Hybrid Analysis of the Ragnar Locker Ransomware

1st Marthe Brendefur
Faculty of Applied Computing and Technology
Noroff University College
Kristiansand, Norway
marthe.brendefur@stud.noroff.no

***Abstract*—Ransomware is one of the most prominent and costly threats enterprises face. Attacks are estimated to cost organisations more than 20 billion dollars globally in 2021 [1]. The rapid development of malicious software and custom attacks requires the defensive side to evolve continuously. When an attack occurs, digital forensics techniques are applied to locate, gather, and preserve information relevant to an investigation. The Hybrid Malware Framework was developed to facilitate a holistic approach to analyse ransomware forensically [2]. This paper details the hybrid analysis of a Ragnar Locker strain. Ragnar Locker is a ransomware family that was first spotted in the wild in December 2019 and has since targeted several large companies. The attackers use an extract and extort tactic, meaning that they steal the victim's data to add extra pressure on the company. Findings include termination criteria, files exempt from encryption, forensic artefacts, and that the strain does not communicate over the internet. The uncovered information can be used to improve the defences against future Ragnar Locker attacks.**

***Index Terms*—Ragnar Locker, Ransomware, Hybrid Analysis, Ransomware Forensics, Malware Analysis, Static Analysis, Dynamic Analysis, MITRE ATT&CK**

I. Introduction

Ransomware is a form of malicious software that denies users access to their data. The name derives from the attackers' practice of demanding a ransom to be paid to decrypt the data. According to Purplesec [1], approximately 187.9 million ransomware attacks occurred in 2019. Ransomware has been the most significant cyber threat since 2016, much due to the high attack frequency and the damage it inflicts upon victims.

The Ragnar Locker ransomware family is used in big game hunting (BGH) operations targeting the energy, shipping - and travel sectors. The threat actor behind Ragnar Locker often utilises weaknesses in IT supply chains and managed service providers (MSPs) in their operations. The attackers, for instance, leverage MSP solutions to get access to networks, bypass security solutions, and spread their attacks. Moreover, the group pioneered a technique where ransomware is deployed inside a virtual machine on the victim system to bypass security mechanisms. Other prominent cybercriminal groups later adopted the technique. The encryption of the target system signifies the last stage of the intrusion. Exfiltrated data is then used as extra pressure to make the victim pay the ransom demand [3]. If the demand is not met, the threat actors publish proof of their intrusion on a designated data leak site (DLS). Ragnar Locker's modus operandi makes it suitable for analysis to uncover trending techniques applied by ransomware operators.

The strain will be analysed using the multi-disciplinary Hybridised Malware Framework (HMF) developed for malware forensics by Schmitt [2]. The framework concatenates existing methods for reverse-engineering malicious software with forensic examination techniques. The key element added by the HMF is network analysis, which helps map what and how a malware strain communicates. Investigating Ragnar Locker using a hybrid approach will uncover more of the strains' traits than what is possible with just using one technique.

Extracted Indicators of Compromise (IOC) and forensic artefacts can help security professionals detect and mitigate the threat posed by Ragnar Locker. Such findings can be used to write YARA rules and generate an overview of the adversary's tactics, techniques, and procedures (TTP). Custom ransomware often bypasses hash signature-based detection mechanisms, necessitating agile security solutions and defence-in-depth. By eliciting behavioural traits, one can distinguish actions performed by the ransomware and incorporate this knowledge into the defence.

The paper is structured as follows: Relevant research regarding ransomware and Ragnar Locker is presented under the Related Work-section. The methods used for analysing Ragnar locker is detailed in the Data Gathering-section, and the outcome is presented in Results. The Conclusion summarises the paper.

II. Related Work

Comprehensive ransomware attack analyses are seldomly offered freely online as breaches and threat actor capabilities are considered sensitive. However, some cursory analyses of dominating threat actors and their TTPs exist. Such a group operates Ragnar Locker. The group have previously been affiliated with the now-defunct Maze Cartel, and Ragnar Locker's achievements have been featured on MountLocker's DLS. Despite these affiliations, there are no indicators that the Ragnar Locker group cooperates with access brokers [4]. The group likely facilitate and conduct all stages of an operation themselves. Ragnar Locker has targeted several profiled organisations through 2020. The cybercriminals publicly extort their non-paying victims on their DLS ”Wall of Shame” [3]. Ragnar Locker is classified as an advanced ransomware due to how it is coded and the techniques it employ [5]. Ragnar Locker has a somewhat unique code structure and implement obfuscation and evasion techniques. The capabilities of a strain reflect the operational maturity. Advanced ransomware often has a covert and overt phase [6]. Attackers aim to stay undetected while moving laterally and exfiltrating data. The visibility in the overt phase is caused by the encryption of data, which, together with the ransom demand, comprise the main characteristics of this form of malicious software [6].

A. Attack Vectors

An attack vector is the method used to gain unauthorised access to a system [7]. Threat actors can use several approaches to launch an attack, ranging from exploits and flaws to physical measures [7]. A significant amount of attacks start with some form of social engineering [8]. Social engineering is when a threat actor exploits human nature, often by deceiving the victim to perform an action or reveal information [8]. The most common method is phishing, which is when an adversary gets an unsuspecting person to click on a malicious link or document through an email. However, operations where threat actors gained access through technical exploits increased drastically through 2020. Such operations consist of scanning the target for vulnerabilities, gathering information about the technical structure, and exploiting the weaknesses [7]. One feature commonly exploited is the Remote Desktop Protocol (RDP). RDP enables a device to be connected to remotely. After the initial compromise, threat actors will try to establish persistence on the system.

B. Attack Infrastructure

Threat groups must have a suitable infrastructure in place to facilitate extensive attacks [9]. Command-and-Control servers (C2 or C&C) are core elements in such a setup. The C2 structure generally consists of numerous servers situated at different locations, and threat actors tend to reuse much of their established infrastructure in different attacks. C2s have several use-cases, such as temporary storage of stolen data, communication with beacons in breached environments, key generation, and masking the attacker's location [9]. The attack infrastructure is the backbone of most large operations.

C. Analysis Methods

The practice with hybrid analysis arose to interpret better how malicious software operates on a system. The hybrid analysis combines the strengths of static and dynamic analysis techniques [10]. The hybrid approaches involve additional focus areas, such as memory forensics and network analysis.

1) Analysis Framework: Eskandari, Khorshidpour, and Hashemi [11] developed a technique where the dynamic analysis is performed first to map the actual Application Programming Interface (API) calls made. The extracted results from the dynamic analysis are combined and further analysed using static methods. One of this approach's main benefits is that it minimises time spent on dynamic examination and bypasses the challenges with packing. Code that is packed will exist in a compressed state while stored and extract itself in memory once executed [12]. However, most approaches to hybrid analysis start with static analysis. The methodology applied by the HMF is to start with static analysis, followed by the dynamic, network, and forensic analyses [2]. To summarise, the uncovered behavioural traits of the analysed species are plotted into the MITRE ATT&CK framework. MITRE have indexed offensive techniques and assigned them an ID. Applying standardised definitions is convenient when generating overviews and comparing TTPs.

2) Static Analysis: Static analysis is an umbrella term for techniques that inspect malware without executing it. Static analysis techniques are useful for examining the source code, binary strings, Windows Portable Executable (PE) files, and the execution paths of the malicious software [10]. One of the main benefits of static analysis is that it is relatively safe, as the malicious program is never executed [10]. It can also be reasonably quick, which is convenient during triage. However, if the code is packed or obfuscated, an in-depth analysis using static techniques will be complicated and time-consuming.

3) Dynamic Analysis: Dynamic analysis is performed by executing programs and analysing their behaviour at runtime. Ransomware is commonly run and monitored in a controlled environment, such as a sandbox. One of the main benefits of dynamic analysis is that the researcher can observe how the strain interacts with the system, such as API calls made and behaviour in memory [12]. Many ransomware strains are aware of their environment. If such a strain detects that it is run in a sandbox or debugger, it will likely refrain from executing the malicious payload [12].

4) Forensics Analysis: A digital forensic examination is a process that uses science and technology to assess digital objects to answer questions about events that occurred [13]. Artefacts that exist due to malicious software strains are often concatenated into a list with identifiable traits, or Indicators of Compromise (IOC). IOCs are a subtype artefact; an artefact is a piece of forensic data related to an event, while an IOC is a forensic item directly related to the threat [14]. Collecting, systemising, and implementing IOCs is a part of the threat intelligence cycle.

D. Existing Analysis of Ragnar Locker

Examinations of completed attacks have shown that the Ragnar Locker operators gain access to a targeted entity's network through RDP and MSP exploits. The group acquires and exfiltrates the targets data before manually deploying the Ragnar Locker payload on the victim's domain [3].

Ragnar Locker will terminate itself on systems where the strings in `LCIDLOCALE\_SYSTEM\_DEFAULT` matches that of a language used in the Commonwealth of Independent States (CIS) countries [3]. After checking the language settings, Ragnar Locker starts the encryption process. The ransomware enumerates directories on the victim's system and copies the ransom note named ”RGNR_⟨ID⟩.txt” to all directories except for system directories and web browsers, which are ignored [3]. The threat actors still need the operating system (OS) to function so the victim can access a web browser and pay the ransom. The files are encrypted using an RSA 2048-bit public key, and “_RAGNAR_” is added as a footer within the files [15]. At this stage, the extension of the encrypted files has changed to ”.ragnar_⟨ID⟩”, where ⟨ID⟩ represent an eight-digit victim ID [15]. Tavares [3] found that the Ragnar Locker strain did not check if a file had already been encrypted, so if invoked again, the file would be encrypted a second time. However, the analysis performed by Blaze Information Security [15] contradicts this, stating that Ragnar Locker will not encrypt a file if the footer “_RAGNAR_” is detected in it. As the algorithm used will be the same in both instances, the only consequence would be that encrypted data must be decrypted an equal number of times.

III. Data Gathering

Due to the economic and operational costs of having data encrypted, it is imperative that ransomware is detected at an early stage. By examining a Ragnar Locker strain using the HMF, this paper aims to uncover traits that can be used to profile the Ragnar Locker ransomware family in a Windows 10 environment. The analysed strain has a SHA-256 value of `9bdd7f965d1c67396afb0a84c78b4d12118ff377db7efdca4a1340933120f376`. The analysis will be conducted on a Windows 10 virtual machine. To account for Ragnar Locker’s reaction to a closed environment, network connectivity will be simulated to overcome the strains’ evasion techniques. However, none of the existing public analyses of the stain suggests that it communicates over the network. The strain will first be inspected using static analysis techniques, followed by dynamic, network, and forensic analyses.

IV. Results

The strain does not appear to be masquerading as legitimate software, nor claim to be anything other than Ragnar Locker. As it does not hide or attempt to lure a victim into executing it, this may indicate that the operators manually deploy Ragnar Locker on the compromised system. According to the DLS, the operators view themselves as penetration testers and bug-hunters. The group claims to search for weaknesses in corporate networks and are willing to provide penetration reports to victims who pay the ransom. They also offer to help fix the issues. Such a mindset corresponds with the emerging trend of operators manually conducting some of the attack procedures.

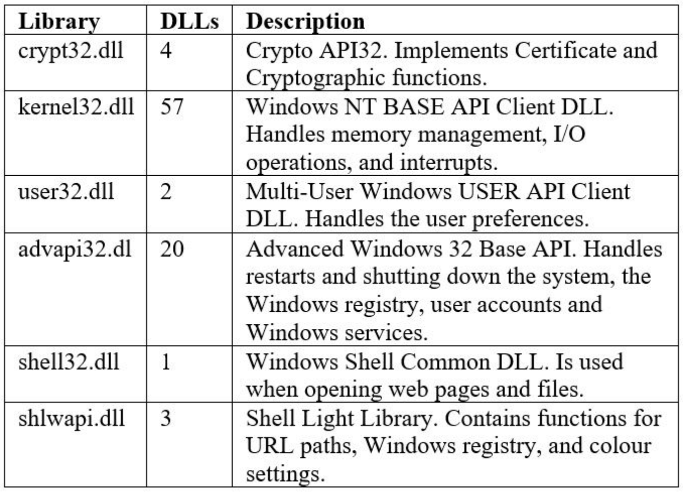
A. Static Analysis

The static analysis was performed using single-purpose tools to determine PE information, entropy, and strings. The x32dbg debugger and the IDA Freeware disassembler were used to analyse the strain’s structure and functions at the assembly level.

1) PE Structure: According to the file header, the strain was compiled at 16:36:20 Friday 31. January 2020. The executable is a 32-bit PE file, indicating that it is created for 32-bit processors. Although 64-bit processors are more common for the Windows 10 OS, using the 32-bit architecture ensures better backwards combability. 32-bit programs also use fewer memory resources, making them suitable for computers with 4GB of RAM or less. The use of 32-bit indicates that the developers designed Ragnar Locker to infect a wide range of OS versions. The strain is compressed when at rest and unpacks itself in memory when executed. Profile-guided Optimisation (PGO), a C++ compiler from Intel, is likely the compression method used. PGO compresses files in three steps and is suitable for processor-intensive applications with few variations in applied data sets.

2) Windows API Class: The Ragnar Locker strain imports six native Windows libraries, as seen in Table I. The Windows application programming interfaces (APIs) are standard for Windows machines. Windows APIs are used to invoke lower-level functions. Relying on these API Classes help Ragnar Locker blend with the environment and reduce the need to import tools.

TABLE I
DLL libraries used by Ragnar Locker.

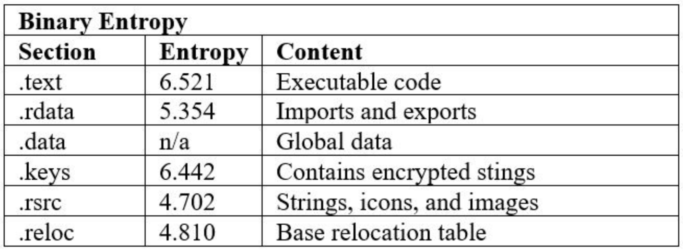


Library	DLLs	Description
crypt32.dll	4	Crypto API32. Implements Certificate and Cryptographic functions.
kernel32.dll	57	Windows NT BASE API Client DLL. Handles memory management, I/O operations, and interrupts.
user32.dll	2	Multi-User Windows USER API Client DLL. Handles the user preferences.
advapi32.dl	20	Advanced Windows 32 Base API. Handles restarts and shutting down the system, the Windows registry, user accounts and Windows services.
shell32.dll	1	Windows Shell Common DLL. Is used when opening web pages and files.
shlwapi.dll	3	Shell Light Library. Contains functions for URL paths, Windows registry, and colour settings.

3) Entropy: Entropy analysis can help determine whether an executable file is compressed or obfuscated. When analysing digital files, the entropy is measured by calculating the randomness of bytes. The result is measured on a scale from 0 to 8, where 0 indicates low entropy and 8 high entropy. Ordinary files usually have an entropy of around 5. By examining the Ragnar Locker strain using PeStudio and Detect It Easy, it becomes evident that parts of the strain are obfuscated or compressed. The `.text`-section has an entropy of 6.521, and the `.keys`-section has an entropy of 6.442. The `.text`-section contains the program code, and the high entropy indicates that the code is packed. As for the `.keys-section`, it is expected that the entropy is high, as it contains encryption data. The sections and their entropy is listed in Table II.

4) Termination Criteria: The strain will abort its execution based on the computer locale. Ragnar Locker uses the `GetLocaleInfoW()` function to determine the OS language settings. If it detects that it is in an environment that matches one of the pre-defined countries, it will obtain its own process

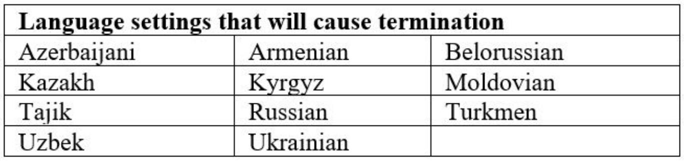
TABLE II
ENTROPY OF THE DIFFERENT SECTIONS OF RAGNAR LOCKER.



Binary Entropy		
Section	**Entropy**	**Content**
.text	6.521	Executable code
.rdata	5.354	Imports and exports
.data	n/a	Global data
.keys	6.442	Contains encrypted stings
.rsrc	4.702	Strings, icons, and images
.reloc	4.810	Base relocation table

and terminate itself. The OS language settings that will make Ragnar Locker stop executing are presented in Table III.

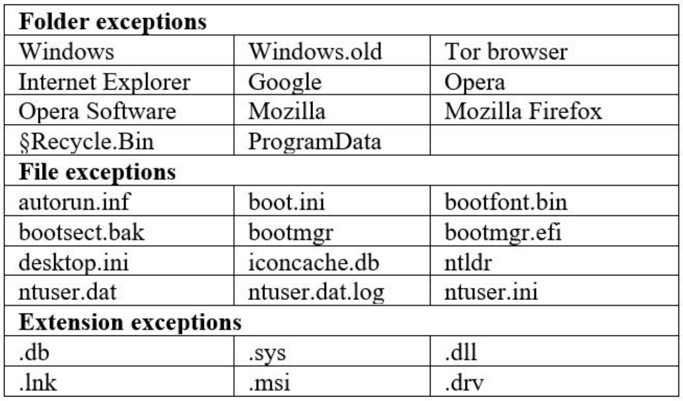
TABLE III
OS LANGUAGES THAT WILL CAUSE RAGNAR LOCKER TO TERMINATE.



Language settings that will cause termination		
Azerbaijani	Armenian	Belorussian
Kazakh	Kyrgyz	Moldovian
Tajik	Russian	Turkmen
Uzbek	Ukrainian	

5) File Encryption: The Ragnar Locker strain will iterate through the volumes and drives on the computer. Logical volumes are also mapped and assigned a drive letter. The files that are to be encrypted is added to the memory stack. The `.keys`-section is called several times to decrypt the strings that reference the files exempt encryption. Folders, files, and extensions refrained from encryption is listed in Table IV.

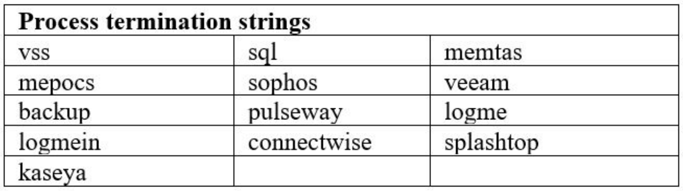
TABLE IV
FOLDERS, FILES, AND EXTENSIONS EXEMPT ENCRYPTION.



Folder exceptions		
Windows	Windows.old	Tor browser
Internet Explorer	Google	Opera
Opera Software	Mozilla	Mozilla Firefox
§Recycle.Bin	ProgramData	
File exceptions		
autorun.inf	boot.ini	bootfont.bin
bootsect.bak	bootmgr	bootmgr.efi
desktop.ini	iconcache.db	ntldr
ntuser.dat	ntuser.dat.log	ntuser.ini
Extension exceptions		
.db	.sys	.dll
.lnk	.msi	.drv

Ragnar Locker will also utilise strings to search for certain running processes. Processes that match these strings will be terminated. This behaviour is likely a security mechanism to avoid detection, as the strings are related to several known MSP, anti-virus (AV), and security solution providers. The strings called are listed in Table V.

TABLE V
PROCESSES TERMINATED BY RAGNAR LOCKER.



Process termination strings		
vss	sql	memtas
mepocs	sophos	veeam
backup	pulseway	logme
logmein	connectwise	splashtop
kaseya		

Once Ragnar Locker finishes enumerating the OS and adds file references to the stack, the encryption process will begin. The strain has a hardcoded public key, and the private key is never present in the system. Ragnar Locker copies the ransom note named ”RGNR_ ⟨ID⟩.txt” to all directories except for the ignored folders. The encrypted files get the extension ”.ragnar_ ⟨ID⟩”. The ID is a hash of the computers NETBIOS name [16]. Once the encryption process is finished, all running processes are terminated, and a notepad window opens to display the ransom note. The ransom note is hardcoded in the strain. The analysed strain appears to be tailor-made for PSE Credit Union. PSE Credit Union is not mentioned on the DLS, nor have the company made a public statement or implied that they have suffered a ransomware attack.

B. Dynamic Analysis

The strain has been run in both a local virtual environment and commercial sandboxes to observe how it behaves when executed. The sandboxes used is VirusTotal, Joe Sandbox, and AnyRun. The findings from the static analysis were validated when the sample was executed locally. One function offered by VirusTotal is to check the detection rate of the strain against a large base of AV software. 64 out of 71 AV engines flagged it as a malicious file the 31.01.2021, giving it a 90.14% detection rate one year after its creation.

1) Initiated processes: The Ragnar Locker strain initiates three processes: `vssadmin.exe`, `wmic.exe`, and `notepad.exe`. The processes are started to perform actions on the system that Ragnar Locker cannot conduct independently.

`Vssadmin.exe` is the main Windows function that administers the volume shadow copies. In `vssadmin.exe`, the command `vssadmin delete shadows /all /quiet` is run. This command deletes all shadow copies on systems running Windows 8.1 and Windows Server 2008 and newer. The volume shadow copies are deleted to prevent a user from recovering information by rolling the system back to a known good version. Ragnar Locker launches `wmic.exe` in addition to `vssadmin.exe`. `Wmic.exe` is a command-line utility used to access the Windows Management Instrumentation and can only be used by the local system administrator. When the strain has obtained access to the command-line, it runs the command `shadowcopy delete`. As some systems have mechanisms in place to secure the shadow copies, this double function is likely a redundancy mechanism for Ragnar Locker. When initiating the same process using two different methods, the threat actors increase their chance to succeed.

Notepad.exe is initiated after the system is encrypted and is used to display the ransom note.

C. Network Analysis

The examined strain does not appear to communicate with external entities. The static analysis found that the strain does not contain or call any web-related APIs. Executing Ragnar Locker whilst monitoring the internet activity confirmed that no attempts to connect to the internet was made. Wireshark was used to listen to the network traffic from the infected machine. To circumvent potential sleep-mechanisms, the network was monitored for three hours. Analyses from the web-based sandboxes further confirmed this finding. The lack of network connectivity further supports the hypothesis of Ragnar Locker being manually deployed by the attackers. The sole purpose of the ransomware is to encrypt the system and inform the victim of the compromise.

D. Forensic Artefacts

When conducting a digital forensics examination of a system, it is essential to note that each interaction with the system will change data on it. As Ragnar Locker is a ransomware strain, it will produce many artefacts, both in the registry and the file system. The number of artefacts will depend on the system.

1) Registry Analysis: Ragnar Locker adds 17 keys to the Windows registry. It adds the `HKLM\SOFTWARE\Microsoft\IdentityCRL\ThrottleCache`-key and appends five accounts to it. The ThrottleCashe is used to store request counts. All the accounts added have the SID S-1-5-18, which originally belongs to the local system's service accounts. Throttles are akin to permissions, so these actions indicate that the strain tries to change the privilege of the accounts it is using. It then adds the provisioning package `UbYfyvbG8E+B6zqO.0` to the `HKLM\SOFTWARE\Microsoft\Provisioning\Sessions`-key. The provisioning packages contains settings used to configure the device. Adding a provisioning package in this location indicates that the strain will change runtime settings on applications, which allow the ransomware to customise its environment. The remaining ten keys are added with the SID S-1-5-21-⟨*domain*⟩-1000, which is a user account on the domain. Five of the keys is associated to the `Microsoft\Speech_OneCore\Recognizers`, and the remaining five values edit the `ApplicationViewManagement\W32`. Three registry keys are deleted, the old `Provisioning\Sessions`-key and two keys related to system updates. The Ragnar Locker strain also adds several values to existing keys. One significant value is stored in the newly created `Provisioning\Sessions`-key. The `NextSessionID` is the same as the key added previously, and the `BeginTime` within that key holds the timestamp of when Ragnar Locker was executed. Knowing when the strain was executed will help build a timeline of events.

E. MITRE ATT&CK

Ragnar Locker uses legitimate user accounts to gain access and maintain persistence on systems. Using legitimate accounts also help the attackers evade defence and detection mechanisms, as the activity is less likely to stand out. To further mask their activity as benign, the attackers use ordinary user accounts and do not try to escalate their privileges to a level that would attract attention. The strain makes use of Windows APIs when executing behaviours. These will largely blend with benign API calls made. Ragnar Locker monitors good API and DLL calls and spawns' processes from them. It may also try to inject arbitrary code in the Windows Explorer process. Process spawning and injection are used to conserve a low profile on the system and escalate privileges. To maintain persistence on the system, Ragnar Locker is equipped with functionalities to infect the boot sector. Such functions are often referred to as bootkits. Bootkits reside in the Master Boot Record, on the layer below the file system, making them hard to discover and remove. The strain is aware of its environment, and queries disk information and compare the user and computer. These actions are common to use when determining whether the system is real or virtual. It also run the `GetProcessHeap` function to check if it is run in a debugger. Ragnar Locker deletes the system shadow copies to hinder the system from being recovered to a known good state. The process of assessing files for encryption is automated, which means that MITRE flags the *Automated Collection* and *Data from Local System*-tags. The strain contains a public RSA-key and calls the Windows Crypto API but shows no sign of attempting to connect to a C2 when run. If Ragnar Locker runs uninterrupted on a system, and its termination criterion is not triggered, Ragnar Locker will encrypt the file system. Table VI presents a high-level overview of the tactics and techniques as classified by MITRE ATT&CK.

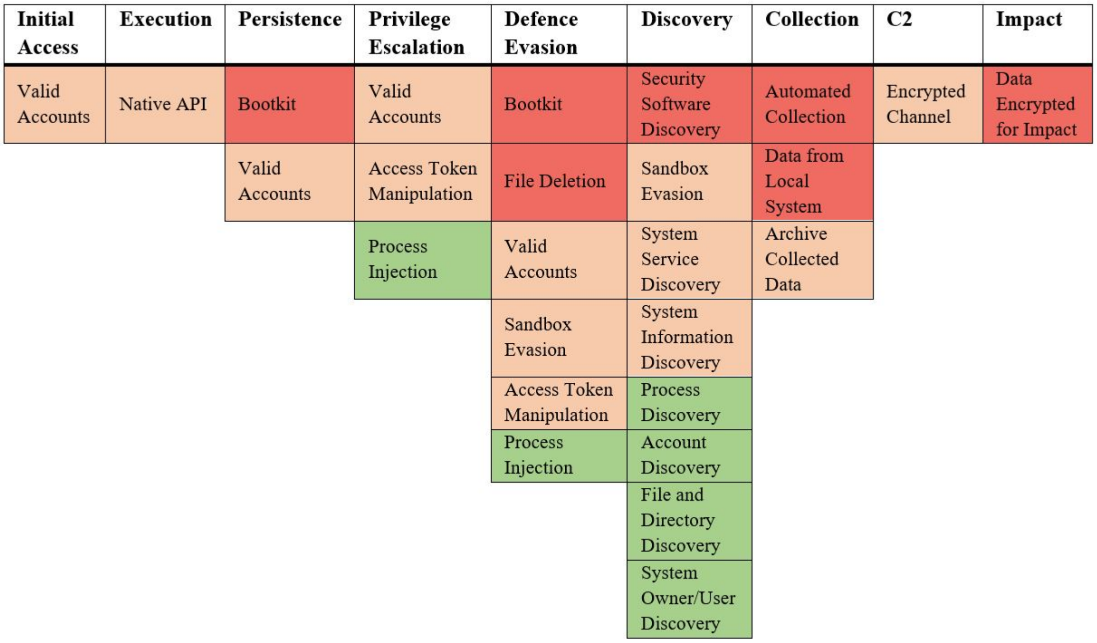
V. Conclusion and Future Work

The growth of agile and tailored ransomware is likely to continue, necessitating procedures that will facilitate thorough analysis of detected strains. Openness and sharing of threat intelligence are essential for building technical defences and mitigate the risk before breaches occur.

The Hybrid Malware Framework coalesce existing methods for reverse engineering malicious code into one forensic oriented approach. This paper is an extension of the testing and validation of this framework. This paper details the analysis of an early strain from the Ragnar Locker ransomware family. The threat actor behind Ragnar Locker tailor their attacks to each victim and are actively present when conducting lateral movement and data exfiltration. The number of manual processes applied to each attack is one factor that leads to exorbitant ransom demands. The ransomware is deployed when the attackers decide to end the engagement.

The strain was analysed using static analysis, dynamic analysis, and network analysis techniques. The Ragnar Locker strain appears to be manually executed on the system, supporting the thesis of the attackers being hands-on present during

TABLE VI
MITRE ATT&CK OVERVIEW



Initial Access	**Execution**	**Persistence**	**Privilege Escalation**	**Defence Evasion**	**Discovery**	**Collection**	**C2**	**Impact**
Valid Accounts	Native API	Bootkit	Valid Accounts	Bootkit	Security Software Discovery	Automated Collection	Encrypted Channel	Data Encrypted for Impact
		Valid Accounts	Access Token Manipulation	File Deletion	Sandbox Evasion	Data from Local System		
			Process Injection	Valid Accounts	System Service Discovery	Archive Collected Data		
				Sandbox Evasion	System Information Discovery			
				Access Token Manipulation	Process Discovery			
				Process Injection	Account Discovery			
					File and Directory Discovery			
					System Owner/User Discovery			

all stages of the attack. It uses the standard Windows APIs and DLLs and does not initiate any network connections. Several folders, files, and file extensions are exempt from encryption. The reason for this is that the attackers need the computer to function to enable the victim to pay the ransom. If Ragnar Locker detects that the OS uses one of the CIS-countries' language settings, it will terminate itself.

Systematically analysing ransomware strains' key traits enables researchers to compare results, both within one family and against other species. Such an approach will facilitate the construction of solid and effective prevention mechanisms. By uncovering similarities between strains and families, one can possibly predict behaviour characteristics in future strains. Many security solutions rely on signature-based detection, which only enables them to detect known threats.

The analysis presented in this paper present some of Ragnar Locker's traits. The uncovered technical indicators can be used to generate YARA rules. Future work should analyse newer strains of Ragnar Locker using the same framework. This would enable the security community to detail the evolution of the Ragnar Locker ransomware family.

References

[1] Purplesec, *2020 Ransomware Statistics, Data, & Trends*, 2021.

[2] V. Schmitt, "A comparative study of CERBER, MAKTUB and LOCKY," Ph.D. dissertation, Grahamstown, Jan. 2019, pp. 1–167.

[3] P. Tavares, *Ragnar Locker malware: what it is, how it works and how to prevent it*, Jun. 2020.

[4] CrowdStrike, "Global Threat Report 2021," *CrowdStrike*, p. 75, 2021.

[5] N. Lord, *What is Advanced Malware?* Sep. 2018.

[6] A. L. Young and M. Yung, "On Ransomware and Envisioning the Enemy of Tomorrow," *Computer*, vol. 50, no. 11, pp. 82–85, Nov. 2017.

[7] C. B. Simmons, S. G. Shiva, H. Bedi, and D. Dasgupta, in *9th Annual Symposium on Information Assurance*, University of Memphis, New York, 2014.

[8] A. Zimba and M. Chishimba, "Understanding the Evolution of Ransomware: Paradigm Shifts in Attack Structures," *Computer Network and Information Security*, vol. 1, pp. 26–39, 2019.

[9] T. Steffens, *Attribution of Advanced Persistent Threats*. Berlin, Heidelberg, 2020.

[10] H. S. Galal, Y. B. Mahdy, and M. A. Atiea, "Behavior-based features model for malware detection," vol. 12, no. 2, pp. 59–67, May 2016.

[11] M. Eskandari, Z. Khorshidpour, and S. Hashemi, "HDM-Analyser: A hybrid analysis approach based on data mining techniques for malware detection," vol. 9, no. 2, pp. 77–93, May 2013.

[12] U. Bayer, A. Moser, C. Kruegel, and E. Kirda, "Dynamic analysis of malicious code," 1, vol. 2, Aug. 2006, pp. 67–77.

[13] M. Abulaish and N. A. H. Haldar, *Digital Forensics and Forensic Investigations*. Apr. 2020.

[14] D. Greten, *Indicators of Compromise (IOCs) and Artifacts: What's the Difference?* Jun. 2020.

[15] Blaze Information Security, *Dissecting Ragnar Locker: The Case Of EDP*, Jul. 2020.

[16] FBI, "Indicators of Compromise Associated with Ragnar Locker Ransomware," Tech. Rep., 2020.