

EvilPayout: Attack Against Iran's State Broadcaster

research.checkpoint.com/2022/evilpayout-attack-against-irans-state-broadcaster/

February 18, 2022



February 18, 2022

In the past few months, a new wave of cyberattacks has been flooding Iran. These attacks are far from minor website defacements – the recent wave is hitting national infrastructure and causing major disruptions to public services.

This article provides an in-depth technical analysis of one of the attacks against the Iranian national media corporation, Islamic Republic of Iran Broadcasting (IRIB) which occurred in late January 2022.

Key findings

- On January 27, Iranian state broadcaster IRIB became the subject of a targeted cyberattack that resulted in several state-run TV channels broadcasting footage of opposition leaders and calling for the assassination of the supreme leader. Check Point Research team investigated this attack and was able to retrieve the files and forensics evidence related to the incident from publicly available resources.
- We found malicious executables whose purpose was to air the protest message, in addition, we discovered evidence that a wiper malware was used. This indicates that the attackers' aim was also to disrupt the state's broadcasting networks, with the damage to the TV and radio networks possibly more serious than officially reported.
- Among the tools used in the attack, we identified malware that takes screenshots of the victims' screens, several custom-made backdoors, and related batch scripts and configuration files used to install and configure the malicious executables. We could not find any evidence that these tools were used previously, or attribute them to a specific threat actor.
- In this article, we provide a technical analysis of the tools related to the attack, as well as the attackers' tactics.

Background

Cyberattacks Hit Iran

In July 2021, an attack hit the Iranian national railway and cargo services, and caused "unprecedented disruptions" to the country's trains. Just a day later, media outlets reported that the website of Iran's Ministry of Roads and Urban Development, in charge of transportation, was taken down in a 'cyber disruption', preventing access to their official portal and sub-services. As if forcing railway employees to update the train schedule manually – across all train stations – wasn't enough, the message

displayed on the train schedule boards referred perplexed passengers to the Supreme Leader’s office phone number. The previously unknown group called ‘Predatory Sparrow’ quickly claimed responsibility for the attacks. Besides that, Check Point Research investigated these attacks and the tools they deployed, and found similar tactics and techniques were used in previous operations against private companies in Syria, linking all of those attacks to anti-regime group called Indra.

Since then, cyber-attacks continue to hit national Iranian entities. Inspecting the targets, it appears that each one was carefully selected to send a tailored message. In August 2021, the hacktivist group Tapandegan, previously known for hacking and displaying protest messages on the electronic flight arrival and departure boards in the Mashad and Tabriz international airports in 2018, released security camera footage from the Evin prison, a Tehran facility in which many political prisoners are held. The videos, which show prisoner abuse, were acquired by a group called Edalat-e Ali (‘Ali’s justice’) in protest against human rights violations. In October 2021, every gas station in Iran was paralyzed by an attack that disrupted the electronic payment process. The incident led to extremely long queues at gas stations for two days and prevented customers from paying with the government-issued electronic cards used to purchase subsidized fuel. When the card was swiped for payment, the Supreme Leader office phone number appeared on the screen, taunting the highest ranking office in the regime yet again. Iranian officials claimed that foreign actors, such as Israel and the US, were behind the attack. However, Predatory Sparrow claimed responsibility for this attack as well.

In November 2021, Iranian airline Mahan Air announced that it foiled an attempted attack against its internal systems, with no harm done. Curiously, this time a group called ‘Hooshyaran-e Vatan’ (Vigilant of the Nation) claimed responsibility, and over the next two months published documents allegedly stolen in the hack that link the airline to the IRGC (Islamic Revolutionary Guard Corps).

Recently, on February 7, 2022, the Edalat-e Ali group released footage from closed-circuit cameras in another Iranian prison, Ghezel Hesar.

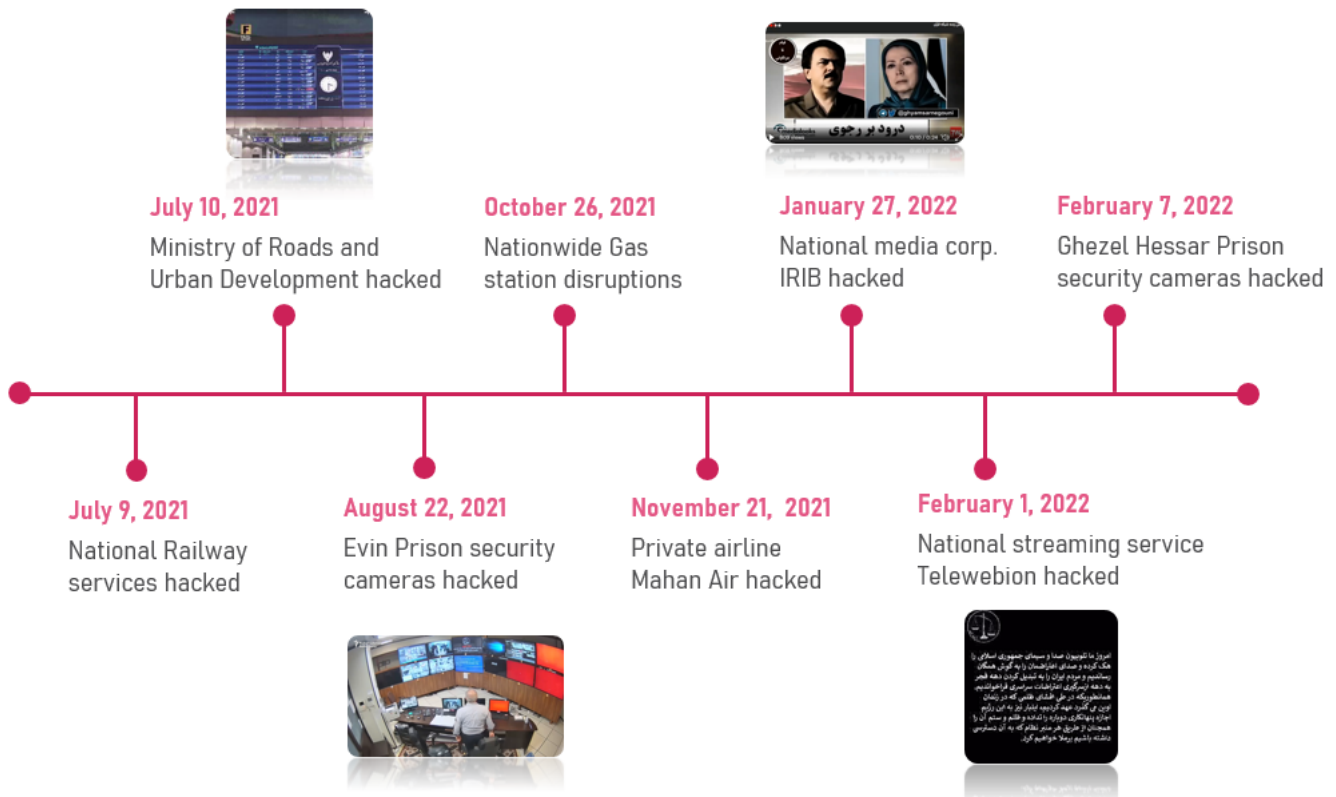


Figure 1 – Timeline of latest cyberattacks in Iran.

The Voice and Vision of the Islamic Republic of Iran

On January 27, only two weeks before the anniversary of the 1979 Islamic Revolution, reports were published that the IRIB, Iran’s national broadcaster, was hacked. The Islamic Republic of Iran Broadcasting, also called ‘The Voice and Vision of the Islamic Republic of Iran’, is a state-operated monopoly in charge of all radio and television services in Iran. The cyberattack resulted in state-run TV channels broadcasting what was described by IRIB officials as “the faces and voices of hypocrites.”

'Hypocrites' is a term used by the Iranian regime to refer to the Mujahedin-e-Khalq (MEK, also called the People's Mujahedin of Iran), an exiled militant organization and the biggest political opposition group, which advocates overthrowing the current regime and installing its own government, relying on an alternative interpretation of Islam. In the hijacked video, the faces of MEK leaders Maryam and Masoud Rajavi appeared, followed by the image of Ayatollah Khamenei crossed out with red lines and the declaration "Salute to Rajavi, death to (Supreme Leader) Khamenei!". The deputy head of technical affairs for IRIB, Reza Alidadi, stated that "only the owners of the technology in use by the corporation would have been able to carry out an attack relying on the system features installed on the systems and the exploited backdoor." He further stated that similar attacks have hit other state-operated radio channels.



Figure 2 – Frame from the video with the opposition leaders' faces broadcast by state-run Iranian TV channels as a result of the cyber attack.

Although not the part of this investigation, it is worth mentioning that several days later, on February 1, the web-based streaming platform of IRIB, Telewebion, was hijacked yet again to broadcast protest messages urging citizens to rise up against the Supreme Leader and stating that "the regime's foundations are rattling". Cleverly, the incident took place in the middle of a live broadcast of the Iran-UAE soccer match. This time, politically motivated group Edalat-e Ali, responsible for the attacks targeting prison facilities' security cameras, claimed responsibility. This claim is plausible, as the video broadcast during the hack features the group's logo on the top left corner.

IRIB attack artifacts

According to Iranian state-run news network Akharin Khabar (Latest News), "the technical and broadcasting systems are completely isolated, they are equipped with acceptable security protocols and are not accessible via the Internet." In the same post, it was reported that security forces associated with the regime's state broadcasting network considered sabotage as the most likely scenario, with the Iranian officials calling the attack "extremely complex."

It is still not clear how the attackers gained initial access to these networks. We were able to retrieve only the files related to the later stages of these attacks, responsible for:

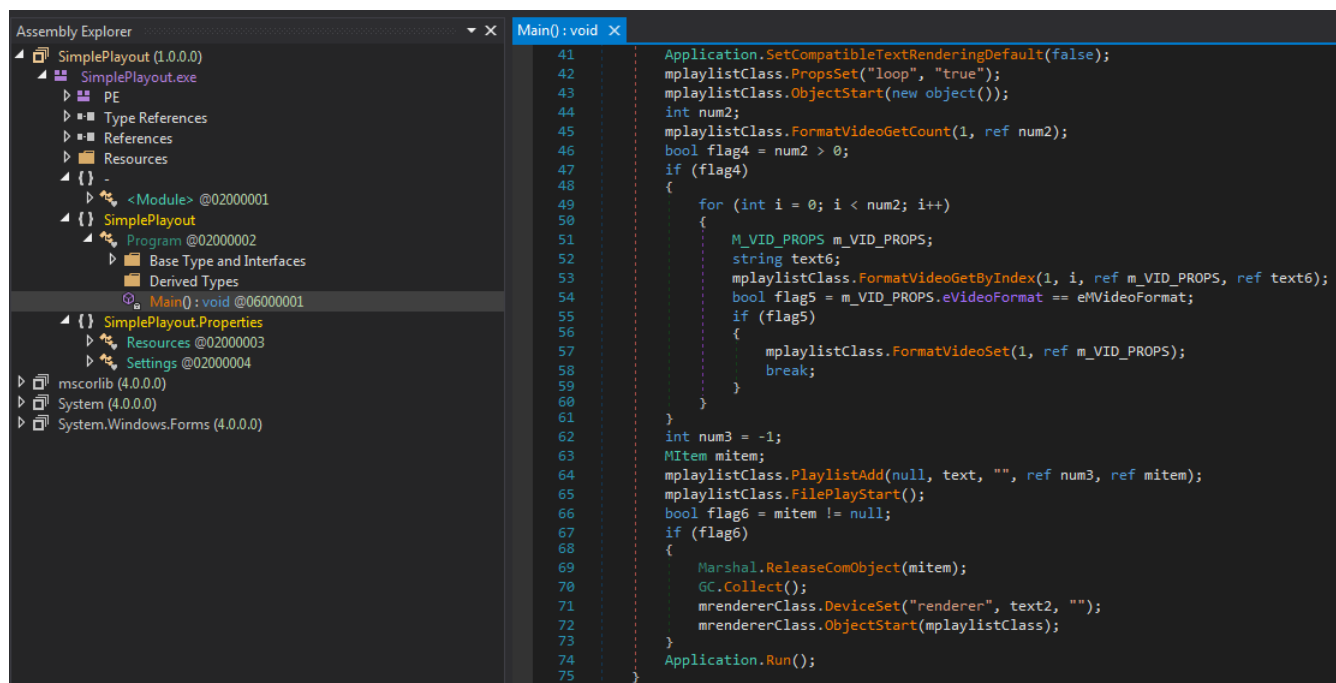
- Establishing backdoors and their persistence.
- Launching the "malicious" video or audio track.
- Installing the wiper malware in an attempt to disrupt operations in the hacked networks.

All of these samples were uploaded to VirusTotal (VT) from multiple sources, mostly with Iranian IPs, and included short batch scripts that install or launch payloads, several forensics artifacts like Windows Event Log files or memory dumps, and the payloads themselves. The latter are mostly .NET executables, with no obfuscation but a timestamped compilation date in the

future. In addition to having the same language and same VT submitters, these files also share other similarities, such as PDB paths, common commands, names, code reuse, and general coding style.

Hijacking broadcast signals

From the MP4 video file that was used to interrupt the TV stream, and was uploaded to VT as `TSE_90E11.mp4`, we were able to pivot to other artifacts related to the broadcast hijacking, supposedly run on servers that broadcast TV programs (playouts). To play the video file, the attackers used a program called `SimplePlayout.exe`, a .NET-based executable compiled in debug mode with the PDB path `c:\work\SimplePlayout\obj\Debug\SimplePlayout.pdb`. This executable has a single functionality: to play a video file in a loop using the .NET MPlatform SDK by Medialooks.



```
41 Application.SetCompatibleTextRenderingDefault(false);
42 mplaylistClass.PropsSet("loop", "true");
43 mplaylistClass.ObjectStart(new object());
44 int num2;
45 mplaylistClass.FormatVideoGetCount(1, ref num2);
46 bool flag4 = num2 > 0;
47 if (flag4)
48 {
49     for (int i = 0; i < num2; i++)
50     {
51         M_VID_PROPS m_VID_PROPS;
52         string text6;
53         mplaylistClass.FormatVideoGetByIndex(1, i, ref m_VID_PROPS, ref text6);
54         bool flag5 = m_VID_PROPS.eVideoFormat == eMVideoFormat;
55         if (flag5)
56         {
57             mplaylistClass.FormatVideoSet(1, ref m_VID_PROPS);
58             break;
59         }
60     }
61     int num3 = -1;
62     MItem mitem;
63     mplaylistClass.PlaylistAdd(null, text, "", ref num3, ref mitem);
64     mplaylistClass.FilePlayStart();
65     bool flag6 = mitem != null;
66     if (flag6)
67     {
68         Marshal.ReleaseComObject(mitem);
69         GC.Collect();
70         mrendererClass.DeviceSet("renderer", text2, "");
71         mrendererClass.ObjectStart(mplaylistClass);
72     }
73 }
74 Application.Run();
75 }
```

Figure 3 – Part of the SimplePlayout code using MPlatform SDK to play the video file.

First, the SimplePlayout program looks for a configuration file called `SimplePlayout.ini` which contains two lines: the video file path, and a number representing the video format. The respective `SimplePlayout.ini` file uploaded together with SimplePlayout specifies the values that correspond to the MP4 file located at `c:\windows\temp\TSE_90E11.mp4` and a video format of HD 1080i with a refresh rate of 50 Hz.

To kill the video stream already playing, the attackers used a batch script called `playjfalcfgcdq.bat`. It kills the running process and deletes the executable of **TFI Arista Playout Server**, a software which the IRIB is known to use for broadcasting, and subsequently uninstalls the **Matrox DSX** driver, a part of the software for media processing in virtualized broadcast infrastructures,

To combine all the malicious components, another script `layoutabcpxtveni.bat` does several things:

- Renames the MP4 video file located at `c:\windows\temp\TSE_90E11.003` to `TSE_90E11.mp4`. This file was probably dropped there by one of the backdoors, which we discuss later.
- Kills the running process of `QTV.CG.Server.exe`, possibly a part of Autocue QTV broadcasting software, and overwrites the original server located at `D:\CG 1400\QTV.CG.Server.exe` with SimplePlayout, the tool used by the attackers to play their video.
- Copies `c:\windows\SimplePlayout.exe` to `SimplePlayout.ini` in the same directory where `QTV.CG.Server.exe` resides. At least this sample of the batch script contains a typo, as the actors probably meant to copy `SimplePlayout.ini` next to the malicious executable.
- Runs `SimplePlayout.exe` from both the initial and replaced locations.

In another set of related artifacts that we discovered, the attackers utilize the WAV file containing the 25 seconds audio track titled `TSE_90E11.001`, similar to the file name of the MP4 file used in the hijacked TV stream. An executable called `Avar.exe` is based on NAudio, an [open-source](#) .NET audio library, and is responsible for playing the WAV file. Unlike the `SimplePlayback.exe`, `Avar.exe` does not rely on the configuration file. Instead, it contains the path to the WAV file hardcoded as `C:\windows\temp\TSE_90E11.001`. After it executes, `Avar.exe` attempts to enumerate through all active audio devices and play the WAV file on each one.

Finally, a batch script named `avapweiguyyyw.bat` puts the pieces together. It kills a process called `ava.exe` and replaces the executable at `C:\Program Files\MIT\AVA\ava.exe` with `Avar.exe`. The use of the name **Ava** in the files and folders might suggest that these files were intended for IRIB's AVA radio, although the fact it was also [impacted](#) by this attack has not been confirmed officially.

The Wiper

We found two identical .NET samples named `msdskint.exe` whose main purpose is to wipe the computer's files, drives, and MBR. This can also be deduced from the PDB path: `C:\work\wiper\Wiper\obj\Release\Wiper.pdb`. In addition, the malware has the capability to clear Windows Event Logs, delete backups, kill processes, change users' passwords, and more. Both samples were uploaded to VT by the same submitters and in the same timeframe as the previously discussed artifacts.

The image shows a screenshot of Visual Studio. On the left, the 'Assembly Explorer' pane displays a list of methods from the Wiper assembly, including `Wiper()`, `BreakSelectedUsers`, `BuildKillMBR`, `ChangePasswordAndDeleteUsers`, `DeleteLogs`, `DeleteSelectedUsers`, `DestroyMBR`, `DestroyVSS`, `GetActiveMachineUsers`, `GetAllSessions`, `GetAllUsers`, `GetCurrentSession`, `GetCurrentUser`, `GetLastWiped`, `IsExcluded`, `KillMBRCode`, `KillOtherUserSessions`, `KillProcesses`, `ListPartitionsByPriority`, `Output`, `Output(string, string)`, `RandomString`, `ReadAllLines`, `Run`, `RunCmd`, `SaveLastWiped`, `Setup`, `WipeDrives`, `WipeFile`, and `WipeFiles`. The `WipeFile` method is selected. On the right, the code editor shows the implementation of the `WipeFile` method:

```

1 // Wiper.Wiper
2 // Token: 0x06000020 RID: 32 RVA: 0x000033F0 File Offset: 0x000015F0
3 private static void WipeFile(string filePath, bool restricted, int light_wipe = 0)
4 {
5     Wiper.Output(filePath);
6     if (Wiper.IsExcluded(filePath))
7     {
8         Wiper.Output("-- Excluded");
9         return;
10    }
11    try
12    {
13        FileInfo fileInfo = new FileInfo(filePath);
14        if (restricted && !Wiper.importantExtensions.Contains(fileInfo.Extension.ToLower()))
15        {
16            Wiper.Output("-- Skipped");
17        }
18        else
19        {
20            Wiper.WipeFromDisk(fileInfo, filePath, light_wipe);
21        }
22    }
23    catch (Exception ex)
24    {
25        Wiper.Output(ex.Message);
26    }
27 }
28

```

Figure 4 – Overview of the wiper capabilities.

The wiper has three modes to corrupt the files, and fills the bytes with random values:

- default – Overwrite the first 200 bytes of each chunk of 1024 bytes in the file.
- `light-wipe` – Overwrite a number of chunks specified in the configuration.
- `full_purge` – Overwrite the entire file content.

The wiper gets its configuration for the wiping process in one of these ways: in command-line arguments, or from the hardcoded default configuration and exclude list in the file `mececipe.ini`. The default configuration contains a pre-defined list of exclusions related to Windows OS and Kaspersky and Symantec security products, which are widely used in Iran:

```

"-light-wipe", "3",
"-stop-iis",
"-logs",
"-shadows",
"-processes",
"*sql",
"-mbr",
"-fork-bomb",
"-wipe-all",
"-wipe-stage-2",
"-wipe-exclude", "C:\\\\Windows",
"-wipe-exclude", "C:\\\\$Recycle.Bin",
"-wipe-exclude", "C:\\\\$WinREAgent",
"-wipe-exclude", "C:\\\\Config.Msi",
"-wipe-exclude", "C:\\\\Recovery",
"-wipe-exclude", "C:\\\\Program Files\\IBM\\*",
"-wipe-exclude", "C:\\\\System Volume Information",
"-wipe-exclude", "C:\\\\Program Files\\Symantec*",
"-wipe-exclude", "C:\\\\Program Files (x86)\\Symantec*",
"-wipe-exclude", "C:\\\\Program Files\\Kaspersky*",
"-wipe-exclude", "C:\\\\Program Files (x86)\\Kaspersky*",
"-wipe-exclude", "C:\\\\Program Files\\Microsoft*",
"-wipe-exclude", "C:\\\\Program Files (x86)\\Microsoft*",
"-wipe-exclude", "C:\\\\Program Files\\Windows*",
"-wipe-exclude", "C:\\\\Program Files (x86)\\Windows*"

```

If the malware has no arguments, it runs as a service named “ `Service1` ”.

The main wiper function computes the FNV1A32 hash of every argument and uses that to determine the action:

Arguments	Options	Action
“-mbr”	–	Enable <code>DestroyMBR</code> flag
“-fork-bomb”		Start two more instances of the wiper, with the “-fork-bomb” argument as well
“-sessions”	–	Kill other users sessions with the cmd commands: <code>logoff {0}</code> and <code>rwinsta {0}</code>
“-delete-users”	file_path or list of users (* = all users)	Delete the specified users using the cmd command: <code>net user {0} /delete</code>
“-break-users”	file_path or list of users (* = all users)	Break the specified users by changing their password to an 8-bytes random string appended with “ <code>aA1!</code> ”
“-logs”	–	Delete events from Windows Event Log using the cmd command: <code>for /F %1 in ('wevtutil.exe el') DO wevtutil.exe cl %1</code>
“-passwords”	–	None
“-shadows”	–	Destroy shadow copies using the cmd command: <code>echo delete shadows all > 1.s && diskshadow /s 1.s && del 1.s</code>
“-start-iis”	–	Start Internet Information Services (IIS) with <code>iisreset /start</code>
“-stop-iis”	–	Stop Internet Information Services (IIS) with <code>iisreset /stop</code>
“-config”	file_path	Read the arguments from the specified config file
“-light-wipe”	size	Corrupt only specified size of 1024-byte chunks in a file
“-wipe-exclude”	list of directories	Add the directories that the wiper won't wipe
“-delete”	–	Enable <code>delete_files</code> flag which means deleting the files after their corruption
“-processes”	file_path or list of processes (* = all processes)	Kill the specified processes using the cmd command: <code>taskkill /PID {0} /f</code>

Arguments	Options	Action
"-wipe-stage-2"	-	Enable <code>wipe_stage_2</code> flag which means wiping the files by default method and then delete them
"-purge"	-	Enable <code>full_purge</code> flag which means corrupting the whole file and not only chunks
"-wipe-only"	file_path or list of files	Add a list of files to wipe
"-wipe-all"	-	Wipe all the files with supported extensions

`DestroyMBR` flag enables the malware to **wipe the MBR** by writing a hardcoded base64-encoded binary to the file `precg.exe` and then running it. `precg.exe` is an MBRKiller based on the Gh0stRAT MBR wiper.

The main wiping procedure starts by searching for the last file that was wiped. The malware writes its path to the file named `lastfile` (or `lastfile2` in the case of `wipe_stage_2`). Then, every file is checked to see if it is excluded or its extension is not in the predefined list:

```
".accdb", ".cdx", ".dmp", ".h", ".js", ".pnf", ".rom", ".tif", ".wmdb", ".acl", ".cfg", ".doc", ".hlp", ".json", ".png", ".rpt", ".tiff", ".wmv", ".acm", ".chk", ".docx", ".hpi", ".lnk", ".pps", ".rsp", ".tlb", ".xdr", ".amr", ".com", ".dot", ".htm", ".log", ".ppt", ".sam", ".tmp", ".xls", ".apln", ".cpl", ".drv", ".html", ".lst", ".pptx", ".scp", ".tsp", ".xlsx", ".asp", ".cpx", ".dwg", ".hxx", ".m4a", ".pro", ".scr", ".txt", ".xml", ".avi", ".dat", ".eml", ".ico", ".mid", ".psd", ".sdb", ".vbs", ".xsd", ".ax", ".db", ".exe", ".inc", ".nls", ".rar", ".sig", ".wab", ".zip", ".bak", ".dbf", ".ext", ".ini", ".one", ".rar", ".sql", ".wab~", ".bin", ".dbx", ".fdb", ".jar", ".pdf", ".rdf", ".sqlite", ".wav", ".bmp", ".dll", ".gif", ".jpg", ".pip", ".resources", ".theme", ".wma", ".config", ".mxf", ".mp3", ".mp4", ".cs", ".vb", ".tib", ".aspx", ".pem", ".crt", ".msg", ".mail", ".enc", ".msi", ".cab", ".plb", ".plt"
```

The `full_purge` mode that overrides all the bytes of the file is always enabled for the files from `<code>the purge_extensions</code>` list:

```
".json", ".htm", ".log", ".html", ".lst", ".txt", ".xml", ".vbs", ".inc", ".ini", ".sql"
```

If the `delete_files` flag is enabled, the wiper also deletes the files after overwriting them.

We found additional forensics artifacts, submitted together with the wiper samples, that prove that the wiper was indeed executed in a TV environment:

- The `lastfile2` containing the path to the last wiped file: `C:\users\tpa\videos\captures\desktop.ini`. This file is created only if the wiper was run in `wipe_stage_2` mode, which deletes the files after the wiping procedures.
- The `breakusufjkjdil.bat` file, which shows that at least one instance of the wiper was supposed to run with the intent to kill existing user sessions and change passwords for all the users: `"c:\windows\temp\msdskint.exe" -break-users * -sessions`
- The Event Viewer Application log file shows events related to the wiper service `Service1`. The logs contain a timestamp which is a few hours after the attack:

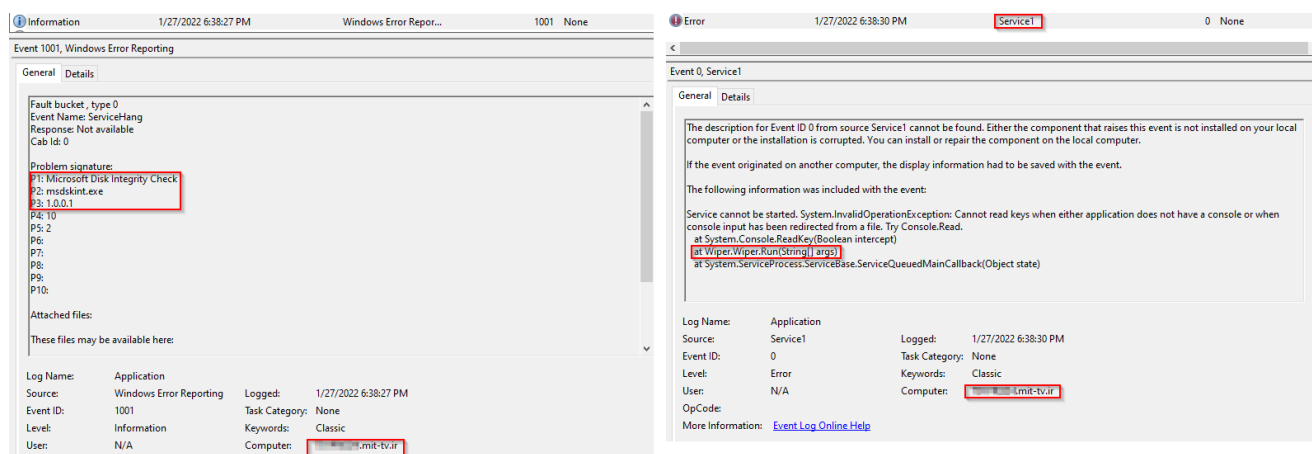


Figure 5 – Windows Event Viewer logs shows the wiper execution in the Iranian TV environment.

Backdoors

WinScreeny

The name of this tool comes from the PDB path: `C:\work\winscreeny\winscreeny\obj\Debug\winscreeny.pdb`. The main purpose of the backdoor is to make screenshots of the victim's computer. We found two samples of this backdoor: the first one is the release version uploaded to VT with the name `mslicval.exe`, and the second one is the debug version named `pregc2.exe`. Needless to say, these files were submitted to VT together with the other artifacts that we discovered.

The backdoor can be run in different ways, based on the command-line argument:

- None – Runs a SimpleTCPService that listens on port 18000.
- `service` – Runs as a service named Service1. At start, the service creates a scheduled task with the command: `schtasks /create /TN \"Microsoft\Windows\NET Framework\NETASM\"/TR \"<file_path>\" /ST <current_time + 1:10> /SC ONCE /F`.
- `setup` – Tries to gain privileges using the `LsaAddAccountRights` API function and then run itself as a service.

The malware listens for packets on port 18000, and for each packet, it checks if the message contains the `scr=` command sent with the POST method. If these conditions are met, the malware saves a screenshot to a file named `screeny-<timestamp>.png` and a “done” message is returned to the attacker if it succeeded.

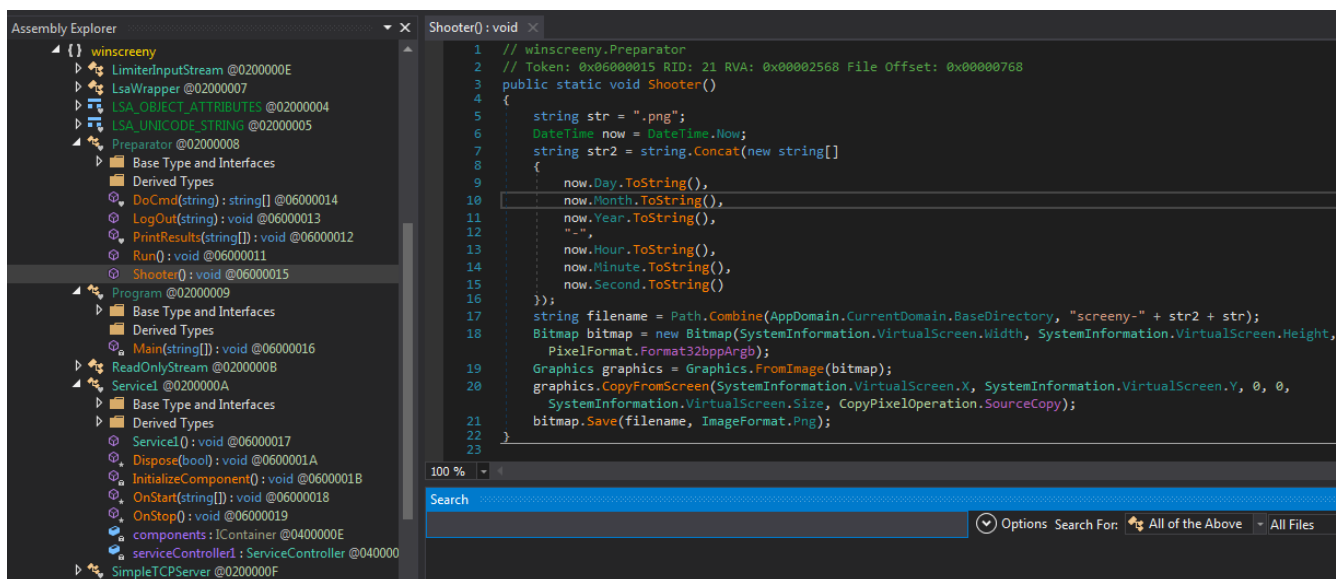


Figure 6 – Winscreeny screenshot capture code.

Interestingly, the release version of this malware is also capable of command execution: it supports the `s=` command which gets a base64-encoded string XORed with 1-byte key 0x24. The decoded string is run by `cmd` and the execution result is returned to the server. The code that handles this feature is also reused in the `HttpService` backdoor that we discuss later.

HttpCallbackService

`HttpCallbackService` is a Remote Administration Tool (RAT) with a familiar PDB path:

`C:\work\simpleserver\HttpCallbackService\obj\Release\HttpCallbackService.pdb`. Its C&C URL can be specified in two different ways: a command-line argument or the configuration file `callservice.ini`. Next, the received value is appended with a short string: `?m=` if the URL ends with `“.aspx”` or `“.php”`; `m=`, if the URL ends with `“/”`, or `/m=` in any other case.

Unfortunately, we didn't find any configuration or other artifacts related to `HttpCallbackService`, so the C&C server in this attack remains unknown.

Every 5 seconds, `HttpCallbackService` sends a request to the C&C URL using the `webClient.DownloadString` method to receive the list of commands split by `'\r\n'`. If the malware doesn't receive any commands in the last 5 minutes and the `isStayAliveMode` flag is disabled, this time frame is increased to 1 minute.

These are the commands supported by the RAT:

Command	Arguments	Action
"upload"	upload_path, base64-encoded content	Upload a file to the victim's computer. The server may send the file in chunks, each of them sequentially decoded from base64 and appended to the file
"download"	file name	Download file from the victim's computer to C&C server, the file is base-64 encoded and sent in chunks of 102400 bytes
"stay-alive"	-	Enable <code>isStayAliveMode</code> flag and change the timer to 5 seconds
"cool-down"	-	Disable the <code>isStayAliveMode</code> flag
Default	command string	Run the command in cmd and return the result to the C&C server

When the results of the commands are uploaded to the server, the data is sent to a slightly different URL: the C&C URL defined previously, now appended with "1". The data is sent using the `WebClient.UploadValues` method in the following format:

- `download=<file_name>\r\n-----\r\n<base64 of chunk>` for the download command
- `<command>\r\n-----\r\n<result>` for the cmd command.

HttpService

HttpService is another backdoor that listens on a specified port: it can be a command-line argument, the pre-defined port depending on the sample, or the value from the configuration file: `<exe_name>.ini`. We found several samples with the default ports 19336, 19334, 19333, as well as two different configuration files uploaded to VT, with 19336 and 19335 values.

Each sample has a hardcoded version. The files that we discovered belong to three different versions: 0.0.5, 0.0.11v4H and 0.0.15v4H. The version 0.0.5 listens to the specified port with a Simple TCP server, whereas 0.0.11v4H and 0.0.15v4H are based on the Simple HTTP Server. All of them use the HTML Agility Pack for HTML parsing and IonicZip library for compression actions.

The highest version (0.0.15v4H) of the backdoor has multiple capabilities, including command execution and manipulation with the files.

Command execution: The command "`cmd`" makes the backdoor run the specified command with `cmd.exe` and return the result in this format: `<div style='color: red'><result_string></div>`. In addition, the backdoor can launch an interactive cmd shell when it receives the "`i=`" command, whose arguments can be:

- "`1`" – Get the output from the shell and send it back to the C&C.
- "`2`" – End the interactive shell and clean up.
- default – Decode and decrypt the XORed string and then run the command in the shell and save the output.

Similar to WinScreeny, the malware also has the "`s=`" command with the string XORed with 1-byte key `0x24` as an argument. The decoded string is run by `cmd.exe` and the result is returned to the server.

Proxy connections: After the "`p=`" or "`b=`" command is received, the backdoor uses the victim's computer as a proxy to the URL it gets as an argument. The backdoor communicates with this URL, redirects the request of the C&C server, and waits for a response to send it back to the C&C.

Download and upload files: The "`f=`" or "`1=`" command allows the backdoor to download a file from the path given as an argument or write a file given as an argument with the content of the message body. After it receives the "`m=`" command, the malware writes the body of the message to the path `<base_directory><client_address>.out`, reads data from `<base_directory><client_address>.in`, and sends it to the C&C. If the file does not exist, the malware creates the file and writes to it the current date and time.

Run SQL commands: The "`con=`" / "`c=`" command receives the SQL DB connection string and SQL query, and returns the result to the server.

Manipulate the local files: The "`<path>`" command checks if the file/directory exists and then does one of these three things, based on the query value:

- “ `zip` ” – Creates a zip file from the directory contents and returns it to the C&C.
- “ `unzip` ” – Unzips the file using the path provided by the C&C.
- “ `del` ” – Deletes the file.

Interestingly, in all three cases, the malware sends back the entire directory contents (including sub-directories) as an HTML page that contains the `Zip` , `Unzip` and `Delete` buttons, depending on the type of the file. This is how the interface looks on the attackers’ side:

```

..
~ 500kb testdir | Zip
05.07.2021 19:26 | ~ 10kb testfile.txt | Delete
05.07.2021 19:27 | ~ 50kb testfile.zip | Unzip here | Delete

```

Figure 7 – HTML page with the directory listing returned to the C&C server.

ServerLaunch dropper

The sample of HttpServer version 0.0.5 was submitted together with its dropper, called `dwDrvInst.exe` , which mimics the remote access software executable by DameWare. The tool’s PDB path has the same pattern, `C:\work\ServerLaunch\Release\ServerLaunch.pdb` . However, the tool is written in C++, not .NET like all the others, and was compiled on December 2, 2021, almost 2 months prior to the attack.

ServerLaunch contains three executable in resources, which it drops to `ionic.zip.dll` , `httpservice2` and `httpservice4` , all in `C:\Users\Public\` . The malware then starts both `httpservice2` and `httpservice4` with no arguments. Each of them has a different pre-defined port to listen on, which likely allows the attackers to ensure some sort of redundancy of the C&C communication.

Connecting the files to the attack

We’ve discussed several different tools and some of artifacts related to their execution. It is clear that all these tools were created by the same actor and are connected. For example, the screenshot tool Winscreeny doesn’t contain the functionality to upload the created screenshots back to the attackers, which likely means that it relies on other backdoors to perform this operation. The recurring `Service1` name for all the tools indicates that different backdoors, if running on the same machine were mostly executed with command-line arguments or provided configuration files.

Taking into account that the samples are related to each other, we can substantiate the connection between these files and the IRIB cyberattack:

- The whole cluster of activity is interconnected and was submitted to VT mostly from Iranian IPs all at the same timeframe, likely by incident responders.
- The audio and video files utilized by the tools are the same as those broadcast live on hacked Iranian TV. The Twitter account @GhyamSarnegouni (“Uprising to overflow”) featured in this video contains a few recordings of different TV channels streams that feature both the video and the audio tracks we’ve discussed.
- Multiple artifacts such as Matrox DSX, Autocue QTV, TFI Arista Playout Server, etc. that were referenced in the samples indicate that these files were intended for a broadcast environment.
- Among the forensics artifacts submitted together with video and executables, we discovered Windows Event Viewer files that contain evidence that the samples were attempted to be executed in the Iranian TV network environment, a domain not resolved publicly. The timestamp of these specific logs is after the time of the actual incident.

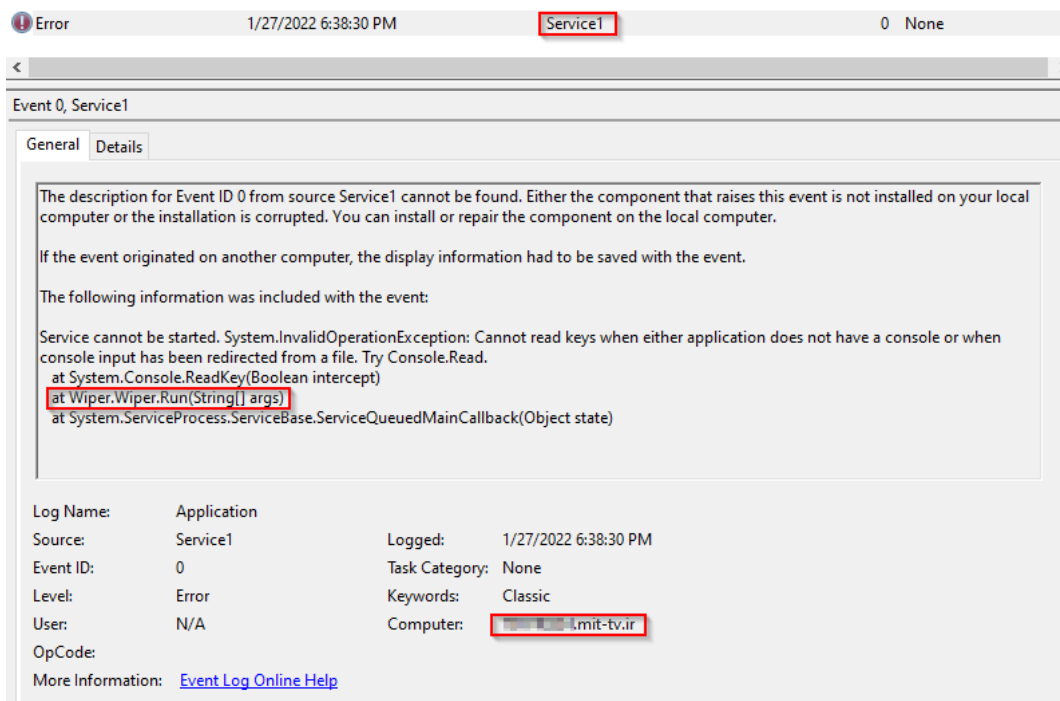


Figure 8 – Screenshot of the Application log that contains the wiper execution evidence.

Numerous other forensics evidence from this VT file cluster contains other artifacts directly related to IRIB. For example, an internal tool called `MIT_FreeSizeService` (md5:307e7440a15c8eed720566f067a2e96b) bears the IRIB logo, and the memory dump of the `MetaSAN` software called `executable.4504.exe` (md5:1fc57ccec4668bbcbebaa9c734a437ba) features memory strings that indicate the software was run on the machine from the `MIT-TV` domain.

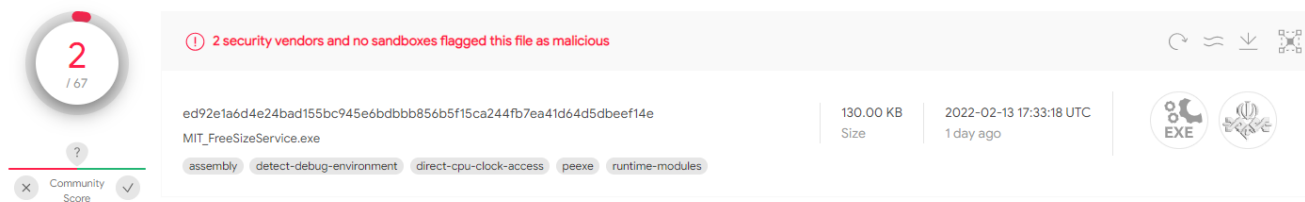


Figure 9 – VT submission of the unknown tool featuring the MIT (same as the domain name) string and containing the IRIB logo

Attribution

Iranian officials appear to be confident that MEK is behind this attack, with the deputy head of technical affairs for Islamic Republic of Iran Broadcasting claiming the same. However, the opposition group itself denies any involvement, stating that “the group had become aware of the incident only when it happened but that the hacking might have been the work of supporters in Iran.”

The hacktivist group Predatory Sparrow, which claimed responsibility for the attacks against the national railway services, the transportation ministry, and the Iranian gas stations, affiliated itself with the IRIB attack via its Telegram channel. On the morning before the attack, they wrote “Wait for the good news from our team. Do not switch the channel.” Later the same evening, they posted a video from one of the disrupted TV channels, introducing it as a “cyber-attack on the country’s radio and television organization by the Predatory Sparrow team.” Currently, no technical proof of the group’s attribution to the attack has been discovered. The video displayed on the channel is available online and refers to a different Telegram account `@GhyamSarnegouni`, so the claims should be treated with caution.



Figure 10 – Posts from ‘Predatory Sparrow’s Telegram channel, in which the group claims responsibility for the attack.

Conclusion

In this article, we analyzed a set of tools that likely was used in a cyberattack against the IRIB, which disrupted several state-run TV and radio channels. The use of wiper malware in the attack against a state entity in Iran begs us to compare the tools with those belonging to Indra, who, among other attacks, is responsible for unleashing a wiper in the Iranian Railways and Ministry of Roads systems. Although these wipers are coded and behave very differently, some implementation details such as execution based on batch files, or the password changing patterns ([random sequence]aA1! for this attack and Aa153![random sequence] in Indra’s case), suggests that the attackers behind the IRIB hack may have been inspired by previous attacks happened in Iran.

As in the case with Indra, it appears that the actor may have many capabilities that have yet to be explored. On the one hand, the attackers managed to pull off a complicated operation to bypass security systems and network segmentation, penetrate the broadcaster’s networks, produce and run the malicious tools that heavily rely on internal knowledge of the broadcasting software used by victims, all while staying under the radar during the reconnaissance and initial intrusion stages.

On the other hand, the attackers’ tools are of relatively low quality and sophistication, and are launched by clumsy and sometimes buggy 3-line batch scripts. This might support the theory that the attackers might have had help from inside the IRIB, or indicate a yet unknown collaboration between different groups with different skills.

Meanwhile, almost two weeks after the attack happened, MEK-affiliated news published a status report of the attack claiming that the “regime’s radio and TV networks have not returned to a normal status” and provided an elaborate list of affected devices with the statement “more than 600 servers, advanced digital production, archiving, and broadcasting of radio and television equipment have been destroyed, and their software has been damaged.” There is no way for us to verify these claims, but if at least some of them are true, the extent of destruction caused by the wiper and other malicious tools that we’ve discovered (and those that are yet unknown), exceeded expectations.

IOCs

Attack files:

hash	name	description
1607f31ac66dfec739dc675ade921582acb8446c2ac7d6d1bc65a3e993fc5b54	msdskint.exe	Wiper
42ed646eed4f949c456c637a222e7d94dd8ac67ed5ebda5e63c7b7979076d9cf	msdskint.exe	Wiper

hash	name	description
8bdf6e262966a59a7242d279e511dd694467f07d1d76c456a0c26d0db2ec48a8	HttpService2.exe	HttpService
427c105859c3dc62ece790e41a42b0f6ae587496a07d3bd190143179cdf6c6bd	HttpService4.exe	HttpService
e3d61cbbf4e41295dd52acff388d1d8b1d414a143d77def4221fd885aae6cd83	HttpService2.exe	HttpService
096bae94e09059e2e3106503353b1b4f7116fa667600ca2ab3fa7591708e645a	HttpService4.exe	HttpService
13a016b8f502c81e172c09114f25e4d8a8632768aefd56c5f6d147e9b6466216	HttpService4.exe	HttpService
ea740894227ae1df923997edb7bda3a00f523bfff7cc02d3b5e6b3de19d672fc	HttpCallbackService.exe	HttpCallbackService
62b692be251feb63af2723a68975976b749cab20014ffaa6488af80f4f03e0a1	dwDrvInst.exe	ServerLaunch
41e0c19cd6a66b4c48cc693fd4be96733bc8ccbe91f7d92031d08ed7ff69759a	pregc2.exe	Winscreeny
e9e4a8650094e4de6e5d748f7bc6f605c23090d076338f437a9a70ced4a9382d	mslicval.exe	Winscreeny
d788ebc7ee98c222f46d7ca2347027643784a78b5954c9a31734ec1b197bc2aa	Avar.exe	Avar
1155dd06e0b108bde3addcbbd5d1da4dc18ca245c39ce7d967f8971eb0f88dbb	SimplePayout.exe	SimplePayout
a25215c9adce51a3ecfe34c802d3e7d865cf410d8be10101e3b41f6ba11347a4	TSE_90E11.mp4	MP4 video file
4cc21810d786dca94e01d0714d37e3f097ff6e3813bf6e17a9bd86cd9a4ceb2b	TSE_90E11.001	WAV file
7ea7b20b87ded3c297ec0890ee8a396427d70caf983b42f479d8fad38629b684	playoutabcpxtveni.bat	
bc8de80a28c8ae55415ccdfece270f6548f067fc2a00e799baf0279d4d560807	breakusufjkjdl.bat	
197f13580ec249fa84b1e54f978c5cab60f22561a2fab2ff60bdb2d5bfa25512	avapweiguyyyw.bat	
efc8f12c53d1730fa8ac00cfa60e63ab43d90f42879ef69d7f6fb9978246f9cb	playjfalcfgcdq.bat	
a2d493c2cb25fc03f5d31cf3023b473d71d38b972eccdb7873f50d2344ea7753	simplepayout.ini	
c305b3cb96a34258a3e702526de6548b2de99449c0839a9aea518acc7c861ab	436748-HttpService4.exe.ini	
8b74c08c33cd8a0cc1eaf822caead6b54bc39e4839e575f3c0ece4bb8992408	436751-HttpService4_2.exe.ini	

Forensics artifacts:

hash	name	description
0daa0aefdc6d0641eb06e62bc8c92a0696aa8089258cb2d3552ac137d53237ec	sec.evtx	security event log from one of the machines
a3b9bd57e6b281610e570be87883d907992bdf7be3bcd37885ee2cf97d930cd3	application.evtx	applications event log from one of the machines
067ae6ecfd108a79a32eb1a76a262868d8f3a9a7924b26091f0e2229152bdd9d	lastfile2	path to the last file wiped and deleted by the wiper