).

Lifecycle Of a Zero-Day Vulnerability

As seen in Figure 1, there are seven stages in the life cycle of a zero-day vulnerability. Usually, a zero-day vulnerability originates as a software bug. Attackers may be able to take advantage of the bug if the software vendor fails to identify it. A security issue is typically released to the public upon discovery of a vulnerability by software vendors (Fidler, 2024; Fatima-Tuz-Zahra et al., 2020). Subsequently, the vendors release a patch for the zero-day vulnerability and update anti-virus signatures to reflect the updated knowledge of zero-day attacks.

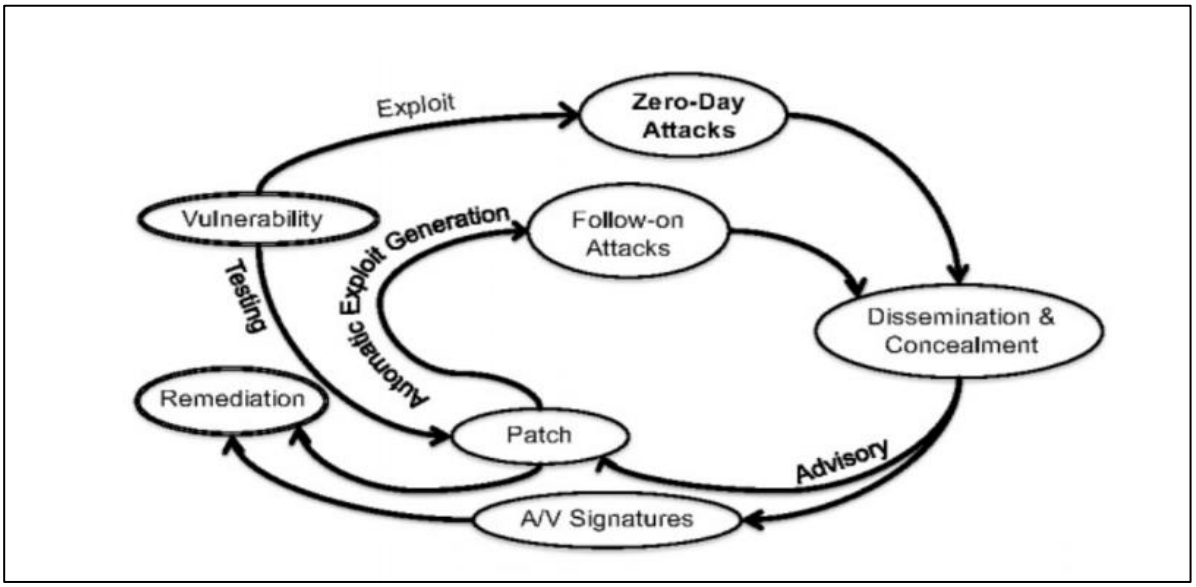

Figure 1. Zero-day Vulnerability Life Cycle (Bilge and Dumitraş, 2012).

Even after a vendor has identified and fixed a vulnerability, new exploits may still be created and used against targets that have not yet installed the patch. It may take several years for remediation to occur after this cycle of patching and exploiting, at which point the vulnerability stops impacting systems. The life cycle of a zero-day vulnerability can also be marked in a timeline as shown in Figure 2 (Bompos, 2020; Gopi et al., 2021).

 doi:10.20944/preprints202407.2338.v1

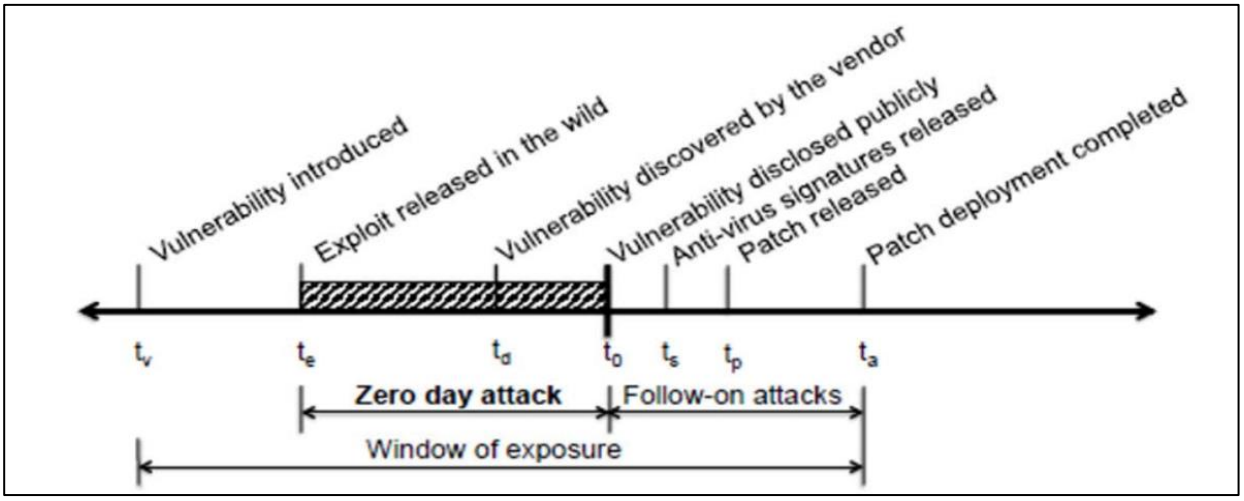

Figure 2. Attack Timeline (Bilge and Dumitraş, 2012).

Architecture and Technologies of Zero-Day Exploit

A zero-day exploit is achieved by deploying a malicious code – worm, inside the exploitation process that spreads and searches for vulnerabilities within a system or network (Akello, 2024). It replicates itself to spread through the network massively to identify all the existing vulnerabilities.

According to Pratama and Adi Rafrastara (2012), the structure of the worm includes infection propagation, remote control and update interface, life-cycle manager, payload and self-tracking. While these are the main functions in the structure of the worm, it is important to be aware that in a worm architecture, there is also an exploitation and defence system (Goni, 2020; Humayun et al., 2022) that would form an attack or hide its track from detection. The basic components in the architecture of the worm will be as follows:

Self-Tracking and Reporting
Defense System
Payload Execution
Life-Cycle Manager
Remote Control and Update Interface
Infection Propagation
Target Locator

Figure 3. Architecture of Worm.

The first two layers of the worm architecture include the target locator and infection propagation. These two layers are built to target the vulnerabilities that are within the system and networks (Wan et al., 2012; Alkinani et al., 2021, M. Lim, at.al., 2019, M Ibrahim, et.al., 2021). The target locator is the start of the exploitation where the worm is spread through the system or network to search for vulnerabilities or targets. Once the target is found the worm will replicate themselves onto it. Then, in the infection propagation layer. This component allows the worm to transfer itself to a new node that is compatible to get control of the remote system (Wang et al., 2014; Azam, Tajwar,

 doi:10.20944/preprints202407.2338.v1

et al., 2023, Sharifonnasabi, 2022, et.al., Sindiramutty, S. R. et al.,). The worm that is transferred into a new node will trick the victim into executing it based on social engineering techniques.

The next layer is the remote control and update interface. Remote control allows the hacker to control the worm network (the worm and their replicates). The hacker could send control messages to the copies of the worm which is threatening as the attacker could attach tools such as DDoS onto the worm and form an attack on the system. As for the update interface, it is a component that allows the worm to update its code and form an exploit onto the compromised system. This would result in an outbreak as it might cause the system to be introduced with more exploits and vulnerabilities after the intrusion of the worm. The life-cycle manager is a component that patches the worm to give it a life-cycle. The attacker would want their malware to only run for a period, so building a life-cycle manager will let the attackers have their worm "commit suicide" when the time has reached or continue running it endlessly in the system.

Another layer in the architecture of the computer worm is the payload execution. This layer allows the worm to carry and execute the payload based on the attacker's objective. The type of payload will determine how the worm will be designed. This layer is crucial for the attackers as the payload execution has the technology to impact the system.

The last two layers in the architecture are the defence system and the self-tracking and reporting layer (Xu and Meng, 2023; Azam, Dulloo, Majeed, Wan, Xin and Sindiramutty, 2023, Ray, S.K, 2015, et. al., Ray, S.K, 2009, et. al.,). This layer is built to cover up the tracks that were left behind by the worm and to gain back information related to the system that was attacked. The defensive layer uses technologies such as encryption to hide its track from the detection of the system's security system. This will allow the worm to stay longer inside the system to further exploit the system. To let the attackers gain insights into the condition of the system, the worm will send back reports through the ways that were decided by the attackers when designing the self-tracking and reporting layer. With this, the attackers could track the progress of the worm inside the system.

Technologies and Phases of Zero-Day Exploit

Originally, zero-day exploits occurred due to the intention of searching for vulnerabilities in software to be exploited. Hackers' attempts to detect any possible vulnerabilities in a system result in the organization being in a vulnerable state for a long period. The zero-day exploitation could be divided into two main phases, which are the discovery phase and the exploitation phase (Riofrío et al., 2021; Chesti et al., 2020, Zerdoumi, 2022, et.al.).

During the discovery phase, there are three steps. Firstly, hackers recognise and audit the vulnerability that they discover with the help of technologies such as fuzzers where they inject random data into the software to detect bugs Fadolalkarim and Fadolalkarim, 2024; Wen et al., 2023), reverse engineering with tools such as disassemblers to disassemble and analyse the binary codes, and binary analysis by understanding the binary code without accessing the source code of the system with a framework such as Radare2. This step is often done, to perform the search for vulnerabilities and find errors. The next step will be triage, where it is traced back to the crash to determine the root cause in the code that is vulnerable to use as an advantage for the hackers to exploit. The final step in the discovery phase is the trigger. It is the step that hackers create a reliable trigger to exploit the vulnerabilities (Ablon and Bogart, n.d.). During this phase, hackers might deploy a worm into the system to gain more information about the vulnerabilities that could be used for exploitation later on.

The exploitation phase consists of the following steps: debug, exploit and deploy. Debugging is the step where the hackers evaluate and refine the exploits to ensure their effectiveness and the functionality of them. Tools such as OllyDbg or WinDbg (specifically works for the Windows operating system), are debuggers that could evaluate and debug exploits. Here, the worm could be utilised by the attackers to further spread the worm across the systems to ensure exploitation is made in various places. The next step, exploit, is the step after the effective and accurate method is determined, hackers would have to run a test for it to review its effectiveness and the Proof of Concept (PoC) of the exploitation. To prevent any discovery by the system due to the tracks and footprint that

 doi:10.20944/preprints202407.2338.v1

were left behind, hackers will cover their tracks after any testing (Riofrío et al., 2021) with BleachBit or CCleaner that could provide functions such as system cleaning and browser cleaning (Shahine, 2023; Alferidah and Jhanjhi, 2020). Technologies such as virtualization are used to test the exploits safely in an environment, due to the ability to isolate the test environment from the host system. Lastly came the step deploy, where the hackers deploy the exploitation into the real system or sell it to the black market.

The figure below which was created by Lillian Ablon and Andy Bogart shows the zero-day exploit life cycle:

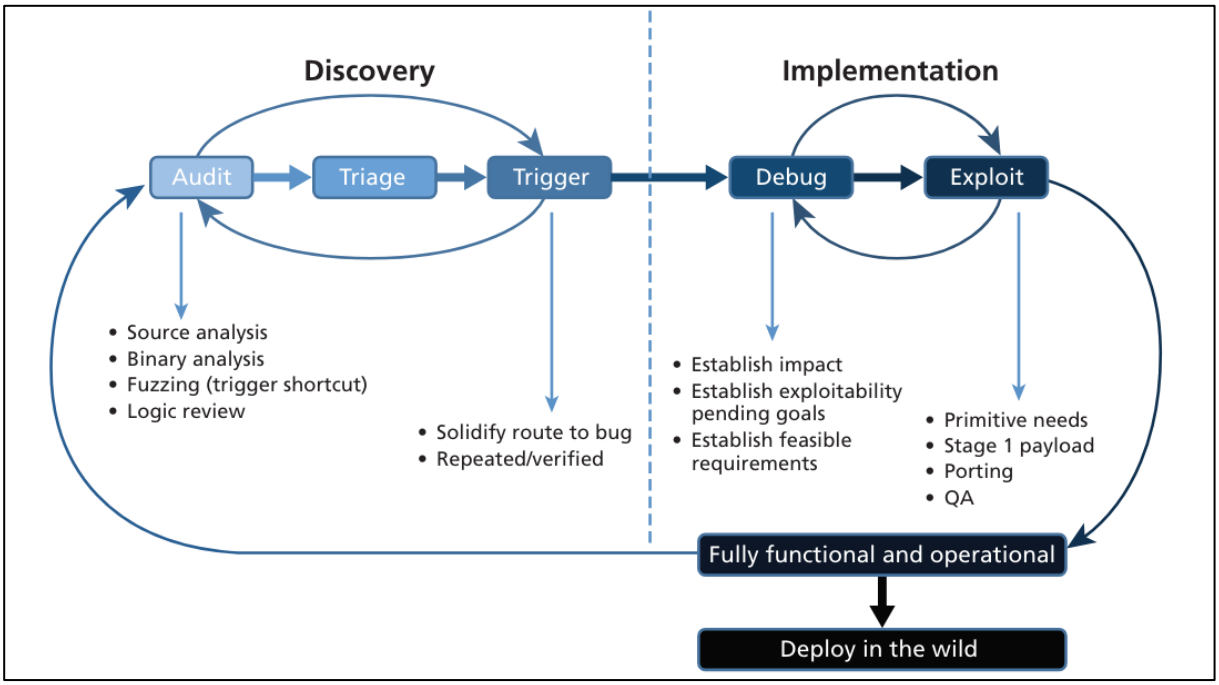

Figure 4. The Phase of Exploit Development. (Ablon and Bogart, n.d.).

Discussion on Security Issues and Countermeasures

Case 1: HAFNIUM targeting Exchange Servers with 0-Day Exploits

Case Overview

In an era where digital communications are crucial to business operations, cybersecurity is really important to maintain organizational integrity. Microsoft Exchange, one of the most widely used communications platforms, serves as an important pillar in many organizations. In early January 2021, suspicious activities were detected on Microsoft Exchange servers, which after investigation, turned out to be zero-day exploits. Based on observed victimology, tactics and procedures, Microsoft attributed this attack to HAFNIUM, a group assessed to be state-sponsored and operating out of China (Security and Intelligence, 2023).

The HAFNIUM attack compromised a huge amount of organizations globally, affecting sectors as diverse as infectious disease researchers, law firms, higher education institutions, and defense contractors. The breach allowed attackers to access sensitive communications, exfiltrate data, and install additional malware for long-term exploitation. By March 5 2021, the estimated number of affected organizations was more than 30,000 in the US alone and hundreds of thousands globally (Grigutytė and Grigutytė, 2024; Kumar et al., 2021).

There were several zero-day vulnerabilities associated with this incident. These kinds of vulnerabilities are highly desirable for attackers due to their effectiveness and stealth since they allow for undetected intrusions into critical systems. In response to the escalating threat, Microsoft swiftly released patches to mitigate these vulnerabilities and issued detailed guides to help organizations defend against ongoing attacks.

 doi:10.20944/preprints202407.2338.v1

Security Issues

Table 1. Vulnerability types for case 1.

CVE	Vulnerability Type
CVE-2021-26855	Server-Side Request Forgery (SSRF)
CVE-2021-26857	Insecure Deserialization
CVE-2021-26858	Arbitrary File Write
CVE-2021-27065	Arbitrary File Write

Microsoft released out-of-band advisories on March 2 in response to the above four zero-day vulnerabilities in Microsoft Exchange Server that have been actively exploited.

CVE-2021-26855

It is a server-side request forgery (SSRF) vulnerability in Microsoft Exchange. This vulnerability allowed the hackers to send arbitrary HTTP requests and authenticate as the Exchange server. Microsoft states that the Exchange Server that is vulnerable must have the ability of accepting untrusted connections via port 443. This vulnerability can be exploited remotely and doesn't require any type of authentication, specific knowledge, or access to the target environment. All that the attacker needs to know is the IP address or fully qualified domain name (FQDN) of an Exchange Server and the email account they wish to target (Adair, 2021).

CVE-2021-26857

It is an insecure deserialization vulnerability in Microsoft Exchange. The Exchange Unified Messaging Service, which offers voicemail capability along with other services, is specifically where the problem lies. An attacker would need to first exploit another vulnerability or authenticate with administrator privileges to the vulnerable Exchange Server in order to take advantage of this vulnerability. If the exploitation is successful, the attacker will be able to execute any code as SYSTEM (Security and Intelligence, 2023b).

CVE-2021-26858

It is an Exchange arbitrary file write vulnerability that occurs post-authentication. HAFNIUM might leverage this vulnerability to write a file to any path on the server if they were able to authenticate with the Exchange server. They might possess stolen administrator credentials or use the CVE-2021-26855 SSRF vulnerability to authenticate. This would allow them to achieve remote code execution (RCE) (Security and Intelligence, 2023b).

CVE-2021-27065

This vulnerability is also similar to CVE-2021-26858, in the sense that it also requires authentication through exploiting CVE-2021-26855 SSRF vulnerability or by using stolen legitimate administrator credentials. Consequently, they would achieve remote code execution (RCE), allowing them to write files to any paths on the server (Security and Intelligence, 2023b).

Once HAFNIUM operators had gained initial access by taking advantage of these vulnerabilities, they deployed web shells on the hacked server. Web shells give hackers the ability to steal information and carry out other malicious deeds that compromise systems even more. Reportedly, the attackers were also able to obtain the offline address book (OAB) for Exchange. It would be useful for a determined threat actor to have this information in order to conduct additional target reconnaissance (Narang, 2021).

Grigutytė and Grigutytė (2024b) also explains that Microsoft revealed a new type of attack that takes advantage of the vulnerabilities on March 12, 2021. The infected systems were being used by hackers to spread the ransomware known as DearCry. The ransomware that infected the target and demanded a $16,000 ransom payment.

 doi:10.20944/preprints202407.2338.v1

Threats Posed by Security Issues

Data Breach

Exploiting these Microsoft Exchange vulnerabilities could lead to significant data breaches. They could allow unauthorized access to sensitive data (Robinson, 2024b; Sindiramutty et al., 2024). This could include personal information, corporate secrets, financial details, and other confidential data stored in emails and associated attachments. The risk is even more serious since Exchange plays an integral role in organizational communication. It often houses a vast array of sensitive information. If attackers gain access, they could potentially leak this data. Such a scenario could lead to financial loss, reputational damage, and legal repercussions for failure to protect user data.

Installation of Malware

Once the attackers exploit vulnerabilities such as CVE-2021-26855 to gain initial access, they can install various types of malware. These malwares could serve multiple malicious purposes (Grigutytė and Grigutytė, 2024c). It might be used to spy on user activities. It could also corrupt data or even use the compromised system as a base for further attacks. This situation could also lead to the establishment of a persistent presence within the network. This would allow attackers ongoing access to the network and its resources. This presence can be difficult to get rid of and might lead to ongoing issues with data integrity and system stability.

Ransomware Attacks

After the Microsoft Exchange zero-day exploits, it was reported that there were cases of ransomware being deployed. Such an example was the DearCry ransomware (Grigutytė and Grigutytė, 2024c). This type of attack can be devastating. It not only prevents users from accessing critical data but also forces the hand of the organization to pay a ransom to restore access to their own information. All this with no guarantee that the attackers will honour the terms once payment is made.

Deeper Network Infection (Lateral Movement Within Networks)

Attackers might try to move deeper into a network after gaining initial access. Regarding the impact, once the attackers gain control of an Exchange server via CVE-2021-26855, it would be possible for them to use it to attack other systems within the network. This could include gaining access to administrative credentials and escalating their privileges. They could then use those privileges to move to other critical and sensitive systems. Lateral movement is dangerous because it allows attackers to extend their reach within the network. Under such conditions, security risks are higher and an attacker could cause a much broader impact and become invisible.

Case 2: Log4Shell: An Arbitrary Code Execution Zero-Day in Log4j

Case Overview

Log4j is an open-source logging library for the Java programming language, part of the Apache Logging Services project maintained by the Apache Software Foundation. It enables developers to log messages from their applications in a consistent and flexible manner, facilitating debugging, monitoring, and auditing. The library is widely used in the industry, from embedded applications and network appliances to software by vendors such as Aruba Networks, Fortigate, Cisco, and Sophos. (Cybersecurity and Infrastructure Security Agency, 2022) Additionally, it is integrated into commercial products by vendors like Amazon AWS (Amazon, 2021) and Cloudflare (Cloudflare, 2021), and also in Java-based games such as Minecraft (Minecraft, 2021)

On November 24, 2021, a remote code execution (RCE) vulnerability in the library was privately disclosed to the Apache Software Foundation by Chen Zhaojun, a security researcher at Alibaba Cloud. It was discovered that this vulnerability had existed in the library undetected since 2013. Before an official CVE identifier was assigned on December 9, 2021, the vulnerability was publicly released on Twitter along with a working proof of concept on GitHub. This proof of concept demonstrated how an attacker could execute arbitrary code on a server by exploiting improper handling of log messages that allowed user input to be logged. Security researchers from Nguyen et al. (2023) tracked this disclosure and coined the term "Log4Shell." The CVE number CVE-2021-44228

 doi:10.20944/preprints202407.2338.v1

was assigned to this vulnerability, and it received the highest available CVSS score of 10 from Apache due to the nature of the exploit. (Nguyen *et al.*, 2023; Hussain et al., 2024)

The Log4Shell exploit took advantage of a flaw in the Log4j library's handling of log messages. Specifically, it exploited the library's ability to interpret and process user-provided input within log entries. By crafting a malicious log message containing a JNDI (Java Naming and Directory Interface) lookup string, an attacker could trick Log4j into fetching and executing code from a remote server. This flaw allowed attackers to perform remote code execution (RCE) on any system using an affected version of Log4j, resulting in significant security risks.

Once the vulnerability was disclosed, a proof of concept was released, demonstrating how simple it was for an attacker to exploit the issue. The exploit involved sending a specially crafted request to an application that logs user input using Log4j. When the application processed this log message, Log4j would initiate a JNDI lookup, leading to the retrieval and execution of malicious code from an attacker-controlled server. This process effectively gave the attacker control over the compromised system, enabling them to perform a wide range of malicious activities, such as stealing data, installing malware, or disrupting services (Nathaniel, 2022; Elmo, 2023).

Java's widespread use in the business world is widely established, with 90% of Fortune 500 organizations using it in some capacity, according to Oracle. Checkpoint found that 40% of business networks were vulnerable to the Log4Shell attack. Cloudflare confirmed that this vulnerability had been exploited in the wild as early as December 1 and 2, 2021, nine days before it was publicly disclosed (Spafford, Metcalf and Dykstra, 2023). Within 72 hours of the vulnerability being publicly reported, over a million attack attempts were attempted, with various variants of the exploit emerging, the majority of which originated in the United States (Qualys, 2022)

Security Issues

CVE-2021-44228

The attacker begins by crafting a malicious string that includes a JNDI lookup. For instance, this string might look like ${jndi:ldap://somedomain.xyz/endpoint}. The key part of this string is the jndi:ldap:// URL, which directs the logging framework to initiate a lookup via the Lightweight Directory Access Protocol (LDAP). The malicious string is then delivered to a vulnerable application in any user input that gets logged by Log4j. This can be through various methods such as HTTP headers, form submissions, or any other means where user input is recorded by the server's logging infrastructure. For example, an attacker might send an HTTP request with the malicious string in the User-Agent header. Once the application logs the malicious input using Log4j, the logging framework interprets the string and triggers the JNDI lookup due to the ${jndi:ldap://...} part of the log message. The JNDI lookup reaches out to the specified LDAP server, which in this case is controlled by the attacker. The attacker's LDAP server responds with a reference to a Java class that the target server should load and execute. The vulnerable server then loads and executes the attacker's Java class, granting the attacker the ability to execute arbitrary code on the server. (Fleury and Reverbel, 2003)

Threats Posed by Security Issues

Remote Code Execution (RCE)

Remote Code Execution (RCE) is a type of security vulnerability that allows an attacker to execute arbitrary code on a remote system. This capability is particularly dangerous because it enables the attacker to run any command or code of their choice on the compromised system, leading to full system compromise, opening the door to a huge list of nefarious activities an attacker could do (Microsoft, 2021)

Botnets

Attackers used the Log4Shell exploit to recruit compromised systems into botnets. These botnets could then be used to launch coordinated attacks, such as Distributed Denial of Service (DDoS) attacks, against other targets (Tamayo, López and Caraguay, 2024). For example, the Mirai botnet,

 doi:10.20944/preprints202407.2338.v1

which has historically been used to launch massive DDoS attacks, saw new variants incorporating Log4Shell exploit capabilities (Pauley, Barford and McDaniel, 2023b).

Ransomware

Prominent ransomware groups such LockBit Conti (Broadcom, 2021) quickly included Log4Shell into their attack strategies. Bypassing conventional security mechanisms, these parties took use of the weakness to first access target networks. Once inside, they moved laterally throughout the affected environment, collecting private information and encrypting it. The encryption process made critical information and systems unreachable to the victims. Then attackers sought decryption keys in exchange for ransom payments—often in bitcoin. By greatly accelerating the speed and scope of attacks, Log4Shell's incorporation into ransomware operations let ransomware operators target a wide range of businesses in many different sectors.

As per Broadcom and Trend Micro, the quick modification of ransomware techniques to incorporate Log4Shell exploitation exposed the developing complexity of attackers. These teams showed a great capacity to maximize their influence by using recently found zero-days. Log4j is a major problem for cybersecurity teams all around since its broad usage increases the possible damage in many different apps and services. Organizations had to take quick and comprehensive measures including system patching, network segmentation, and monitoring and detection capability improvement to lower the danger Log4Shell and subsequent ransomware attacks presented.

Espionage

The Log4Shell vulnerability was exploited by advanced persistent threat (APT) companies and state actors for espionage (Möller, 2023; Azeem et al., 2021). To gather intelligence, these groups targeted government institutions, large enterprises, and key infrastructure, stealing sensitive information. For example, the Hafnium group, which is believed to be linked to the Chinese government, was believed to have targeted key businesses exploiting the vulnerability (Microsoft, 2021; Sindiramutty, 2024). These individuals gained unauthorized access to private data via Log4Shell, allowing them to monitor and assemble valuable information on political, economical, and strategic issues. Nation-state actors employing Log4Shell demonstrated the serious implications of zero-day vulnerabilities for national security. These groups often remained present in compromised networks, and meticulously exploited the weakness to infiltrate systems undetected.

Discussion on Security Countermeasures

Case 1: HAFNIUM Targeting Exchange Servers with 0-Day Exploits

Countermeasures Taken

Rapid Release of Patches

Microsoft released emergency patches for the vulnerabilities to close the security gaps. This was crucial to stop the immediate exploitation of these vulnerabilities across affected systems globally. They did not waste time in rolling out emergency security updates in March for Exchange Server versions 2013, 2016, and 2019 to address the vulnerabilities exploited by HAFNIUM. Furthermore, later in the year Microsoft released security updates (Rhodes and Bettany, 2016). They recommended that all updates should be installed via an elevated command prompt to avoid installation issues. Moreover, they also provided extensive guidance on troubleshooting potential problems post-update. Microsoft also provided tools and scripts, such as the Microsoft Support Emergency Response Tool (MSERT) to help organizations determine if they had been compromised prior to patching.

Detailed Guidance and Tools

Microsoft made sure to provide detailed guidance and tools for detecting and mitigating any potential threats from these vulnerabilities. They provided well detailed scripts to check for signs of compromise. Tools like the Microsoft Safety Scanner were provided to remove known threats, and the Exchange On-premises Mitigation Tool (EOMT) helped in automating some of the mitigation processes.

On the Microsoft Security blog, the Exchange Server team provided a script to scan Exchange log files for indicators of compromise (Security and Intelligence, 2023b).

 doi:10.20944/preprints202407.2338.v1

- CVE-2021-26855 exploitation can be detected via some Exchange HttpProxy logs detailed in the script.
- CVE-2021-26858 exploitation can be detected via the Exchange log files detailed in the script.
- CVE-2021-26857 exploitation can be detected via the Windows Application event logs.
- CVE-2021-27065 exploitation can be detected via the detailed Exchange log files.

Isolation of Exchange Server

To reduce the risk of exploitation of the vulnerabilities, Microsoft explained that the Exchange Server can be isolated from the public internet by blocking inbound connections over port 443. This could reduce chances of exploitation until patches were applied (Msrc, 2021b).

The counter measures that were employed were quite effective and very methodological in that besides seeking to eliminate exploitation of such vulnerabilities in the future, preventive efforts were made to ensure that such vulnerabilities will not pose a threat in the future. The fast development and distribution of patches along with comprehensive instructions and resources for managing the risk are clear signs of a well-orchestrated effort addressing a major threat.

But the incident also shows that presence and organization cannot be a one-time exercise followed by business-as-usual; organizations have to remain vigilant constantly and adhere to best practices such as timely patching, threat intelligence, and implementing defense-in-depth or multiple-layered security as a mechanism to counter new-age cyber threats. Organizational cybersecurity measures also need continuous assessment and upgrades, which is evident when such significant weaknesses were identified in widely used software.

Countermeasures Proposed

There are several countermeasures that Microsoft could have taken regarding the four exploited vulnerabilities.

CVE-2021-26855: To defend against vulnerabilities like this, which exploited port 443 on the Microsoft Exchange servers, organizations can adopt several effective countermeasures. First, implementing strict network segmentation ensures that Exchange servers are isolated to specific network segments or VLANs, limiting access to only necessary systems. Coupled with firewall rules configured to permit traffic on port 443 exclusively from trusted IP addresses or ranges, this mitigates the risk of unauthorized access attempts. Additionally, requiring all remote access through a VPN establishes encrypted tunnels for authenticated users, preventing direct exposure of Exchange web services to the internet and adding an extra layer of security through access control. These measures are specifically designed to secure access to Exchange services (e.g., Outlook Web Access, Exchange ActiveSync) over port 443, which is primarily used for HTTPS communication rather than SMTP (Simple Mail Transfer Protocol) traffic used for actual mail delivery, which by design needs access to the internet.

CVE-2021-26857: Use Intrusion Detection System (IDS) for organization-wide traffic auditing and to identify any exchange server activities that seem suspicious. Connecting these systems to Security Information and Event Management (SIEM) solutions can help improve the detection of threats by providing a consolidated view of security threats. It would be safer to conduct regular audits of activities performed by SYSTEM-level processes to ensure that only legitimate activities are carried out. Behavioural analysis tools could also be used to detect deviations from normal operations, therefore indicating potential misuse of high privileges.

CVE-2021-26858 and **CVE-2021-27065**: Implement network segmentation to isolate the servers that are connected to the internet from the rest of the internal network. This is done to limit the exposure of critical systems and data to external threats. Firewalls and gateways should be configured to strictly control incoming and outgoing traffic to these servers. Furthermore, strict file system ACLs should be implemented to limit reading and writing permissions to processes and users who absolutely need it.

Case 2: Log4Shell: An Arbitrary Code Execution Zero-Day in Log4j

Countermeasures Taken

 doi:10.20944/preprints202407.2338.v1

By the Apache Foundation

The Apache Foundation quickly released multiple updates to the Log4j library to address the vulnerability.

- **Log4j 2.15.0:** The initial fix disabled the JNDI lookups by default for log messages.
- **Log4j 2.16.0**: This version removed support for message lookups and limited JNDI functionality to only the localhost by default, to further reduce the attack surface.
- **Log4j 2.17.0 and 2.17.1:** Additional fixes and improvements were made to address related issues and ensure more comprehensive protection.

In addition to the quick successive updates to the library, Apache also provided detailed security advisories and documentation to guide users on how to mitigate the vulnerability. This included step-by-step instructions on how to update Log4j and additional configuration changes to enhance security. They worked closely with security researchers, industry partners, and the broader community to identify and patch related vulnerabilities. They encouraged open communication to quickly address any emerging issues (Riegler *et al.*, 2023).

By Individuals and Organizations

Organisations and individuals rapidly updated their Log4j dependencies to the latest secure versions released by Apache. For instance, large enterprises like IBM, Dell, AWS, and Microsoft issued urgent advisories to their customers to apply patches. As an immediate mitigation in situations where they might not be able to install the patches, the following advisories were made (Dell, 2021).

- Set the system property log4j2.formatMsgNoLookups to true to disable the message lookups function completely.
- Remove the JndiLookup class from the classpath by deleting it from the JAR file (e.g., using the zip command).

Organisations also implemented network-level protections to block exploit attempts. Web Application Firewall rules were updated, and users were advised to update their own to detect and block malicious JNDI strings in incoming requests.

Countermeasures Proposed

Zero-day exploits target undisclosed software vulnerabilities, posing a challenge as they circumvent established security protocols (Nair and Mhavan, 2023). Achieving absolute security in any software is unrealistic, but proactive measures can mitigate risks to some extent. The aftermath of Log4Shell highlighted the difficulty in patching a widely adopted library embedded in numerous software applications (Changsan and Boonyopakorn, 2023). Delays in deploying necessary patches and implementing mitigations left systems potentially vulnerable, a situation that still persists today. Below are suggested measures to lessen the impact of future exploits

Implementing a Zero Trust architecture

Implementing a Zero Trust architecture could have mitigated the Log4Shell vulnerability by focusing on granular access controls and strict verification of software components (Rains, 2020). In a Zero Trust model, each component, including third-party libraries like Log4j, would undergo rigorous vetting and continuous monitoring regardless of its origin or perceived trustworthiness. Access to sensitive systems or data would be strictly controlled based on user identity, device security posture, and behaviour analysis. For instance, even if a vulnerable version of Log4j were present in the network, Zero Trust principles would limit its access to only necessary functions and restrict communication to trusted resources, thereby containing potential exploitation attempts and minimising the impact of Log4Shell across the network. (He, Y., Huang, D., Chen, L., Ni, Y., & Ma, X., 2022)

Using unsupervised anomaly detection algorithms

Integrating unsupervised anomaly detection algorithms within Intrusion Detection Systems (IDS) would offer a proactive approach to identifying zero-day attacks, such as the Log4Shell exploit (Lyu, Gharakheili and Sivaraman, 2024; Shah, Jhanjhi and Laraib, 2022). These algorithms analyse

 doi:10.20944/preprints202407.2338.v1

patterns in log entries and network traffic to detect unusual behaviours indicative of exploitation attempts. By modelling normal system behaviour over time, these algorithms can identify deviations that may signify zero-day attacks, even when specific attack patterns are unknown or have not been previously observed (Saurabh et al., 2024). IDS equipped with unsupervised learning can dynamically adjust detection thresholds based on real-time data analysis, improving detection accuracy and reducing false positives. This approach would enhance the security by complementing signature-based detection methods and enabling early detection of new threats.

Security Audits of Widely Used Libraries

Performing rigorous security audits of widely used libraries, such as Log4j, is crucial to identifying and mitigating potential vulnerabilities (Badyal, 2024; Attaullah et al., 2022). Log4j, being extensively adopted across numerous software ecosystems and relied upon by countless enterprise service providers, underscores the critical need for thorough scrutiny. The discovery that the vulnerability in Log4j existed since 2013 highlights the importance of ongoing audits. Users and developers must allocate dedicated time and resources to systematically examine the security posture of these libraries. Such audits ensure proactive identification of security flaws, enabling prompt remediation and reducing the risk of exploitation. This proactive approach is essential to maintaining the integrity and security of software systems that rely on widely adopted libraries like Log4j.

Maintain an Inventory of Software Libraries

Maintain a comprehensive inventory of libraries utilised during software development and deployment. Certain users invested extensive time verifying their software stack, which often includes applications from third-party vendors, to ascertain the presence of the log4shell vulnerability. Conducting thorough audits of your system's components facilitates swift identification of potential vulnerabilities.

Automated Patch Management

Automated patch management plays a crucial role in mitigating the risk of zero-day exploits, as exemplified by incidents like Log4Shell. By automating the detection, deployment, and verification of software updates, organisations can significantly reduce the window of vulnerability between the discovery of a security flaw and its remediation. In the case of Log4Shell, delays in patching allowed malicious actors to exploit the vulnerability for days before widespread awareness and mitigation efforts were fully implemented. Automated patch management systems would have expedited the deployment of fixes across affected systems, potentially preventing exploitation and minimising the impact on organisations.

Conclusions

This review has thoroughly examined the challenges and responses associated with zero-day exploits using the significant cases of HAFNIUM's attacks on Microsoft Exchange servers and the Log4j vulnerability as focal points. These discussions have revealed critical insights into the nature of cyber threats and the evolving landscape of cybersecurity.

Key Takeaways

1. Rapid Patch Deployment: The immediate release and application of patches were crucial in countering the impacts of these exploits.
2. Advanced Detection Systems: Implementing sophisticated intrusion detection systems (IDS) can help in early identification of exploits.
3. Proactive Security Posture: Encouraging a proactive security approach through regular audits, constant monitoring, and the use of behavioral analytics is essential.
4. Network Segmentation and Access Controls: Reducing the amount of damage that may be caused by the exploit through network segmentation and strict access controls is vital.

Importance of Ongoing Vigilance

 doi:10.20944/preprints202407.2338.v1

The threats related to cyber space are dynamic in nature as the attackers adapt new ways of targeting. Organizations must maintain the most recent security implementation and continuously upgrade the cybersecurity team's knowledge. By learning from past incidents and anticipating future challenges, organizations can better prepare themselves against the inevitable next wave of zero-day exploits. This review has thoroughly examined the challenges and responses associated with zero-day exploits, using the significant cases of HAFNIUM's attacks on Microsoft Exchange servers and the Log4j vulnerability as focal points. These discussions have revealed critical insights into the nature of cyber threats and the evolving landscape of cybersecurity.

Findings

The HAFNIUM and Log4j cases are indicative of the sensitivity and ease with which digital systems are attacked through zero-day exploits that rely on previously uncovered vulnerabilities in the used software. Such incidents show that zero-day threats are as dangerous as any threat that can be utilised before the owning organization, or the security world can act on it. Scalability of such attacks has been highlighted recently by sources like Adair, 2021 or Microsoft, 2023: The former of the two sources states about a tens of thousands of worldwide servers affected by the HAFNIUM. The same applies to the Log4j vulnerability covered by various sources including and but not limited to LunaSec (2021).

Solutions

Rapid Patch Deployment: The immediate release and application of patches were crucial in countering the impacts of these exploits. Microsoft's response to the HAFNIUM attacks through swift patch rollout set a standard for how quickly organizations need to respond in order to deal with emerging threats, thus preventing widespread damage.

Advanced Detection Systems: Sophisticated intrusion detection systems (IDS) should be implemented. These can spot unusual network or file system activity can help in early identification of exploits before they cause significant harm. The integration of these systems with Security Information and Event Management (SIEM) solutions improves the overall security by the offering big picture visibility into threat information and allowing for the rapid response.

Proactive Security Posture: A proactive security approach should be adopted, encouraging harmony security approach and conducting frequent audits, constant monitoring, and the usage of behavioural analytics. These help in establishing a regular or normal range of organization activities and be able to identify abnormal conditions that might be indicative of a breach.

Network Segmentation and Access Controls: Reducing the amount of damage that may be caused by the exploit by choice of network perimeter and strict measures of access will prevent the attacker from accessing other areas of the network and hence limit the lateral movements.

Future Outlook: It is important to note that the threats related to cyber space are dynamic in nature as the attackers adapt new ways of targeting. So, organizations not only have to maintain the most recent security implementation in organizations but also must drive the organization's culture to upgrade the cybersecurity team's knowledge continuously. However, as the technology grows, and deep-learning techniques are integrated into cybersecurity approaches, the probability to prevent the unsuccessful usage of zero-day vulnerabilities might significantly grow.

Overall, this means that the battle against zero-day exploits is far from over, and that a variety of mechanisms has to be employed. In this regard, defence entails the orchestration of technology, the employment of highly trained personnel, and the proactively formulated policies. By learning from past incidents and anticipating future challenges, organizations can better prepare themselves against the inevitable next wave of zero-day exploits.

References

1. Ablon, L. and Bogart, A. (n.d.). Zero Days, Thousands of Nights The Life and Times of Zero-Day Vulnerabilities and Their Exploits. [online] Available at: https://web.archive.org/web/20180720224154id_/https://www.rand.org/content/dam/rand/pubs/research_reports/RR1700/RR1751/RAND_RR1751.pdf [Accessed 10 Jun. 2024].
2. Adair, S. (2021) Operation Exchange Marauder: Active exploitation of multiple Zero-Day Microsoft Exchange vulnerabilities.

3. Adepu, S. *et al.* (2019) 'Attacks on smart grid: power supply interruption and malicious power generation,' *International Journal of Information Security*, 19(2), pp. 189–211. https://doi.org/10.1007/s10207-019-00452-z.
4. Akello, N.B.O. (2024) 'Organizational information security threats: Status and challenges,' *World Journal of Advanced Engineering Technology and Sciences*, 11(1), pp. 148–162. https://doi.org/10.30574/wjaets.2024.11.1.0152.
5. Alferidah, D.K. and Jhanjhi, N. (2020) 'Cybersecurity Impact over Bigdata and IoT Growth,' *2020 International Conference on Computational Intelligence (ICCI)* [Preprint]. https://doi.org/10.1109/icci51257.2020.9247722.
6. Alkinani, M.H. *et al.* (2021) '5G and IoT Based Reporting and Accident Detection (RAD) System to Deliver First Aid Box Using Unmanned Aerial Vehicle,' *Sensors*, 21(20), p. 6905. https://doi.org/10.3390/s21206905.
7. Ali, I., Jhanjhi, N. Z., Amsaad, F., & Razaque, A. (2022). The role of Cutting-Edge Technologies in Industry 4.0. *Chapman and Hall/CRC eBooks*, *10*, 9781003203087-4.
8. Ashfaq F, Ghoniem RM, Jhanjhi NZ, Khan NA, Algarni AD. Using Dual Attention BiLSTM to Predict Vehicle Lane Changing Maneuvers on Highway Dataset. *Systems*. 2023; 11(4):196. https://doi.org/10.3390/systems11040196
9. Attaullah, M. *et al.* (2022) 'Initial stage COVID-19 detection system based on patients' symptoms and chest X-Ray images,' *Applied Artificial Intelligence*, 36(1). https://doi.org/10.1080/08839514.2022.2055398.
10. Azam, H., Dulloo, M.I., Majeed, M.H., Wan, J.P.H., Xin, L.T. and Sindiramutty, S.R. (2023) 'Cybercrime Unmasked: Investigating cases and digital evidence,' *International Journal of Emerging Multidisciplinaries. Computer Science and Artificial Intelligence*, 2(1). https://doi.org/10.54938/ijemdcsai.2023.02.1.255.
11. Azam, H., Dulloo, M.I., Majeed, M.H., Wan, J.P.H., Xin, L.T., Tajwar, M.A., *et al.* (2023) 'Defending the digital Frontier: IDPS and the battle against Cyber threat,' *International Journal of Emerging Multidisciplinaries. Computer Science and Artificial Intelligence*, 2(1). https://doi.org/10.54938/ijemdcsai.2023.02.1.253.
12. Azam, H., Tajwar, M.A., *et al.* (2023) 'Innovations in Security: A study of cloud Computing and IoT,' *International Journal of Emerging Multidisciplinaries. Computer Science and Artificial Intelligence*, 2(1). https://doi.org/10.54938/ijemdcsai.2023.02.1.252.
13. Azeem, M. *et al.* (2021) 'FOG-Oriented secure and lightweight data aggregation in IOMT,' *IEEE Access*, 9, pp. 111072–111082. https://doi.org/10.1109/access.2021.3101668.
14. Badyal, N. (2024) Scalable techniques for risk assessment of open-source libraries. https://doi.org/10.32657/10356/173692.
15. Bilge, L. and Dumitraş, T. (2012) 'Before we knew it,' : *Proceedings of the 2012 ACM Conference on Computer and Communications Security* [Preprint]. https://doi.org/10.1145/2382196.2382284.
16. Bompos, K. (2020). Naval Postgraduate School Monterey, California Thesis Development Time Of Zero-Day Cyber Exploits In Support Of Offensive Cyber Operations
17. Changsan, U. and Boonyopakorn, P. (2023) 'Log4shell Investigate Based On Generic Computer Forensic Investigation Model,' 2023 20th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON) [Preprint]. https://doi.org/10.1109/ecti-con58255.2023.10153283.
18. Chesti, I.A. *et al.* (2020) 'Evolution, Mitigation, and Prevention of Ransomware,' *2020 2nd International Conference on Computer and Information Sciences (ICCIS)* [Preprint]. https://doi.org/10.1109/iccis49240.2020.9257708.
19. Conti ransomware threat group adopts Log4j exploit to compromise VMware vCenter servers (no date). https://www.broadcom.com/support/security-center/protection-bulletin/conti-ransomware-threat-group-adopts-log4j-exploit-to-compromise-vmware-vcenter-servers.
20. Dell (2022) How to remove JndiLookup class from the log4j-2.x core jar file. https://www.dell.com/support/kbdoc/en-us/000194596/how-to-remove-jndilookup-class-from-the-log4j-2-x-core-jar-file.
21. Elmo, D., II (2023) The Open Charge Point Protocol (OCPP) version 1.6 Cyber Range a training and testing platform. https://corescholar.libraries.wright.edu/etd_all/2788/.
22. en-us (no date). https://help.minecraft.net/hc/en-us/articles/4416199399693-Security-Vulnerability-in-Minecraft-Java-Edition.
23. Fadolalkarim, D. and Fadolalkarim, D. (2024) 'DCAFixer: an automatic tool for bug detection and repair for database Java client applications,' *IEEE Transactions on Dependable and Secure Computing/IEEE Transactions on Dependable and Secure Computing*, pp. 1–16. https://doi.org/10.1109/tdsc.2024.3396667.
24. Fatima-Tuz-Zahra, N. *et al.* (2020) 'Proposing a Hybrid RPL Protocol for Rank and Wormhole Attack Mitigation using Machine Learning,' *2020 2nd International Conference on Computer and Information Sciences (ICCIS)* [Preprint]. https://doi.org/10.1109/iccis49240.2020.9257607.
25. Fidler, M. (2024) Zero progress on Zero days: How the last ten years created the modern spyware market. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4626426.
26. Fleury, M. and Reverbel, F. (2003) 'The JBoss extensible server,' in *Lecture notes in computer science*, pp. 344–373. https://doi.org/10.1007/3-540-44892-6_18.

27. GeeksforGeeks. (2020). Zero-day Exploit (Cyber Security Attack). [online] Available at: https://www.geeksforgeeks.org/zero-day-exploit-cyber-security-attack/.
28. Gill, S.H. *et al.* (2022) 'Security and privacy aspects of cloud Computing: A Smart Campus case study,' *Intelligent Automation and Soft Computing/Intelligent Automation & Soft Computing*, 31(1), pp. 117–128. https://doi.org/10.32604/iasc.2022.016597.
29. Goni, A. (2020) 'International Journal of Research and Scientific Innovation,' *International Journal of Research and Scientific Innovation* [Preprint]. https://doi.org/10.51244/ijrsi.
30. Gopi, R. *et al.* (2021) 'Enhanced method of ANN based model for detection of DDoS attacks on multimedia internet of things,' *Multimedia Tools and Applications*, 81(19), pp. 26739–26757. https://doi.org/10.1007/s11042-021-10640-6.
31. Gouda, W. *et al.* (2022) 'Detection of COVID-19 based on chest x-rays using deep learning,' *Healthcare*, 10(2), p. 343. https://doi.org/10.3390/healthcare10020343.
32. Grigutytė, M. and Grigutytė, M. (2024) Microsoft Exchange zero-day vulnerability explained. https://nordvpn.com/blog/microsoft-exchange-exploits/.
33. He, Y. et al. (2022) 'A survey on Zero Trust architecture: Challenges and future trends,' Wireless Communications and Mobile Computing, 2022, pp. 1–13. https://doi.org/10.1155/2022/6476274.
34. Humayun, M. *et al.* (2022) 'A Transfer Learning Approach with a Convolutional Neural Network for the Classification of Lung Carcinoma,' *Healthcare*, 10(6), p. 1058. https://doi.org/10.3390/healthcare10061058.
35. Hurst, W. and Shone, N. (2024) 'Critical infrastructure security: Cyber-threats, legacy systems and weakening segmentation,' in *Elsevier eBooks*, pp. 265–286. https://doi.org/10.1016/b978-0-323-99330-2.00010-6.
36. Hussain, K. *et al.* (2024) 'Threats and Vulnerabilities of Wireless Networks in the Internet of Things (IoT),' *2024 IEEE 1st Karachi Section Humanitarian Technology Conference (KHI-HTC)* [Preprint]. https://doi.org/10.1109/khi-htc60760.2024.10482197.
37. Kaspersky (2018). What is Zero Day Exploit? [online] www.kaspersky.com. Available at: https://www.kaspersky.com/resource-center/definitions/zero-day-exploit.
38. Kumar, M.S. *et al.* (2021) 'Blockchain based peer to peer communication in autonomous drone operation,' *Energy Reports*, 7, pp. 7925–7939. https://doi.org/10.1016/j.egyr.2021.08.073.
39. Lyu, M., Gharakheili, H.H. and Sivaraman, V. (2024) 'A survey on Enterprise Network Security: asset behavioral monitoring and Distributed attack Detection,' *IEEE Access*, p. 1. https://doi.org/10.1109/access.2024.3419068.
40. M. Lim, A. Abdullah, N. Z. Jhanjhi, M. Khurram Khan and M. Supramaniam, "Link Prediction in Time-Evolving Criminal Network With Deep Reinforcement Learning Technique," in *IEEE Access*, vol. 7, pp. 184797-184807, 2019, doi: 10.1109/ACCESS.2019.2958873.
41. Muhammad, Ibrahim, Humayun Mamoona, N. Z. Jhanjhi, and M. N. Talib. "Intelligent Computing and Innovation on Data Science." (2021): 425-434.
42. Möller, D.P.F. (2023) 'Threats and threat intelligence,' in *Advances in information security*, pp. 71–129. https://doi.org/10.1007/978-3-031-26845-8_2.
43. Msrc (2021) Guidance for responders: Investigating and remediating on-premises Exchange Server vulnerabilities | MSRC Blog | Microsoft Security Response Center.
44. Nair, D. and Mhavan, N. (2023) 'Augmenting Cybersecurity: A survey of intrusion detection systems in combating zero-day vulnerabilities,' in *Contemporary studies in economic and financial analysis*, pp. 129–153. https://doi.org/10.1108/s1569-37592023000110a007.
45. Narang, S. (2021). CVE-2021-26855, CVE-2021-26857, CVE-2021-26858, CVE-2021-27065: Four Zero-Day vulnerabilities in Microsoft Exchange Server exploited in the Wild. https://www.tenable.com/blog/cve-2021-26855-cve-2021-26857-cve-2021-26858-cve-2021-27065-four-microsoft-exchange-server-zero-day-vulnerabilities.
46. Nathaniel, H. (2022) *Defense Technical Information Center*. https://apps.dtic.mil/sti/citations/trecms/AD1200567.
47. Nguyen, T.G. *et al.* (2023) 'Multi-Granularity detector for vulnerability fixes,' *IEEE Transactions on Software Engineering*, 49(8), pp. 4035–4057. https://doi.org/10.1109/tse.2023.3281275.
48. owasp.org. (n.d.). Fuzzing | OWASP. [online] Available at: https://owasp.org/www-community/Fuzzing#:~:text=A%20fuzzer%20is%20a%20program.
49. Pauley, E., Barford, P. and McDaniel, P. (2023a) 'The CVE Wayback Machine: Measuring Coordinated Disclosure from Exploits against Two Years of Zero-Days,' *ACM* [Preprint]. https://doi.org/10.1145/3618257.3624810.
50. Pratama, A. and Adi Rafrastara, F. (2012). Computer Worm Classification. International Journal of Computer Science and Information Security, [online] 10(4), pp.21–24. Available at: https://www.researchgate.net/publication/299580232_Computer_Worm_Classification.
51. Rains, T. (2020) Cybersecurity Threats, malware Trends, and Strategies, Packt Publishing eBooks. http://uh2.scholarvox.com/catalog/book/88897769.

52. Rhodes, C. and Bettany, A. (2016) *Windows installation and update troubleshooting*, *Apress eBooks*. https://doi.org/10.1007/978-1-4842-1827-3.
53. Riegler, M. *et al.* (2023) 'A model-based mode-switching framework based on security vulnerability scores,' *Journal of Systems and Software/the Journal of Systems and Software*, 200, p. 111633. https://doi.org/10.1016/j.jss.2023.111633.
54. Riofrío, X., Astudillo-Salinas, F., Tello-Oquendo, L. and Merchan-Lima, J. (2021). The Zero-day attack: Deployment and evolution. Latin-American Journal of Computing, [online] 8(1), pp.38–53. Available at: https://lajc.epn.edu.ec/index.php/LAJC/article/view/208/148 [Accessed 22 Apr. 2024].
55. Robinson, P. (2024) Microsoft Exchange Server attack: The Hafnium Breach. https://www.lepide.com/blog/the-hafnium-breach-microsoft-exchange-server-attack/.
56. Saurabh, K. *et al.* (2024) 'HMS-IDS: threat intelligence integration for Zero-Day exploits and advanced persistent threats in IIoT,' *Arabian Journal for Science and Engineering* [Preprint]. https://doi.org/10.1007/s13369-024-08935-5.
57. Security, M. 365 and Intelligence, M.T. (2023) HAFNIUM targeting Exchange Servers with 0-day exploits.
58. Shah, I.A., Jhanjhi, N.Z. and Laraib, A. (2022) 'Cybersecurity and blockchain usage in contemporary business,' in *Advances in information security, privacy, and ethics book series*, pp. 49–64. https://doi.org/10.4018/978-1-6684-5284-4.ch003.
59. Shahine, M. (2023). CCleaner vs BleachBit: A Deep Dive. [online] Internet Safety Statistic. Available at:
60. Shandilya, S.K. *et al.* (2024) 'Achieving Digital Resilience with Cybersecurity,' in *EAI/Springer Innovations in Communication and Computing*, pp. 43–123. https://doi.org/10.1007/978-3-031-53290-0_2.
61. Sharifonnasabi Fatemeh , Jhanjhi Noor Zaman , John Jacob , Obeidy Peyman , Band Shahab S. , Alinejad-Rokny Hamid , Baz Mohammed, Hybrid HCNN-KNN Model Enhances Age Estimation Accuracy in Orthopantomography, Frontiers in Public Health, VOLUME 10, 2022, DOI=10.3389/fpubh.2022.879418.
62. Sindiramutty, S.R. (2024) 'Autonomous Threat Hunting: a future paradigm for AI-Driven Threat intelligence,' *arXiv (Cornell University)* [Preprint]. https://doi.org/10.48550/arxiv.2401.00286.
63. Sindiramutty, S. R., Jhanjhi, N. Z., Tan, C. E., Khan, N. A., Shah, B., & Gaur, L. (2024). Securing the Digital Supply Chain Cyber Threats and Vulnerabilities. In N. Jhanjhi & I. Shah (Eds.), Cybersecurity Measures for Logistics Industry Framework (pp. 156-223). IGI Global. https://doi.org/10.4018/978-1-6684-7625-3.ch007
64. Sindiramutty, S.R. *et al.* (2024) 'Cybersecurity measures for logistics industry,' in *Advances in information security, privacy, and ethics book series*, pp. 1–58. https://doi.org/10.4018/979-8-3693-3816-2.ch001.
65. Singh, N., Buyya, R. and Kim, H. (2024) 'Securing Cloud-Based Internet of Things: Challenges and mitigations,' *arXiv (Cornell University)* [Preprint]. https://doi.org/10.48550/arxiv.2402.00356.
66. Spafford, E., Metcalf, L. and Dykstra, J. (2023) Cybersecurity myths and misconceptions: Avoiding the Hazards and Pitfalls That Derail Us. Addison-Wesley Professional.
67. Ray, S. K., Sinha, R., & Ray, S. K. (2015, June). A smartphone-based post-disaster management mechanism using WiFi tethering. In *2015 IEEE 10th conference on industrial electronics and applications (ICIEA)* (pp. 966-971). IEEE.
68. Ray, S. K., Pawlikowski, K., & Sirisena, H. (2009). A fast MAC-layer handover for an IEEE 802.16 e-based WMAN. In AccessNets: Third International Conference on Access Networks, AccessNets 2008, Las Vegas, NV, USA, October 15-17, 2008. Revised Papers 3 (pp. 102-117). Springer Berlin Heidelberg.
69. Tamayo, J., López, L.I.B. and Caraguay, Á.L.V. (2024) 'Detection of distributed denial of service attacks carried out by botnets in Software-Defined networks,' *arXiv (Cornell University)* [Preprint]. https://doi.org/10.48550/arxiv.2401.09358.
70. Touré, A. *et al.* (2024) 'A framework for detecting zero-day exploits in network flows,' *Computer Networks*, p. 110476. https://doi.org/10.1016/j.comnet.2024.110476.
71. Wan, M. et al. (2012) 'Locator/Identifier Separation: Comparison and analysis on the mitigation of worm propagation,' the International Journal of Computational Intelligence Systems/International Journal of Computational Intelligence Systems, 5(5), p. 868. https://doi.org/10.1080/18756891.2012.733218.
72. Wang, Y. *et al.* (2014) 'Modeling the Propagation of worms in networks: A survey,' *IEEE Communications Surveys and Tutorials/IEEE Communications Surveys and Tutorials*, 16(2), pp. 942–960. https://doi.org/10.1109/surv.2013.100913.00195.
73. Wen, B.O.T. *et al.* (2023) 'Detecting cyber threats with a Graph-Based NIDPS,' in *Advances in logistics, operations, and management science book series*, pp. 36–74. https://doi.org/10.4018/978-1-6684-7625-3.ch002.
74. www.ibm.com. (n.d.). What is a Zero-Day Exploit? | IBM. [online] Available at: https://www.ibm.com/topics/zero-day#:~:text=A%20zero%2Dday%20exploit%20is.
75. Xu, G. and Meng, L. (2023) 'A novel algorithm for identifying influential nodes in complex networks based on local propagation probability model,' *Chaos, Solitons & Fractals/Chaos, Solitons and Fractals*, 168, p. 113155. https://doi.org/10.1016/j.chaos.2023.113155.
76. Zerdoumi, S., Hashem, I.A.T. & Jhanjhi, N.Z. A new spatial spherical pattern model into interactive cartography pattern: multi-dimensional data via geostrategic cluster. *Multimed Tools Appl* **81**, 22903–22952 (2022). https://doi.org/10.1007/s11042-021-11339-4

 doi:10.20944/preprints202407.2338.v1