vmware® THREAT ANALYSIS UNIT

Exposing Malware in Linux-Based Multi-Cloud Environments

Technical Threat Report

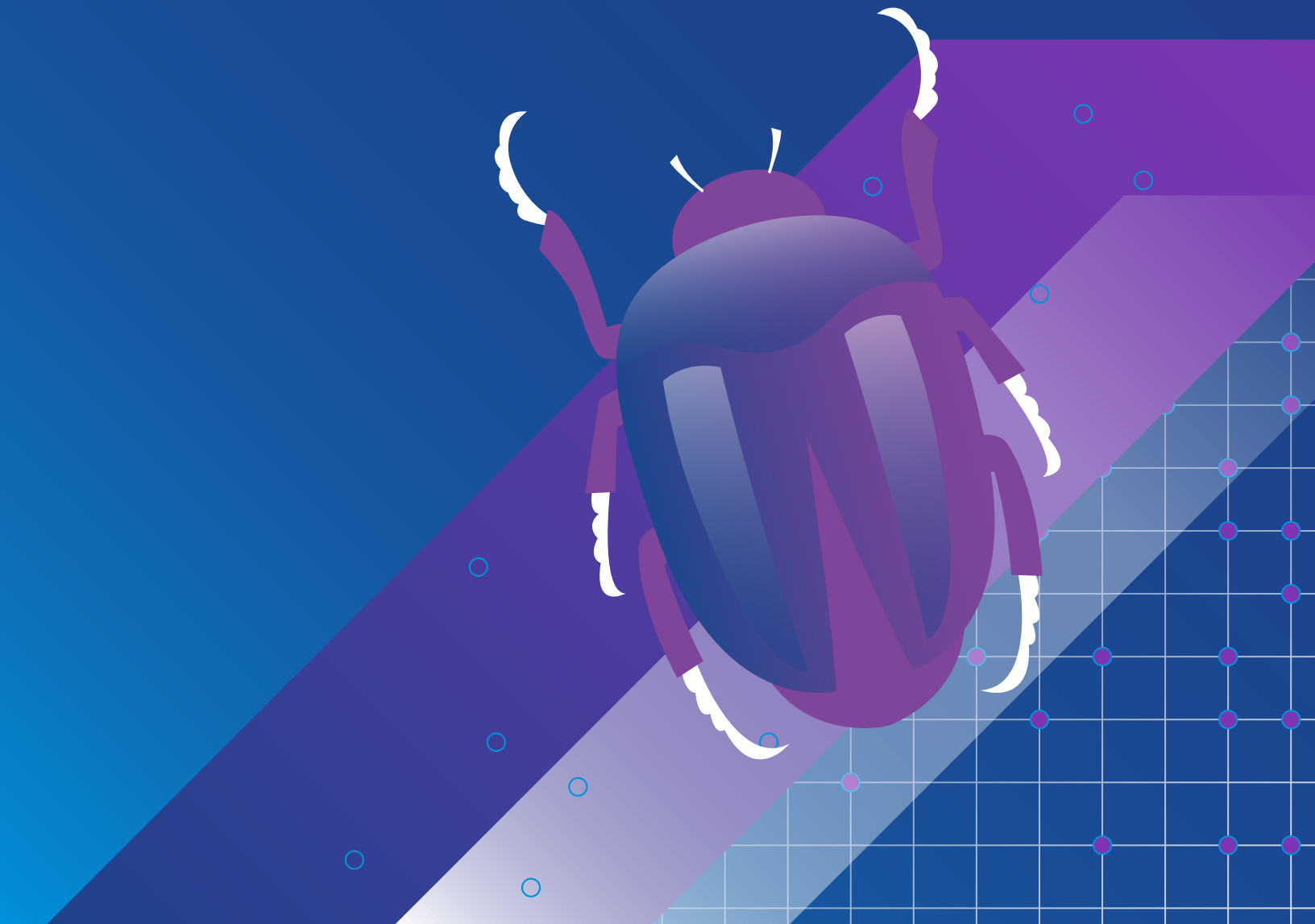

Table of contents

Executive summary

In the past five years, Linux® has become the most common operating system (OS) in multi-cloud environments. It has even bypassed Windows on Microsoft Azure to power more than 78 percent of the most popular websites.[1] Malicious actors have taken notice and are increasingly targeting vulnerable Linux-based systems in multi-cloud environments to infiltrate corporate and government networks.

Threat actors know that current malware countermeasures are mostly focused on addressing Windows-based threats, leaving many public and private cloud deployments vulnerable to Linux-based attacks. These public and private clouds are high-value targets for cybercriminals, providing access to critical infrastructure services and substantial computational resources.

In fact, cloud infrastructures and data centers host key components, such as email servers and customer databases, that have been the target of high-profile intelligence-gathering breaches. The large-scale campaign carried out in early 2021, which targeted Exchange servers,[2] and the Cybersecurity and Infrastructure Security Agency (CISA) alert about BlackMatter,[3] which targeted the U.S. food and agriculture sector, are good examples of how attacks to vulnerable cloud infrastructure can disrupt an organization's value-delivery pipeline.

These threats take advantage of weak authentication, vulnerabilities and misconfigurations in container-based infrastructures to infiltrate the environment with remote access tools (RATs). Once the attackers have obtained a foothold in their target cloud environment, they often look to perform two types of attacks: execute ransomware or deploy cryptomining components.

Organizations need to bolster their ability to identify and defend against these types of attacks. Given the distributed, dynamic and heterogeneous nature of today's enterprise workloads and networks, organizations need to extend telemetry across the entire infrastructure—from endpoints to multi-cloud environments. This will allow organizations to better monitor traffic and identify abnormal behavior to mitigate the impact of attacks on the enterprise, while increasing overall efficiencies and reducing operational costs.

This report, based on VMware's experience with a diverse customer base, offers a comprehensive look at Linux-based malware threats to multi-cloud environments. It highlights the unique characteristics of this class of threats and provides guidance on how combining endpoint detection and response (EDR) and network detection and response (NDR) solutions can help organizations stay ahead of the threats Linux-based malware poses.

78%

of the most popular websites are powered by Linux[1]

Linux-based malware is a fast-growing threat to multi-cloud environments, including data centers, that must be addressed to protect an organization's assets and operations. This report details the research that the VMware Threat Analysis Unit™ has done into the latest threats to Linux-based systems, documenting key findings that can help organizations better understand and prepare to defend themselves against these rising threats.

Key findings

Linux-based systems are fast becoming an attacker's way into high-value, multi-cloud environments.

- Linux is the most common OS across multi-cloud environments.[4]
- Malware targeting Linux-based systems is increasing in volume and complexity, but it is still less sophisticated than Windows threats.
- There is a lack of focus on the detection of threats that target Linux-based systems, making existing tools inadequate.
- The main threats in most multi-cloud environments are ransomware, cryptominers and RATs.
- Existing attack characterization techniques based on static information, such as strings and APIs, are useful but easily evaded by sophisticated threats.
- Defense evasion is the most common tactic used in ransomware and cryptominers. Various encryption or obfuscation techniques, such as Base64 encoding and AES-based encryption, are used by attackers to conceal code and data.

Ransomware is becoming more sophisticated.

- Ransomware has recently evolved to target Linux host images that are used to spin workloads in virtualized environments.
- Ransomware attacks against cloud deployments are targeted, not opportunistic.
- Ransomware attacks against cloud environments are often combined with data exfiltration, implementing a double-extortion scheme that improves their odds of success.
- The detection of sophisticated threats targeting Linux-based systems requires dynamic analysis and continuous host monitoring—capabilities that work well with the Linux kernel.

Cryptojacking attacks use XMRig to mostly mine Monero.

- Cryptojacking attacks focus on monetizing stolen CPU cycles to mine cryptocurrencies.
- Most cryptojacking attacks focus on mining the Monero cryptocurrency (or XMR). XMRig is the most commonly used tool for cryptomining, and research found that 89 percent of cryptominers used XMRig-related libraries.

RATs are an increasing threat to Linux-based systems.

- As Cobalt Strike is such a ubiquitous threat on Windows, its expansion to other operating systems, such as Linux, is notable. It demonstrates the desire of threat actors to use readily available remote-control tools to target as many platforms as possible.
- VMware Threat Analysis Unit discovered more than 14,000 active Cobalt Strike team servers on the internet since the end of February 2020.
- The most popular protocol for the Cobalt Strike beacon is HTTPS.
- Close to 90 percent of the Cobalt Strike server population is version 4 or later.
- The total percentage of cracked and leaked Cobalt Strike customer IDs is 56 percent. This means that more than half of the Cobalt Strike users are using illegitimately obtained versions of the commercial software.
- Vermilion Strike is just the first of many malware targeting Linux-based systems that will mimic the actions of other well-known RATs to simplify an adversary's work.

Ransomware and cryptominers

Ransomware

The impact of a ransomware attack can range from being a nuisance (e.g., having to restore data from backups and clean up the network) to being devastating (e.g., having to pay large sums of money to regain access to key assets). Unfortunately, when talking about cloud environments, the results tend to be more on the devastating side. Recently, cybercriminals have started calculating the damage they might cause to the valuation of a company going through a financial event to make the potential impact of their attack clear and incentivize ransom payments.[5] At the same time, they've been honing their tactics with increasingly sophisticated techniques to target victim organizations.

Large-scale ransomware attacks on cloud deployments, however, have some distinct characteristics that make them harder to spot and stop. First, ransomware attacks against cloud deployments are targeted, not opportunistic. Unlike general ransomware, which plays a numbers game (e.g., sending malicious email attachments to millions of users in hopes that some will inadvertently click and infect their systems), attacks on cloud infrastructures are more targeted and carefully planned.

These attacks are designed for maximum impact. The cybercriminals make sure that their target systems have been fully compromised before starting the file encryption process. This allows them to simultaneously encrypt the whole network to inundate response resources and make incident response more difficult.

Ransomware has recently evolved to target the Linux host images used to spin up workloads in virtualized environments. This new and worrisome development shows how attackers look for the most valuable assets in cloud environments to inflict the maximum damage on their target. Examples include the Defray777 ransomware family, which encrypted host images on VMware ESXi servers,[6] and the DarkSide ransomware family, which crippled Colonial Pipeline's networks and caused a nationwide gasoline shortage in the U.S.[7]

The basic profile of a ransomware attack is so popular that even non-technical people know how it works: a network is compromised, sensitive files are encrypted, and a ransom note is presented to the victim that asks for money/cryptocurrency in exchange for a decryption key that will unlock access to the files. (For a peek into what victim organizations go through, check out the blog post, HelloKitty: The Victim's Perspective.[8])

Many ransomware attacks against cloud environments involve exfiltrating data, allowing the attacker to implement a double-extortion scheme. Essentially, the attacker uses the data they collect as leverage to incent the victim into paying the ransom. If the victim does not comply, the attacker leaks the information, making it public on the internet to damage the victim's reputation.[9]

Ransomware families

For this report, the VMware Threat Analysis Unit analyzed nine ransomware families that target Linux systems, characterizing their evolution and cross-pollination. A brief description of the families covered is included in the sidebar. We started analyzing the different characteristics of the ransomware samples of each of these nine families by looking at the static information extracted from their ELF files. While threats can be a combination of shell scripts, Python scripts, and binaries, this report focuses on the binaries. Binaries are usually the components that carry out the file system encryption in a ransomware attack.

Characterizing similarity

As a first step, we used telfhash[10] to characterize the code sharing among samples. The approach used by telfhash applies a locality-sensitive hashing function, namely TLSH, to the symbols contained in the ELF files. In the case of statically linked and stripped files that contain no symbols, the hashing function is applied to the target addresses of the call instructions observed in the binary code of the program. One of the advantages of telfhash is that it is architecture-independent, so it can operate on binaries compiled for different platforms, such as x86 32bit, x86 64bit, and ARM.

Because of the locality-sensitive nature of TLSH, files with similar sets of symbols produce similar telfhash values. These values can be used as a similarity metric to identify closely related samples and cluster together malware families. Note that telfhash does not produce a normalized value between two fixed values (e.g., 0.0 to 1.0), but instead defines two samples as similar when their distance is below a certain number (the default is 50).

Figure 1 shows the confusion matrix of the ransomware families. Lighter colors indicate samples that were more similar. Note that this graph orders the samples by time of appearance to highlight their evolution over time.

Ransomware families

REvil

The REvil ransomware, also known as Sodinokibi, originally targeted Windows hosts, but released a Linux version in spring 2021.[29] Interestingly, this threat relies on the esxcli command-line tool to stop the current ESXi virtual machines (VMs). It then encrypts their on-disk images to prevent the recovery of running VMs. Recently, REvil actors have been targeted by a coordinated take-down operation[30] that may impact future variants.

DarkSide

The actors behind DarkSide initially distributed REvil ransomware but grew tired of sharing the profits with the REvil ransomware-as-a-service (RaaS) operator, so decided to create their own ransomware.[31] The DarkSide ransomware has been used to target a wide variety of organizations across North America and Europe. Most famously, the U.S. fuel distribution company, Colonial Pipeline, was held ransom by DarkSide, dramatically affecting gasoline distribution on the East Coast.[32] DarkSide initially targeted Windows hosts but quickly evolved to include Linux targets—and in particular, those running on ESXi servers.[33] These servers are usually targeted after the threat actors gain access to a VMware vCenter® deployment, often by means of stolen credentials.

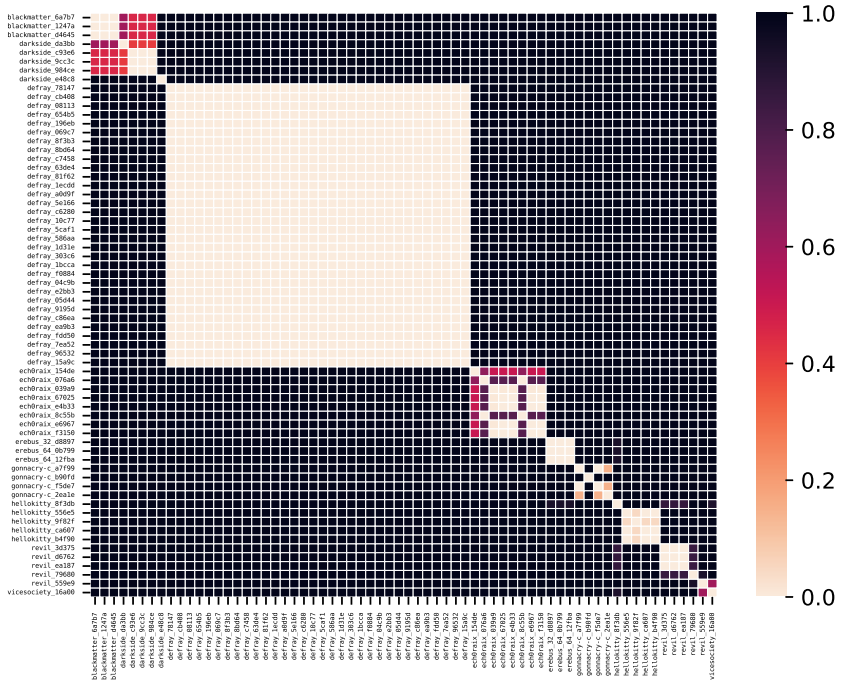

Figure 1: Code similarity between ransomware samples, based on telfhash (lighter color/lower distance corresponds to higher similarity).

From the confusion matrix in Figure 1, it is clear that samples that belong to the same family have a strong telfhash similarity. This indicates that one can build an effective classification system using telfhash. However, in some cases, the similarities diminish as time passes, showing that evolution within a family might lead to new code being introduced—possibly invalidating signatures developed for early versions of a threat.

From the graph, we see how DarkSide and BlackMatter samples share substantial portions of code, and that ViceSociety shares some code fragments with REvil. This accounts for the relationship between threats that have been highlighted in previous reports.

To further characterize the relationships between families of ransomware, we developed a similarity measure that leverages the term frequency–inverse document frequency (TF-IDF) algorithm applied to the hashes of the strings contained in the samples. This allows us to find the strings that best characterize each family. The confusion matrix in Figure 2 is based on the cosine similarity computed over the TF-IDF values for each string in the ransomware dataset we analyzed.

Ransomware families

BlackMatter

BlackMatter is considered an evolution of the DarkSide ransomware.[34] Interestingly, the actors behind BlackMatter made sure to publicly announce that they were not targeting specific verticals, such as healthcare, oil and gas, government, and critical infrastructure companies—possibly following the backlash that the Colonial Pipeline attack created, and the unwanted attention that the DarkSide operators received.

Defray777

Defray ransomware is another Linux-based threat that targets ESXi VMs.[35] An interesting property of some of its samples is that it doesn't strip or tamper with ELF binaries, which makes it easier to analyze. This ransomware family is closely related to RansomEXX[36]—to the point that sometimes the two families are considered to be variations of the same threat.

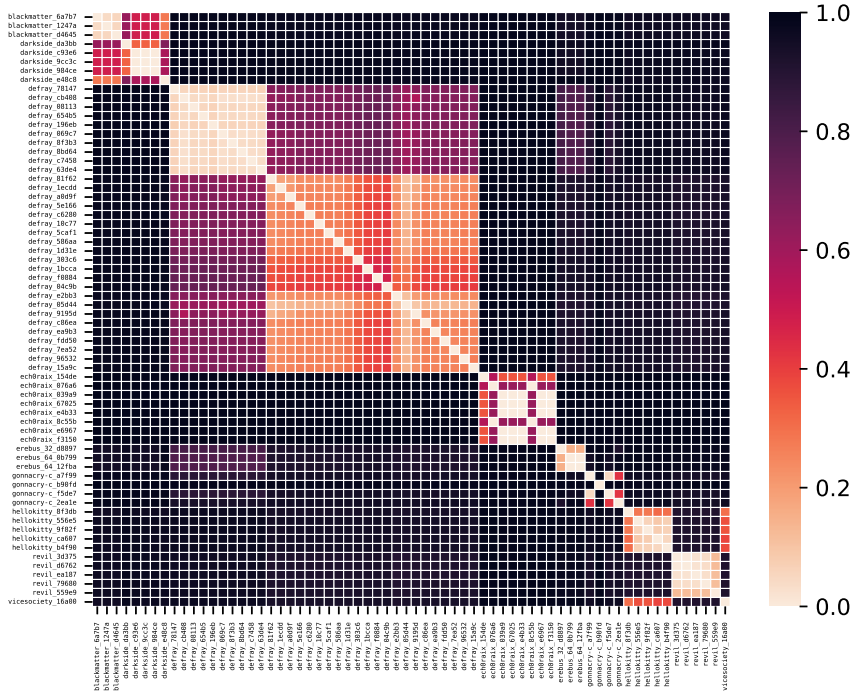

Figure 2: Similarity among ransomware samples based on TF-IDF applied to string hashes (lighter color/lower distance corresponds to higher similarity).

The use of string hashes for similarity shows that, in some cases, strings are better than telfhash to capture similarity among samples. For example, the REvil family is better characterized using this technique instead of relying on telfhash.

The use of the TF-IDF algorithm over the strings in the samples provides an opportunity for the creation of YARA rules for the identification of samples. We evaluated the YARA rules generated using our corpus of ransomware samples found running on Linux-based systems against the samples in the benign dataset, and we obtained a very low 0.01 percent false positive rate.

Another interesting way to characterize Linux-based ransomware threats is to look at the directories they avoid during the encryption process. For example, the **`/proc`** file system is a pseudo-file system that provides access to kernel-level information, using a file system-like interface. Attempting to encrypt the files under this directory could create an unstable system, jeopardizing the attack. The same issue is true for other directories, such as **`/bin`**, **`/usr/bin`**, and **`/lib`**, so they are explicitly avoided by ransomware targeting Linux-based systems.

Ransomware families

HelloKitty

The actors behind HelloKitty ransomware have achieved notoriety after successfully attacking CD Projekt Red, the makers of the Cyberpunk 2077 video game. It's another example of a Windows-based threat that evolved and expanded into the Linux world, targeting Linux-based systems and ESXi servers.[37] Like other samples that target ESXi VMs, HelloKitty uses the esxcli tool to stop the VMs currently running before encrypting their files.

ViceSociety

The actors behind ViceSociety ransomware are believed to have broken away from the HelloKitty group. Not surprisingly, their malware shows substantial similarities with the HelloKitty ransomware. This ransomware family was responsible for attacking the United Health Centers in the San Joaquin Valley in California, among other targets, which resulted in the leaking of sensitive patient records.[38]

We also looked at other paths that could indicate the use of specific tools as part of the attack. Most notably, the **`/sbin/esxcli`** command is used in ESXi environments to shut down existing virtual workloads so their on-disk images can be encrypted. Therefore, the presence of a reference to this tool is likely an indication of ransomware-like behavior.

Characterizing behavior

We used CAPA[11] to detect the capabilities of the Linux-based ransomware samples. Figure 3 shows the typical behaviors detected by CAPA aligned to the MITRE ATT&CK tactics and techniques framework. As we see from Figure 3, **defense evasion** is the most common tactic used in the samples. We also found various encryption or obfuscation techniques, such as Base64 encoding and AES-based encryption, used by the attacks to hide their code and data.

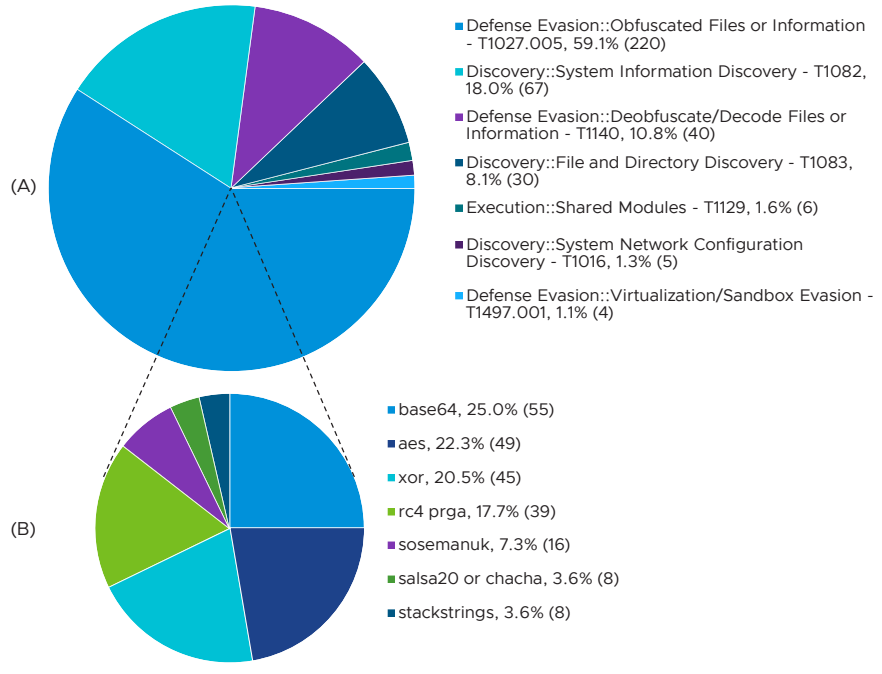

Figure 3: Malicious behaviors of the ransomware samples in the dataset: (A) Typical MITRE ATT&CK tactics and techniques leveraged by the samples; (B) Various encryption/obfuscation techniques used for defense evasion.

Ransomware families

Erebus

Erebus is a relatively older ransomware family. It initially targeted Windows hosts but evolved in 2016 to include a Linux variant.[39] This threat is unique because of its multilingual nature. While the actors behind the ransomware have stopped their activity, it is still an interesting sample that shares some behaviors with other ransomware families.

GonnaCry

GonnaCry is an open source ransomware sample written in C and Python.[40] While the Python version is mostly used as a way to showcase some of the behaviors associated with ransomware, the C version has actually been observed in the wild.

eCh0raix

eCh0raix ransomware targets QNAP network-attached storage (NAS) devices with weak credentials.[41] This family is written in the Go language, and its features are simpler than other ransomware families. For example, it appears that this threat does not have a way to distinguish among victims.[42]

Detecting and mitigating the threat

The solution to the ransomware threat is a combination of approaches, mechanisms and policies. Beyond having a solid data backup and recovery process, deploying an EDR solution that monitors the actions performed by processes on cloud workloads is critical to a defense-in-depth strategy. This must be complemented by an effective segmentation and NDR system that can recognize network-based evidence of attacks and ideally block the malware before it can take hold of the target hosts.

Cryptominers

Cybercriminals are not indifferent to the frenzy surrounding cryptocurrencies. The advantage of targeting cryptocurrencies is that successful attacks can be immediately and directly turned into cyber cash. Cybercriminals get instant reward without the need to perform cumbersome scams using stolen information, such as personal data, or by having to interact with victims of ransomware.[12]

Attacks targeting cryptocurrencies typically take one of two approaches. The first is to include wallet-stealing functionality in malware, sometimes posing as cryptocurrency-based applications.[13] The second is to monetize stolen CPU cycles to successfully mine cryptocurrencies, sometimes called cryptojacking. One of the first cryptojacking attacks was against Tesla's public cloud[14]—a Kubernetes deployment was hijacked and dedicated to mining the currency, while the computational costs were paid by Tesla. This notorious event was just the first in a series of incidents that targeted the CPU cycles of cloud environments. A report from Palo Alto Networks[15] indicated that cryptojacking affected "at least 23 percent of organizations that maintain cloud infrastructures."

Monero is the coin of the realm

Most cryptojacking attacks focus on mining the Monero currency, also known as XMR. Monero is an attractive target because it is known for preserving the privacy of its users, thanks to its use of Ring Confidential Transactions (RingCT), which hides the amount of each transaction, and stealth addresses that make transactions much more difficult to trace.[16]

Another driver behind the popularity of Monero among cryptojacking operators is the fact that it can be mined without needing specialized hardware. This differs from mining Bitcoin, which requires specific hardware, such as application-specific integrated circuits (ASICs), that can cost thousands of dollars.[17] With Monero, cryptominers can simply use the CPU or GPU cycles of ordinary computers, making compromised cloud workloads suitable for mining this cybercurrency.

Cryptominer families

XMRig

XMRig[43] is an open source miner available for Windows, Linux and macOS. While this miner is not a threat by itself, a variant of this component is often deployed as part of cryptojacking attacks to perform the mining. XMRig is often used to mine the Monero cryptocurrency, which is the preferred target because it can be mined without needing specialized hardware.

Sysrv

Sysrv is a botnet written in Go with cryptomining capabilities[44] that has been recently deployed against Kubernetes pods running WordPress.[45] The actors behind Sysrv rely on shell scripts to obtain persistency and hide the presence of the miner (e.g., by providing a modified version of the top command that doesn't show the CPU-hungry processes). This threat has also worm-like capabilities, attempting to spread to different hosts by leveraging SSH keys found on compromised hosts, performing password dictionary attacks against vulnerable services, and using an ever-increasing database of exploits against known remote code execution (RCE) vulnerabilities.

XMRig

The most common application used to mine Monero currency is the open source XMRig miner.[18] This popular application can mine different types of cryptocurrencies, but it's mostly used to mine Monero. Many of the cryptomining samples from Linux-based systems have some relationship to the XMRig application. Therefore, when XMRig-specific libraries and modules in Linux binaries are identified, it is likely evidence of potential cryptomining behavior.

If a binary is dynamically linked and not stripped, the task of identifying XMRig-related libraries and modules is trivial. However, stripped, statically linked binaries are challenging to analyze. In Table 1, we noted that all the cryptominers in our dataset are stripped. As a result, we needed to use function signature models, such as the ones produced by FLIRT,[19] to identify known libraries in C/C++ binaries, and tools, such as redress[20] to identify relevant modules in Go binaries.

We developed FLIRT signatures for the libraries used by XMRig when compiled on various Linux distributions. We also developed Go module detectors to identify relevant crypto-related modules.

When we checked for the presence of these components (written in both C/C++ and Go), we found that **89 percent of cryptomining attacks used XMRig-related libraries**. None of the benign Linux binaries linked those components, making the presence of these libraries and modules an effective way to identify cryptomining behavior.

Mining pools

When a cryptomining program is deployed on a compromised host, the program connects to a mining pool. By joining a mining pool, the malware can contribute to the overall mining process and share the benefits of collective mining—the computing power of a single host would likely be insufficient to achieve any meaningful results. Some common, well-known mining pools are **minexmr[.]com**, **nanopool[.]com**, and **supportxmr[.]com**.

Cryptominer families

TeamTNT

TeamTNT threat actors target open Kubernetes pods and Docker deployments to deploy XMRig cryptominers.[46] To evade detection, this threat hijacks the library loading mechanism to hide specific directories in the /proc file system, which are associated with the processes running the cryptominers.[47]

Mexalz

Mexalz threat actors are likely based in Romania,[48] exploiting weak credentials to compromise hosts and deploy cryptomining malware, which is mostly customized versions of XMRig.[49] The name of this threat is derived from the host that they use to store the components of the attacks, mexalz[.]us.

89%
of cryptomining attacks used XMRig-related libraries

In Table 1, we show the cryptomining pools associated with the families analyzed in this report.

Mining Pool	Port	Family	Note
194.145.227[.]21	5443	Sysrv	Proxy
80.211.206[.]105	6666	WatchDog	Private pool
monerohash[.]com		Mexalz, TeamTNT, XMRig	Public pool
moneroocean[.]stream		TeamTNT	Public pool
pool.hashvault[.]pro		XMRig	Public pool
pool.minexmr[.]com	5555	Sysrv	Public pool
pool.supportxmr[.]com	443	Mexalz, TeamTNT	Public pool
xmr-eu1.nanopool[.]org	14444	Sysrv	Public pool
xmr-eu2.nanopool[.]org	14444	Sysrv, WatchDog	Public pool
xmr.f2pool[.]com	13531	Sysrv, WatchDog	Public pool
xmr.pool.gntl[.]co.uk	40009	WatchDog	Public pool

Table 1: Mining pools observed in the analyzed cryptominers.

Cryptominer families

Omelette

Omelette[50] is a cryptomining worm that exploits known vulnerabilities in Exim, WebLogic and Confluence to install modified versions of XMRig. It spreads to other systems by exploiting trust relationships (e.g., the "known_hosts" SSH config file).

WatchDog

WatchDog represents one of the longest running cryptomining operations.[51] The name of this threat, which is written in Go, is derived from the presence of a component that is tasked to monitor the execution of the actual cryptomining program, similar to the Linux utility "watchdog." This operation has been running for more than two years, starting in January 2019, targeting both Windows and Linux hosts.

Kinsing

Threat actors behind the Kinsing cryptominer family are known to target vulnerable container-based deployments.[52] More specifically, the attack exploits Docker API endpoints that are open to the world to download and install a number of shell scripts that aid in persistence and lateral movement, and ultimately lead to the download of a cryptomining component.

In Table 2, we show the wallets that the analyzed cryptominers used to collect the mined currency. Even though these wallets can be used to identify relationships between different campaigns, the anonymous and transient nature of these indicators is a real barrier to meaningful tracking and attribution.

Monero Wallet Address	Family	Note
428uyvSqdpVZL7HHgpj2T5SpasCcoHZNTTzE3Lz2H5ZkiMzqayy19sYDcBGDCjobTfLBnc3t-c9rG4Y8gXQ8fJiP5tqeBda	TeamTNT	
43Xbgtym2GZWBk87XiYbCpTKGPBTxYZZWi44SWrkqqvzPZV6Pfmjv3UHR6FD-wvPgePJyv9N5PepeajfmKp1X71EW7jx4Tpz	WatchDog	Active
43zqYTWj1JG1H1idZFQWwJZLTos3hbJ5iR3tJpEtwEi43UBbzPeaQxCRysdjYTt dc8aHao7csi-Wa5BTP9PfNYzyfSbbrwoR	WatchDog	
49dnvYkWkZNPrDj3KF8fR1BHLBfiVArU6Hu61N9gtrZWgbRptntwht5JUrXX1ZeofwPw-C6fXNxPZfGjNEChXttwWE3WGURa	Sysrv	
4BrL51JCc9NGQ71kWhnYoDRffsDZy7m1HUU7MRU4nUMXAHNFBEJhkTZV9H-daL4gfuNBxLPc3BeMkLGaPbF5vWtANQn3gTBhyMDeJJJSvog	Mexalz	Active
82etS8QzVhqdiL6LMbb85BdEC3KgJeRGT3X1F3DQBnJa2tzgBJ54bn4aNDju-WDtpygBsRqcfGRK4gbbw3xUy3oJv7TwpUG4	Sysrv, WatchDog	Active
85X7JcgPpwQdZXaK2TKJb8baQAXc3zBsnW7JuY7MLi9VYSamf4bFwa7SEAK-9Hgp2P53npV19w1zuaK5bft5m2NN71CmNLoh	TeamTNT	
87q6aU1M9xmQ5p3wh8Jzst5mcFfDzKEuuDjV6u7Q7UDnAXJR7FLeQH2UY-FzhQatde2WHuZ9LbxRsf3PGA8gpnGXL3G7iWMv	WatchDog	
88ZrgnVZ687Wg8ipWyapjCVRWL8yFMRaBDrxtiPSwAQrNz5ZJBRozBSJrCYffurn1Qg7Jn-7WpRQSAA3C8aidaeadAn4xi4k	TeamTNT	

Table 2: Monero wallet addresses observed in the analyzed cryptominers.

Interestingly, at the time of writing, the Monero wallet address 82etS8QzVh…3oJv7TwpUG4 was actively mining on xmr.nanopool[.]org, as shown in Figure 4.

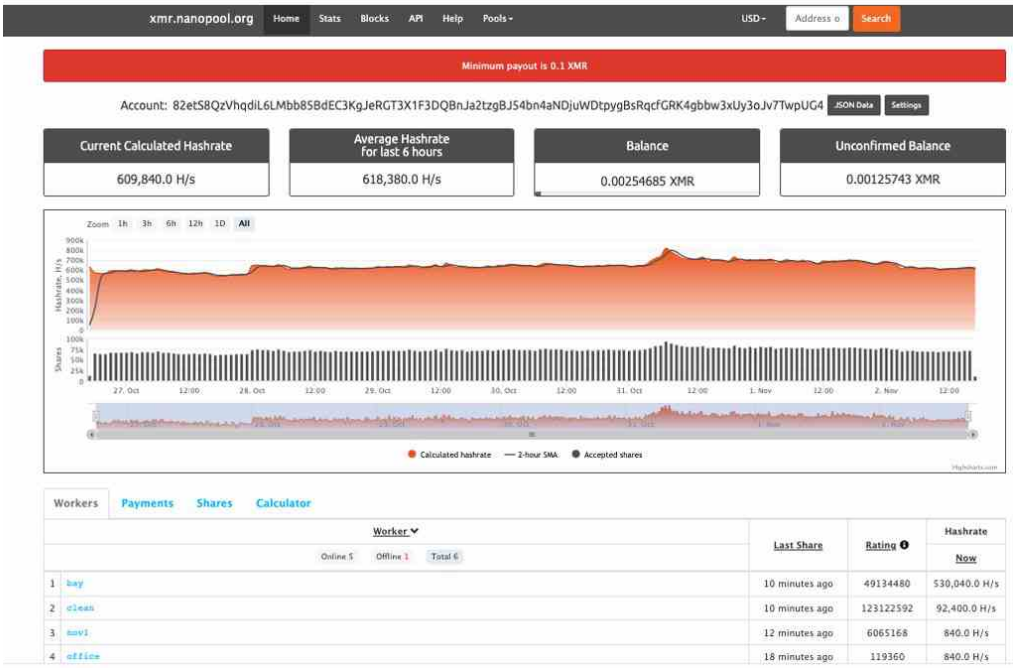

Figure 4: Current cryptomining operation on xmr.nanopool.org.

Even more interesting, the first payment was made in October 2020, more than a year ago (see Figure 5). This shows how long-lasting and difficult it is to eradicate these mining operations.

Figure 5: History of payments to the cryptominer's Monero wallet.

Characterizing similarity

We used both telfhash and our TF-IDF-based metric to characterize the similarity across the analyzed cryptominer families; the findings are shown in Figure 6. Because many of the cryptominers were packed, the analysis produced very poor results. For example, the similarity among samples of the Sysrv family was not captured.

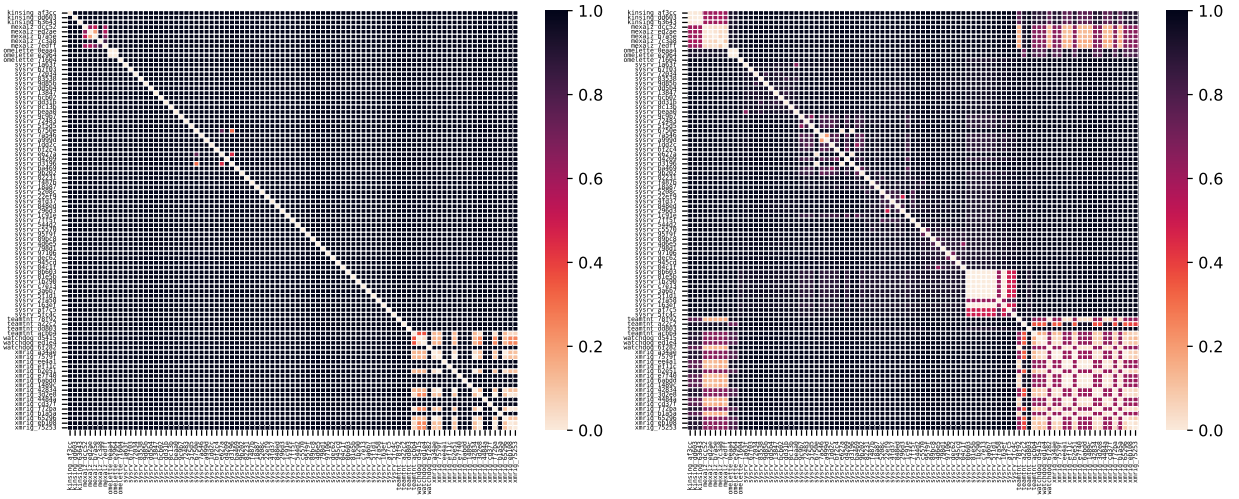

Figure 6: Code-based and string-based similarity between cryptominer samples (lighter color/lower distance corresponds to higher similarity).

Applying dynamic analysis makes it possible to unpack most of the samples, and Figure 7 shows the code and string similarity between samples after the unpacking process. The confusion matrix shows how some clusters of samples are clearly identifiable and how the samples evolve over time (see, for example, the various Sysrv family clusters). The results also show that, while telfhash is considered the state of the art for code similarity among ELF files, TF-IDF, which looks for similarity based on string hashes, provides a more useful characterizing metric.

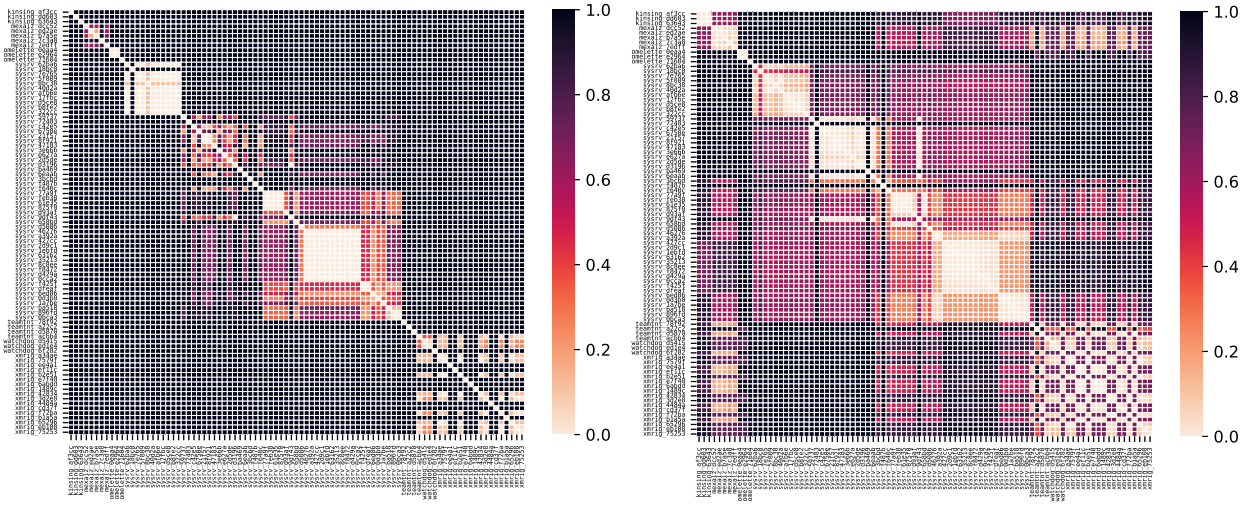

Figure 7: Code-based and string-based similarity between cryptominer samples after unpacking (lighter color/lower distance corresponds to higher similarity).

When we ordered the samples by similarity, not by time, the clusters became even more evident, as shown in Figure 8. In particular, the similarity between some TeamTNT samples, Mexalz and XMRig, as well as the similarities between WatchDog and XMRig, are clearly emphasized.

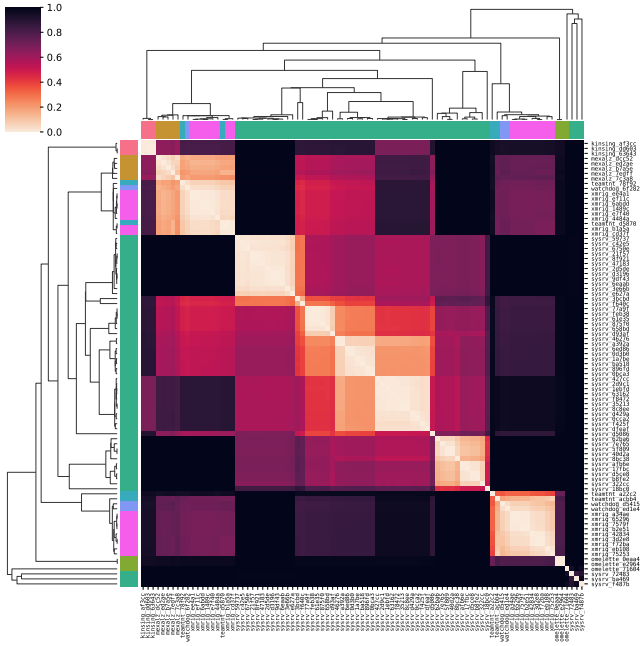

Figure 8: Cryptominers ordered based on string similarity (instead of family and time), showing the dendrograms that cluster together similar samples (lighter color/lower distance corresponds to higher similarity).

In addition to unpacking the cryptominer samples, we also monitored their execution in our sandbox. We found that a subset of the samples was de-obfuscating and executing additional code at runtime. We extracted these code fragments and analyzed their similarity against the model we built for the existing cryptominers. Note that the similarity is based only on the strings. The dynamically generated code is not in ELF format, therefore telfhash cannot be applied. This further shows that TF-IDF analysis on strings is an effective and flexible approach with wider applicability than telfhash.

The results of the analysis are shown in Figure 9. It shows that the generated code has a few clusters that represent small variations of the deployed code.

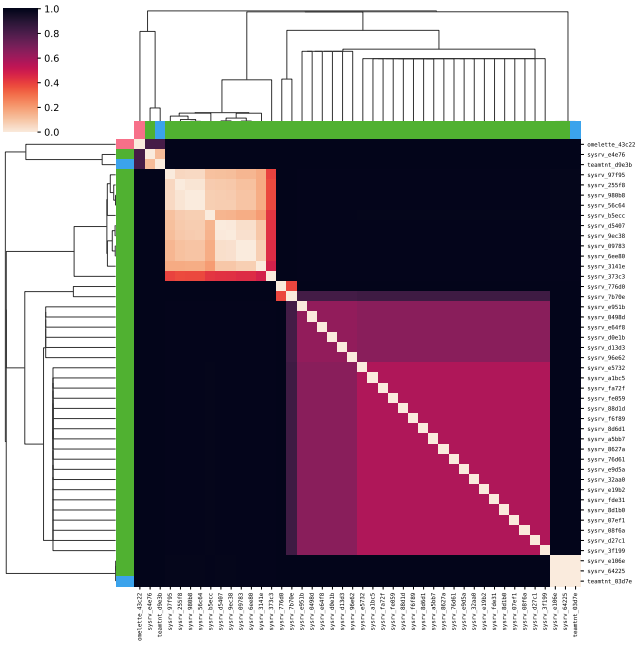

Figure 9: Similarity analysis of dynamically generated code.

Characterizing behavior

We also used CAPA to examine the behaviors of cryptomining samples. The malicious behaviors of the cryptomining samples are shown in Figure 10. Similar to the behaviors observed in the ransomware samples, defense evasion is the most commonly used technique by cryptominers. In terms of the encryption methods associated with defense evasion, it appears the techniques cryptominers use to obfuscate data are more diversified (Figure 10 (B)) in comparison to the ransomware samples discussed earlier (Figure 3 (B)).

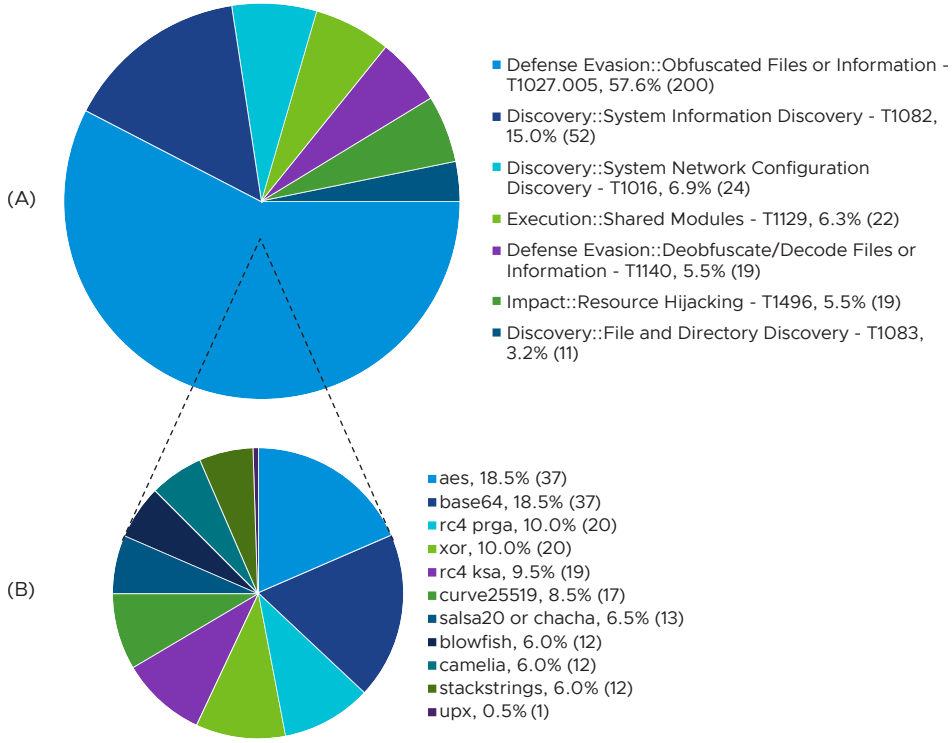

Figure 10: Malicious behaviors of the cryptominer samples in the dataset: (A) Typical MITRE ATT&CK tactics and techniques leveraged by the samples; (B) Various encryption/obfuscation techniques used for defense evasion.

Detecting and mitigating the threat

A cryptojacking attack might result in higher energy bills, stalled operations, or higher cloud computing costs. Unfortunately, these attacks can be tricky to detect because they do not completely disrupt the operations of cloud environments, like ransomware does, or raise alarms, like a data breach might when unauthorized or anomalous access to sensitive data is detected.

The best way to detect cryptojacking attacks is to use network traffic analytics (NTA) to identify internal hosts that are communicating the results of mining work to the outside, as this communication is required to monetize the attack. The communications to look for are connections to mining pools. However, many cryptomining

malware samples connect to a command-and-control host that acts as a network proxy to avoid being detected. In these cases, more sophisticated anomaly detection techniques are necessary to identify the threat. EDR solutions may be able to identify abnormal CPU usage patterns that can be directly associated with the calculations related to blockchain mining. Once again, the concerted monitoring of cloud environments using both host-based and network-based detection techniques can help keep these attacks at bay.

Methodology

Static and dynamic analysis

Linux programs, malicious or not, can be analyzed with a number of different techniques that can be categorized into two classes: static and dynamic analysis.

Static analysis looks at Linux binaries without executing them. These techniques either extract the meta-information provided by the ELF binary format or investigate the code and data segments that are part of the binary (e.g., looking for strings used by the program). For example, pyelftools[21] is a Python library that can extract information from ELF files. Another example is the FLIRT technology provided by the IDA Pro disassembler[22] that enables the identification of specific libraries in statically linked, stripped binaries, where all the symbols associated with the source code have been removed. Another interesting tool is redress,[23] which can be used to analyze Linux programs written in the Go language.

The advantage of static analysis techniques is that they are typically fast and don't require the execution of the program's code, which might include malicious actions. For example, the CAPA tool[24] can extract interesting behaviors without having to execute a sample. However, the main disadvantage of static analysis is that it is relatively easy to foil, using layers of obfuscation, packing and encryption. For example, the most used code-similarity function for ELF files, telfhash,[25] is of limited effectiveness when the files are packed.

Dynamic analysis looks at the actual execution of a program, usually in a controlled environment, such as a sandbox. These techniques are often resource-intensive and require extreme care to avoid the "spill out" of malicious actions or the fingerprinting of the analysis environment. However, the main advantage of dynamic analysis techniques is that they have the potential to expose the hidden behavior of malicious programs (e.g., encrypted portions of code).

The Linux kernel has recently introduced a technology, called eBPF,[26] that allows for the monitoring of programs with minimal overhead. Tools built on top of eBPF, such as Tracee,[27] can identify specific malicious behaviors, such as the unpacking of code or the presence of suspicious sequences of system calls.

In this report, we have applied a composition of static and dynamic techniques to characterize various families of malware observed on Linux-based systems.

Malware datasets

As part of this report, we are releasing a curated dataset of metadata associated with Linux binaries. All the samples in this dataset are public and therefore they can be easily accessed using VirusTotal[28] or various websites of major Linux distributions.

We started by collecting more than 11,000 benign samples from several Linux distributions, namely Ubuntu, Debian, Mint, Fedora, CentOS and Kali. We then collected a dataset of samples for two classes of threats, namely ransomware and cryptominers. These datasets have been manually labeled and vetted. This is a resource-intensive process that provides better labels than the ones provided by most automated analysis tools, which can be noisy and imprecise. Finally, we collected a dataset of malicious ELF binaries from VirusTotal that we used as a test malicious dataset. We started collecting the dataset in June 2021 and concluded in November 2021.

For each sample in each dataset, we looked at the general characteristics of the program, such as the architecture, the type of linking (static or dynamic), the presence or absence of symbols, the use of UPX for packing, and other traits. Figure 11 displays the characteristics of the files in our datasets, which show how different the characteristics of each group are.

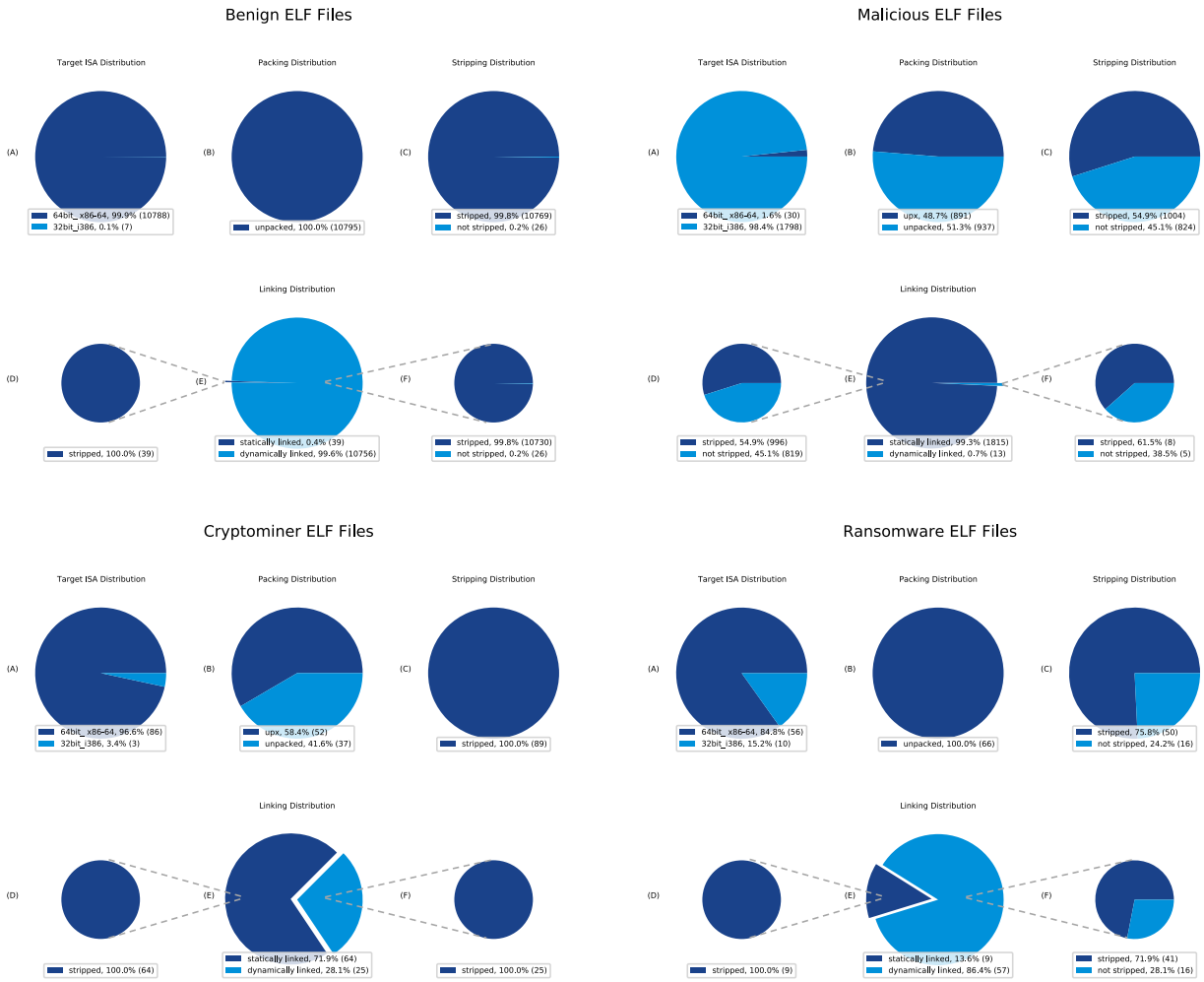

Figure 11: Characteristics of the samples in the datasets.

Remote access tools – Implants

An important aspect of an adversary's activity is how they compromise systems to gain control. This control allows them to persist within the environment, establishing a staging server they can use to pivot and target additional systems. Once an adversary has gained initial access to an environment, they enter the most difficult portion of the attack. They will need to find a way to leverage this limited access to gain a stronger foothold, while attempting to map out and find resources to accomplish their goal.

Attackers look to install an implant on a compromised system that gives them partial control of the machine. An implant, also known as a beacon, is what is generally regarded as the malware component of an attack. Its goal is to simply make regular network connections out to the command-and-control (C2) server to obtain new commands to execute and pass along the results.

In the context of an attack, what is an implant? Malware, webshells, remote access Trojans, and even known-good RATs can all be implants that install themselves on a compromised system to allow for remote access. An implant is deployed in a persistent manner to allow an adversary to keep control of that infected system. This is typically done in two ways: passive and active. A passive implant awaits an external connection, such as a webshell on a compromised server. Conversely, an active implant will continually try to send beacon messages to a preset C2 server and await further instructions. Analysis of RATs in the wild shows that most act as active implants that will communicate with known C2 servers to gather further instructions. Very few RATs make use of passive implants that rely on a service to listen for commands. While such malware could create its own listener, often they install a component into an existing service, such as a web server, to piggyback on existing network connectivity. However, the adversary would need to perform additional research of the infected system and face the risk of insufficient account privileges to install. Because a passive implant has no guarantee that the system will allow incoming connections, an active implant is the preferred tactic for malware.

From a malware analysis perspective, what is interesting is determining the number of implants that interact via well-known HTTP protocols, and those that create their own specialized protocols. An implant's purpose is far-reaching, depending on its author. Many are used for very simplistic purposes, such as to show files on the system, download new files, upload existing files, or execute commands.

Others allow for more advanced tactics, such as mapping out the local network and pivoting from the infected system into new systems on the network. Overall, depending on the adversary and their ability, an implant can contain a great amount of functionality, as documented in Figure 12.

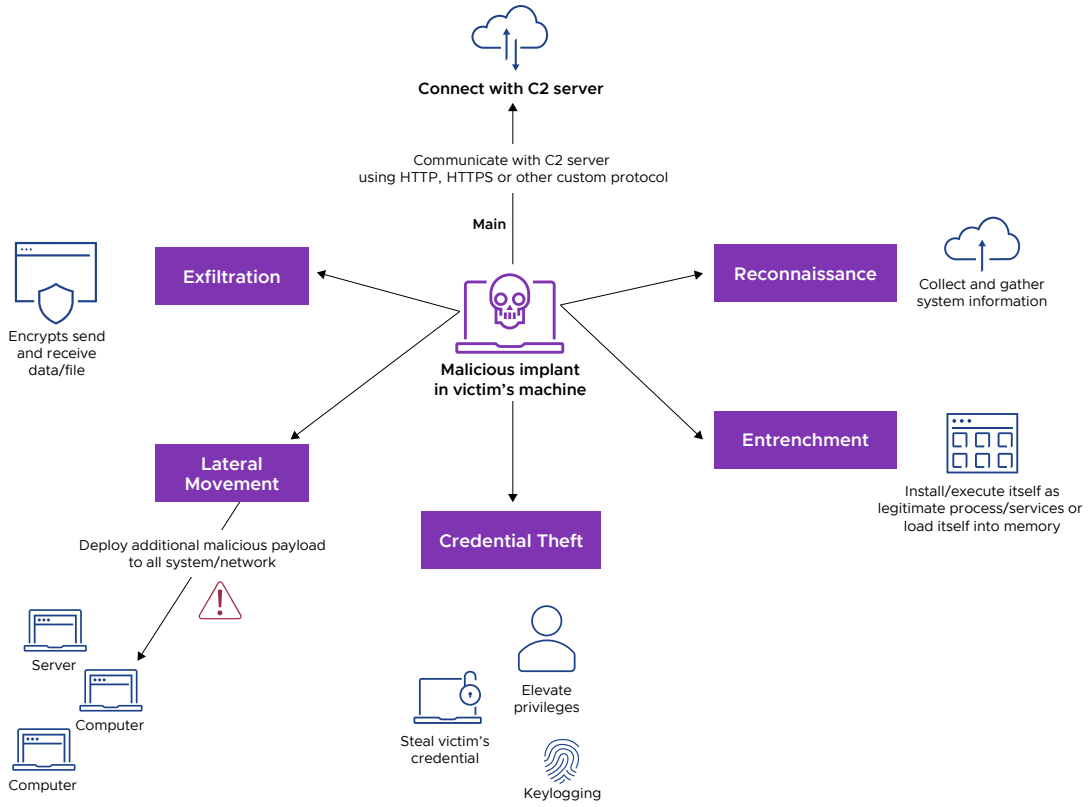

Figure 12: Malicious implant diagram.

Tactics used by implants

Implants often perform reconnaissance on systems in the area. For instance, they may scan an entire set of IP addresses to collect systems information and grab TCP port banner data. This can also allow the implant to collect IP addresses, hostnames, active user accounts, and specific operating systems and software versions of all the systems it detects.

Implants also rely on their ability to entrench themselves within their infected systems to persist. Their hope is to become background noise within a typical day's activity, showing up as just another Windows service or application to operate undetected. They can hide themselves in various ways, but on Linux-based multi-cloud environments, we often see their activities performed as routine cron jobs. Like the Scheduled Tasks within Windows, cron allows Linux, macOS and Unix environments to schedule processes to be executed at regular intervals. In this way, malware can implant itself onto a compromised system with a restart frequency of 15 minutes, so it can relaunch if it is ever terminated.

One of the more advanced tactics used by implants is that of lateral movement. Also known as pivoting, this allows an adversary to install additional implants within the environment, allowing them to jump to another system internally. From here, they can start gathering additional data about the environment from systems that may have additional access. In incident responses seen in the past, there are specific systems that have higher privileges to connect into protected enclaves for more sensitive data. These protected enclaves do not have remote access but could be accessed via multiple pivots, allowing a patient and careful adversary the ability to test access and find the weak center of the infrastructure.

Implants could be active for weeks, or even months, as the adversary figures out how best to carry out their objectives. In a typical smash-and-grab ransomware scenario, the actor may get lucky—they could sit dormant on a Windows server they've compromised and scan logons for just a few days before they catch a domain administrator. Once they've got these stolen credentials, they can quickly pivot and use them to force ransomware across an entire environment using built-in PowerShell scripting capabilities. However, this task is more difficult in a multi-cloud environment. The actor typically must remain quiet and unnoticed for as long as possible while they discover all the required resources.

Implants are also invaluable for collecting and exfiltrating data. Unlike large-scale data collection, which is typically done through tools such as rclone, implants may have the ability to exfiltrate directory listings of existing files and individual files. This allows a malware operator to collect superficial data for a malicious analyst to identify critical assets to steal.

Attack stages

These implants allow for additional stages of attack. While a first stage may be to exploit a vulnerability and install an implant, a second stage may be to download additional malware. This is often seen in Windows attacks via malware, such as TrickBot and QBot. The same exists in multi-cloud environments, as adversaries tend to specialize their toolsets based on functionality. The initial implant may have very specific capabilities, but it can be leveraged to download and execute ransomware or a cryptomining application, such as those mentioned earlier in this report.

Within the data center, adversaries may have to juggle access and connections to a variety of operating systems and services. In some cases, it could be necessary to pivot between systems instead of making direct connections, especially if some services are virtually segmented and only allow connections from a hard-coded set of systems. Lucky for them, it is completely possible to

Implants used by threat actors

Cobalt Strike and Vermilion Strike

Cobalt Strike is one of the most well-known and well-regarded red team tools on the market, sold as a legitimate service to select businesses. It uses HTTP, HTTPS and DNS to exfil information from a compromised network. Vermilion Strike is an open source threat emulation software based on Cobalt Strike's protocols, making it compatible with Cobalt Strike servers.

Merlin

Merlin[74] is a post-exploit C2 tool that communicates using the HTTP/1.1, HTTP/2 and HTTP/3 protocols. The server and implant are written in Go, so they can be cross-compiled to run on any OS platform.

traverse an environment using simple command lines and built-in accessibility tools, such as SSH. However, mapping out an environment, as well as the resources found on each system, is difficult for someone using direct connection techniques. This creates the need for an attack management tool.

Attack management tools

Attack management tools are good for both attackers and red team operations. They provide the ability to graphically organize assets, store collections of notes, and easily transmit properly structured commands to multiple compromised systems. At a basic level, they allow the actor to map out accessible IP addresses and hostnames within an environment and identify their operating systems. The actor can then target specific systems for exploitation, track legitimate user accounts and credentials, and document which systems have what assets.

A deeper look at Cobalt Strike – Vermilion Strike

The primary implant that this report focuses on is Cobalt Strike and its recent Linux-based variant, Vermilion Strike. To fully understand its impact on a multi-cloud environment, this report will dive into the implant's capabilities and methods of operation.

Cobalt Strike is a commercial penetration testing and red team tool. It allows a red team to simulate real attacks during their testing. Unfortunately, threat actors have found the tool useful, as well, thanks to its robust feature set, which makes it easy to remotely control victim machines once they are infected. Cobalt Strike uses an implant named beacon that, once deployed on a machine, will phone home to retrieve tasks to execute. Many threat actors simply use the Cobalt Strike beacon for their final payload delivery. The beacon implant is a Windows-only application.

In September 2021, Intezer reported[53] that they had discovered malware that appeared to be a Linux re-implementation of the Cobalt Strike beacon implant. Open source versions of Cobalt Strike's beacon implant exist, such as Geacon[54] and CrossC2,[55] but Vermilion Strike appears to be the first re-implementation of the Cobalt Strike protocol in the wild. Because Cobalt Strike is such a ubiquitous threat on Windows, its expansion to other operating systems, such as Linux, is notable. It demonstrates the desire of threat actors to use readily available remote-control tools to target as many platforms as possible.

Vermilion Strike appears to be implemented against version 3 of the Cobalt Strike team server. Its C2 configuration and communication appear to be the same as Cobalt Strike, but it only supports a handful of commands. These commands are explained in more detail in the technical analysis in Figure 13.

Implants used by threat actors

SSH backdoor implant

An SSH backdoor implant[75] can be loaded when a malicious actor exploits a control web panel (CWP) server administration web application and downloads a sshins installer binary. This drops a malicious shared library, /lib64/libs.so, and writes the name of the dropped file in the directory as /etc/ld.so.preload. When the OpenSSH service restarts, the malicious library will load and have the ability to inject its own code whenever sshd calls bind(). It then uses this hook to periodically beacon to the C2 server and exfiltrate sensitive data, such as CPU and OS information, OpenSSH configuration, and other critical data.

Linux C2 malware – RedXOR

RedXOR[76] masquerades as a polkit daemon. It is named for its network data encoding scheme, which is based on XOR. It communicates with a C2 server over a TCP socket and makes the traffic look like HTTP traffic. The C2 server sends commands to the implant via a command code that is returned in a JSESSIONID cookie. These commands can include collect system information, upload file, open file, execute shell command, and other tasks.

```
File Name   :   294b8db1f2702b60fb2e42fdc50c2cee-
                6a5046112da9a5703a548a4fa50477bc
File Size   :   89,416 bytes
MD5         :   3db3e55b16a7b1b1afb970d5e77c5d98
SHA256      :   294b8db1f2702b60fb2e42fdc50c2cee-
                6a5046112da9a5703a548a4fa50477bc
Fuzzy       :   1536:tMCIVGxHiGZsz9ZLTSKKTrcAFgtzgrSWUnCTOPS:tMCIVUbi
                z9VT1KTwAFgtzgrFO
Magic       :   ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
                dynamically linked, interpreter /lib64/
                ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[s
                ha1]=2322a87e5a86ac36f71d745a4b290772f4b3614e,
                stripped
```

Figure 13: Analysis of Vermilion Strike file on Linux.

Vermilion Strike configuration details

The malware has multiple encoded blocks of data that it loads at start-up. The first is a 4,096-byte block of data that is XOR-encoded with the key **0x69**. After decoding this first block, we can see that it is a group of type-length-value (TLV) encoded values consistent with a Cobalt Strike beacon. The use of the **0x69** decryption key implies that this beacon might be similar to Cobalt Strike 3.x beacons. Cobalt Strike 4.x beacons make use of the **0x2e** key to decrypt.

Next, the malware decodes five separate blocks of string data. These chunks of data have 12 bytes of a header and a variable length chunk of XOR-encoded data. As shown in Figures 14 and 15, the first 4 bytes always appear to be "**80 80 00 00**". The second 4 bytes make up a 4-byte key to be used with XORing the encrypted data. The third 4 bytes indicate the length of the data. In the example shown in Figure 14 and 15, we can see a key of "**e2 16 a2 de**" used to decode 181 (**0xb5**) bytes of data.

Implants used by threat actors

ACBackdoor malware

ACBackdoor[77] provides arbitrary execution of shell commands, arbitrary binary execution, persistence, and the ability to update malware on a compromised system. It communicates with a C2 server using the HTTPS protocol to send the information it collects as a Base64-encoded payload.

BlackTech – ELF_Plead

ELF_Plead[78] is a Linux version of a RAT used by the threat actor BlackTech. The configuration is RC4-encrypted, and a 32-byte encryption key can be found before the encrypted configuration. It uses a custom protocol to communicate with a C2 server. The data sent to the C2 server is RC4-encrypted and then LZO-compressed. The ELF_Plead command can provide arbitrary shell command execution and send/receive files, among other things.

```
00000000    80 80 00 00 e2 16 a2 de b5 00 00 00 cd 75 c3 fe    |....â.¢Þµ...ÍuÃþ|
00000010    cd 49 fd ab 96 7b 8c b9 8b 70 82 f1 92 7f da bb    |ÍIý«.{.¹.p.ñ..Ú»|
00000020    8e 38 c5 b7 84 36 8d b9 cc 66 cb a6 87 7a 82 f1    |.8Å·.6.¹ÌfË¦.z.ñ|
00000030    86 79 d6 f0 85 7f c4 fe cd 63 d2 ba 83 62 c7 ad    |.yÖð..ÄþÍcÒº.bÇ.|
00000040    cc 64 d1 ad c2 39 c4 a9 8e 7f cc b5 c2 39 c1 b3    |ÌdÑ.Â9Ä©..ÌµÂ9Á³|
00000050    c2 39 c1 a6 c2 39 d2 b7 9a 73 ce fe cd 7b c3 aa    |Â9Á¦Â9Ò·.sÎþÍ{Ãª|
00000060    81 7e 82 f1 94 7f d1 b7 96 38 c8 ad c2 39 ce b1    |.~.ñ..Ñ·.8È.Â9Î±|
00000070    83 72 82 f1 92 63 d1 b6 c2 39 d2 aa 88 36 8d b4    |.r.ñ.cÑ¶Â9Òª.6.´|
00000080    cc 77 c6 fe cd 71 c3 f0 88 65 82 f1 87 78 fd 8b    |ÌwÆþÍqÃð.e.ñ.xý.|
00000090    b1 39 c3 b2 8e 38 c8 ad c2 39 c3 bd 96 7f d4 b7    |±9Ã².8È.Â9Ã½..Ô·|
000000a0    96 6f 82 f1 ab 53 9b 9d 8d 7b d2 bf 96 40 cb bb    |.o.ñ«S...{Ò¿.@Ë»|
000000b0    95 5a cb ad 96                                     |.ZË..|
```

Figure 14: Vermilion Strike encrypted string data.

```
00000000    2f 63 61 20 2f 64 70 69 78 65 6c 20 2f 5f 5f 75    |/ca /dpixel /__u|
00000010    74 6d 2e 67 69 66 20 2f 70 69 78 65 6c 2e 67 69    |tm.gif /pixel.gi|
00000020    66 20 2f 67 2e 70 69 78 65 6c 20 2f 64 6f 74 2e    |f /g.pixel /dot.|
00000030    67 69 66 20 2f 75 70 64 61 74 65 73 2e 72 73 73    |gif /updates.rss|
00000040    20 2f 66 77 6c 69 6e 6b 20 2f 63 6d 20 2f 63 78    | /fwlink /cm /cx|
00000050    20 2f 70 69 78 65 6c 20 2f 6d 61 74 63 68 20 2f    | /pixel /match /|
00000060    76 69 73 69 74 2e 6a 73 20 2f 6c 6f 61 64 20 2f    |visit.js /load /|
00000070    70 75 73 68 20 2f 70 74 6a 20 2f 6a 2e 61 64 20    |push /ptj /j.ad |
00000080    2f 67 61 2e 6a 73 20 2f 65 6e 5f 55 53 2f 61 6c    |/ga.js /en_US/al|
00000090    6c 2e 6a 73 20 2f 61 63 74 69 76 69 74 79 20 2f    |l.js /activity /|
000000a0    49 45 39 43 6f 6d 70 61 74 56 69 65 77 4c 69 73    |IE9CompatViewLis|
000000b0    74 2e 78 6d 6c                                     |t.xml|
```

Figure 15: Vermilion Strike decrypted string data.

These five chunks of data are either separated by a space or comma, and contain the rest of the configuration data the malware uses when communicating with its C2 server:

- DNS servers
- GET URLs
- POST URLs
- Subdomains to use with DNS C2 traffic

Finally, the malware parses the beacon configuration. Although the malware uses the same structure as a real Cobalt Strike beacon, it only loads the configuration types shown in Table 3 from the beacon data.

Type	Name	Description
01	BeaconType	How to communicate with the C2
02	Port	What port to use when communicating with the C2
03	SleepTime	How often to check in with the C2
07	PublicKey	An RSA public key used to encrypt communication with the C2
08	C2Server	A list of server names and GET URL paths to use to check in
09	UserAgent	The User-Agent string to use in HTTP communication
10	HttpPostUri	The POST URL to use to send responses to the C2
13	HttpPost_Metadata	Additional data to set in POST requests to the C2

Table 3: Vermilion Strike configuration types.

Vermilion Strike setup

After loading the configuration, the malware proceeds to initialize the additional values it needs to communicate with the C2. These steps are similar to what a real Cobalt Strike beacon does.[56] The setup consists of the following steps:

1. Generate an array of 16 random bytes
2. Generate a SHA256 of the bytes
3. Use the first half of the SHA256 for AES keys
4. Use the second half of the SHA256 for HMAC keys
5. Load the RSA key from the beacon configuration
6. Collect and encrypt victim machine information:
   - Random number
   - PID
   - OS name
   - IP address
   - User name
   - Host name
   - Malware version number

The version number used in this specific sample of the malware is `1.0.1.LR.`

Cobalt Strike/Vermilion Strike C2 communication

After loading the configuration and performing the additional setup, the malware enters its main processing loop. Each time through the main loop, the malware will attempt to check in with the C2 server and then go to sleep for the time specified in the **`SleepTime`** beacon configuration. Depending on the **`BeaconType`** value, as previously mentioned, the C2 communication method will change. The malware currently supports hybrid HTTP DNS, HTTPS, and HTTP communication. There is an additional ICMP communication method in the code but no configuration option to select it. As Intezer noted, this might indicate this code is under development.

In the case of HTTP or HTTPS communications, a GET request is made to the server to check in. A cookie value is set with Base64-encoded data collected from the victim machine. The server responds to the GET request with any queued-up commands. The commands shown in Table 4 are supported by the malware.

Command	Name	Description
02	shell	Execute the command
04	sleep	Change how often the beacon calls home
05	cd	Change directory on host
10	upload	Upload a file to host (first chunk)
11	download	Download a file
19	cancel	Cancels a download that is currently in progress
39	pwd	Displays the current working directory
53	ls	List files in a folder
55	drives	List drives on current system
67	upload	Upload a file to host (subsequent chunks)

Table 4: Vermilion Strike supported commands.

The malware will execute the commands sent to it from the server and then send a POST request back with the requested information.

```
File Name          :  7129434afc1fec276525acfeee5bb08923ccd9b32269638a54c7b452f5493492
File Size          :  238,080 bytes
MD5                :  4baec501cd3c6318c8bceb4cf5c8b394
SHA256             :  7129434afc1fec276525acfeee5bb08923ccd9b32269638a54c7b452f5493492
Compiled Time      :  Wed Jun 26 02:59:19 2019 UTC
PE Sections (5)    :  Name        Size         MD5
                      .text       165,888      856639ce9212eb1329c8a59f89f0f97e
                      .rdata      51,200       590ccfa17cf705285509a4ae3ae50f38
                      .data       7,168        bfcb5a68d595cf49d2b372f35bbaacc5
                      .rsrc       512          09a004fff9ae1f2b5ff7ded5bcfaf389
                      .reloc      12,288       f6d8de448cad7e9a2587b75d8894c69d
Original DLL       :  gigabigsvc.dll
DLL Exports (1)    :  Ordinal Name
                      1    ServiceMain
Magic              :  PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
```

Figure 16: Vermilion Strike Windows file.

Vermilion Strike Windows and Linux differences

As an aside, the Windows version of the malware is almost identical to the Linux version with only a few differences. For instance, when collecting and encrypting victim machine information, the Windows version uses the following malware version numbers:

- 1.0.1.WR
- W1.0.1

Additionally, before entering the main loop, the Windows version will start a thread that attempts to read information from a named pipe and send the data to the C2 server using standard communication methods. Table 5 shows three additional commands the Windows version understands that are not available in the Linux variant due to differences in how the two operating systems manage named pipes.

Command	Name	Description
20	create_pipe	Create a named pipe
21	resume_pipe	ResumeThread on the named pipe thread
23	suspend_pipe	SuspendThread on the named pipe thread

Table 5: Vermilion Strike commands for named pipes.

These additional commands and threads seem to be related to Cobalt Strike functionality that allows the beacon to be injected into other processes and use a named pipe to communicate back to the main beacon when sending responses to the C2. Although the commands exist for controlling the named pipe thread in Linux, there doesn't seem to be any ability for the beacon to inject itself into other processes.

Metadata

The Windows version also provides additional metadata that might be useful. First is a PDB path, shown below. A PDB value is a file path stored within a Windows executable that has debugging enabled. This refers back to the original path on the malware developer's computer where the malware was compiled, which can provide insightful clues to the actor.

C:\workspace\spy\cobaltstrike-client-vc2008\Release\gigabigsvc.pdb

Second is the compilation timestamps of the beacons and stager binaries. The following unique timestamps are seen on the Windows binaries:

- 2019-06-26 02:59:19
- 2019-06-26 02:59:26
- 2020-09-12 14:35:36
- 2020-09-12 14:36:10

The fact that a lot of the compilation timestamps of the samples come from 2019 is a clue that this Cobalt Strike clone might have been written to be compatible with version 3.x.

Vermilion Strike compatibility with Cobalt Strike team server

Based on the analysis of malware samples, it appears this malware is most compatible with Cobalt Strike 3.x. After testing, a sample can connect and retrieve commands from the server but does not properly send back responses. There appears to be two problems.

The first problem is even though the beacon configuration is read and the POST URI decoded, the malware does not use this value. Instead, it selects at random from an array of POST URIs, which are loaded from the .rodata section. When sending responses to the server, it most often gets a **404** error returned, as seen in Figure 17.

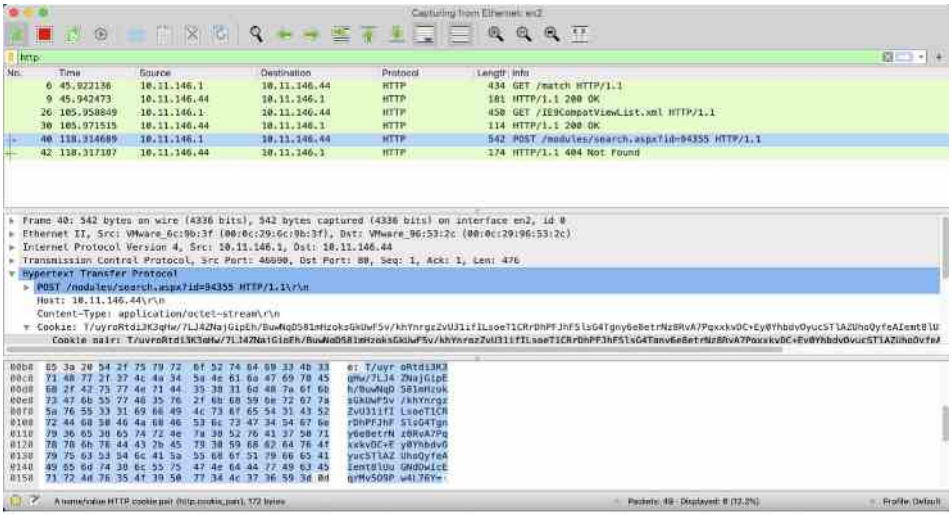

Figure 17: C2 POST error.

The second issue is that the POST URI is supposed to include a session ID, but the malware just uses a random number for this session ID. The result is that, even if the correct POST URI is picked at random, the server doesn't process the results because the session ID is unknown.

The VMware Threat Analysis Unit Cobalt Strike threat intelligence collection

Security practitioners often rely on the reputation of IP addresses to determine if traffic to and from that indicator of compromise (IOC) is malicious. However, the reputation is not effective for catching fresh malware C2 servers. In the example shown in Figure 18, antivirus (AV) engines detected an IP address as harmless (0/87)[57] on VirusTotal in September 2021, but in the VMware Threat Analysis Unit, we were able to identify it as a Cobalt Strike team server (C2).

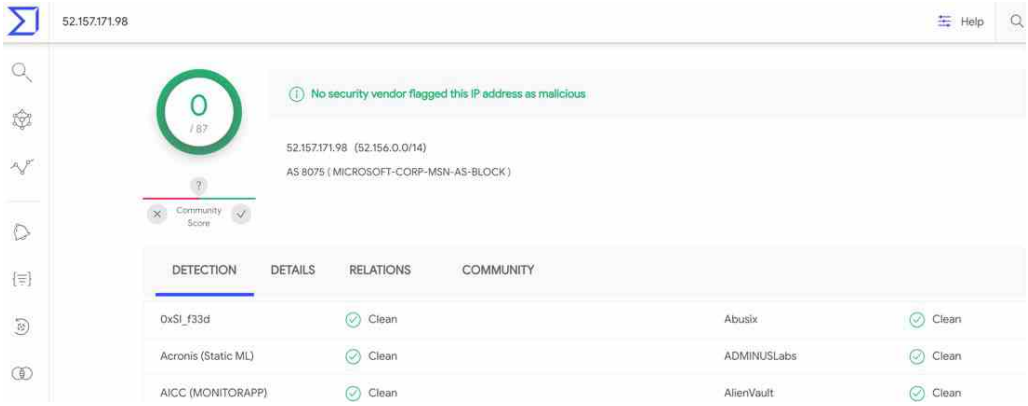

Figure 18: VirusTotal result against one IP address.

We looked at the DNS protocol, which we had reversed, and saw the protocols of high-profile malware families were emulated, especially those used for cyberespionage to discover real-time C2 instances on the internet. We utilized this intelligence to not only detect the threats but to also support incident response cases. The following section describes our findings on these Cobalt Strike threats.

Protocol overview and our approach

Cobalt Strike is split into a client and server component called a team server. An operator using a client GUI program connects to a team server after authenticating with a password through the TLS protocol.

A Cobalt Strike stager then downloads a main RAT module, the beacon, from the team server. The beacon will receive task (command) information from the team server and send back the results of the executed command. Both stager and beacon protocols for C2 communications are implemented in HTTP/HTTPS/DNS.

Additionally, Cobalt Strike allows third-party programs to act as a communication layer for the beacon's payload. In those cases, the beacon session is forwarded to a team server's External C2[58] service function, using a third-party client and controller. (For more Cobalt Strike protocol details, please see our presentation[59] for the Japan Security Analyst Conference (JSAC) 2021.)

To identify Cobalt Strike team servers, we focused on the staging process of each protocol, (HTTP/HTTPS/DNS/External C2) to ensure we:

- **Did not circumvent any technological measure, such as authentication for the discovery.** The client authentication protocol was not in our options because the emulation could be a login attempt by an unauthorized user to the team servers.
- **Avoided false positives.** Previous research (e.g., Fox-IT[60] and ZoomEye[61]) didn't differentiate the Cobalt Strike team servers and NanoHTTPD servers because they rely solely on the HTTP response header data. Similarly, we obtained responses from the team servers by sending requests based on the beacon protocol, such as HTTP/HTTPS GET requests, with the correct URI paths or arbitrary DNS A record queries. The team servers answered with a 200 OK status code in HTTP/HTTPS or a dns_idle value in DNS. However, we knew that a simple confirmation, without RSA/AES encryption in the beacon protocol would produce many false positives.

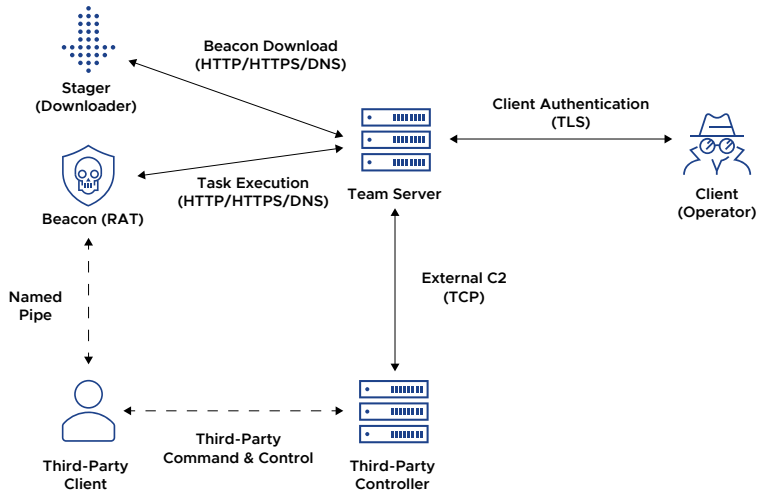

Figure 19: Cobalt Strike protocols overview.

- **Discovered team servers silently.** We emulated the beacon session encryption and then downloaded task information from the servers. However, once the emulation code checked in at the servers, an entry was created on the Cobalt Strike GUI console. Such a noticeable method is unfavorable for this purpose.

Therefore, we took the same approach as the one used by the Cobalt Strike developers[62] to emulate the stager protocol. Our implementation downloads beacon executables from Cobalt Strike team servers, and then decodes and parses their configuration blocks to obtain further information. The protocol requires no authentication, and the method will not lead to any false positives. We recognize that some security researchers and organizations analyze the Cobalt Strike team servers in the wild, using the stager protocol, but that doesn't cover **DNS and External C2 protocols**, and threat actors tend to prefer them to HTTP/HTTPS.

Additionally, since version 3.5.1, Cobalt Strike has an option to disable the hosting of payload stages for HTTP/HTTPS/DNS protocols (except External C2). Specifically, users are able to disable the stager protocol by just setting a line in the Malleable C2 Profile:[63]

```
set host_stage "false";
```

When disabled, it is impossible to detect the server by our method, but it gives Cobalt Strike users an alternative mechanism to deliver their beacons.

Observations since February 2020

Between February 2020 to November 2021, we discovered more than **14,000** active Cobalt Strike team servers on the internet.

Populations by protocol, version and customer ID

The percentage of each stager/beacon protocol is shown in Figure 20. The most popular protocol is HTTPS; the HTTP ratio increased 31 percent in January 2021 to 37 percent in November. We hypothesize Cobalt Strike users try to avoid a detection technique based on TLS handshakes, called JARM.[64]

Our discovery system guesses Cobalt Strike versions based on the collected beacon's configuration values. For example, if a SETTING_WATERMARK value (i.e., the customer ID[65]) is included in the configuration, the version must be 3.9 and later at minimum. In addition, a SETTING_DOMAIN_STRATEGY value indicates that the version is 4.3 and later. From our sample datasets, we found that close to 90 percent are version 4 and later (Figure 21). Regarding customer IDs, we found at least five customer Cobalt Strike IDs were cracked and leaked:

- 1873433027
- 305419896
- 16777216
- 1359593325
- 0 (trial)

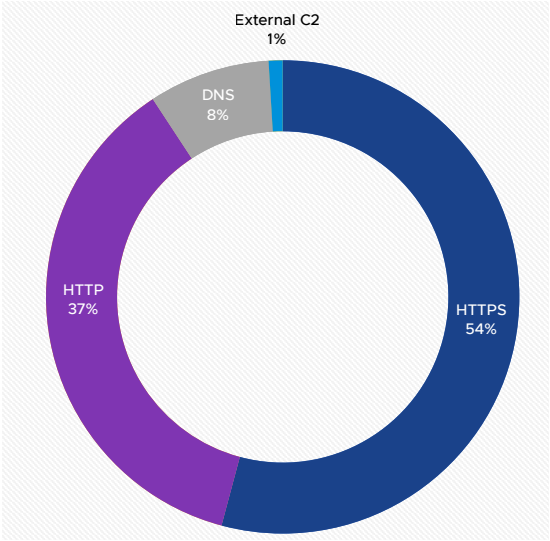

Figure 20: Population by protocol.

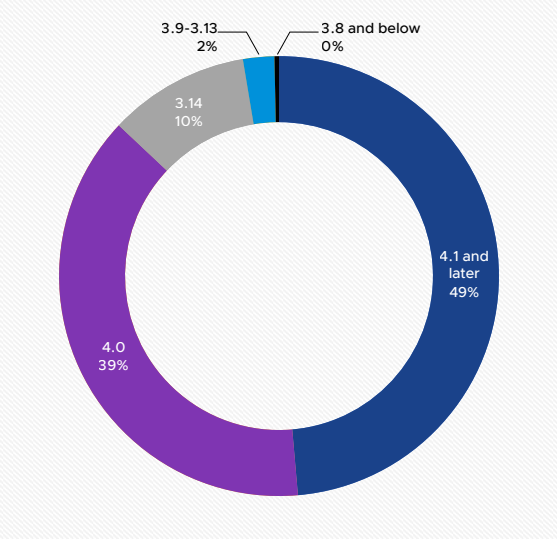

Figure 21: Cobalt Strike server population by version.

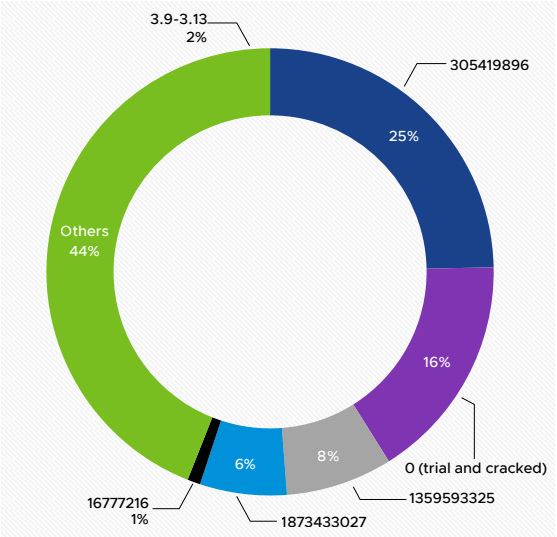

Figure 22: Population by customer ID.

As shown in Figure 22, the total percentage of cracked and leaked customer IDs is 56 percent. This means that **more than half of the observed Cobalt Strike users are using illegitimately obtained versions of the commercial software**.

Additionally, it should be noted that cracked trial license cases are increasing lately. Since Cobalt Strike version 3.6, the encryption of the beacon protocol is disabled[66] in the trial license. It can be checked by looking at the config value SETTING_CRYPTO_SCHEME. If it's 1, it's disabled. However, we noticed there are a lot of team servers with the value 0, even if the customer ID is 0 and the version is newer than 3.6, such as in the following parsed config output:

```
...
word CRYPTO_SCHEME (1 = disable encryption) at 0x746: 0 (0x0)
...
dword WATERMARK at 0x798: 0 (0x0)
...
```

We counted these servers as a part of the cracked customer IDs in Figure 22.

Change in the number of team servers obtained by a single scan

We discovered Cobalt Strike team servers targeting multiple protocols and ports. In Figure 23, we display the changes in the result of a single scan, which focuses on typical ports (HTTPS/443, HTTP/80, DNS/53 and ExternalC2/2222), and a scan that covers other ports as well.

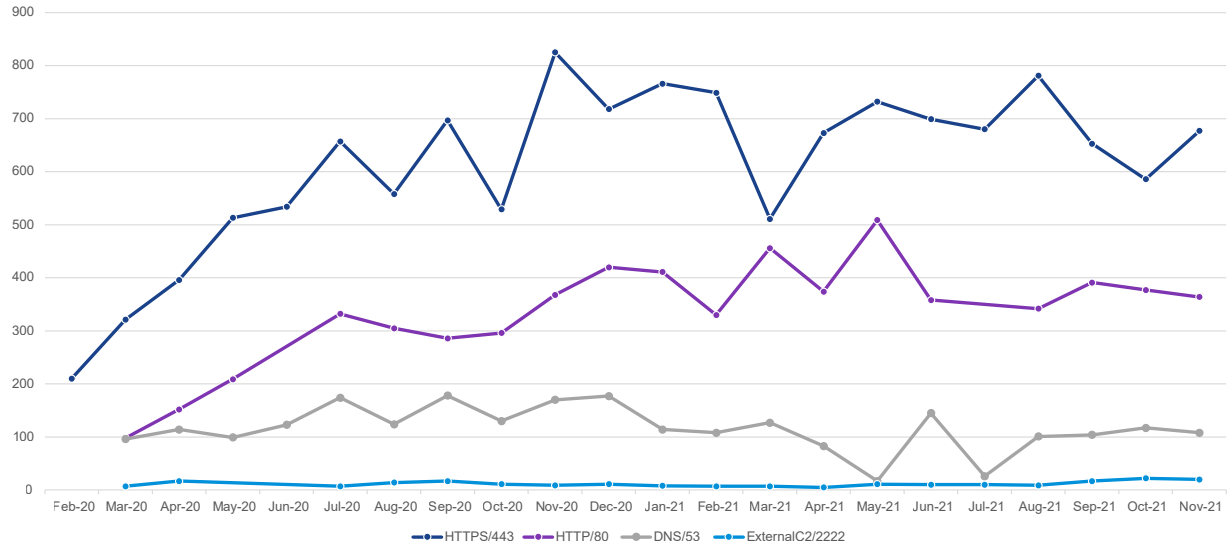

Figure 23: Change in the result of a single scan.

Since December 2020, the numbers look to decrease. However, we understand a part of Cobalt Strike users just disabled the stager protocol, rather than halting its usage. Security researchers should pay attention to the stager-disabled team servers to detect any changes.

Domain fronting observation (e.g., Microsoft Azure, Fastly)

Domain fronting[67] is a technique that obfuscates the intended destination of HTTPS traffic. Domain fronting takes advantage of routing schemes in content delivery networks (CDNs) and other services.

Specifically, the Cobalt Strike beacon configuration has different hostnames in C2 hostname (SETTING_DOMAINS) and HOST header values. Users can set the header values in either of two locations: in the HTTP config or the GUI menu. The Malleable C2 Profile[68] setting is part of the HTTP transform data of the config, while the config value SETTING_HOST_HEADER is part of the GUI menu setting. The latter setting will overwrite the former one.

We found the **most popular CDN abused by Cobalt Strike was Microsoft Azure, followed by Fastly**.

CDN	Host header value	C2 hostname examples
Microsoft Azure	*.azureedge.net	*.microsoft.com, *.msn.com, *.skype.com, *.visualstudio.com, *.azure.com
Fastly	*.global.prod.fastly.net	*nytimes.com, *yelp.com, *bbc.com, *usatoday.com, *forbes.com, *theguardian.com, *cnn.com, *stackexchange.com, *reddit.com

Table 6: Azure/Fastly domain fronting settings observed in Cobalt Strike team servers.

We found multiple Azure-fronted team servers with cracked and leaked customer IDs that were likely to be managed by threat actors, not red teamers. VMware researchers worked directly with Microsoft in January 2021 and, two months later, Microsoft stated[69] that the company decided to make a change to their policy to ensure that domain fronting will be stopped and prevented within Azure. Since then, the new Azure-fronted cases have been on a downward slope. From August to October 2021, they have been reduced to zero. Based on our monitoring, we hypothesize that Cobalt Strike users that were using Azure have migrated to Fastly or other services hiding a real IP address, such as Cloudflare Workers.

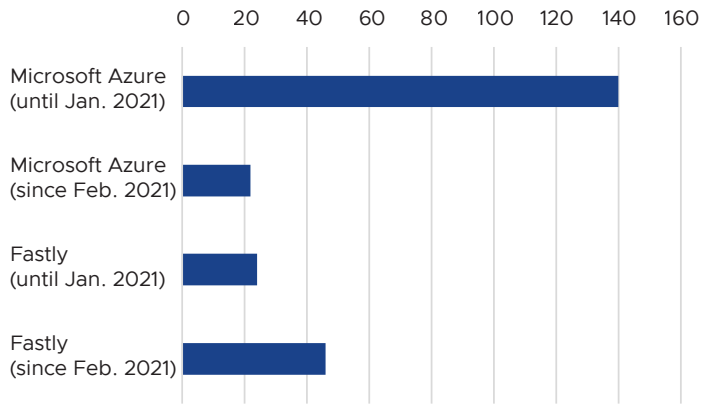

Figure 24: The change in number of new domain-fronted team servers.

Attribution to the specific threat actors

We parsed configuration blocks of beacons downloaded from team servers to categorize them into clusters based on config values. The clusters could be attributed to specific threat actors or campaigns. We primarily focused on two values in the configuration for this purpose: SETTING_PUBKEY and SETTING_WATERMARK.

SETTING_PUBKEY is an RSA public key in a DER format utilized in the beacon protocol encryption. The RSA key pair is created as a file, .cobaltstrike.beacon_keys, when starting and logging on to a team server for the first time. After that, if the team server directory is copied to another host, the copied one will have the same key pair file and the public key will be reused. Therefore, the two servers, whose beacons have the same public key, are likely to be managed by the same person or organization, unless the key pair file is leaked. If the file is leaked, the number of servers using the same public key can be huge.[70] We can exclude such a key for clustering.

SETTING_WATERMARK is a customer ID extracted from the authorization file, cobaltstrike.auth on a team server. According to Cobalt Strike's developers,[71] the ID is assigned to every team server of Cobalt Strike, version 3.9 and later, and is changed when running the update program. If a team server's WATERMARK is matched with another one, it means they are operated by a single actor. Even if the software package is not valid (leaked or cracked), the activities utilizing an invalid package are probably malicious. This is how we can differentiate servers managed by valid customers and ones abused by criminals.

Other than these values, some values in the configuration, such as HTTP header information, and the jar path hash value (SETTING_PROCINJ_STUB) may be beneficial for attribution work.

The undisclosed team servers owned by the threat actors (e.g., APT41)

Based on the SETTING_PUBKEY sharing, we were able to identify the undisclosed team servers owned by APT41 in the two attack campaigns.

Case 1: Campaign ColunmTK

Group-IB[72] discovered and shared a cyberattack on Air India and attributed it with moderate confidence to APT41 in June 2021. The campaign was codenamed ColunmTK. Based on the published network indicators, we found two undisclosed Cobalt Strike team servers.

The identification procedure is simple:

1. Search for the known IP addresses/domains in our datasets.
2. Obtain customer ID and public key MD5 values from the records.
3. Search for other IP addresses sharing the same values.

In this case, we utilized the four public key MD5 values extracted from three known servers. The customer IDs were not available for clustering as they were all cracked and leaked.

C2 IP address	Protocol/port	First seen	Last seen	Customer ID	Public key MD5
104.224.169.214	HTTP and DNS/5353, HTTP/80, HTTPS/443, DNS/53	2020/03/16	2020/09/03	0 (trial and cracked), 305419896 (cracked and leaked)	90419b03b90efe0c2c708294b40ced50, 64e69b07e15940bdb21e44bd3d7d9da4
185.118.164.198	HTTPS/443	2021/01/12	2021/01/12	305419896 (cracked and leaked)	99683533be317f513a70f40fb-db61cd6
185.118.166.66	HTTPS/443, HTTPS/8443	2020/12/06	2021/04/09	305419896 (cracked and leaked)	9cdb3fca6156c6cbed2f01d6431b3dfb

Table 7: Known team servers reported by Group-IB and observed by the VMware Threat Analysis Unit system.

By using the public key MD5 values, we obtained two undisclosed server IPs, which are shown in Table 8. We were able to even catch a new server (45.144.31.31) deployed after the Group-IB report publication.

C2 IP address	Protocol/port	First seen	Last seen	Public key MD5
149.248.62.83	HTTPS/443	2020/04/04	2020/04/04	90419b03b90efe0c2c708294b40ced50
45.144.31.31	HTTPS/8443	2021/06/14	2021/06/14	9cdb3fca6156c6cbed2f01d6431b3dfb

Table 8: Two undisclosed servers sharing the public keys.

Case 2: Cobalt Strike loader used by APT41

LAC,[73] a cybersecurity company in Japan, reported the campaign by APT41 using Cobalt Strike loaders in May 2021. Our system detected seven other servers, which are likely to be managed by the same actor.

C2 IP address	Protocol/port	First seen	Last seen	Customer ID	Public key MD5
104.168.30.164	HTTPS/443, HTTP/80	2020/12/04	2020/12/18	305419896 (cracked and leaked)	531c720aae6e053b9db9be8e7b56f78f
185.118.166.205	DNS/53	2020/12/12	2021/09/11	305419896 (cracked and leaked)	df50953714f29628a7f6a6c97eb0bc2e

Table 9: Known team servers reported by LAC and observed by the Threat Analysis Unit system.

C2 IP address	Protocol/port	First seen	Last seen
45.144.29.242	HTTPS/443, HTTP/80	2021/05/20	2021/06/03
185.250.151.18	HTTP/80	2020/12/31	2020/12/31
45.142.214.242	HTTPS/443, HTTP/80, HTTPS/8443, DNS/53	2021/04/25	2021/07/02
45.142.214.56	HTTPS/443, HTTP/80	2021/06/18	2021/07/19
45.67.229.168	HTTPS/443	2020/12/03	2020/12/03
45.153.231.194	DNS/53	2021/02/03	2021/02/03
194.156.98.214	HTTP/80	2021/01/26	2021/01/26

Table 10: Seven undisclosed servers sharing the public key 531c720aae6e053b9db9be8e7b56f78f.

Moreover, it should be noted that the earliest server was active at least six months ahead of both reports. The team server discovery, using protocol emulations, enables us to take proactive countermeasures.

Identifying potential targets

We collected manual proxy settings of the beacon configurations from team servers. The number of settings is small; however, they can be useful for identifying the victim organizations, based on the internal domain name, username and password. The victim information, based on the proxy settings, is listed in Table 11.

Victim industry	Victim country	Time period	Cracked/leaked customer ID?
Financial services	ES	2020/03-2020/06	No
Vertical transportation	CH	2020/04	No
Automotive	DE	2020/04-2020/08	No
Energy (oil and gas)	NL	2020/06	No
Airport	-	2020/06	Yes
Insurance	JP	2020/09	No
Networking equipment	FI	2020/09-2020/12	No
Insurance	US	2020/11	No
Financial services	FI	2020/12	No
Insurance	HK	2020/12-2021/01	Yes
Telecommunications	HK	2021/01-2021/03	Yes
Government	-	2021/03-2021/08	Yes
Financial services	GB	2021/05	No
Financial services	US	2021/06-2021/07	No
Hydrocarbon exploration	US	2021/08	No

Table 11: Victim information included in the manual proxy configuration.

Most team servers with the settings were likely to be owned by red teams, judging from the customer IDs. We contacted the victims and shared our findings – the team server had a cracked/leaked customer ID and was active at that time – so they could take steps to address the ongoing attack that was happening.

Detecting and mitigating the threat

RATs, such as Cobalt Strike and Vermilion Strike, pose a significant threat to enterprises. They are often used as the first stage of an attack, delivering additional information or even malware that allows threat actors to pivot and spread to other internal infrastructure. RATs typically gain an initial foothold via simple attacks, such as phishing emails.

We have discovered Cobalt Strike team servers in the wild for more than a year and a half. The fact that more than half of the servers have cracked and leaked customer IDs tells us that Cobalt Strike has become a commodity tool among criminals. A robust combination of NDR software and EDR solutions can help stop these attacks before they begin.

Looking for unknown applications executing in the environment or abnormal network connections is often an indicator of something larger going on. By actively monitoring and locking down the environment with NDR and EDR solutions, these malicious applications can be stopped before they have a chance to do real harm.

VMware recommendations

Organizations need to think of security as an inherent and distributed part of the modern enterprise, which must be incorporated into all aspects of the environment. Protecting multi-cloud environments starts with complete visibility into all workloads with detailed system context that makes it easier to understand and prioritize mitigation efforts. Information from all sources must be combined in an intelligent fashion that adds value, while enabling the sharing of this contextual data across teams to reduce silos.

This requires an EDR solution that can monitor the actions performed by processes on cloud workloads and implement effective segmentation to contain risks. In addition, organizations need an NDR system that can recognize network-based evidence of attacks and malicious lateral movements to ideally block the malware before it can take hold of the target hosts.

A secure multi-cloud environment requires securing all workload access and communications, both inside and across multiple clouds. Easily operationalizing security across clouds requires a scale-out architecture with software that gives the underlying infrastructure—firewalls, network detection and response, meshes, and load balancers—the same elasticity as modern, distributed applications.

A Zero Trust strategy can help organizations embed security throughout their infrastructure. Zero Trust offers a connected approach—joining users, devices, workloads and networks—to help organizations systematically address the threat vectors that make up their attack surface. Organizations can ensure they are implementing control points and distributing security across the infrastructure to better protect data and operations. With visibility, context, actionable insights, and control points embedded throughout the environment, organizations can start to spot and stop many of today's threats before they can even get started.

How VMware can help

VMware can deliver security as a built-in distributed service across your control points of users, devices, workloads and networks. With VMware, you can implement Zero Trust with fewer tools and silos, and scale response with confidence, speed and accuracy. When security becomes intrinsic, you reduce your attack surface to mitigate security risk, ensure compliance, and simplify security operations.

With VMware Security, you can deliver the speed and security required of the modern enterprise. You can transition to next-gen systems and modern applications, without increasing security complexity, and with dramatically fewer blind spots or choke points. Through vendor, agent and tool consolidation, you can achieve better security outcomes and deliver better employee and customer experiences, while spending less time on administrative tasks.

With VMware, you operationalize more of your security through your IT and development teams by creating a common source of truth and dramatically increasing your capacity to protect and defend your infrastructure. The authoritative context from the visibility, depth and accuracy of VMware's data collection enables security teams to confidently respond to events occurring within the organization's assets. This allows an organization's most critical assets—its people—to focus on high-value activities, using VMware's intelligent risk correlation with proactive prevention, detection and response capabilities.

VMware Security provides many capabilities to protect organizations from advanced threats targeting multi-cloud environments, such as ransomware, cryptominers, and remote access tools, as described in this threat report:

- Organizations focusing on protecting end-user solutions can utilize VMware Workspace ONE®, VMware Horizon®, and VMware Carbon Black Cloud™ to stop advanced threats from entering the environment.
- VMware vSphere®, VMware NSX® Advanced Threat Prevention™, VMware Carbon Black Cloud, CloudHealth® Secure State™, VMware Tanzu®, VMware vRealize® Suite, and VMware NSX provide organizations with robust capabilities to protect against, detect and respond to advanced threats in multi-cloud environments.

By partnering with VMware, organizations can capitalize on enterprise modernization efforts, continuously incorporating security into all aspects of the technology stack to accelerate Zero Trust strategies and achieve a more effective security posture.

References

1 W3Techs. “Usage statistics of operating systems for web sites.” Dec. 2021.

2 Microsoft. "HAFNIUM targeting Exchange Servers." March 2, 2021.

3 Cybersecurity and Infrastructure Security Agency. "Alert (AA21-291A BlackMatter Ransomware." Oct. 18, 2021.

4 CBT Nuggets. "Why Linux runs 90 percent of the public cloud workload." August 10, 2018.

5 Federal Bureau of Investigation. "Ransomware Actors Use Significant Financial Events and Stock Valuation to Facilitate Targeting and Extortion of Victims.” Nov. 1, 2021.

6 VMware. "Deconstructing Defray777 Ransomware." Threat Analysis Unit. March 11, 2021.

7 Mandiant. "Shining a Light on Darkside Ransomware Operations.” Jordan Nuce, Jeremy Kennelly, Kimberly Goody, Andrew Moore,Brendan McKeague, Jared Wilson. May 11, 2021.

8 VMware. “HelloKitty: The Victim’s Perspective.” Threat Analysis Unit. Sept. 9, 2021.

9 VMware. "Moving Left of the Ransomware Boom." Oct. 11, 2021.

10 Github. "trendmicro / telfhash."

11 Github. "mandiant / capa."

12 Security Affairs. "HelloKitty ransomware now targets VMware ESXi servers." Pierluigi Paganini. July 15, 2021.

13 Intezer. “Operation ElectroRAT: Attacker Creates Fake Companies to Drain Your Crypto Wallets.” Avigayil Mechtinger. Jan. 5, 2021.

14 Wired. "Hack Brief: Hackers Enlisted Tesla's Public Cloud to Mine Cryptocurrency." Lily Hay Newman. Feb. 20, 2018.

15 Palo Alto Networks. "Highlights from the Unit 42 Cloud Threat Report, 2H 2020." Oct. 6, 2020.

16 Investopedia. “The 6 Most Private Cryptocurrencies.” Shobhit Seth. July 4, 2021.

17 Miner Daily. "How Much Does it Cost to Mine a Bitcoin?" Sam Ling. May 4, 2021.

18 Github. "xmrig / xmrig."

19 hex-rays. "IDA F.L.I.R.T. Technology: In-Depth."

20 Github. "goretk / redress."

21 Github. "eliben / pyelftools"

22 hex-rays. "F.L.I.R.T."

23 Github. "goretk / redress"

24 Github. "mandiant / capa"

25 Github. "trendmicro / telfhash."

26 ebpf.io.

27 Github. "aquasecurity / tracee."

28 virustotal.com

29 Bleeping Computer. “REvil ransomware’s new Linux encryptor targets ESXi virtual machines.” Lawrence Abrams. June 28, 2021.

30 ARS Technica. "FBI, others crush REvil using ransomware gang's favorite tactic against it.” Tim De Chant. Oct. 22, 2021.

31 Crowdstrike. "Hypervisor Jackpotting, Part 1: CARBON SPIDER and SPRITE SPIDER Target ESXi Servers with Ransomware to Maximize Impact." Eric Loui-Sergei Frankoff. Feb. 26, 2021.

32 VMware. "Critical Infrastructure Remains at Risk Following Ransomware Attack.” Rick McElroy. May 11, 2021.

33 Mandiant. "Shining a Light on Darkside Ransomware Operations." Jordan Nuce, Jeremy Kennelly, Kimberly Goody, Andrew Moore, Brendan McKeague, Jared Wilson. May 11, 2021.

34 Insikt Group. "BlackMatter Ransomware Emerges as Successor to DarkSide, REvil." July 27, 2021.

35 VMware. "Deconstructing Defray777 Ransomware." Threat Analysis Unit. March 11, 2021.

36 Bleeping Computer. “RansomEXX Ransomware Linux encryptor may damage victims’ files.” Lawrence Abrams. Sept. 30, 2021.

37 Security Affairs. "HelloKitty ransomware now targets VMware ESXi servers." Pierluigi Paganini. July 15, 2021.

38 HIPPAA Journal. "Vice Society Ransomware Gang Attacks United Health Centers of San Joaquin Valley." Sept. 27, 2021.

39 Trend Micro. “Erebus Resurfaces as Linux Ransomware.” Ziv Chang, Gilbert Sison, Jeanne Jocson. June 19, 2017.

40 Github. "tarcisio-marinho / GonnaCry."

41 Anomali Threat Research. "The eCh0raix Ransomware." July 10, 2019.

42 Bleeping Computer. "New eCh0raix Ransomware Brute-Forces QNAP NAS Devices." Sergiu Gatlan. July 10, 2019.

43 Github. "xmrig / xmrig."

44 Cujo AI. “The Sysrv Botnet and How it Evolved.” September 22, 2021.

45 Sysdig. "THREAT ALERT: Crypto miner attack – Sysrv-Hello Botnet targeting WordPress pods." Stefano Chierici. August 26, 2021.

46 Palo Alto Networks. "Hildegard: New TeamTNT cryptojacking Malware Targeting Kubernetes." Jay Chen, Aviv Sasson, Ariel Zelivansky. Feb. 3, 2021.

47 Anomali Threat Research. “Inside TeamTNT’s Impressive Arsenal: A Look into a TeamTNT Server.” Tara Gould. Oct. 6, 2021,

48 hex-rays. "IDA F.L.I.R.T. Technology: In-Depth."

49 Bitdefender. “How We Tracked a Threat Group Running an Active Cryptojacking Campaign." July 14, 2021.

50 AT&T Business. “AT&T Alient Labs analysis of an active cryptomining work.” Fernando Dominguez. Jan. 9, 2020.

51 Unit 42, Palo Alto Networks. "Exposing a Cryptojacking Campaign That's Operated for Two Years." Nathaniel Quist. Feb. 17, 2021.

52 Aqua Security. “Kinsing Malware Attacks Container Environments.” Gal Singer. April 3, 2020.

53 Intezer. "Vermilion Strike: Linux and Windows Re-implementation of Cobalt Strike." Avigayil Mechtinger, Ryan Robinson, Joakim Kennedy. Sept. 13, 2021.

54 Github. "darkr4y / geacon."

55 Github. "gloxec / CrossC2."

56 NCC Group. “Striking Back at Retired Cobalt Strike: A look at a legacy vulnerability.” June 15, 2020.

57 virustotal.com. "52.157.171.98."

58 Cobalt Strike User Guide. "External C2."

59 VMware Carbon Black. “Knock, knock, Neo. Active C2 Discovery Using Protocol Emulation.” Takahiro Haruyama,

60 Fox IT. "Identifying Cobalt Strike team servers in the wild." Feb. 26, 2019.

61 Medium. “Identifying Cobalt Strike team servers in the wild by using ZoomEye (Part 2)." heige, KnownSec 404 Team. April 10, 2020.

62 Cobalt Strike. "Cobalt Strike Team Server Population Study." Raphael Mudge. Feb. 19, 2019.

63 Cobalt Strike User Guide. “Malleable Command and Control.”

64 Github. "Salesforce / jarm."

65 Cobalt Strike User Guide. "License Authorization Files."

66 Cobalt Strike. “Cobalt Strike Trial’s Evil Bit.” Raphael Mudge. Oct. 14, 2015.

67 MITRE ATT&CK. “Proxy: Domain Fronting.”.

68 Cobalt Strike. “High-reputation Redirectors and Domain Fronting.” Raphael Mudge.Feb. 6, 2017.

69 Microsoft. "Securing our approach to domain fronting within Azure." Eric Doerr. March 26, 2021.

70 NVISO Labs. “Cobalt Strike: Using Known Private Keys to Decrypt Traffic – Part 1." Didier Stevens. Oct. 21, 2021.

71 Cobalt Strike User Guide. “License Authorization Files.”

72 Group IB. “Big airline heist.” Nikita Rostovcev. Oct. 6, 2021.

73 LAC Watch. “Targeted attack by "Cobalt Strike loader" that abuses Microsoft's digital signature ~ Attacker group APT41." Yoshihiro Ishikawa. May 21, 2021.

74 Github. "Ne0nd0g / merlin."

75 Juniper Networks. “Linux Servers Hijacked to Implant SSH Backdoor.” Asher Langton. April 26, 2021.

76 Security Boulevard. "Detect C2 ‘RedXOR’ with state-based functionality.” Ben Reardon. April 20, 2021.

77 Intezer. “ACBackdoor: Analysis of a New Multiplatform Backdoor.” Ignacio Sanmillan. Nov. 18, 2019.

78 JPCERT CC. “ELF_PLEAD – Linux MalwareUsed by BlackTech.” Nov. 16, 2020.

Item No: Exposing Malware in Linux-based Multi-Cloud Environments 2/22