CCleanup: A Vast Number of Machines at Risk

By Edmund Brumaghin

Published: 2017-09-18 · Archived: 2026-04-02 10:39:36 UTC

Update 9/18: CCleaner Cloud version 1.07.3191 is also reported to be affected
Update 9/19: This issue was discovered and reported by both Morphisec and Cisco in separate in-field cases and reported separately to Avast.

Update 9/19: There has been some confusion on how the DGA domains resolve.

> The fallback command and control scheme in use by the CCBkdr involves:
> 1. Generating a Monthly Domain name (all of which are controlled by Talos for 2017)
> 2. Request the A records for the domain.
> 3. 16 bits of the true destination IP are encoded in the first A record, 16 bits are encoded in the second A record
> 4. The true destination IP is then computed and connected to.
> To control the connections Talos has to create two IPs such that they can be fed into the application to resolve to the sinkhole IP.
> 32 bits of random data were generated. 16 bits of that were combined with 16 bits of the destination address to create the first A record. The remaining 16 random bits were combined with the remaining bits of the destination address to create the second A record.
> The resulting two A record IP addresses were then assigned to the DNS configuration.
> There was no analysis performed on the selected addresses beyond that they could be combined to create the destination.

Update 9/20: Continued research on C2 and payloads can be found here: /ccleaner-c2-concern

Introduction

Supply chain attacks are a very effective way to distribute malicious software into target organizations. This is because with supply chain attacks, the attackers are relying on the trust relationship between a manufacturer or supplier and a customer. This trust relationship is then abused to attack organizations and individuals and may be performed for a number of different reasons. The Nyetya worm that was released into the wild earlier in 2017 showed just how potent these types of attacks can be. Frequently, as with Nyetya, the initial infection vector can remain elusive for quite some time. Luckily with tools like AMP the additional visibility can usually help direct attention to the initial vector.

Talos recently observed a case where the download servers used by software vendor to distribute a legitimate software package were leveraged to deliver malware to unsuspecting victims. For a period of time, the legitimate signed version of CCleaner 5.33 being distributed by Avast also contained a multi-stage malware payload that rode on top of the installation of CCleaner. CCleaner boasted over 2 billion total downloads by November of 2016

with a growth rate of 5 million additional users per week. Given the potential damage that could be caused by a network of infected computers even a tiny fraction of this size we decided to move quickly. On September 13, 2017 Cisco Talos immediately notified Avast of our findings so that they could initiate appropriate response activities. The following sections will discuss the specific details regarding this attack.

Technical Details

CCleaner is an application that allows users to perform routine maintenance on their systems. It includes functionality such as cleaning of temporary files, analyzing the system to determine ways in which performance can be optimized and provides a more streamlined way to manage installed applications.

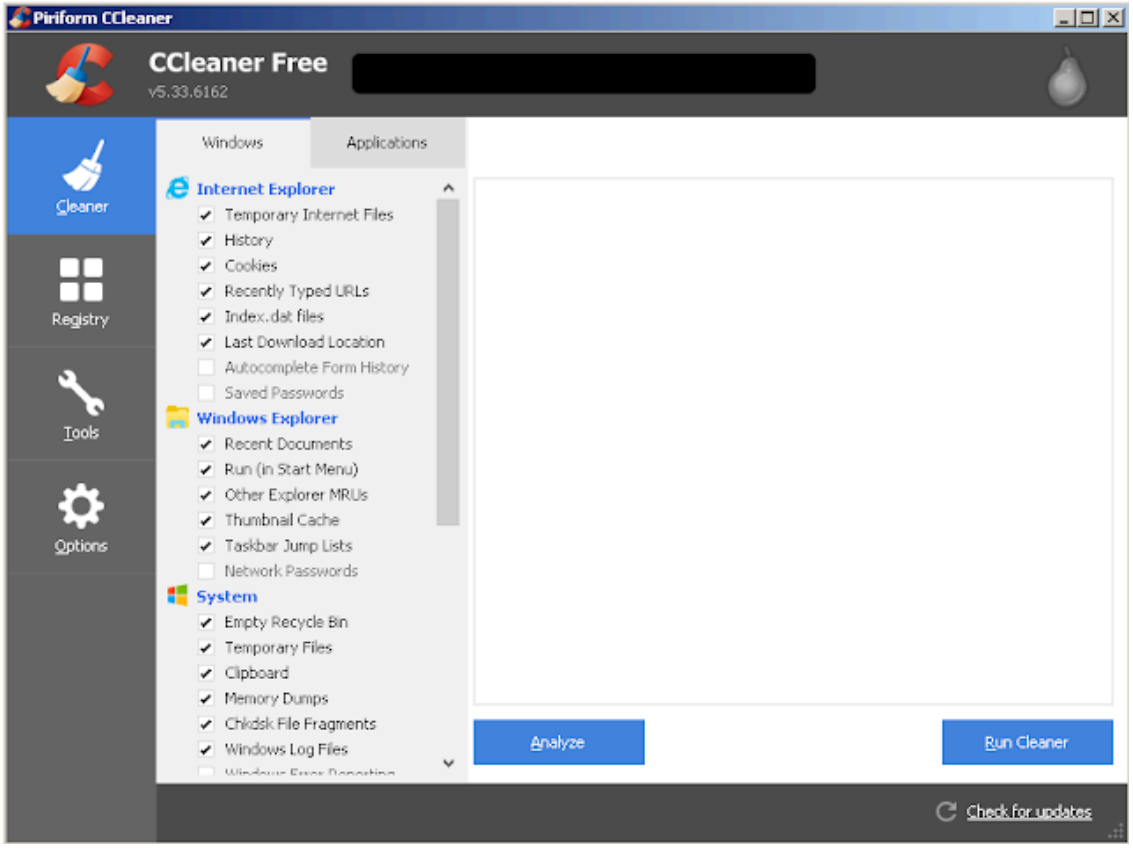

Figure 1: Screenshot of CCleaner 5.33

On September 13, 2017 while conducting customer beta testing of our new exploit detection technology, Cisco Talos identified a specific executable which was triggering our advanced malware protection systems. Upon closer inspection, the executable in question was the installer for CCleaner v5.33, which was being delivered to endpoints by the legitimate CCleaner download servers. Talos began initial analysis to determine what was causing this technology to flag CCleaner. We identified that even though the downloaded installation executable was signed using a valid digital signature issued to Piriform, CCleaner was not the only application that came with the download. During the installation of CCleaner 5.33, the 32-bit CCleaner binary that was included also contained a malicious payload that featured a Domain Generation Algorithm (DGA) as well as hardcoded Command and Control (C2) functionality. We confirmed that this malicious version of CCleaner was being hosted directly on CCleaner's download server as recently as September 11, 2017.

In reviewing the Version History page on the CCleaner download site, it appears that the affected version (5.33) was released on August 15, 2017. On September 12, 2017 version 5.34 was released. The version containing the malicious payload (5.33) was being distributed between these dates. This version was signed using a valid certificate that was issued to Piriform Ltd by Symantec and is valid through 10/10/2018. Piriform was the company that Avast recently acquired and was the original company who developed the CCleaner software application.

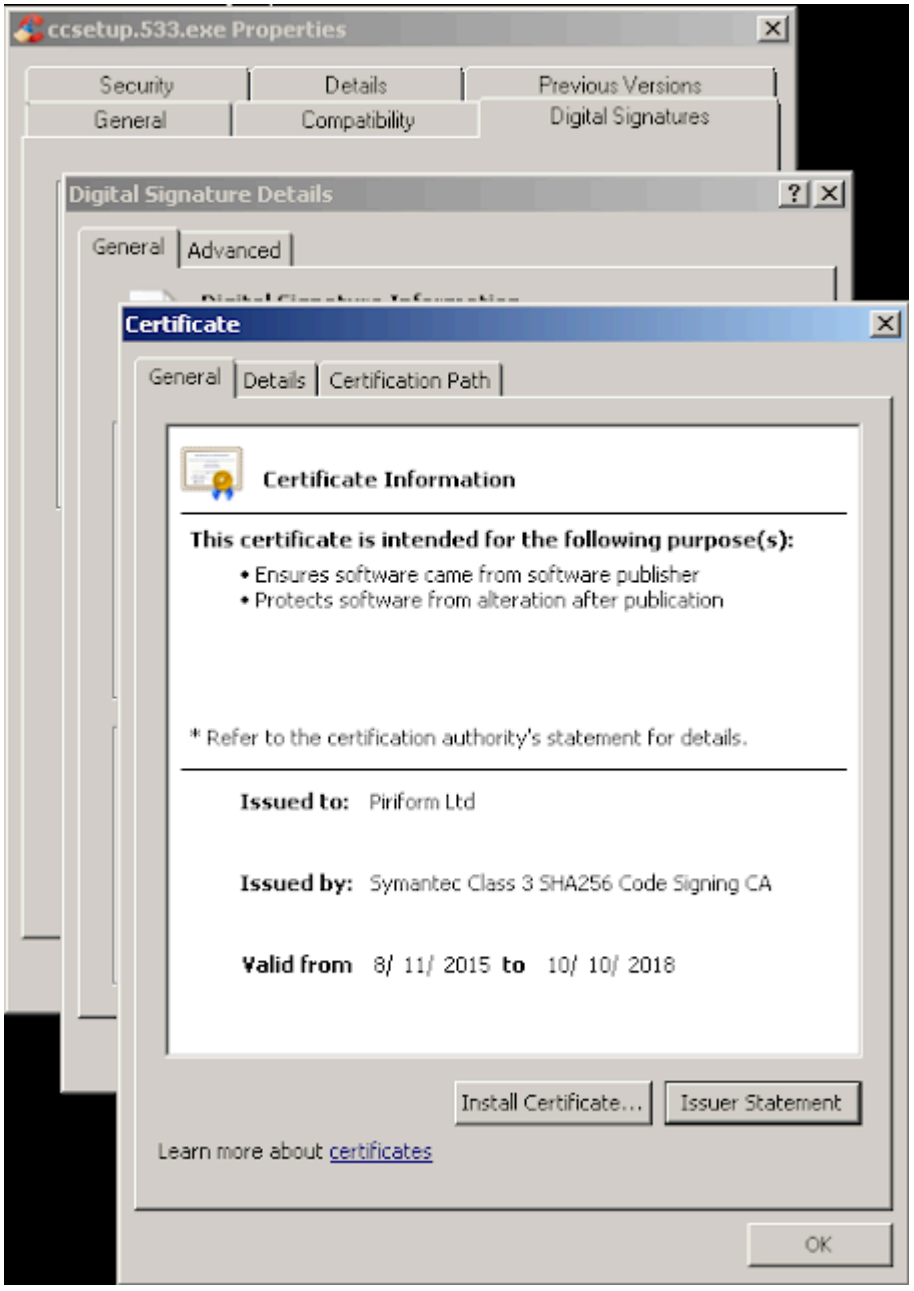

Figure 2: Digital Signature of CCleaner 5.33

A second sample associated with this threat was discovered. This second sample was also signed using a valid digital certificate, however the signing timestamp was approximately 15 minutes after the initial sample was signed.

The presence of a valid digital signature on the malicious CCleaner binary may be indicative of a larger issue that resulted in portions of the development or signing process being compromised. Ideally this certificate should be revoked and untrusted moving forward. When generating a new cert care must be taken to ensure attackers have no foothold within the environment with which to compromise the new certificate. Only the incident response process can provide details regarding the scope of this issue and how to best address it.

Interestingly the following compilation artifact was found within the CCleaner binary that Talos analyzed:

S:\workspace\ccleaner\branches\v5.33\bin\CCleaner\Release\CCleaner.pdb

Given the presence of this compilation artifact as well as the fact that the binary was digitally signed using a valid certificate issued to the software developer, it is likely that an external attacker compromised a portion of their development or build environment and leveraged that access to insert malware into the CCleaner build that was released and hosted by the organization. It is also possible that an insider with access to either the development or build environments within the organization intentionally included the malicious code or could have had an account (or similar) compromised which allowed an attacker to include the code.

It is also important to note that while previous versions of the CCleaner installer are currently still available on the download server, the version containing the malicious payloads has been removed and is no longer available.

Malware Installation and Operation

Within the 32-bit CCleaner v5.33 binary included with the legitimate CCleaner v5.33 installer, '__scrt_get_dyn_tls_init_callback' was modified to call to the code at CC_InfectionBase(0x0040102C). This was done to redirect code execution flow within the CCleaner binary to the malicious code prior to continuing with the normal CCleaner operations. The code that is called is responsible for decrypting data which contains the two stages of the malicious payload, a PIC (Position Independent Code) PE loader as well as a DLL file that effectively functions as the malware payload. The malware author had tried to reduce the detection of the malicious DLL by ensuring the IMAGE_DOS_HEADER was zeroed out, suggesting this attacker was trying to remain under the radar to normal detection techniques.

The binary then creates an executable heap using HeapCreate(HEAP_CREATE_ENABLE_EXECUTE,0,0). Space is then allocated to this new heap which is where the contents of the decrypted data containing the malware is copied. As the data is copied to the heap, the source data is erased. The PE loader is then called and begins its operation. Once the infection process has been initiated, the binary erases the memory regions that previously contained the PE loader and the DLL file, frees the previously allocated memory, destroys the heap and continues on with normal CCleaner operations.

The PE loader utilizes position independent coding practices in order to locate the DLL file within memory. It then maps the DLL into executable memory, calls the DLLEntryPoint to begin execution of the DLL being loaded and the CCleaner binary continues as normal. Once this occurs the malware begins its full execution, following the process outlined in the following sections.

CBkrdr.dll

The DLL file (CBkdr.dll) was modified in an attempt to evade detection and had the IMAGE_DOS_HEADER zeroed out. The DLLEntryPoint creates an execution thread so that control can be returned to the loader. This thread is responsible for calling CCBkdr_GetShellcodeFromC2AndCall. It also sets up a Return Oriented Programming (ROP) chain that is used to deallocate the memory associated with the DLL and exit the thread.

CCBkrdr_GetShellcodeFromC2AndCall

This function is responsible for much of the malicious operations that Talos observed while analyzing this malware. First, it records the current system time on the infected system. It then delays for 601 seconds before continuing operations, likely an attempt to evade automated analysis systems that are configured to execute samples for a predefined period of time or determine whether the malware is being executed in a debugger. In order to implement this delay functionality, the malware calls a function which attempts to ping 224.0.0.0 using a delay_in_seconds timeout set to 601 seconds. It then checks to determine the current system time to see if 600 seconds has elapsed. If that condition is not met, the malware terminates execution while the CCleaner binary continues normal operations. In situations where the malware is unable to execute IcmpCreateFile, it then falls back to using Sleep() to implement the same delay functionality. The malware also compares the current system time to the value stored in the following registry location:

HKLM\SOFTWARE\Piriform\Agomo:TCID

If the value stored in TCID is in the future, the malware will also terminate execution.

```
00EC2543 2BC FF D6              call    esi ; time
00EC2545 2BC 8B F8              mov     edi, eax
00EC2547 2BC C7 04 24 59 02 00+mov     [esp+2B8h+delay], 601 ; delay
00EC254E 2BC E8 84 FF FF FF     call    DelayForSeconds
00EC2553 2BC 53                 push    ebx             ; Time
00EC2554 2C0 FF D6              call    esi ; time
00EC2556 2C0 2B C7              sub     eax, edi
00EC2558 2C0 59                 pop     ecx
00EC2559 2BC 3D 58 02 00 00     cmp     eax, 600
00EC255E 2BC 59                 pop     ecx
00EC255F 2B8 72 1B              jb      short BailOut
```

Figure 3: Delay Routine

The malware then checks to determine the privileges assigned to the user running on the system. If the current user running the malicious process is not an administrator the malware will terminate execution.

```
00EC2572 2B8 FF 15 90 10 EC 00 call    IsUserAnAdmin
00EC2578 2B8 85 C0             test    eax, eax
00EC257A 2B8 75 07             jnz     short RunIfAdmin

00EC2583
00EC2583                               RunIfAdmin:
```

Figure 4: Privilege Check

If the user executing the malware does have administrative privileges on the infected system, SeDebugPrivilege is

enabled for the process. The malware then reads the value of 'InstallID' which is stored in the following registry location:

HKLM\SOFTWARE\Piriform\Agomo:MUID

If this value does not exist, the malware creates it using '((rand()*rand() ^ GetTickCount())'.

Once the aforementioned activities have been performed, the malware then begins profiling the system and gathering system information which is later transmitted to the C2 server. System information is stored in the following data structure:

```
00000000 CCBkdr_System_Information struc
00000000 InstallID       dd
00000004 NtMajorVersion  db
00000005 NtMinorVersion  db
00000006 IsWow64Process  db
00000007 unk_zero        db
00000008 ComputerName    db 64
00000048 ComputerNameDnsDomain db 64
00000088 IpAddresses     dd 6
000000A0 Records         CCBkdr_Record 254 ; Installed processes according to
000000A0                                   ;
SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall
000000A0                                   ; Running processes
```

Figure 5: CCBkdr_System_Information Data Structure

Once the system information has been collected, it is encrypted and then encoded using modified Base64. The malware then establishes a Command and Control (C2) channel as described in the following section.

Command and Control (C2)

While analyzing this malware, Talos identified what appears to be a software bug present in the malicious code related to the C2 function. The sample that Talos analyzed reads a DGA computed IP address located in the following registry location, but currently does nothing with it:

HKLM\SOFTWARE\Piriform\Agomo:NID

It is unknown what the purpose of this IP address is at this time, as the malware does not appear to make use of it during subsequent operations. In any event, once the previously mentioned system information has been collected and prepared for transmission to the C2 server, the malware will then attempt to transmit it using an HTTPS POST request to 216[.]126[.]225[.]148. The HTTPS communications leverage a hardcoded HTTP Host header that is set to speccy[.]piriform[.]com, a legitimate platform which is also created by Piriform for hardware monitoring. This could make dynamic analysis more difficult as the domain would appear to be legitimate and perhaps even expected depending on the victim infrastructure. The requests also leverage HTTPS but ignore all security errors as the server currently returns a self-signed SSL certificate that was issued to the subdomain defined in the Host header field. In cases where no response is received from the C2 server, the malware then fails back to a Domain Generation Algorithm (DGA) as described in the section 'Domain Generation Algorithm' of this post.

Once a C2 server has been identified for use by the malware, it then sends the encoded data containing system profile information and stores the C2 IP address in the following registry location:

HKLM\SOFTWARE\Piriform\Agomo:NID

The malware then stores the value of the current system time plus two days into the following registry location:

HKLM\SOFTWARE\Piriform\Agomo:TCID

Data received from the C2 server is then validated to confirm that the received data is in the correct format for a CCBkdr_ShellCode_Payload structure. An example is shown below:

```
00000000 CCBkdr_ShellCode_Payload struc
00000000 Size            dd
00000004 EncryptedInstallID dd
00000008 EncodedEncryptedDataBuffer db *
```

Figure 6: CCBkdr_ShellCode_Payload Data Structure

The malware then confirms that the value of EncryptedInstallID matches the value that was previously transmitted to the C2 server. It then allocates memory for the final shellcode payload. The payload is then decoded using modified Base64 and stored into the newly allocated memory region. It is then decrypted and called with the addresses of LoadLibraryA and GetProcAddress as parameters. Once the payload has been executed, the memory is deallocated and the following registry value is set to the current system time plus seven days:

HKLM\SOFTWARE\Piriform\Agomo:TCID

The received buffer is then zeroed out and deallocated. The CCBkdr_ShellCode_Payload structure is also deallocated and the malware then continues with normal CCleaner operations. A diagram describing the high level operation of this malware is below:

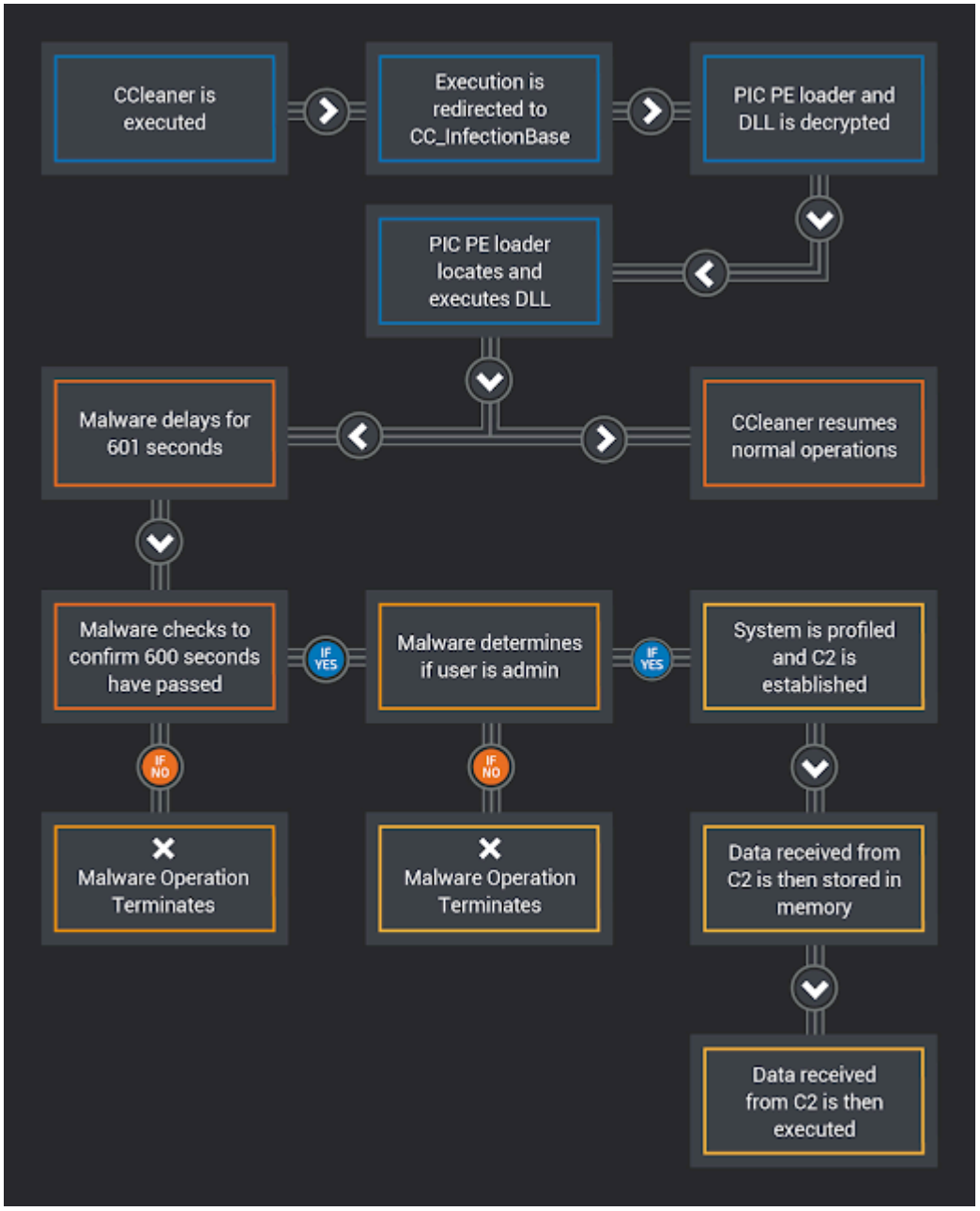

Figure 7: Malware Operation Process Flow

Domain Generation Algorithm

In situations where the primary C2 server does not return a response to the HTTP POST request described in the previous section, the malware fails back to using a DGA algorithm. The algorithm used by this malware is time-based and can be calculated using the values of year and month. A list of DGA domains is below:

Year-Month	DGA Domain
2017-02	ab6d54340c1a[.]com
2017-03	aba9a949bc1d[.]com
2017-04	ab2da3d400c20[.]com
2017-05	ab3520430c23[.]com
2017-06	ab1c403220c27[.]com
2017-07	ab1abad1d0c2a[.]com
2017-08	ab8cee60c2d[.]com
2017-09	ab1145b758c30[.]com
2017-10	ab890e964c34[.]com
2017-11	ab3d685a0c37[.]com
2017-12	ab70a139cc3a[.]com

Figure 8: 12 Month DGA Genearation The malware will initiate DNS lookups for each domain generated by the DGA algorithm. If the DNS lookup does not result in the return of an IP address, this process will continue. The malware will perform a DNS query of the active DGA domain and expects that two IP addresses will be returned from the name server managing the DGA domain's namespace. The malware will then compute a secondary C2 server by performing a series of bit operations on the returned IP address values and combine them to determine the actual fallback C2 server address to use for subsequent C2 operations. A diagram showing this process is below:

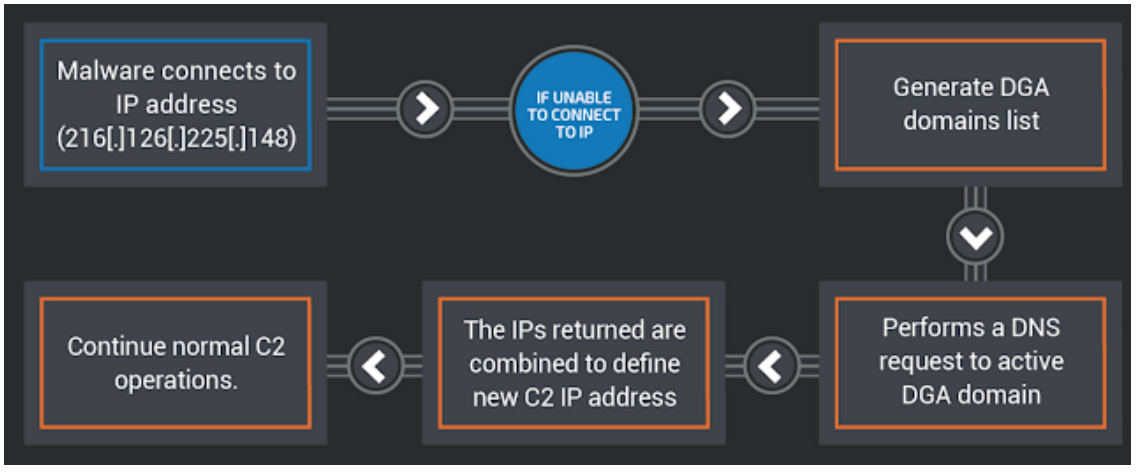

Figure 9: C2 Process Diagram

Cisco Talos observed during analysis that the DGA domains had not been registered, so we registered and sinkholed them to prevent attackers from being able to use them for malicious purposes.

Potential Impact

The impact of this attack could be severe given the extremely high number of systems possibly affected. CCleaner claims to have over 2 billion downloads worldwide as of November 2016 and is reportedly adding new users at a

rate of 5 million a week.

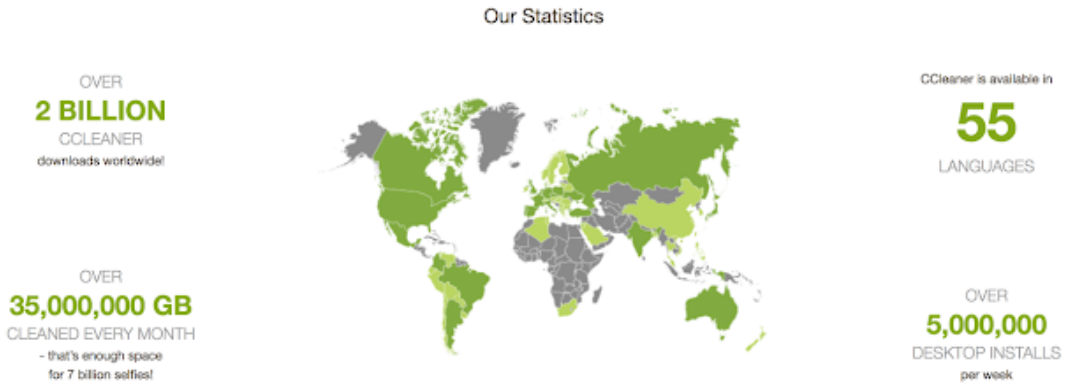

Figure 10: CCleaner Consumer Demographics

If even a small fraction of those systems were compromised an attacker could use them for any number of malicious purposes. Affected systems need to be restored to a state before August 15, 2017 or reinstalled. Users should also update to the latest available version of CCleaner to avoid infection. At the time of this writing that is version 5.34. It is important to note that according to the CCleaner download page, the free version of CCleaner does not provide automated updates, so this might be a manual process for affected users.

In analyzing DNS-based telemetry data related to this attack, Talos identified a significant number of systems making DNS requests attempting to resolve the domains associated with the aforementioned DGA domains. As these domains have never been registered, it is reasonable to conclude that the only conditions in which systems would be attempting to resolve the IP addresses associated with them is if they had been impacted by this malware. While most of the domains associated with this DGA have little to no request traffic associated with them, the domains related to the months of August and September (which correlates with when this threat was active in the wild) show significantly more activity.

Looking at the DNS related activity observed by Cisco Umbrella for the month of July 2017 (prior to CCleaner 5.33 being released) we observed very little in the way of DNS requests to resolve the IP address for DGA domain associated with this malware:

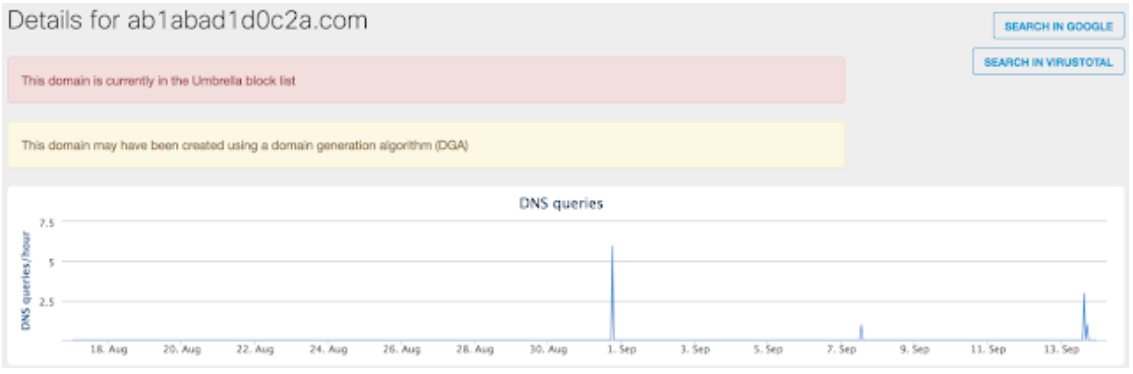

Figure 11: DNS Activity for July 2017 DGA Domain

As mentioned earlier in this post, the version of CCleaner that included this malware was released on August 15,

2017. The following graph shows a significant increase in the amount of DNS activity associated with the DGA domain used in August 2017:

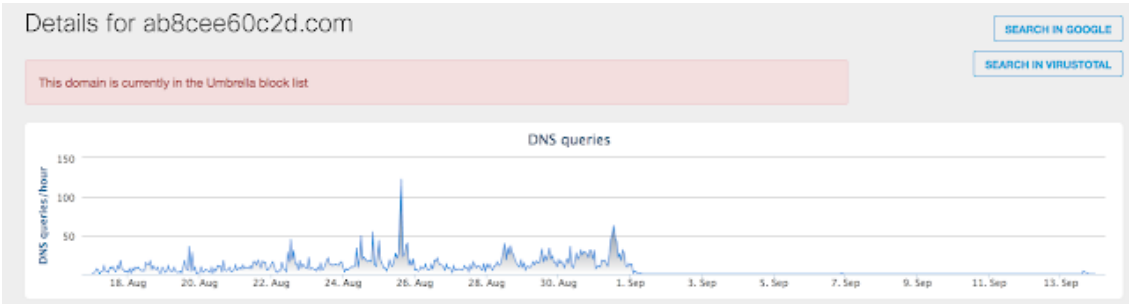

Figure 12: DNS Activity for August 2017 DGA Domain

Likewise, the DGA domain associated with September 2017 reflects the following activity with regards to attempts to resolve the IP associated with it:

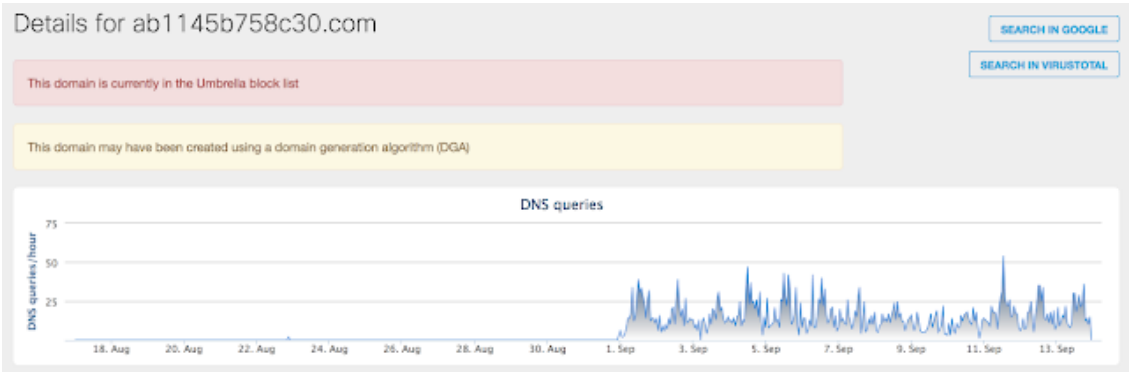

Figure 13: DNS Activity for September 2017 DGA Domain

Note that in on September 1, 2017 it appears that the DNS activity shifted from the DGA domain previously used in August, to the one used in September, which matches the time-based DGA algorithm described in the "Domain Generation Algorithm" section of this blog post. After reaching out to Avast we noted that the server was taken down and became unavailable to already infected systems. As a result, we saw a significant increase in the amount of requests that were being directed at the failback DGA domains used by the malware.

Figure 14: Traffic Spike Following Server Takedown

It is also worth noting that at the time of this post, antivirus detection for this threat remains very low (The detections are at 1/64 at the time of this writing).

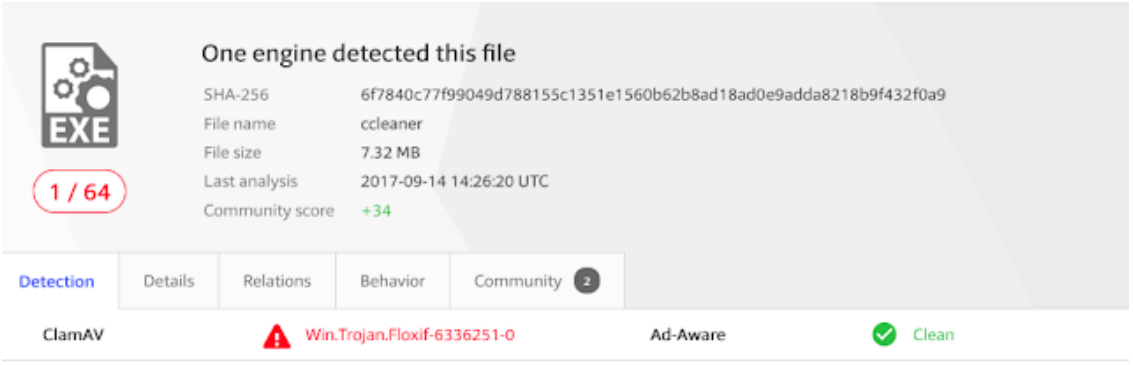

Figure 14: VirusTotal Detections for CCleaner Binary

As part of our response to this threat, Cisco Talos has released comprehensive coverage to protect customers. Details related to this coverage can be found in the "Coverage" section of this post.

Conclusion

This is a prime example of the extent that attackers are willing to go through in their attempt to distribute malware to organizations and individuals around the world. By exploiting the trust relationship between software vendors and the users of their software, attackers can benefit from users' inherent trust in the files and web servers used to distribute updates. In many organizations data received from commonly software vendors rarely receives the same level of scrutiny as that which is applied to what is perceived as untrusted sources. Attackers have shown that they are willing to leverage this trust to distribute malware while remaining undetected. Cisco Talos continues to monitor all aspects of the threat landscape to quickly identify new and innovative techniques used by attackers to target organizations and individuals around the world.

Coverage

The following ClamAV signatures have been released to detect this threat: 6336251, 6336252.

Additional ways our customers can detect and block this threat are listed below.

PRODUCT	PROTECTION
AMP	✔
CloudLock	N/A
CWS	✔
Email Security	N/A
Network Security	N/A
Threat Grid	✔
Umbrella	✔
WSA	✔

Advanced Malware Protection (AMP) is ideally suited to prevent the execution of the malware used by these threat actors.

CWS orWSA web scanning prevents access to malicious websites and detects malware used in these attacks.

AMP Threat Grid helps identify malicious binaries and build protection into all Cisco Security products.

Umbrella, our secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs, and URLs, whether users are on or off the corporate network.

Indicators of Compromise (IOCs)

File Hashes

6f7840c77f99049d788155c1351e1560b62b8ad18ad0e9adda8218b9f432f0a9
1a4a5123d7b2c534cb3e3168f7032cf9ebf38b9a2a97226d0fdb7933cf6030ff
36b36ee9515e0a60629d2c722b006b33e543dce1c8c2611053e0651a0bfdb2e9

DGA Domains

ab6d54340c1a[.]com
aba9a949bc1d[.]com
ab2da3d400c20[.]com
ab3520430c23[.]com
ab1c403220c27[.]com
ab1abad1d0c2a[.]com
ab8cee60c2d[.]com
ab1145b758c30[.]com
ab890e964c34[.]com
ab3d685a0c37[.]com
ab70a139cc3a[.]com

IP Addresses

216[.]126[.]225[.]148
