China-Nexus Threat Group ‘Velvet Ant’ Leverages a Zero-Day to Deploy Malware on Cisco Nexus Switches

: 8/22/2024

Sygnia uncovers the China-Nexus group ‘Velvet Ant’ leveraging a zero-day exploit (CVE-2024-20399) on Cisco Switch appliances, escalating evasion tactics to maintain long-term network persistence.

22 August 2024

CONTENTS

Key Takeaways

- Earlier in 2024, Sygnia observed ‘Velvet Ant’ leveraging a zero-day exploit (CVE-2024-20399) to compromise and control on-premises Cisco Switch appliances. These types of vulnerabilities are used by threat actor to operate on compromised devices in a way that is completely hidden to the enterprise security stack.
- As part of the ‘Velvet Ant’ multi-year intrusion, the transition to operating from internal network devices marks yet another escalation in the evasion techniques used in order to ensure the continuation of the espionage campaign.
- The zero-day exploit allows an attacker with valid administrator credentials to the Switch management console to escape the NX-OS command line interface (CLI) and execute arbitrary commands on the Linux underlying operating system. Following the exploitation, ‘Velvet Ant’ deploy tailored malware, which runs on the underlying OS and is invisible to common security tools.
- The modus-operandi of ‘Velvet Ant’ highlights risks and questions regarding third-party appliances and applications that organizations onboard. Due to the ’black box‘ nature of many appliances, each piece of hardware or software has the potential to turn into the attack surface that an adversary is able to exploit.
- By enhancing logging, implementing continuous monitoring, and conducting systematic threat hunts on key organizational choke points, organizations can better detect and counteract advanced persistent threats such as ‘Velvet Ant’. For additional, detailed prevention and detection guidelines, see Sygnia’s vulnerability advisory.

Introduction

Sygnia recently published a blog post about a China-Nexus threat group, providing an in-depth analysis of Velvet Ant TTPs that have been seen in the wild. While the previous blog demonstrates the attack flow and compromise, this blog highlights the technique used by Velvet Ant to compromise Cisco Switch appliances and use them to perform stealthy attacks.

In an intrusion that spanned over multiple years, Velvet Ant escalated their tactics to stealthily maintain persistence within networks. For over three years, they evaded detection, gradually infiltrating new Windows systems, servers, and laptops. Gradually, they shifted their operations to legacy Windows systems, such as Windows 2003 servers. These older systems, with their default inadequate logging and inability to support modern security technologies, provided an ideal environment for the attackers to continue their activities undetected.

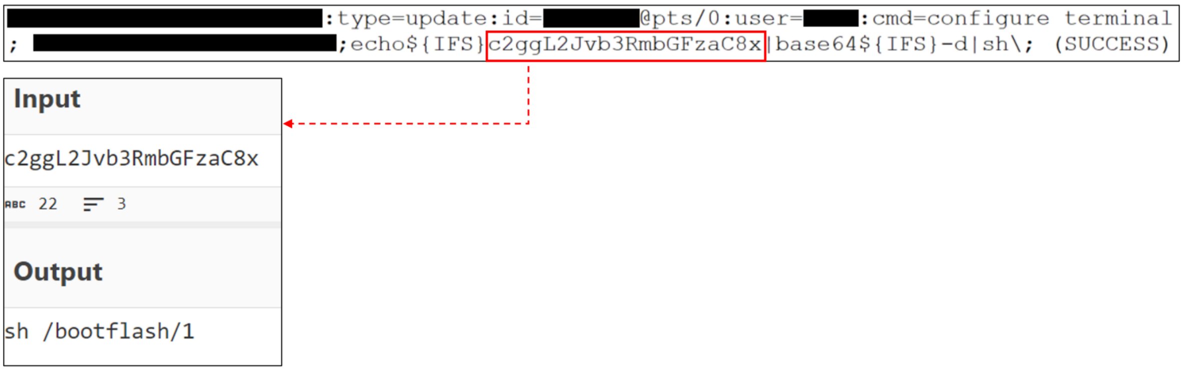
Next, Velvet Ant adapted their attack approach again and moved to an operational tactic that leveraged network devices such as legacy F5 BIG-IP appliances, as described in the earlier blog post. This tactic allowed the group to obtain a new vantage point – one that is not accessible to the victim, as it is a black box that enables the attackers to avoid detection.

In recently observed attacks , Velvet Ant transitioned to operating from Cisco Nexus switch appliances and exploited a zero-day vulnerability, in order to access the underlying Linux layer of the switch to install their malware – named ‘VELVETSHELL’ by Sygnia. These switch appliances do not give the user access to the underlying operating system, making scanning for indicators of compromise nearly impossible. This shift towards network appliances emphasizes the group’s sophistication and determination to maintain persistence in a compromised environment in order to continue conducting espionage activities.

Jailbreaking a Cisco Switch Appliance using a 0-Day NX-OS CLI Exploit (CVE-2024-20399)

During response activities following a recent Velvet Ant intrusion, a suspicious anomaly was detected on a Cisco switch appliance, prompting deeper investigation. Upon accessing the device, Sygnia observed the threat actor performing reconnaissance activities, including issuing extended ping commands to probe additional network devices, and mapping the routing paths across various VRFs (Virtual Routing and Forwarding). Moreover, the threat actor's use of the Cisco switch as a main pivot to access additional network devices allowed for clear identification of additional activities originating from known compromised locations.

By investigating the accounting logs of the affected system, Sygnia discovered several suspicious Base64-encoded commands that were executed using valid administrative credentials. These commands were identified as being not merely unusual administrative commands, but rather part of an exploit leveraging a command injection vulnerability. The threat actor utilized this technique to execute a malicious script to load and execute a backdoor binary on the device, thereby bypassing standard security mechanisms.



```
[redacted]:type=update:id=[redacted]@pts/0:user=[redacted]:cmd=configure terminal
; [redacted];echo${IFS}c2ggL2Jvb3RmbGFzaC8x|base64${IFS}-d|sh\; (SUCCESS)
```

Input

```
c2ggL2Jvb3RmbGFzaC8x
```

Output

```
sh /bootflash/1
```

Figure 1: Snippet from the accounting log of the Cisco Nexus switch, showing the command injection vulnerability used by Velvet Ant.

Cisco NX-OS in a Nutshell

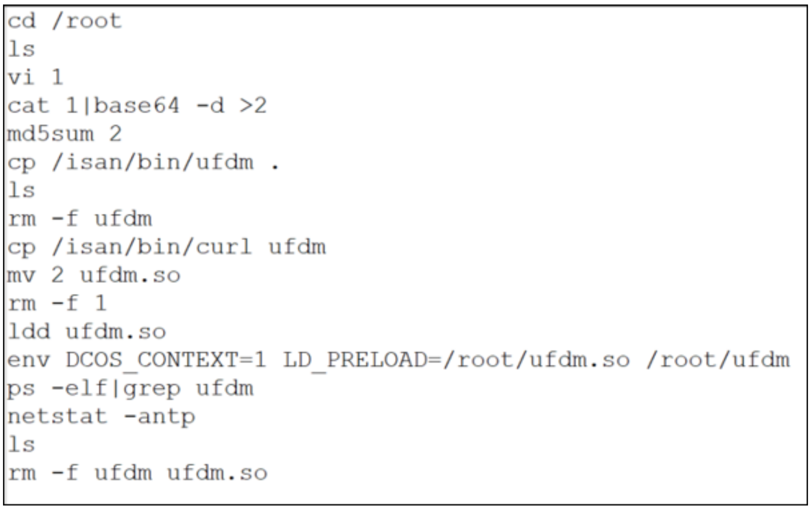
Cisco NX-OS is a network operating system designed specifically for Cisco's Nexus-series switches. It operates with a distinct layered architecture, consisting of an 'application' level and an underlying Linux-based OS level. The 'application' level is what a network administrator would interact with through the CLI, providing commands tailored for network management tasks such as configuring routing, managing interfaces, and monitoring network performance.

By design, end users are restricted to the application level to ensure a secure and controlled environment. This layer is robust and is equipped with numerous security mechanisms to prevent unauthorized access and to maintain the integrity of network operations. However, the underlying Linux OS layer, which forms the foundation of the NX-OS, is typically hidden and inaccessible to end users. It handles the core system functions, running processes and managing resources that are critical to the switch's operation.

Velvet Ant discovered and utilized a zero-day command injection vulnerability in the Cisco NX-OS Software CLI to bypass the restrictive application layer. This vulnerability was later assigned the CVE ID of CVE-2024-20399. By leveraging this vulnerability, the group gained unauthorized access to the underlying Linux OS level. This access provided the attackers with elevated control over the switch, enabling them to execute malicious scripts and manipulate the system beyond the intended administrative capabilities.

NX-OS Linux Layer Post-Exploitation and Malware Deployment

After accessing the compromised system, Velvet Ant focused on deploying the VELVETSHELL malware on the device and obfuscating its presence. The bash history file reveals a methodical approach to post-exploitation tasks. First, the threat actor created a file, and then decoded its Base64-encoded content into another file – which was later renamed 'ufdm.so'- this change suggests that it contained the malicious payload.



```
cd /root
ls
vi 1
cat 1|base64 -d >2
md5sum 2
cp /isan/bin/ufdm .
ls
rm -f ufdm
cp /isan/bin/curl ufdm
mv 2 ufdm.so
rm -f 1
ldd ufdm.so
env DCOS_CONTEXT=1 LD_PRELOAD=/root/ufdm.so /root/ufdm
ps -elf|grep ufdm
netstat -antp
ls
rm -f ufdm ufdm.so
```

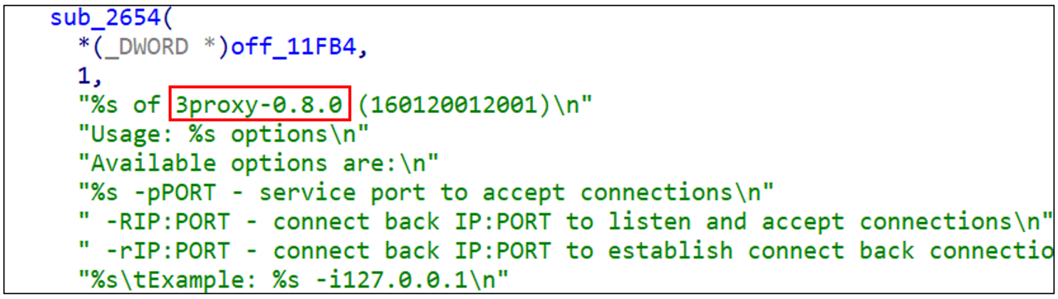
Figure 2: Snippet from the accounting log of the Cisco Nexus switch, showing the command injection vulnerability used by Velvet Ant.

Before the execution of the malware, the threat actor copied the legitimate ‘curl’ binary and renamed it ‘ufdm’ – which is the name of a legitimate binary on Cisco Nexus switch appliances. The ‘LD_PRELOAD’ environment variable was then set to load ‘ufdm.so’, allowing the attacker to inject their code into the masqueraded ‘/root/ufdm’ process, thereby gaining control over the execution flow. The threat actor then checked the running processes and active network connections using the ‘ps’ and ‘netstat’ commands respectively – likely, to ensure that their malware was running as intended, and to assess the system’s network activity. After executing their payload, the threat actor meticulously removed traces of their actions by deleting the renamed ‘ufdm’ and ‘ufdm.so’ files, in an attempt to cover their tracks and avoid detection. This sequence highlights the sophistication and stealth of the threat actor’s operations during the post-exploitation phase.

VELVETSHELL Analysis

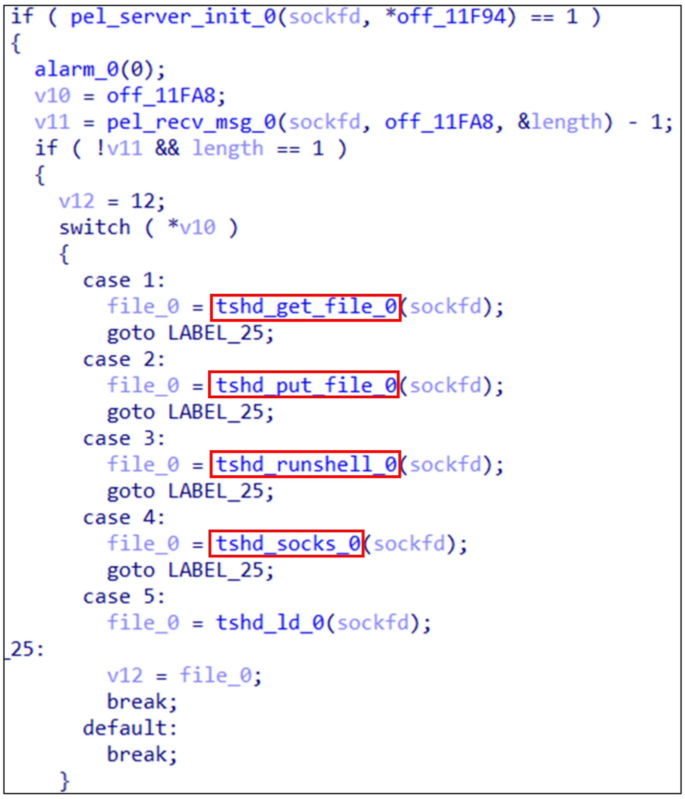
Despite the malware being deleted by the threat actor, Sygnia was able to reconstruct it from the device memory through a detailed forensic process. By processing and analyzing the reconstructed VELVETSHELL malware, it was determined that it is a hybrid customized version of two open-source tools: TinyShell, a Unix backdoor, and 3proxy, a proxy tool. Both tools were utilized separately in the past for nefarious purposes; however, in this case, they were identified as being incorporated into a single binary.

As a hybrid of known tools, and with the additional analysis of the binary for confirmation, the VELVETSHELL malware provides multiple capabilities – such as execution of arbitrary commands, download and upload of files, and establishing tunnels for proxying network traffic. These functionalities provided the threat actor with extensive control over the compromised system, enabling both data exfiltration and persistent access.



```
  sub_2654(
    *(_DWORD *)off_11FB4,
    1,
    "%s of 3proxy-0.8.0 (160120012001)\n"
    "Usage: %s options\n"
    "Available options are:\n"
    "%s -pPORT - service port to accept connections\n"
    " -RIP:PORT - connect back IP:PORT to listen and accept connections\n"
    " -rIP:PORT - connect back IP:PORT to establish connect back connectio
    "%s\tExample: %s -i127.0.0.1\n"
```

Figure 3: Snippet from ‘IDA’ decompile software, showing the reconstructed VELVETSHELL malware functions, which include 3proxy f

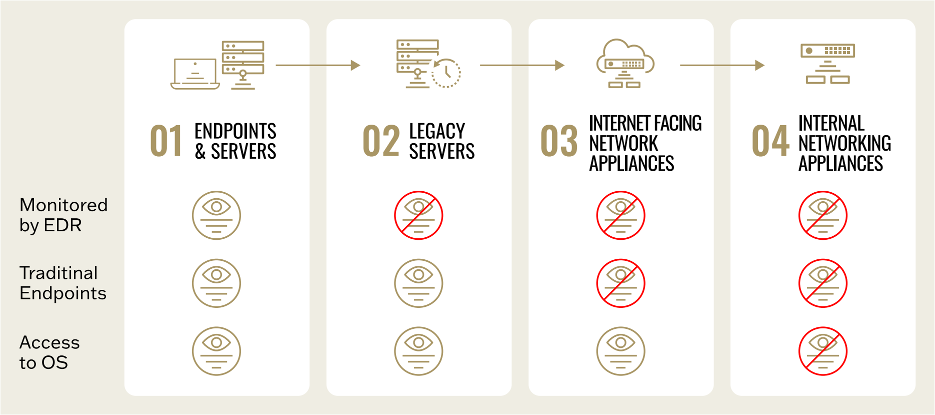


```
if ( pel_server_init_0(sockfd, *off_11F94) == 1 )
{
  alarm_0(0);
  v10 = off_11FA8;
  v11 = pel_recv_msg_0(sockfd, off_11FA8, &length) - 1;
  if ( !v11 && length == 1 )
  {
    v12 = 12;
    switch ( *v10 )
    {
      case 1:
        file_0 = tshd_get_file_0(sockfd);
        goto LABEL_25;
      case 2:
        file_0 = tshd_put_file_0(sockfd);
        goto LABEL_25;
      case 3:
        file_0 = tshd_runshell_0(sockfd);
        goto LABEL_25;
      case 4:
        file_0 = tshd_socks_0(sockfd);
        goto LABEL_25;
      case 5:
        file_0 = tshd_ld_0(sockfd);
25:
        v12 = file_0;
        break;
      default:
        break;
    }
```

Figure 4: Snippet from 'IDA' a decompile software, showing the reconstructed VELVETSHELL malware functions, which include 'TinyShell' functionalities.

A Note on 'Velvet Ant'

Over the years of espionage activities 'Velvet Ant' increased their sophistication, using evolving tactics to continue their cyber operations in a victim network – from operating on ordinary endpoints, shifting operations to legacy servers and finally moving towards network appliances and using 0-days. The determination, adaptability and persistence of such threat actors highlights the sensitivity of a holistic response plan to not only contain and mitigate the threat but also monitor the network for additional attempts to exploit the network.



Appendix I: Indicators of Compromise

Value	Type	Description
/bootflash/id.txt	File path	N/A
/bootflash/1	File path	N/A
/root/ufdm	File path	Renamed curl
/root/ufdm.so	File path	Malicious library
/root/a	File path	N/A
/root/t	File path	N/A
/root/1	File path	N/A
/root/2	File path	N/A