Using the LockBit builder to generate targeted ransomware

By Eduardo Ovalle

Published: 2024-04-15 · Archived: 2026-04-05 21:53:01 UTC

The previous Kaspersky research focused on a detailed analysis of the LockBit 3.0 builder leaked in 2022. Since then, attackers have been able to generate customized versions of the threat according to their needs. This opens up numerous possibilities for malicious actors to make their attacks more effective, since it is possible to configure network spread options and defense-killing functionality. It becomes even more dangerous if the attacker has valid privileged credentials in the target infrastructure.

In a recent incident response engagement, we faced this exact scenario: the adversary was able to get the administrator credential in plain text. They generated a custom version of the ransomware, which used the aforementioned account credential to spread across the network and perform malicious activities, such as killing Windows Defender and erasing Windows Event Logs in order to encrypt the data and cover its tracks.

In this article, we revisit the LockBit 3.0 builder files and delve into the adversary's steps to maximize impact on the network. In addition, we provide a list of preventive activities that can help network administrators to avoid this kind of threat.

Revisiting the LockBit 3.0 builder files

The LockBit 3.0 builder has significantly simplified creating customized ransomware. The image below shows the files that constitute it. As we can see, **keygen.exe** generates public and private keys used for encryption and decryption. After that, **builder.exe** generates the variant according to the options set in the **config.json** file.

```
PS C:\Users\admin\Desktop\LockBit> ls


    Directory: C:\Users\admin\Desktop\LockBit


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
------        2/2/2024   6:18 PM           1114 Build.bat
------        2/2/2024   6:18 PM         480768 builder.exe
------        2/2/2024   6:18 PM           8363 config.json
------        2/2/2024   6:18 PM          31744 keygen.exe
```

LockBit builder files

This whole process is automated with the **Build.bat** script, which does the following:

```
IF exist Build (ERASE /F /Q Build\*.*) ELSE (mkdir Build)
keygen -path Build -pubkey pub.key -privkey priv.key
builder -type dec -privkey Build\priv.key -config config.json -ofile Build\LB3Decryptor.exe
builder -type enc -exe -pubkey Build\pub.key -config config.json -ofile Build\LB3.exe
builder -type enc -exe -pass -pubkey Build\pub.key -config config.json -ofile Build\LB3_pass.exe
builder -type enc -dll -pubkey Build\pub.key -config config.json -ofile Build\LB3_Rundll32.dll
builder -type enc -dll -pass -pubkey Build\pub.key -config config.json -ofile
Build\LB3_Rundll32_pass.dll
builder -type enc -ref -pubkey Build\pub.key -config config.json -ofile
Build\LB3_ReflectiveDll_DllMain.dll
```

The **config.json** file allows enabling impersonation features (**impersonation**) and defining accounts to impersonate (**impers_accounts**). In the example below, the administrator account was used for impersonation. The configuration also allows enabling the encryption of network shares (**network_shares**), killing Windows Defender (**kill_defender**), and spreading across the network via PsExec (**psexec_netspread**). After a successful infection, the malicious sample can delete Windows Event Logs (**delete_eventlogs**) to cover its tracks.

```
{
  "bot": {
    "uid": "00000000000000000000000000000000",
    "key": "00000000000000000000000000000000"
  },
"config": {
    "settings": {
      "encrypt_mode": "auto",
      "encrypt_filename": false,
      "impersonation": true,
      "skip_hidden_folders": false,
      "language_check": false,
      "local_disks": true,
      "network_shares": true,
      "kill_processes": true,
      "kill_services": true,
      "running_one": true,
      "print_note": true,
      "set_wallpaper": true,
      "set_icons": true,
      "send_report": false,
      "self_destruct": true,
      "kill_defender": true,
      "wipe_freespace": false,
      "psexec_netspread": false,
      "gpo_netspread": true,
      "gpo_ps_update": true,
      "shutdown_system": false,
      "delete_eventlogs": true,
      "delete_gpo_delay": 1
    },
    "white_folders": "$recycle.bin;config.msi;$windows.~bt;$windows.~ws;windows;boot;program files;program files (x86);programdata;system volume information;tor brow
    "white_files": "autorun.inf;boot.ini;bootfont.bin;bootsect.bak;desktop.ini;iconcache.db;ntldr;ntuser.dat;ntuser.dat.log;ntuser.ini;thumbs.db;GDIPFONTCACHEV1.DAT;
    "white_extens": "386;adv;ani;bat;bin;cab;cmd;com;cpl;cur;deskthemepack;diagcab;diagcfg;diagpkg;dll;drv;exe;hlp;icl;icns;ico;ics;idx;ldf;lnk;mod;mpa;msc;msp;mssty
    "white_hosts": "WS2019",
    "kill_processes": "sql;oracle;ocssd;dbsnmp;synctime;agntsvc;isqlplussvc;xfssvccon;mydesktopservice;ocautoupds;encsvc;firefox;tbirdconfig;mydesktopqos;ocomm;dbeng
    "kill_services": "vss;sql;svc$;memtas;mepocs;msexchange;sophos;veeam;backup;GxVss;GxBlr;GxFWD;GxCVD;GxCIMgr",
    "gate_urls": "https://test.com/",
    "impers_accounts": "Administrator:test@123",
    "note": "
          ~~~ Testing the LockBit builder~~~"
  }
}
```

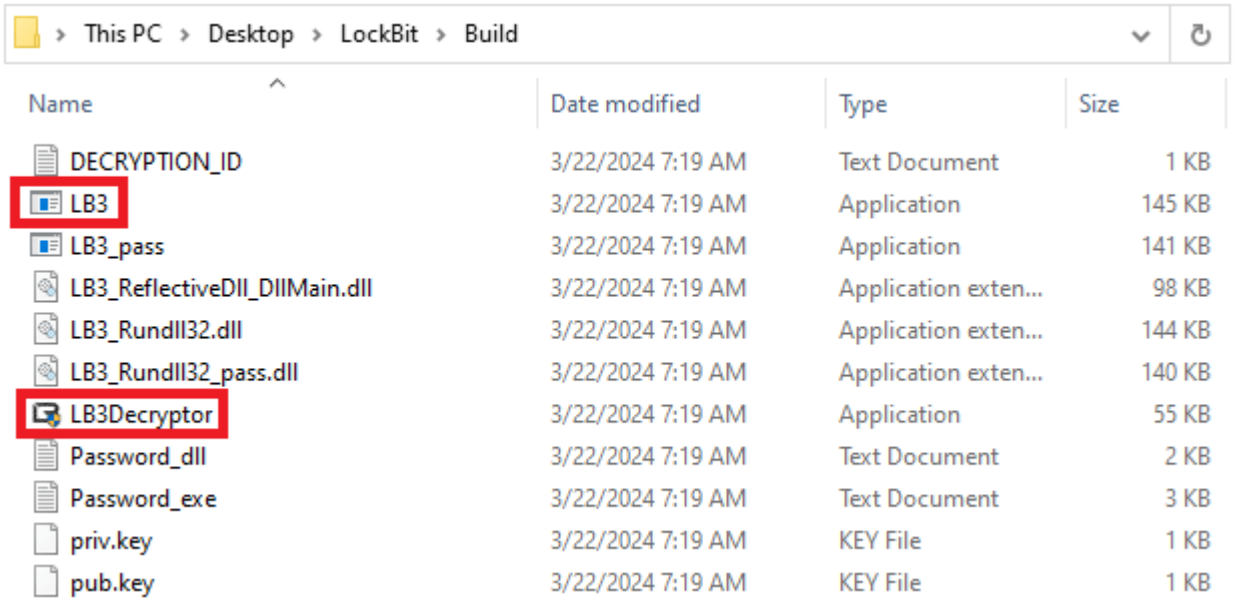
Custom configuration

Besides this, the builder allows the attacker to choose which files, in which directories, and in which systems they do not want to encrypt. If the attacker knows their way around the target infrastructure, they can generate malware tailored to the specific configuration of the target's network architecture, such as important files, administrative

accounts, and critical systems. The images below show the process of generating customized ransomware according to the above configuration, and the resulting files. As we can see, **LB3.exe** is the main file. This is the artifact that will be delivered to the victim. The builder also generates **LB3Decryptor.exe** for recovering the files, as well as several different variants of the main file. For example, **LB3_pass.exe** is a password-protected version of the ransomware, while the reflective DLL can be used to bypass the standard operating system loader and inject malware directly into memory. The TXT files contain instructions on how to execute the password-protected files.

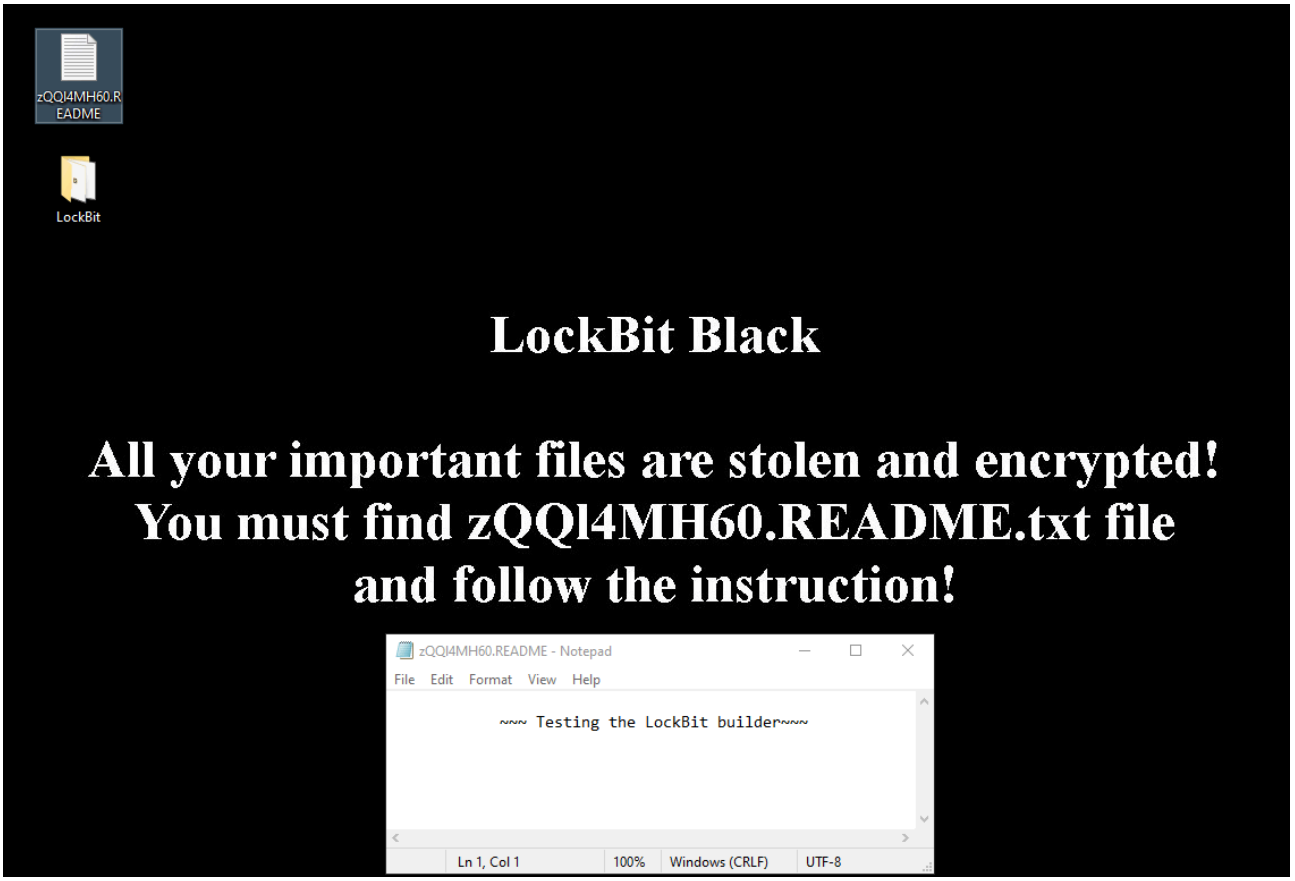
```
Press any key to generate the LockBit files (will overwrite existing files)...
(1/7) Generating keys
(2/7) Building decryptor
(3/7) Building ransomware executable
(4/7) Building ransomware executable that requires password
(5/7) Building ransomware DLL
(6/7) Building ransomware DLL that requires password
(7/7) Building reflective DLL
Done.
```

Creation of a customized LockBit version



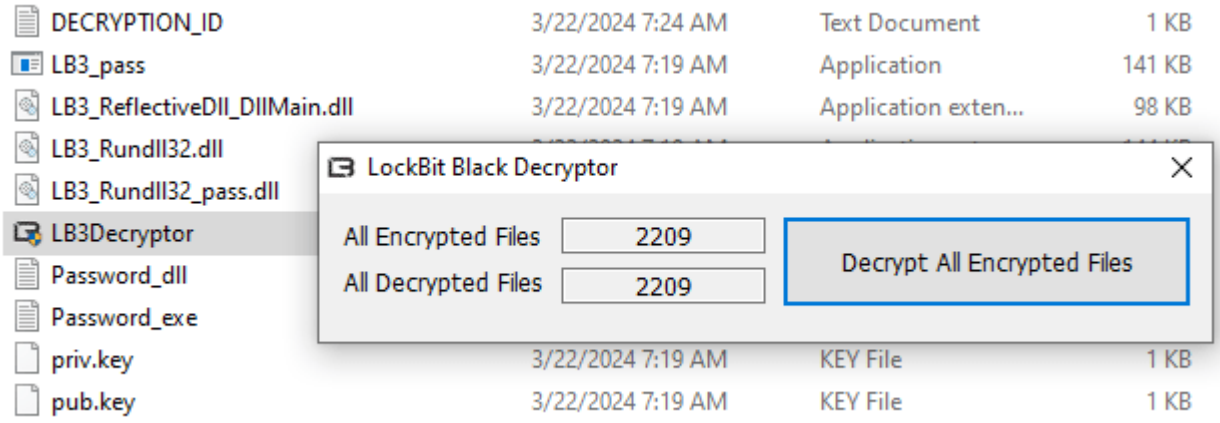
Generated LockBit files

When we executed this custom build on a virtual machine, it performed its malicious activities and generated custom ransom note files. In real-life scenarios, the note will include details on how the victim should contact the attackers to obtain a decryptor. It is worth noting that negotiating with the attackers and paying ransom should not be an option. Besides the ethical issues involved, there is doubt whether a tool for recovering the files will ever be provided.



Custom ransom note

However, as we generated the ransomware sample and a corresponding decryptor ourselves in a controlled lab environment, we were able to test if the latter actually worked. We tried to decrypt our encrypted files and found out that if the decryptor for the sample was available, it was indeed able to recover the files, as shown in the image below.



LB3Decryptor execution

That said, we must once again underscore that even a correctly working decryptor is no guarantee that the attackers will play fair.

The recent LockBit takedown and custom LockBit builds

In February 2024, the international law enforcement task force Operation Cronos gained visibility into LockBit's operations after taking the group down. The collaborative action involved law enforcement agencies from 10 countries, which seized the infrastructure and took control of the LockBit administration environment. However, a few days after the operation, the ransomware group announced that they were back in action.

The takedown operation allowed LEAs to seize the group's infrastructure, obtain private decryption keys and prepare a decryption toolset based on a known-victim ID list obtained by the authorities. The **check_decryption_id** utility checks if the ransom ID enabled for the victim is on the list of known decryption keys:

```
PS C:\Tools\Decryptors\Decryption_Checker_for_LockBit> .\check_decryption_id.exe
Enter the Decryption ID that you received from the threat actor (min. 16 characters): AAAAAAAAAAAAAAAA


Unfortunately, a decryption key for that Decryption ID is currently unavailable.
We recommend checking the No More Ransom website for updates in the coming days, as new decryption keys may become avail
able.

Press any key to continue . . .
PS C:\Tools\Decryptors\Decryption_Checker_for_LockBit>
```

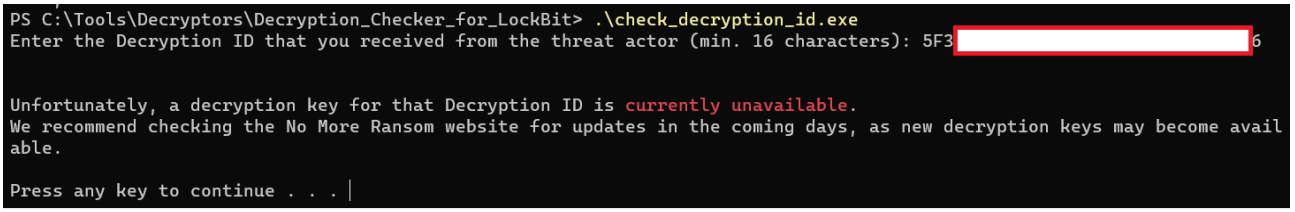
check_decryption_id.exe execution

The **check_decrypt** tool assesses decryptability: while there is a possibility that the files will be recovered, the outcome of the process depends on multiple conditions, and this tool just checks which of these conditions are met in the systems being analyzed. A CSV file is created, listing files that can be decrypted and providing an email address to reach out to for further instructions on restoring the files:

```
PS C:\Tools\Decryptors\Decryption_Checker_for_LockBit> .\check_decrypt.exe --help
usage: check_decrypt.exe [-h] [-l FILELIST_ENCFILE] [--filelist_decinfo FILELIST_DECINFO] [-v]
                         Input_Folder Target_Extension

Assess decryptablity of LockBit 3.0 encrypted files
Please provide an input folder and the file extension of the encrypted files as arguments.

positional arguments:
  Input_Folder          full path to the encrypted files
  Target_Extension      file extension of encrypted files(case-sensitive)

options:
  -h, --help            show this help message and exit
  -l FILELIST_ENCFILE, --filelist_encfile FILELIST_ENCFILE
                        use filelist of encrypted files
  --filelist_decinfo FILELIST_DECINFO
                        use filelist with decryption information
  -v, --version         show version
PS C:\Tools\Decryptors\Decryption_Checker_for_LockBit>
```

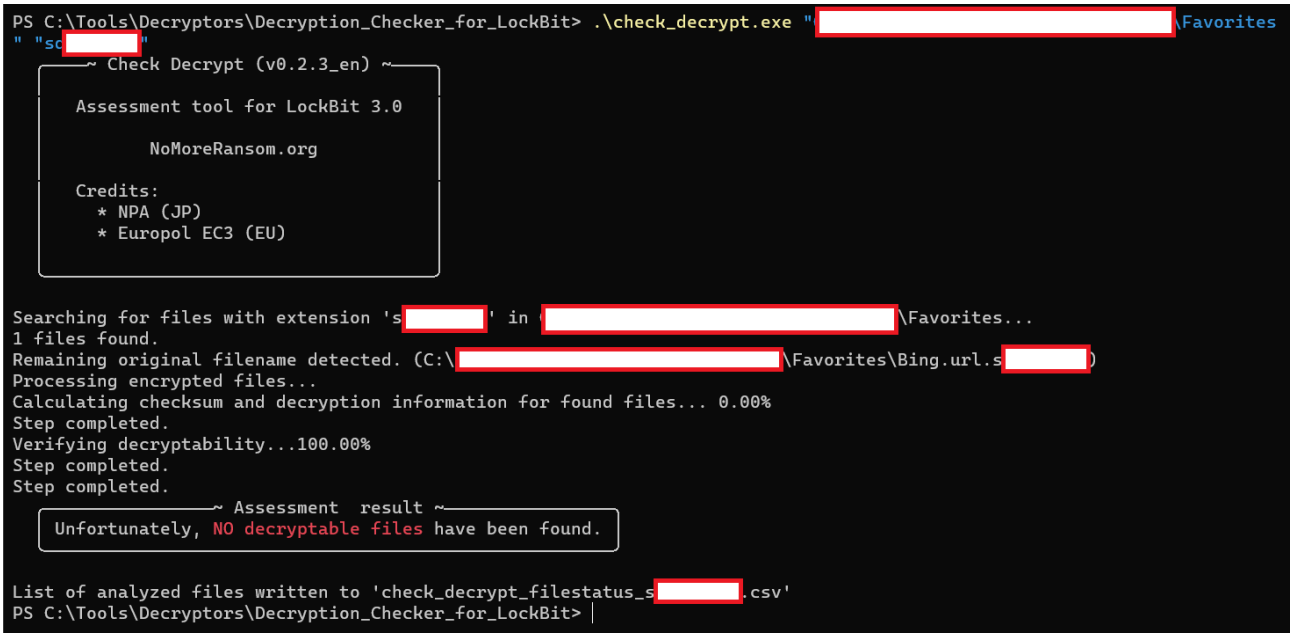
check_decrypt.exe execution

This toolset caught our attention because we had investigated several cases relating to the LockBit threat. We normally recommend that our customers save their encrypted critical files and wait for an opportunity to decrypt them with the help of threat researches or artifacts seized by the authorities, which is merely a matter of time. We ran victim IDs and encrypted files analyzed by our team through the decryption tool, but most of them showed the same result:



Testing the tool on a victim ID obtained by our team

The **check_decrypt** also confirmed that it was not possible to decrypt the files by using the database of known keys:



Testing the check_decrypt.exe tool on encrypted files

Our analysis and previous research confirmed that files encrypted with a payload generated with the help of the leaked LockBit builder could not be decrypted with existing decryption tools, essentially because the independent groups behind these attacks did not share their private keys with the RaaS operator.

Geography of the leaked LockBit builder-based attacks

Custom LockBit builds created with the leaked builder were involved in a number of incidents all over the world. These attacks were most likely unrelated and executed by independent actors. The leaked builder apparently has been used by LockBit ransomware competitors to target companies in the Commonwealth of Independent States, violating the group's number one rule to avoid compromising CIS nationals. This triggered a discussion on the dark web, where LockBit operators tried to explain that they had nothing to do with these attacks.

In our incident response practice, we have come across ransomware samples created with the help of the leaked builder in incidents in Russia, Italy, Guinea-Bissau, and Chile. Although the builder provides a number of customization options, as we have shown above, most of the attacks used the default or slightly modified configuration. However, one incident stood out.

A real-life incident response case involving a custom LockBit build

In a recent incident response engagement, we faced a ransomware scenario involving a LockBit sample built with the leaked builder and featuring impersonation and network spread capabilities we had not seen before. The attacker was able to exploit an internet-facing server that exposed multiple sensitive ports. Somehow, they were able to obtain the administrator password – we believe that it may have been stored in plain text inside a file, or that the attacker may have used social engineering. Then, the adversary generated custom ransomware using the privileged account they had access to. Our team was able to obtain the relevant fields present in the **config.json** file that the attacker used:

```
1  "impersonation": true,
2  "impers_accounts": "Administrator:************",
3  "local_disks": true,
4  "network_shares": true,
5  "running_one": false,
6  "kill_defender": true,
7  "psexec_netspread": true,
8  "delete_eventlogs": true,
```

As we can see, the custom version has the ability to impersonate the administrator account, affect network shares, and spread easily across the network via PsExec.

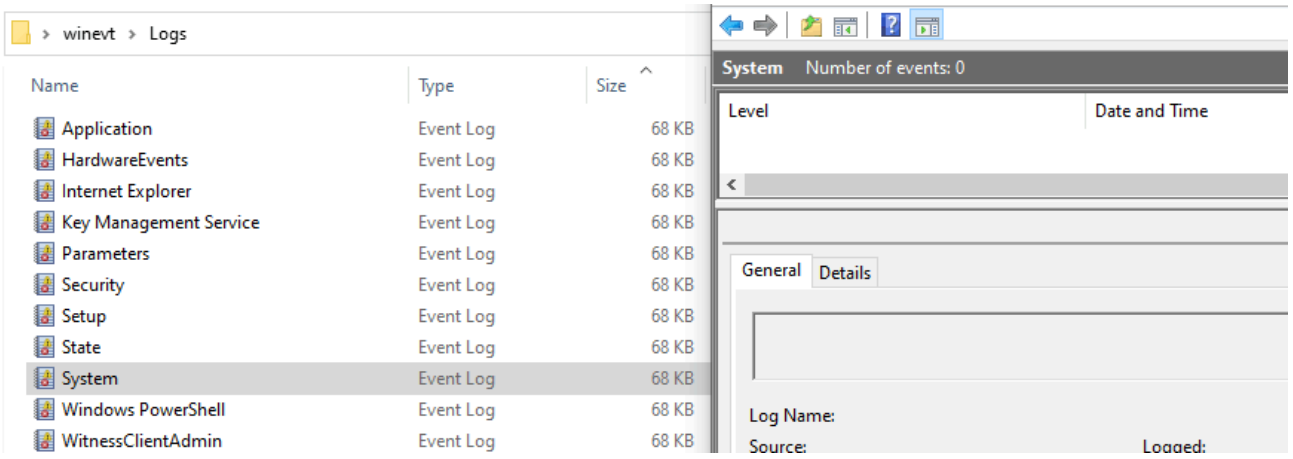
Moreover, it is configured to run more than once on each host. One of the first steps that the executable does when started is check for, and create, a unique mutex based on a hash sum of the ransomware public key in the format: **“Global\%.8x%.8x%.8x%.8x%.8x%.8x”**. If the **running_one** flag is set to true in the configuration and the mutex is already present in the operating system, the process will exit.

In our case, the configuration allowed concurrent executions of several ransomware instances on the same host. This behavior, combined with the use of configuration flags for automatic network propagation with high-privileged domain credentials, led to an uncontrolled avalanche effect: each host that got infected then started trying to infect other hosts on the network, including those already infected. From an incident response point of view, this means finding evidence, if available, of different origins for the same threat. See below the evidence found on one host of remote service creation by PsExec with authentication completed from multiple infected hosts.

date	Id	Action	Service_Name	USER	DOMAIN	SrcIP
2024-03-10 17:18:16.5124277	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.74'
2024-03-10 17:18:16.8551939	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 17:26:55.7603530	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.84'
2024-03-10 17:26:56.1037369	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 17:34:27.6601469	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.71'
2024-03-10 17:34:27.7497878	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 17:48:56.0332683	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.246'
2024-03-10 17:48:56.1716956	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 18:21:39.1390142	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.161'
2024-03-10 18:21:39.4119230	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 18:28:14.2075819	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.225'
2024-03-10 18:28:14.6365296	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 18:35:02.3407125	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.13'
2024-03-10 18:35:02.3765532	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 18:41:35.4176816	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.21'
2024-03-10 18:41:35.8129178	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 19:21:59.5626144	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.85'
2024-03-10 19:21:59.8142775	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 19:34:54.7575938	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.80'
2024-03-10 19:34:55.1338333	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 19:58:57.1415824	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.155'
2024-03-10 19:58:57.9395845	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:09:25.4321141	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.107'
2024-03-10 20:09:25.6227167	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:24:16.1704690	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.79'
2024-03-10 20:24:17.2718780	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:25:27.1213592	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.79'
2024-03-10 20:25:27.7875362	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:36:07.2643122	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.88'
2024-03-10 20:36:07.3363542	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:42:41.9274431	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.103'
2024-03-10 20:42:43.3435539	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:54:23.2367389	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.105'
2024-03-10 20:54:23.4388132	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 20:54:54.8381516	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.105'
2024-03-10 20:54:54.9255787	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 21:01:38.9842427	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.247'
2024-03-10 21:01:39.0865463	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-10 21:02:08.4813842	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.247'
2024-03-10 21:02:08.5568727	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-11 09:35:45.9535944	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.142'
2024-03-11 09:35:48.8233086	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			
2024-03-11 09:40:29.4740367	4624	AUTH_SUCCESS		Administrator	CUSTOMERDOMAIN	192.168.*.79'
2024-03-11 09:40:31.3916367	7045	SERVICE_CREATE	{*-7878-FF50-E38A-*}			

Remote service creation by PsExec

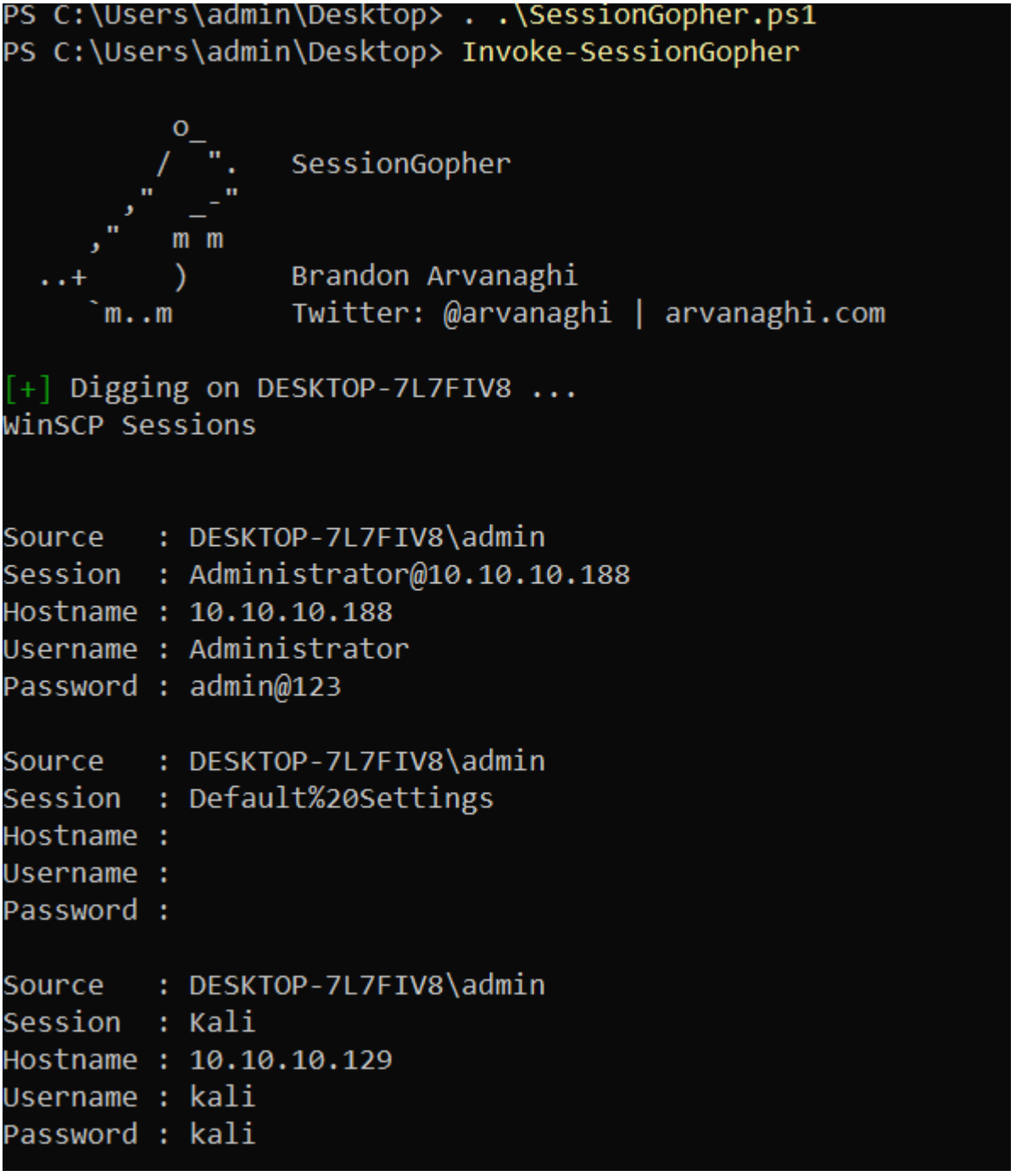
Although this evidence was present in the infected systems, most of the logs had been deleted by the ransomware immediately after the initial infection. Because of that, it was not possible to determine how the attacker was able to gain access to the server and to the administrator password. The remote service creation logs remained because when the malware was performing lateral movement on the network, it generated new logs, which it did not delete, and which were helpful in detecting its spread across the infrastructure.



Event logs cleared

By analyzing some of the traces that were not erased on the initial affected server, we identified compressed Gzip data in a memory stream. The data was encoded in Base64. After decoding and decompression, we found evidence of the use of Cobalt Strike. We were able to identify the C2 server used by the attacker to communicate with the affected machine and promptly sent this indicator to the customer for blacklisting.

We also spotted the use of the SessionGopher script. This tool uses WMI to extract saved session information for remote desktop access tools, such as WinSCP, PuTTY, FileZilla, and Microsoft Remote Desktop. This is accomplished by querying **HKEY_USERS** for PuTTY, WinSCP, and Remote Desktop saved sessions. In **Thorough** mode, the script can identify **.ppk**, **.rdp**, and **.sdtid** files in order to extract private keys and session information. It can be run remotely by using the **-iL** option followed by the list of computers. The **-AllDomain** flag allows running it against all AD-joined computers. As shown in the image below, the script can easily extract saved passwords for remote connections. The results can be exported to a CSV file for later use.



Password extraction using SessionGopher

Although SessionGopher is designed for collecting stored credentials, it was not the tool used by the attackers for initial credential dumping. Instead, they employed SessionGopher to collect additional credentials and services in the infrastructure at a later stage.

Once we identified the C2 domains and some other IP addresses related to the attacker and extracted details about the impersonated accounts and tools implemented for automatic deployment, the customer changed all affected users' credentials and configured security controls to avoid PsExec execution, thus stopping the infection. Monitoring network and user account activities allowed us to identify the infected systems and isolate them for analysis and recovery.

This case shows an interesting combination of techniques used to gain and maintain access to the target network, as well as encrypt important data and impair defenses. Below are the TTPs identified for this scenario.

Tactic	Technique	ID
Impact	Data Encrypted for Impact	T1486
Defense Evasion, Persistence, Privilege Escalation, Initial Access	Valid Accounts	T1078.002
Credential Access	Credentials from Password Stores	T1555
Lateral Movement	Remote Services	T0886
Discovery	Network Service Discovery	T1046
Defense evasion	Clear Windows Event Logs	T1070.001
Defense evasion	Impair Defenses	T1562

Preventive actions against ransomware attacks

Ransomware attacks can be devastating, especially if the attackers manage to get hold of high-privileged credentials. Measures for mitigating the risk of such an attack may vary depending on the technology used by the company. However, there are certain infrastructure-agnostic techniques:

- Using a robust, properly-configured antimalware solution, such as Kaspersky Endpoint Security
- Implementing Managed Detection and Response (MDR) to proactively seek out threats
- Disabling unused services and ports to minimize the attack surface
- Keeping all systems and software up to date
- Conducting regular penetration tests and vulnerability scanning to identify vulnerabilities and promptly apply appropriate countermeasures
- Adopting regular cybersecurity training, so that employees are aware of cyberthreats and ways to avoid them
- Making backups frequently and testing them

Conclusion

Our examination of the LockBit 3.0 builder files shows the alarming simplicity with which attackers can craft customized ransomware, as evidenced by a recent incident where adversaries exploited administrator credentials to deploy a tailored ransomware variant. This underscores the need for robust security measures capable of mitigating this kind of threat effectively, as well as adoption of a cybersecurity culture among employees.

Kaspersky products detect the threat with the following verdicts:

- Trojan-Ransom.Win32.Lockbit.gen

- Trojan.Multi.Crypmod.gen
- Trojan-Ransom.Win32.Generic

And the SessionGopher script, as:

- HackTool.PowerShell.Agent.l
- HackTool.PowerShell.Agent.ad
