Industroyer2: Industroyer reloaded

By ESET Research

Archived: 2026-04-05 14:30:29 UTC

Executive summary

The blogpost presents the analysis of a cyberattack against a Ukrainian energy provider.

Key points:

- ESET researchers collaborated with CERT-UA to analyze the attack against the Ukrainian energy company
- The destructive actions were scheduled for 2022-04-08 but artifacts suggest that the attack had been planned for at least two weeks
- The attack used ICS-capable malware and regular disk wipers for Windows, Linux and Solaris operating systems
- We assess with high confidence that the attackers used a new version of the Industroyer malware, which was used in 2016 to cut power in Ukraine
- We assess with high confidence that the APT group Sandworm is responsible for this new attack

Industroyer2: Industroyer reloaded

ESET researchers responded to a cyber-incident affecting an energy provider in Ukraine. We worked closely with CERT-UA in order to remediate and protect this critical infrastructure network.

The collaboration resulted in the discovery of a new variant of Industroyer malware, which we together with CERT-UA named Industroyer2 – see CERT-UA publication here. Industroyer is an infamous piece of malware that was used in 2016 by the Sandworm APT group to cut power in Ukraine.

In this case, the Sandworm attackers made an attempt to deploy the Industroyer2 malware against high-voltage electrical substations in Ukraine.

In addition to Industroyer2, Sandworm used several destructive malware families including CaddyWiper, ORCSHRED, SOLOSHRED and AWFULSHRED. We first discovered CaddyWiper on 2022-03-14 when it was used against a Ukrainian bank – see our Twitter thread about CaddyWiper. A variant of CaddyWiper was used again on 2022-04-08 14:58 against the Ukrainian energy provider previously mentioned.

At this point, we don’t know how attackers compromised the initial victim nor how they moved from the IT network to the Industrial Control System (ICS) network. Figure 1 shows an overview of the different malware used in this attack.

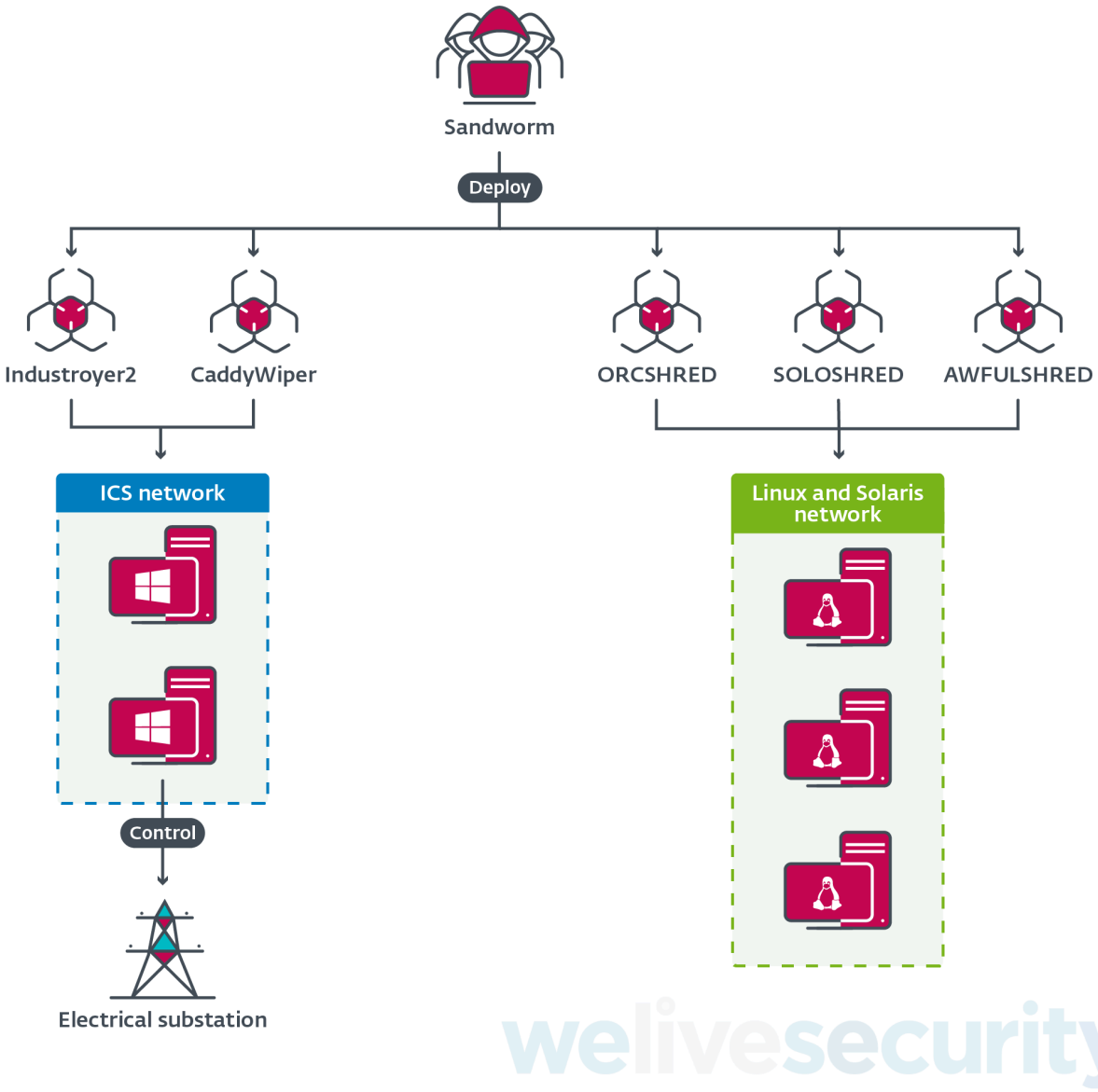

Figure 1. Overview of the malware deployed in the attack

Figure 2 summarizes the chain of events.

- **2022-02-24:** Beginning of the current Russian invasion in Ukraine
- **2022-03-14:** Deployment of CaddyWiper against a Ukrainian bank
- **2022-04-01:** Deployment of CaddyWiper against a Ukrainian governmental entity
- **2022-04-08 14:58 UTC:** Deployment of CaddyWiper on some Windows machines and of Linux and Solaris destructive malware at the energy provider
- **2022-04-08 15:02:22 UTC:** Sandworm operator creates the scheduled task to launch Industroyer2
- **2022-04-08 16:10 UTC:** Scheduled execution of Industroyer2 to cut power in an Ukrainian region
- **2022-04-08 16:20 UTC:** Scheduled execution of CaddyWiper on the same machine to erase Industroyer2 traces

Figure 2. Timeline of events

In 2017, ESET researchers revealed that a piece of malware that we named Industroyer was responsible for the power blackout that impacted Ukraine's capital Kiev in December 2016.

As detailed in our white paper Win32/Industroyer: A new threat for industrial control systems, it is capable of interacting with industrial control systems typically found in electric power systems. This includes IEC-101, IEC-104, IEC 61850 and OPC DA devices.

At that time, we said that "it seems very unlikely anyone could write and test such malware without access to the specialized equipment used in the specific, targeted industrial environment". This was confirmed in 2020 by the United States government when six officers of the Russian Military Unit 74455 of the Main Intelligence Directorate (GRU), were indicted for their role in multiple cyberattacks including Industroyer and NotPetya – see the indictment on justice.gov and our historical overview of Sandworm's operations.

The recently discovered malware is a new variant of Industroyer, hence the name Industroyer2.

Industroyer2

Industroyer2 was deployed as a single Windows executable named 108_100.exe and executed using a scheduled task on 2022-04-08 at 16:10:00 UTC. It was compiled on 2022-03-23, according to the PE timestamp, suggesting that attackers had planned their attack for more than two weeks.

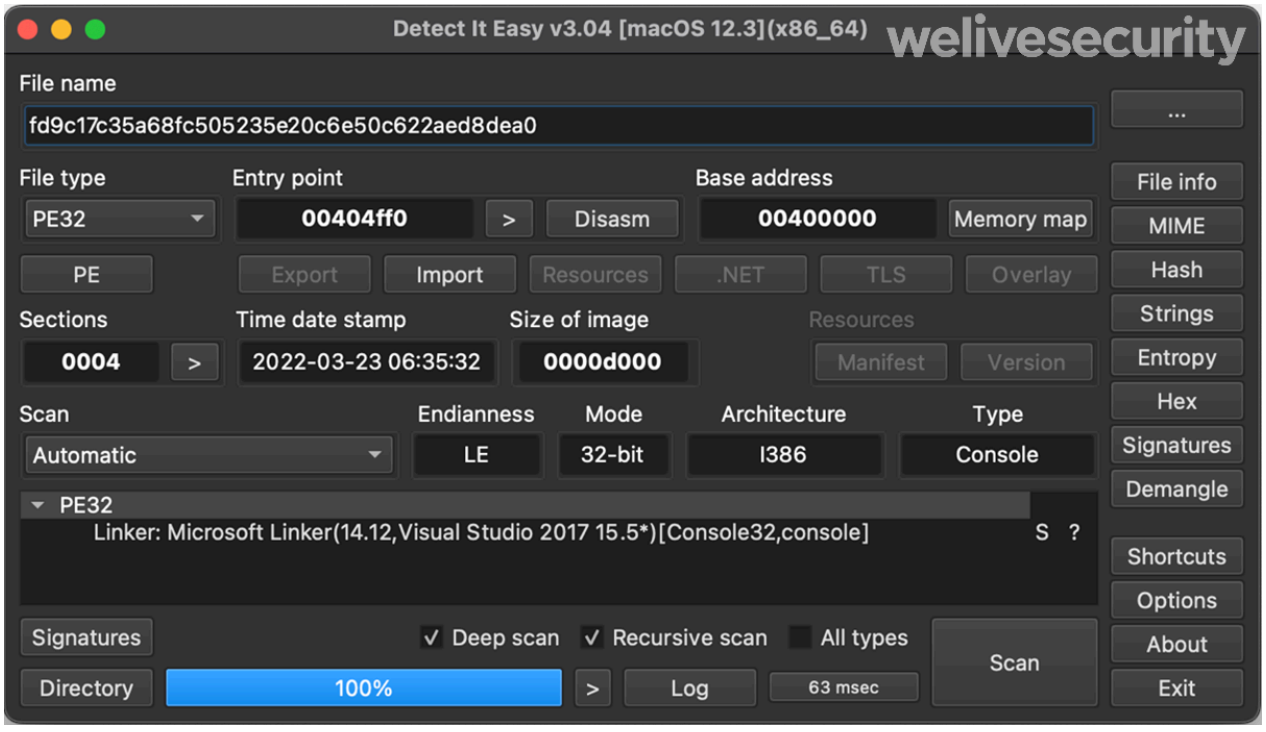

Figure 3. Timestamp and compiler information

Industroyer2 only implements the IEC-104 (aka IEC 60870-5-104) protocol to communicate with industrial equipment. This includes protection relays, used in electrical substations. This is a slight change from the 2016 Industroyer variant that is a fully-modular platform with payloads for multiple ICS protocols.

Industroyer2 shares number of code similarities with the payload 104.dll of Industroyer. We assess with high confidence that the new variant was built using the same source code.

Industroyer2 is highly configurable. It contains a detailed configuration hardcoded in its body, driving the malware actions. This is different from Industroyer, stores configuration in a separate .INI file. Thus, attackers need to recompile Industroyer2 for each new victim or environment. However, given that the Industroyer* malware family has only been deployed twice, with a five year gap between each version, this is probably not a limitation for Sandworm operators.

The new configuration format is stored as a string which is then supplied to the IEC-104 communication routine of the malware. Industroyer2 is able to communicate with multiple devices at once. Specifically, the analyzed sample contains eight different IP addresses of devices – see Figure 4.

The configuration contains values that are used during communication via IEC-104 protocol, such as ASDU (Application Service Data Unit) address, Information Object Addresses (IOA), timeouts, etc.

Before connecting to the targeted devices, the malware terminates a legitimate process that is used in standard daily operations. In addition to that, it renames this application by adding .MZ to the filename. It does so in order to prevent automatic re-start of this legitimate process.

The analysis is still ongoing in order to determine what are the exact actions taken for each device. We believe that this component is able to control specific ICS systems in order to cut power.

Industroyer2 can produce a log file or output its progress to the console window. However, instead of meaningful text messages as in the previous version, the malware writes various error codes – see Figure 5. We believe it is an obfuscation attempt by Sandworm developers to hamper analysis.

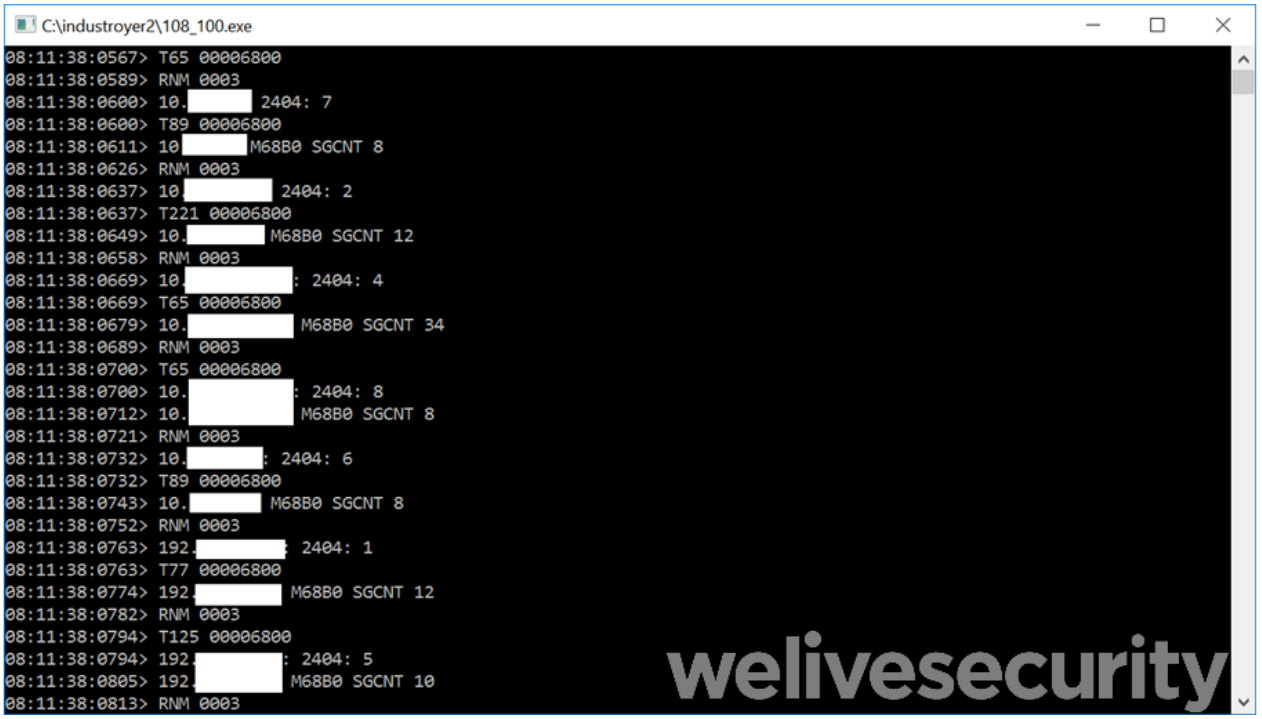

Figure 5. Output produced by Industroyer2 malware (IP addresses redacted by ESET)

CaddyWiper

In coordination with the deployment of Industroyer2 in the ICS network, the attackers deployed a new version of the CaddyWiper destructive malware. We believe it was intended to slow down the recovery process and prevent operators of the energy company from regaining control of the ICS consoles. It was also deployed on the machine where Industroyer2 was executed, likely to cover their tracks.

The first version of CaddyWiper was discovered by ESET researchers in Ukraine on 2022-03-14 when it was deployed in the network of a bank. It was deployed via Group Policy Object (GPO), indicating the attackers had prior control of the target's network beforehand. The wiper erases user data and partition information from attached drives, making the system inoperable and unrecoverable.

New CaddyWiper loading chain

In the network of the energy provider, attackers deployed a new version of CaddyWiper that uses a new loader, named ARGUEPATCH by CERT-UA. ARGUEPATCH is a patched version of a legitimate component of Hex-Rays IDA Pro software, specifically the remote IDA debugger server win32_remote.exe. IDA Pro is not intended to be used in an ICS environment, as its main purpose is for software reverse-engineering including malware analysis. We don’t know why attackers chose to trojanize this piece of software; it might be a troll towards defenders.

ARGUEPATCH was executed by a scheduled task that was intended to be launched once on 2022-04-08 14:58 UTC on one machine and at 16:20 UTC on the machine where Industroyer2 was deployed.

The patched binary loads encrypted shellcode from a file and decrypts it with a key, both are provided on command line. A single-byte XOR key is derived from the input key and used to decrypt the shellcode.

The decrypted shellcode is a slightly modified version of CaddyWiper. A comparison of their main routines is provided in Figure 6 and Figure 7. Note that they do not wipe the domain controller, and they wipe C:\Users\ and disks from D:\ to [:\. The wiping routine is also almost identical: it fills all files with 0.

```
strcpy(s_netapi32, "netapi32.dll");
(LoadLibraryA)(s_netapi32);
Buffer = 0;
result = DsRoleGetPrimaryDomainInformation(0, DsRolePrimaryDomainInfoBasic, &Buffer);
if ( *Buffer != DsRole_RolePrimaryDomainController )
{
  (LoadLibraryA)(s_advapi32);
  strcpy(dir, "C:\\Users");
  Wipe(dir);
  strcpy(drive, "D:\\");
  for ( i = 0; i < 24; ++i )
  {
    Wipe(drive);
    ++drive[0];
  }
```

Figure 6. Main routine of the first sample of CaddyWiper.

```
strcpy(s_netapi32, "netapi32.dll");
v5 = (LoadLibraryA)(s_netapi32);
strcpy(s_DsRoleGetPrimaryDomainInformation, "DsRoleGetPrimaryDomainInformation");
DsRoleGetPrimaryDomainInformation = GetProcAddress(v5, s_DsRoleGetPrimaryDomainInformation);
Buffer = 0;
DsRoleGetPrimaryDomainInformation(0, DsRolePrimaryDomainInfoBasic, &Buffer);
result = Buffer;
if ( *Buffer != DsRole_RolePrimaryDomainController )
{
  strcpy(dir, "C:\\Users");
  Wipe(GetProcAddress, dir);
  drive = '\\:D';
  max = 24;
  do
  {
    Wipe(GetProcAddress, &drive);
    LOBYTE(drive) = drive + 1;
    --max;
  }
  while ( max );
```

Figure 7. Main routine of the CaddyWiper sample deployed at the energy provider

Finally, CaddyWiper calls DeviceIoControl with IOCTL_DISK_SET_DRIVE_LAYOUT_EX and a zeroed InputBuffer for all disks from \\PHYSICALDRIVE9 to \\PHYSICALDRIVE0. This erases extended information of the drive's partitions: the Master boot record (MBR) or the GUID Partition Table (GPT). This renders the machine unbootable.

Active Directory enumeration

Alongside CaddyWiper, a PowerShell script was found both in the energy provider network and in the bank that was compromised earlier.

This script enumerates Group Policies Objects (GPO) using the Active Directory Service Interface (ADSI). The script, shown in Figure 8, is almost identical to a snippet provided in a Medium blogpost.

We believe that attackers deployed CaddyWiper via a GPO and used the script to check the existence of this GPO.

```
(([adsisearcher]'').SearchRoot).Path | %{
    if(([ADSI]"$_").gPlink){
        $a = ((([ADSI]"$_").gPlink) -replace "[[;]" -split "]");
        for ($i=0; $i -lt $a.length;$i++){
            if ($a[$i]){
                Write-Host ([ADSI]($a[$i]).Substring(0, $a[$i].Length - 1)).Path;
                Write-Host ([ADSI]($a[$i]).Substring(0, $a[$i].Length - 1)).DisplayName;
                Write-Host ([ADSI]($a[$i]).Substring(0, $a[$i].Length - 1)).flags;
            }
        }
    }
}
```


Figure 8. PowerShell script to enumerate GPO (beautified)

Linux and Solaris destructive malware (ORCSHRED, SOLOSHRED, AWFULSHRED)

Additional destructive malware for systems running Linux and Solaris was also found on the network of the targeted energy company. There are two main components to this attack: a worm and a wiper. The latter was found in two variants, one for each of the targeted operating system. All malware was implemented in Bash.

The worm

The first component launched by the attacker was a worm, having its file named sc.sh. This Bash script starts by adding a scheduled task (cron job) to launch the wiper component at 2:58pm UTC (assuming the system is in the local time zone, UTC+3), unless it was launched with the “owner” argument. This is likely a way to avoid the initial system used to launch the worm auto-destructing.

```
if [[ $is_owner -eq 0 ]]; then
        echo "Start most security mode!"
        crontab -l > /var/log/tasks

        check_solaris=$(find /etc -name os-release > /var/log/res)
        check_solaris=$(cat /var/log/res)

        if [ -s /var/log/res ]; then
                check_solaris=$(cat /etc/os-release | grep ID=solaris; echo $? > /var/log/res)
                check_solaris=$(cat /var/log/res)

                if [[ $check_solaris -eq 0 ]]; then
                        echo "58 17 * * * /bin/bash /var/log/wsol.sh & disown"  >> /var/log/tasks
                else
                        echo "58 17 * * * /bin/bash /var/log/wobf.sh & disown"  >> /var/log/tasks
                fi
        else
                echo "58 17 * * * /bin/bash /var/log/wobf.sh & disown"  >> /var/log/tasks
        fi

        crontab /var/log/tasks
        rm -f /var/log/tasks
        rm -f /var/log/res
fi
```


Figure 9. Setting up the cron job to launch the wiper at 5:58pm. The correct wiper is picked depending on the installed operating system.

The script then iterates over the networks accessible by the system by looking at the result of ip route or ifconfig -a. It always assumes a class C network (/24) is reachable for each IP address it collects. It will try to connect to all hosts in those networks using SSH to TCP port 22, 2468, 24687 and 522. Once it finds a reachable SSH server, it tries credentials from a list provided with the malicious script. We believe the attacker had credentials prior to the attack to enable the spread of the wiper.

If the system is not already compromised, malware is copied to the new target, and the worm is launched. The worm is *not* launched with the owner argument, so the wiper is scheduled to launch at 2:58pm UTC and destroy all data. If those systems were set to the local time zone, the destruction must've started at the same time as the system compromised with CaddyWiper.

The Linux wiper

The Linux variant of the wiper is lightly obfuscated: variables and function names have been replaced with meaningless 8-letter words. Most literal values were also replaced with variables at the beginning of the file.

Figure 10. Except from the obfuscated script (whitespace optimized).

Figure 11. Deobfuscation of the above obtained by renaming functions and variables and using literals

Ultimately, the Linux wiper destroys the whole content of the disks attached to the system by using shred if available or simply dd (with if=/dev/random) otherwise. If multiple disks are attached, data removal is done in parallel to speed up the process.

Depending on the size, it may take hours for the full disk to be completely erased. To render the system inoperable faster, it first tries to stop and disable HTTP and SSH services. Both services are disabled by using systemctl disable. To ensure service isn't reenabled, the systemd unit file responsible for loading the service is deleted from the disk.

Files from /boot, /home and /var/log are also removed before destroying the full drives. This makes the system inoperable faster, deletes user data and perhaps removes incriminating logs.

The malicious script's last action is to forcibly initiate a reboot using SysRq. Since all drives are filled with random, no operating system will boot.

The Solaris wiper

Unlike the Linux wiper, the Solaris variant is not obfuscated.

Like the Linux variant, the malicious script iterates over all services to stop and disable them if they contain the keyword ssh, http, apache and additionally ora_ or oracle. Those services are very likely used by applications used to control ICS systems. Wiping them would prevent the energy company's operators from retaking control of the substations and roll back Industroyer2 actions.

It uses either systemctl or svcadm depending on what's available. The latter is most likely since Solaris is not running systemd.

File destruction begins by deleting databases. It removes, using shred then rm, all files and directories contained in environment variables starting with ORA. Note that shred makes sure data recovery (without a backup) isn't possible.

Like the Linux variant, files in /boot, /home and /var/log are deleted with priority.

Then the script iterates over disks connected to the system, found in /dev/dsk/. It ignores slices (partitions) and work only on full disks. For each of them, the malicious script overwrites the full content using shred. To minimize the time required to perform the wipe, all disks are erased in parallel.

Lastly, the script self-destructs.

Conclusion

Ukraine is once again at the center of cyberattacks targeting their critical infrastructure. This new Industroyer campaign follows multiple waves of wipers that have been targeting various sectors in Ukraine. ESET researchers will continue to monitor the threat landscape in order to better protect organizations from these types of destructive attacks.

And a big thanks to @_CERT_UA who worked with us on this case and provided samples. You can read their advisory on this campaign here.

For any inquiries about our research published on WeLiveSecurity, please contact us at threatintel@eset.com.
ESET Research now also offers private APT intelligence reports and data feeds. For any inquiries about this service, visit the ESET Threat Intelligence page.

Indicators of Compromise

SHA-1	Filename	ESET detection name	Description
FD9C17C35A68FC505235E20C6E50C622AED8DEA0	108_100.exe	Win32/Industroyer.B	Industroyer2
6FA04992C0624C7AA3CA80DA6A30E6DE91226A16	zrada.exe	Win32/Agent.AECG	ArguePatch
9CE1491CE69809F92AE1FE8D4C0783BD1D11FBE7	pa.pay	N/A	TailJump (Encrypted CaddyWiper)
0090CB4DE31D2D3BCA55FD4A36859921B5FC5DAE	link.ps1	PowerShell/HackTool.Agent.AH	Script which enumerates GPO
D27D0B9BB57B2BAB881E0EFB97C740B7E81405DF	sc.sh	Linux/Agent.PC trojan	OrcShred (Linux worm)
3CDBC19BC4F12D8D00B81380F7A2504D08074C15	wobf.sh	Linux/KillFiles.C trojan	AwfulShred (Linux wiper)
8FC7646FA14667D07E3110FE754F61A78CFDE6BC	wsol.sh	Linux/KillFiles.B trojan	SoloShred (Solaris wiper)

Source: https://www.welivesecurity.com/2022/04/12/industroyer2-industroyer-reloaded/