



**2022
PRAGUE**

28 - 30 September, 2022 / Prague, Czech Republic

EVILPLAYOUT: ATTACK AGAINST IRAN'S STATE TV AND RADIO BROADCASTER

Alexandra Gofman, Israel Gubi & Itay Cohen
Check Point, Israel

alexandrag@checkpoint.com

israelgu@checkpoint.com

itayc@checkpoint.com

ABSTRACT

The anniversary of the 1979 Islamic Revolution is a major annual political celebration in Iran: celebrations include the streets being decorated with huge Iranian flags and balloons, rallies commemorating the achievements of the revolution, and of course state radio and television broadcasting patriotic songs and programmes lauding Iran's scientific and military achievements. On 27 January 2022, only two weeks before this year's highly celebrated event, reports were published [1] revealing that the IRIB, Iran's national broadcaster, had been hacked. The Islamic Republic of Iran Broadcasting (IRIB), 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 cyber attack resulted in state-run TV channels broadcasting what was described by IRIB officials as 'the faces and voices of hypocrites'.

Our investigation sheds light on the previously undiscussed technical implementation of this attack. We found malicious executables used to air the opposition message, as well as multiple scripts and backdoors that enabled their execution. In addition, we discovered evidence that the attackers turned to wiper malware, indicating that the actor's intention was not only to hijack the broadcast but also to irreversibly damage the networks of the state-affiliated TV and radio stations.

INTRODUCTION

'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 [2] overthrowing the current regime and installing its own government, relying on an alternative interpretation of Islam. In the hijack video that was shown on state-run TV channels, the faces of MEK leaders Maryam and Masoud Rajavi appeared, followed by an 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 the attack affected not only the TV channels, but also a number of state-operated radio channels.

The video referenced the *Twitter* and *Telegram* accounts of a previously unknown group called GhyamSarnegouni ('Uprising to overthrow').



Figure 1: A frame, showing the opposition leaders' faces, from the video broadcast on state-run Iranian TV channels as a result of the cyber attack.

IRIB ATTACK ARTIFACTS

According to the Iranian state-run news network Akhbarin Khabar (Latest News) [3], '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 to be the most likely scenario, with the Iranian officials calling the attack 'extremely complex' [4].

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

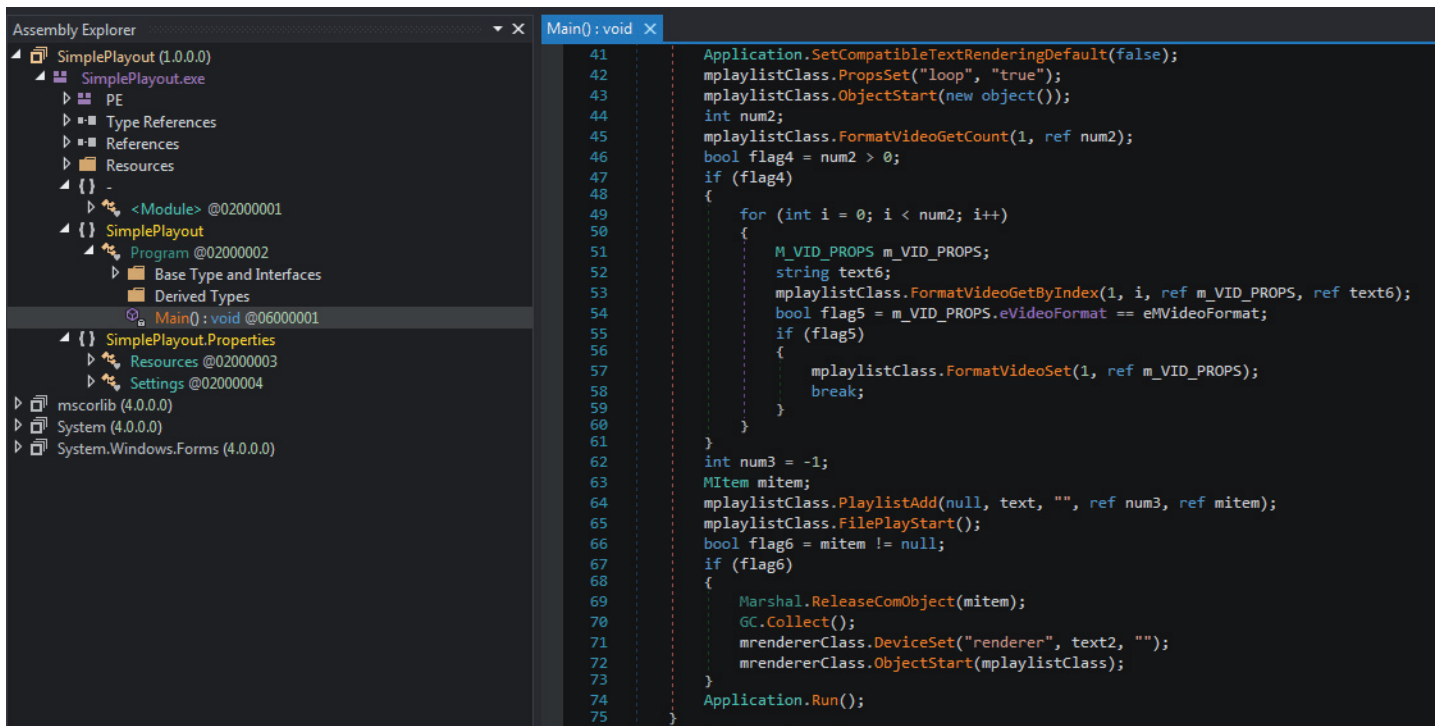
- Establishing backdoors and their persistence in the victims' networks.
- Launching the 'malicious' video or audio track.
- Installing wiper malware in an attempt to disrupt operations in the hacked networks.

All of these samples had been uploaded to *VirusTotal* (VT) from multiple sources, mostly with Iranian IPs, and included short batch scripts that install or launch payloads, several forensic artifacts like *Windows Event Log* files or memory dumps,

and the payloads themselves. The latter were mostly .NET executables, with no obfuscation but a time-stamped compilation date in the future. In addition to having the same language and same VT submitters, the files also have 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 hijacking of the broadcast, supposedly run on servers that broadcast TV programmes (playouts). To play the video file, the attackers used a program called SimplePayout.exe, a .NET-based executable compiled in debug mode with the PDB path c:\work\SimplePayout\obj\Debug\SimplePayout.pdb. This executable has a single functionality: to play a video file in a loop using the .NET MPlatform SDK by Medialooks.



```

Assembly Explorer
└─ SimplePayout (1.0.0.0)
  └─ SimplePayout.exe
    └─ PE
      └─ Type References
      └─ References
      └─ Resources
      └─ {} -
        └─ <Module> @02000001
        └─ {} SimplePayout
          └─ Program @02000002
            └─ Base Type and Interfaces
            └─ Derived Types
            └─ Main(): void @06000001
              └─ {} SimplePayout.Properties
                └─ Resources @02000003
                └─ Settings @02000004
  └─ mscorlib (4.0.0.0)
  └─ System (4.0.0.0)
  └─ System.Windows.Forms (4.0.0.0)

Main(): void
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 }
62 int num3 = -1;
63 MItem mitem;
64 mplaylistClass.PlaylistAdd(null, text, "", ref num3, ref mitem);
65 mplaylistClass.FilePlayStart();
66 bool flag6 = mitem != null;
67 if (flag6)
68 {
69     Marshal.ReleaseComObject(mitem);
70     GC.Collect();
71     mrendererClass.DeviceSet("renderer", text2, "");
72     mrendererClass.ObjectStart(mplaylistClass);
73 }
74 Application.Run();
75

```

Figure 2: Part of the SimplePayout code using the MPlatform SDK to play the video file.

First, the SimplePayout program looks for a configuration file called SimplePayout.ini, which contains two lines: the video file path, and a number representing the video format. The respective SimplePayout.ini file, uploaded to VT together with SimplePayout.exe, 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 50Hz.

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

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

- It 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.
- It kills the running process of QTV.CG.Server.exe, which is possibly a part of the *Autocue QTV* broadcasting software, and overwrites the original server located at D:\CG_1400\QTV.CG.Server.exe with SimplePayout, the tool used by the attackers to play their video.
- It copies c:\windows\SimplePayout.exe to SimplePayout.ini in the directory in which QTV.CG.Server.exe resides. At least this sample of the batch script contains a typo, as the actors probably meant to copy SimplePayout.ini next to the malicious executable.
- It runs SimplePayout.exe from both the initial and the replaced locations.

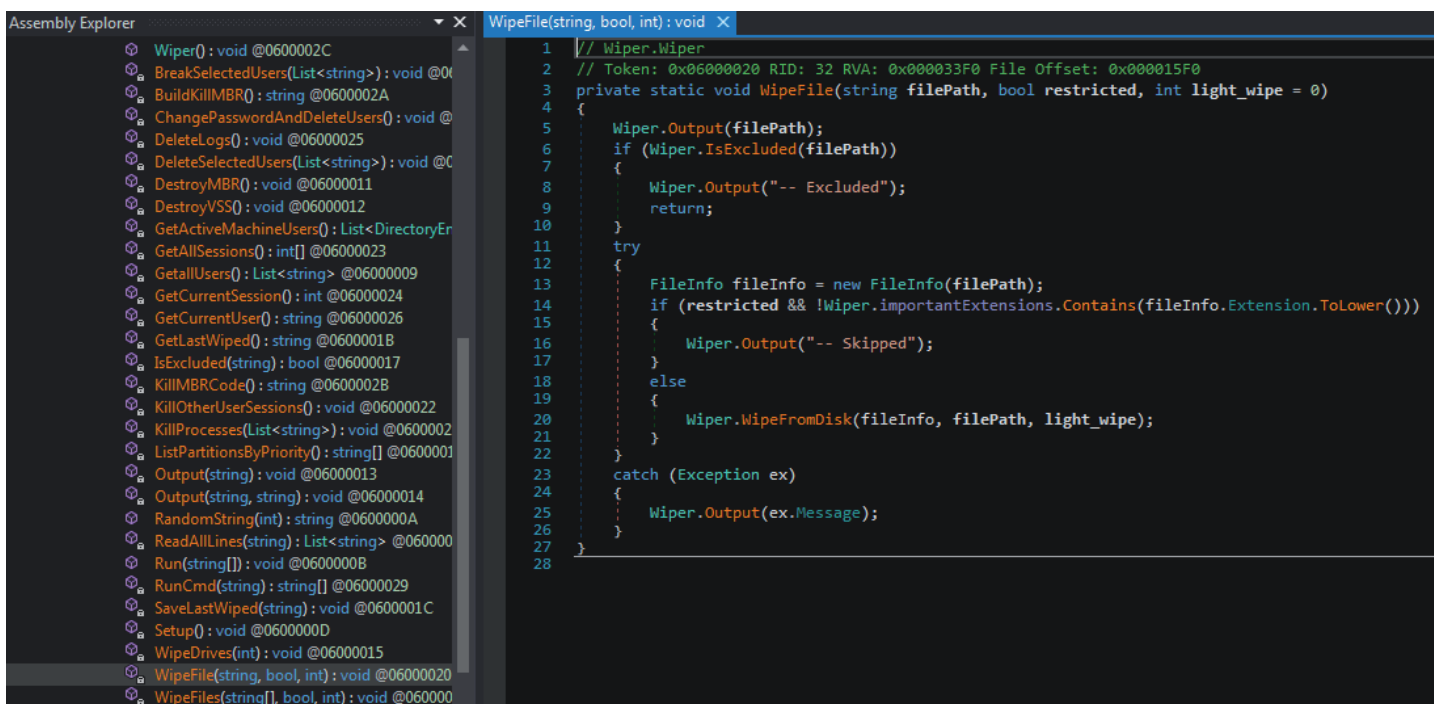
In a set of related artifacts that we discovered, the attackers utilize a WAV file containing the 25-second audio track titled TSE_90E11.001, which is similar to the file name of the MP4 file used in the hijacked TV stream. An executable called Avar.exe is based on *NAudio*, an open-source .NET audio library [5], and is responsible for playing the WAV file.

Unlike `SimplePayout.exe`, `Avar.exe` does not rely on a configuration file. Instead, it contains the path to the WAV file hard coded 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 `avapweiguuyyw.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 it having been impacted by this attack was stated only by MEK-affiliated media [6] and has not been confirmed officially.

THE WIPER

In addition to the files related to video and audio hijacking, we discovered evidence that the attackers used `wiper` malware: it appears, that the actor's intention was not only to hijack the broadcast but also to irreversibly damage the state-affiliated TV and radio stations' networks. 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 of the samples: `C:\work\wiper\Wiper\obj\Release\Wiper.pdb`. In addition, the malware has the ability 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 screenshot shows the Assembly Explorer window with the `WiperFile(string, bool, int): void` method selected. The code is as follows:

```

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 3: Overview of the wiper capabilities.

The wiper has three modes to corrupt files by filling the bytes with random values:

- default – overwrite the first 200 bytes of each chunk of 1,024 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 either in command-line arguments or from a hard-coded default configuration and exclude list in the file `mecliwipe.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 the <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 eight-byte random string appended with 'aA1!'
'-logs'	-	Delete events from <i>Windows Event Log</i> using the cmd command: for /F %tokens% in ('wevtutil.exe el') DO wevtutil.exe cl %tokens%
'-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 the <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>
'-wipe-stage-2'	-	Enable the <code>wipe_stage_2</code> flag, which means wiping the files using the default method, and then delete them
'-purge'	-	Enable the <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

The `DestroyMBR` flag enables the malware to wipe the MBR by writing a hard-coded base64-encoded binary to the file `pregc.exe` and then running it. The `pregc.exe` file 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 the `purge_extensions` 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 forensic artifacts, submitted together with the wiper samples, that prove that the wiper was indeed executed in a TV environment:

- The `lastfile2` file 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:

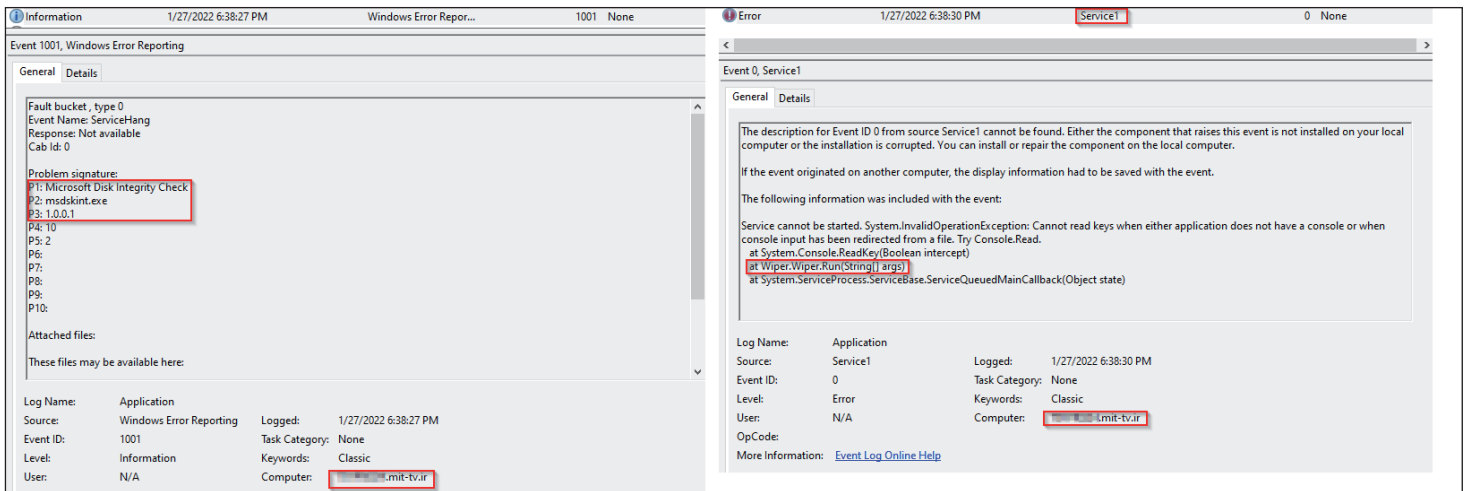


Figure 4: Windows Event Viewer logs show the wiper execution in the Iranian TV environment.

BACKDOORS

WinScreeny

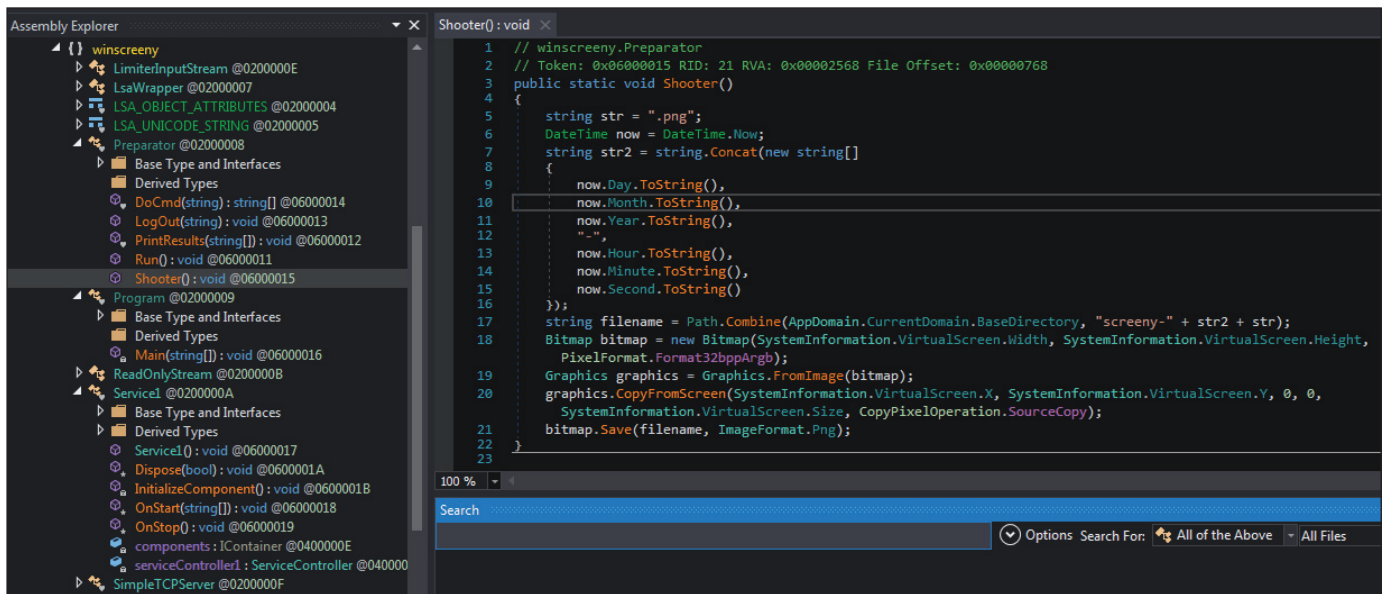
The name of this tool comes from the PDB path: `C:\work\winscreeny\winscreeny\obj\Debug\winscreeny.pdb`. The main purpose of the backdoor is to take screenshots of the victim's computer. We found two samples of this backdoor: the first is the release version uploaded to VT with the name `mslicval.exe`, and the second 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 `SimpleTCPServer` 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 succeeds.



```

1 // winscreeny.Preparator
2 // Token: 0x06000015 RID: 21 RVA: 0x00002568 File Offset: 0x00000768
3 public static void Shooter()
4 {
5     string str = ".png";
6     DateTime now = DateTime.Now;
7     string str2 = string.Concat(new string[]
8     {
9         now.Day.ToString(),
10        now.Month.ToString(),
11        now.Year.ToString(),
12        "-",
13        now.Hour.ToString(),
14        now.Minute.ToString(),
15        now.Second.ToString()
16    });
17    string filename = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "screeny-" + str2 + str);
18    Bitmap bitmap = new Bitmap(SystemInformation.VirtualScreen.Width, SystemInformation.VirtualScreen.Height,
19        PixelFormat.Format32bppArgb);
20    Graphics graphics = Graphics.FromImage(bitmap);
21    graphics.CopyFromScreen(SystemInformation.VirtualScreen.X, SystemInformation.VirtualScreen.Y, 0, 0,
22        SystemInformation.VirtualScreen.Size, CopyPixelOperation.SourceCopy);
23    bitmap.Save(filename, ImageFormat.Png);
24 }

```

Figure 5: 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 XOR'ed with one-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 five seconds, `HttpCallbackService` sends a request to the C&C URL using the `webClient.DownloadString` method to receive a list of commands split by `'\r\n'`. If the malware hasn't received any commands in the last five minutes and the `isStayAliveMode` flag is disabled, this time frame is increased to one 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 the file from the victim's computer to the C&C server. The file is base-64 encoded and sent in chunks of 102,400 bytes.
'stay-alive'	–	Enable the <code>isStayAliveMode</code> flag and change the timer to five seconds.
'cool-down'	–	Disable the <code>isStayAliveMode</code> flag.
Default	command string	Run the command in <code>cmd</code> 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 and 19333, as well as two different configuration files uploaded to VT, with 19336 and 19335 values.

Each sample has a hard-coded version. The files that we discovered belong to three different versions: 0.0.5, 0.0.11v4H and 0.0.15v4H. 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 of the backdoor (0.0.15v4H) has multiple capabilities, including command execution and manipulation of 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 XOR'ed 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 XOR'ed with one-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 `'l='` 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 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:



Figure 6: 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 2 December 2021, almost two months prior to the attack.

ServerLaunch contains three executable 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 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 in 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 account @GhyamSarnegouni ('Uprising to overthrow') referenced in the video contains a few recordings of different TV channel 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 forensic artifacts submitted together with video and executables, we discovered *Windows Event Viewer* files that contain evidence that attempts had been made to execute the samples 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.

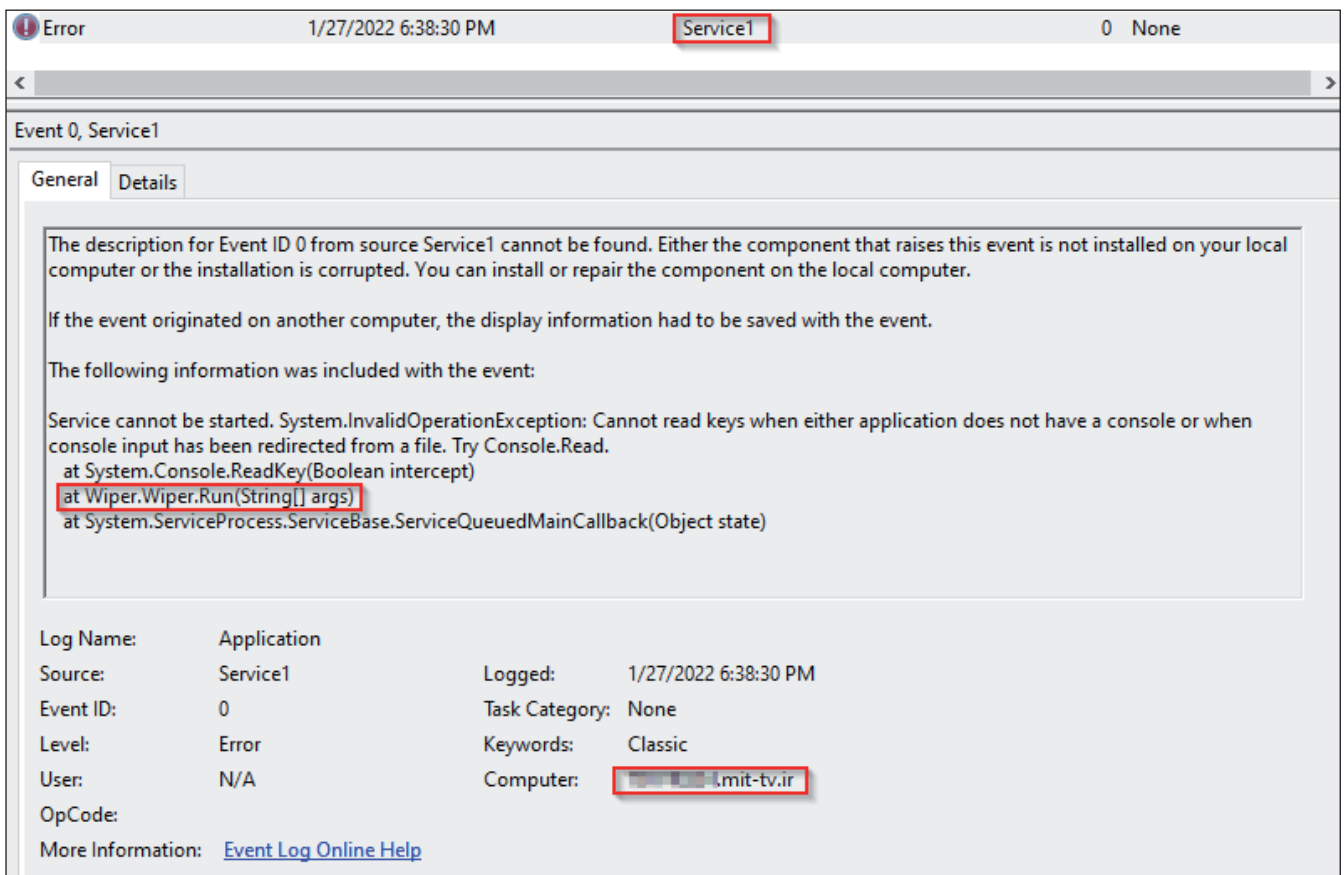


Figure 7: Screenshot of the Application log that contains evidence of the wiper execution.

- Numerous other pieces of forensic evidence from this VT file cluster contain 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` (md5:1fc57ccec4668bbcbcbbaa9c734a437ba) features memory strings that indicate the software was run on a machine from the MIT-TV domain.



Figure 8: VT submission of the unknown tool featuring the MIT (same as the domain name) string and containing the IRIB logo.

ATTRIBUTION AND FURTHER THREAT ACTOR ACTIVITY

Iranian officials appear to be confident that MEK is behind this attack, with the deputy head of technical affairs for the Islamic Republic of Iran Broadcasting claiming the same [7]. However, the opposition group itself denies any involvement, stating that ‘the group had become aware of the incident only when it happened’ and that ‘the hacking might have been the work of supporters in Iran’ [8].

The hacktivist group Predatory Sparrow, which claimed responsibility for multiple attacks that happened in 2021, including an attack against the national railway services and Iranian gas stations, affiliated itself with the IRIB attack via its *Telegram* channel. On the morning before the attack, the group wrote [9] ‘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 [10], introducing it as a ‘cyber-attack on the country’s radio and television organization by the Predatory Sparrow team’. No technical proof of the group’s connection 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 9: Posts from Predatory Sparrow’s Telegram channel, in which the group announced the attack.

The video broadcast during the attack contained the logo and referenced *Twitter* and *Telegram* accounts of a previously unknown group called GhyamSarnegouni (‘Uprising to overthrow’). During the months following the IRIB attack, the actors behind this account continued their activity against the Iranian national infrastructure, meticulously documenting all their achievements in the *Telegram* group.

In March 2022, they hacked the portal of the Ministry of Culture and Islamic Guidance and its affiliated websites, with the front page replaced by – yet again – photos of leaders of the MEK and a photo of Supreme Leader Ali Khamenei with a large red X drawn on his face. The actor also shared in the *Telegram* group some of the documents leaked from the hacked

networks and videos of attackers allegedly wiping the data from the victims' internal systems. Later in April, the group claimed a very similar attack on another Iranian government body, the Ministry of Agriculture Jihad, once again defacing the website, publishing the leaked documents and videos of the data being destroyed.

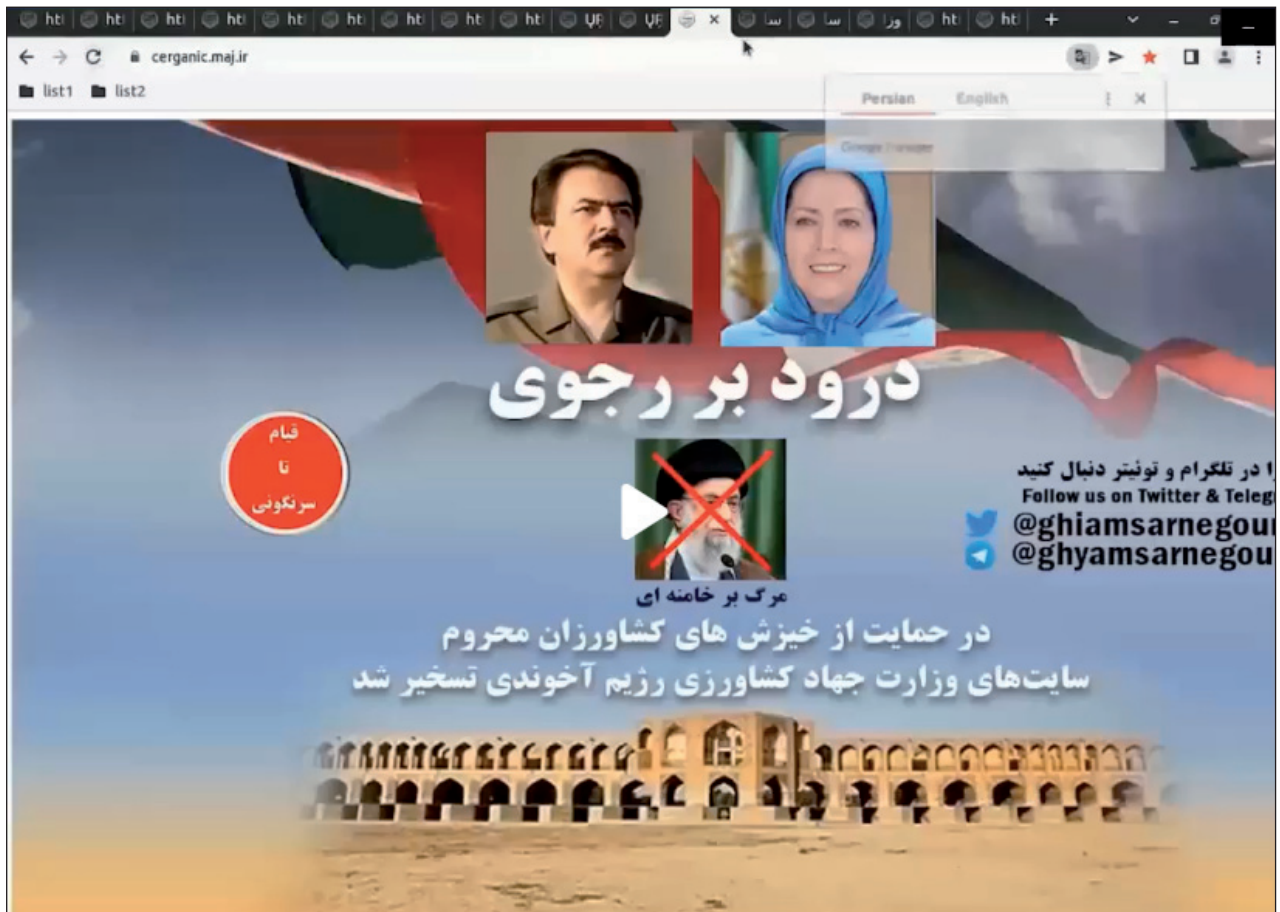


Figure 10: A frame from the video published in the GhyamSarnegouni Telegram group showcasing the defaced website of the Ministry of Agriculture Jihad.

Finally, in early June 2022, on the eve of commemorative ceremonies for the anniversary of the death of the late founder of the Islamic Republic Ruhollah Khomeini, the group hacked into over 5,000 surveillance cameras and online services of Tehran Municipality, once again documenting all their findings with videos.

During these months, we've also seen a few other variants of the wiper uploaded to *VirusTotal* from different Iranian accounts. These samples contained the *pdb* paths `C:\work\forsaken\forsaken\obj\Release\dilemma.pdb` and `C:\work\wiper\forsaken\obj\Release\dilemma.pdb`, indicating that the internal name of the wiper changed from 'Wiper' to 'Dilemma'. In June, Iranian news [11] and Iranian security company *Padvish* issued warnings [12] about this malware 'detected in Iran's infrastructure systems', likely indicating that it was used in at least some of the aforementioned attacks.

CONCLUSION

In this paper, we analysed a set of tools that were used in a cyber attack against the IRIB, which disrupted several state-run TV and radio channels: we described multiple backdoors, malicious scripts and executables whose purpose was to air a protest message. In addition, we discovered evidence that a piece of wiper malware was used, which indicates that the attackers' aim was not only to deliver a protest message, but also to disrupt the state's broadcasting networks, with the damage to the TV and radio networks possibly more serious than officially reported.

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 malicious tools that rely heavily 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 three-line batch scripts. This might support the theory that the attackers had help from inside the targeted networks, or indicate a yet unknown collaboration between different groups with different skills.

Following the actor's activity in the months after the IRIB attack, we discovered that the actor continues to target Iran's governmental networks, with a similar aim to protest against the regime, disrupt the affected networks and destroy their data.

REFERENCES

- [1] Reuters. Iran's state broadcaster says it was hacked for 10 seconds. January 2022. <https://www.reuters.com/world/middle-east/irans-state-broadcaster-says-it-was-hacked-10-seconds-2022-01-27/>.
- [2] Merat, A. Terrorists, cultists – or champions of Iranian democracy? The wild wild story of the MEK. The Guardian. November 2018. <https://www.theguardian.com/news/2018/nov/09/mek-iran-revolution-regime-trump-rajavi>.
- [3] <https://t.me/akharinkhabar/301980>.
- [4] alarabiya news. Iran's state broadcaster says it was hacked for 10 seconds. January 2022. <https://english.alarabiya.net/News/middle-east/2022/01/27/Iran-s-state-broadcaster-says-it-was-hacked-for-10-seconds>.
- [5] <https://github.com/naudio/NAudio>.
- [6] Mahoutchi, F. Iran's state TV disrupted, airing footage of opposition leaders. People's Mojahedin Organization of Iran. January 2022. <https://english.mojahedin.org/news/iran-news/iran-state-tv-radio-massoud-maryam-rajavi-khamenei-irib-20220127/>.
- [7] Motevalli, G.; Sykes, P. Iran State TV Says Exiled Dissidents Hacked Live Broadcasts. Bloomberg. January 2022. <https://www.bloomberg.com/news/articles/2022-01-27/iran-state-tv-says-exiled-dissidents-briefly-hacked-broadcasts>.
- [8] Sinaee, M. MEK Opposition Group Denies It Hacked Iran State TV And Radio. Iran International. January 2022. <https://www.iranintl.com/en/202201270195>.
- [9] <https://t.me/GonjeshkeDarande/143>.
- [10] <https://t.me/GonjeshkeDarande/146>.
- [11] <https://www.farsnews.ir/news/14010321000810/%D8%A8%D8%AF%D8%A7%D9%81%D8%B2%D8%A7%D8%B1-%D8%AC%D8%AF%DB%8C%D8%AF-%D8%AF%D8%B1-%D8%B3%DB%8C%D8%B3%D8%AA%D9%85%E2%80%8C%D9%87%D8%A7%DB%8C-%D8%B2%DB%8C%D8%B1%D8%B3%D8%A7%D8%AE%D8%AA%DB%8C-%D8%B4%D9%86%D8%A7%D8%B3%D8%A7%DB%8C%DB%8C-%D8%B4%D8%AF>.
- [12] HackTool.Win32.APT- PS. <https://threats.amnpardaz.com/malware/apt/hacktool-win32-apt-ps/>.