WS 1.2 - File Disclosure and Server-Side Request Forgery

Polonium - Pwnzer0tt1

gh repo fork WS_1.2 - File Disclosure and Server-Side Request Forgery

Prerequisites

- WS_1.1 - HTTP Protocol And Web Security Overview
- A server to exploit
- Questionable mental sanity

Outline

- File Disclosure
- Server-Side Request Forgery

File Disclosure

File Disclosure

A **file disclosure** is the result of an attack that **disclose/leak** important **files** from a server.

There are multiple vulnerabilities that can lead to a file disclosure, some are: human errors (skill issues), improper server configurations and RCE (Remote Code Execution).

Targets of these type of attacks are servers that store files containing critical information such as: users sensitive data, configuration files, source code of an application.

Everything that works with files can potentially lead to a file disclosure vulnerability.

File Disclosure - Code execution

How to code execute:

1) Find a vulnerable input field
2) Enter a command in the input field
3) Send the command
4) Get the file

Try it: https://ctf.cyberchallenge.it/challenges#challenge-1

File Disclosure - Path traversal attack

Path traversal is a vulnerability that leads to a file disclosure. It happens when a user is able to control the arguments of a function that reads files (e.g. *open()*).

If there are no security checks or sanitizations, an attacker could inject paths that are not meant to be read.

Try it: https://ctf.cyberchallenge.it/challenges#challenge-3

File Disclosure - Other methods

Sometimes the attacker doesn’t need to do anything, because people commit errors.

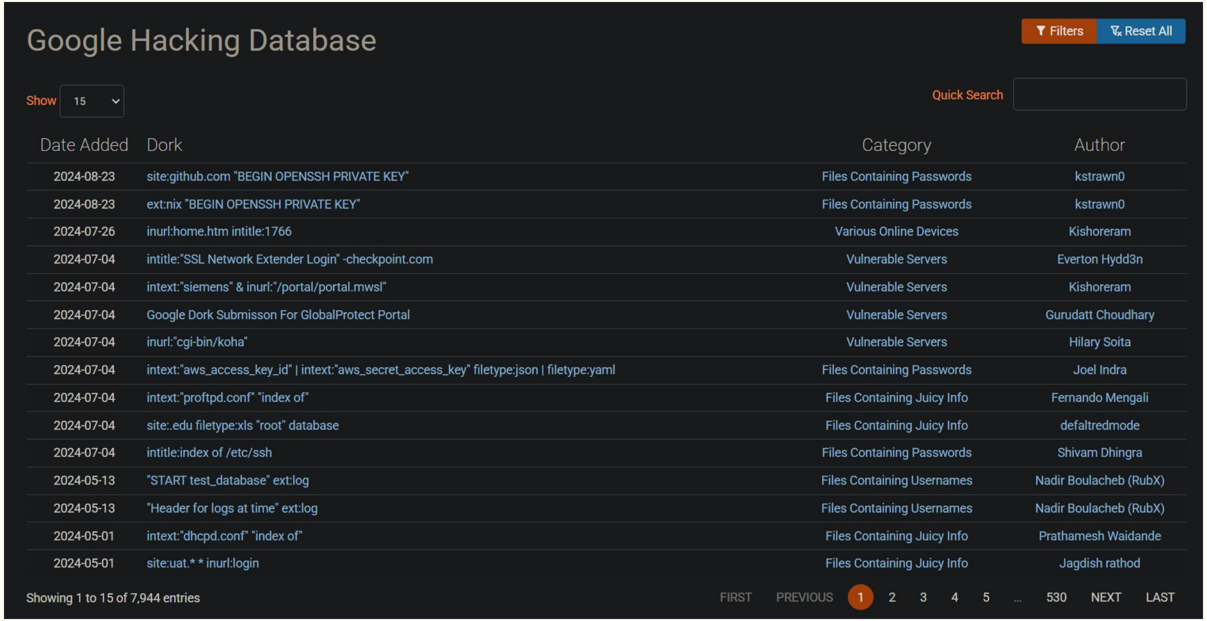
It’s common to find online, open to every user on the internet, reserved informations related to: government agencies, banks, private servers and many more…

There are multiple ways to discover these files:

- Google Dorks (if indexed by Google)
- Paths busting/scanning (Gobuster)

File Disclosure - Google Dorks

It works under the assumption that if a file is accessible by a public URL a crawler (Google) eventually will find it.

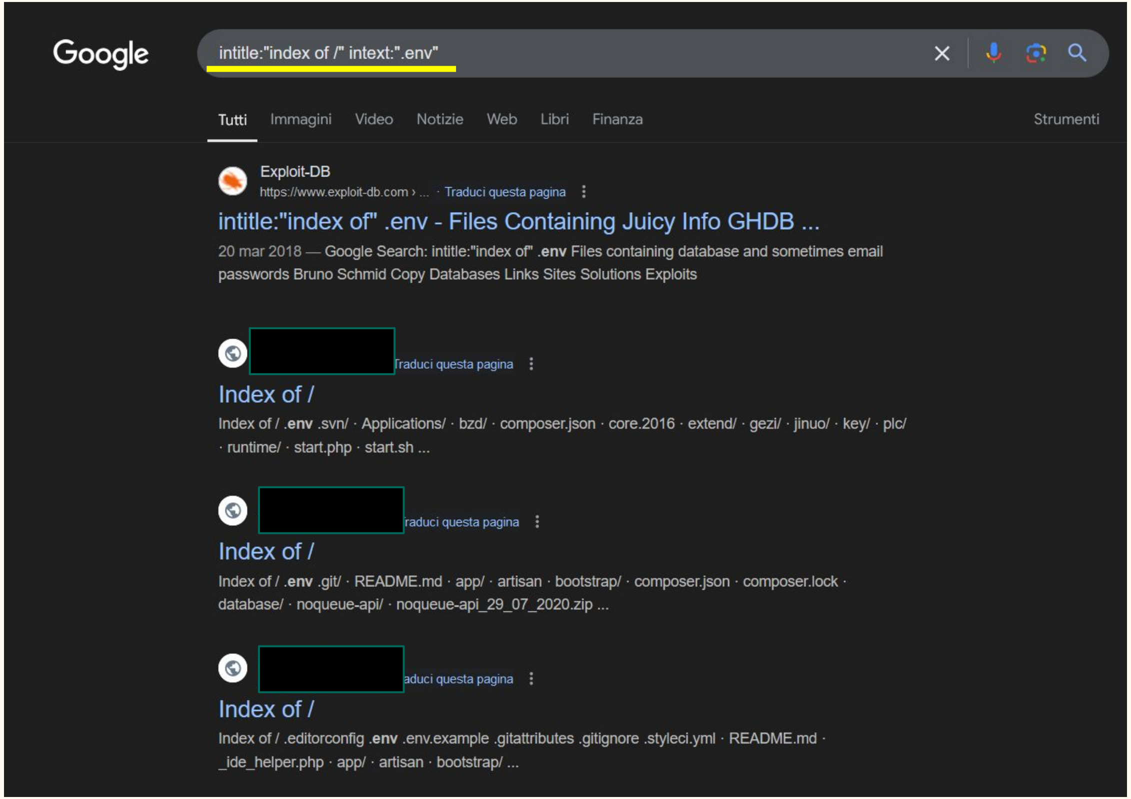


Google Hacking Database

Date Added	Dork	Category	Author
2024-08-23	site:github.com "BEGIN OPENSSH PRIVATE KEY"	Files Containing Passwords	kstrawn0
2024-08-23	ext:nix "BEGIN OPENSSH PRIVATE KEY"	Files Containing Passwords	kstrawn0
2024-07-26	inurl:home.htm intitle:1766	Various Online Devices	Kishoreram
2024-07-04	intitle:"SSL Network Extender Login" -checkpoint.com	Vulnerable Servers	Everton Hydd3n
2024-07-04	intext:"siemens" & inurl:"/portal/portal.mwsl"	Vulnerable Servers	Kishoreram
2024-07-04	Google Dork Submisson For GlobalProtect Portal	Vulnerable Servers	Gurudatt Choudhary
2024-07-04	inurl:"cgi-bin/koha"	Vulnerable Servers	Hilary Soita
2024-07-04	intext:"aws_access_key_id" | intext:"aws_secret_access_key" filetype:json | filetype:yaml	Files Containing Passwords	Joel Indra
2024-07-04	intext:"proftpd.conf" "index of"	Files Containing Juicy Info	Fernando Mengali
2024-07-04	site:.edu filetype:xls "root" database	Files Containing Juicy Info	defaltredmode
2024-07-04	intitle:index of /etc/ssh	Files Containing Passwords	Shivam Dhingra
2024-05-13	"START test_database" ext:log	Files Containing Usernames	Nadir Boulacheb (RubX)
2024-05-13	"Header for logs at time" ext:log	Files Containing Usernames	Nadir Boulacheb (RubX)
2024-05-01	intext:"dhcpd.conf" "index of"	Files Containing Juicy Info	Prathamesh Waidande
2024-05-01	site:uat.* * inurl:login	Files Containing Juicy Info	Jagdish rathod

Showing 1 to 15 of 7,944 entries

File Disclosure - Google Dorks



intitle:"index of /" intext:".env"

Exploit-DB
https://www.exploit-db.com › ... · Traduci questa pagina

intitle:"index of" .env - Files Containing Juicy Info GHDB ...

20 mar 2018 — Google Search: intitle:"index of" **.env** Files containing database and sometimes email passwords Bruno Schmid Copy Databases Links Sites Solutions Exploits

[redacted] Traduci questa pagina

Index of /

Index of / **.env** .svn/ · Applications/ · bzd/ · composer.json · core.2016 · extend/ · gezi/ · jinuo/ · key/ · plc/ · runtime/ · start.php · start.sh ...

[redacted] raduci questa pagina

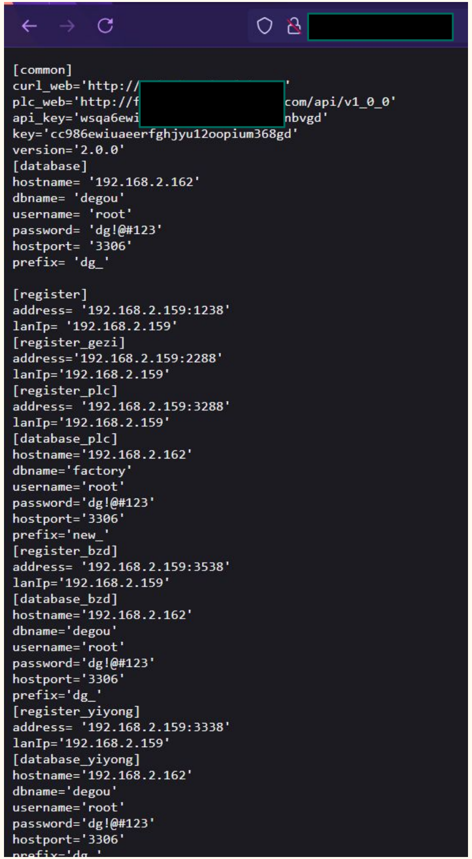
Index of /

Index of / **.env** .git/ · README.md · app/ · artisan · bootstrap/ · composer.json · composer.lock · database/ · noqueue-api/ · noqueue-api_29_07_2020.zip ...

[redacted] aduci questa pagina

Index of /

Index of / .editorconfig **.env** .env.example .gitattributes .gitignore .styleci.yml · README.md · _ide_helper.php · app/ · artisan · bootstrap/ ...


[redacted]

```
[common]
curl_web='http://[redacted]'
plc_web='http://f[redacted]com/api/v1_0_0'
api_key='wsqa6ewi[redacted]nbvgd'
key='cc986ewiuaeerfghjyu12oopium368gd'
version='2.0.0'
[database]
hostname= '192.168.2.162'
dbname= 'degou'
username= 'root'
password= 'dg!@#123'
hostport= '3306'
prefix= 'dg_'

[register]
address= '192.168.2.159:1238'
lanIp= '192.168.2.159'
[register_gezi]
address='192.168.2.159:2288'
lanIp='192.168.2.159'
[register_plc]
address= '192.168.2.159:3288'
lanIp='192.168.2.159'
[database_plc]
hostname='192.168.2.162'
dbname='factory'
username='root'
password='dg!@#123'
hostport='3306'
prefix='new_'
[register_bzd]
address= '192.168.2.159:3538'
lanIp='192.168.2.159'
[database_bzd]
hostname='192.168.2.162'
dbname='degou'
username='root'
password='dg!@#123'
hostport='3306'
prefix='dg_'
[register_yiyong]
address= '192.168.2.159:3338'
lanIp='192.168.2.159'
[database_yiyong]
hostname='192.168.2.162'
dbname='degou'
username='root'
password='dg!@#123'
hostport='3306'
prefix='dg_'
```

File Disclosure - Paths busting

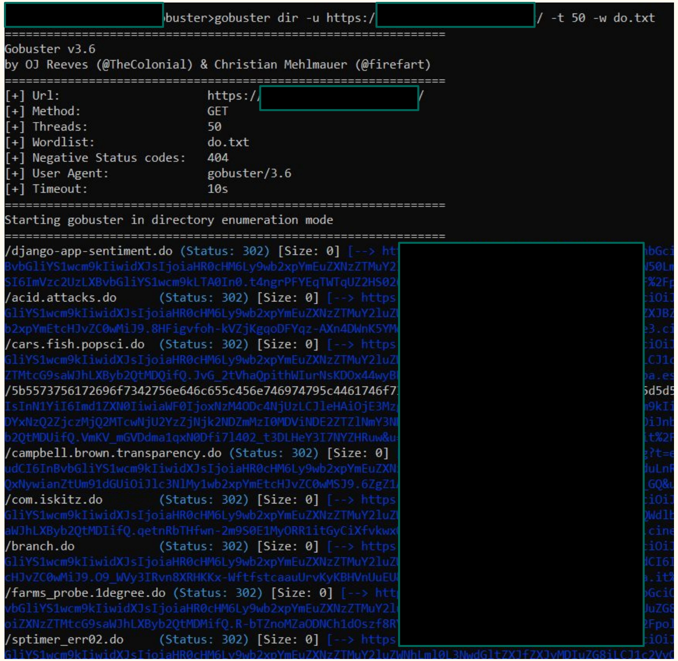
A website created using a known framework or that follows *best practice* will have a list of folders and files named in a standard way, such as: robots.txt, config.toml, .env and others...

If the attacker has specific informations on the target it could try plausible combinations (dictionary attack), otherwise a brute-force attack could be used (not recommended because of the “noise” it generates).

File Disclosure - Gobuster

Source: https://github.com/OJ/gobuster

It’s a tool used to brute-force URLs, DNS subdomains, Amazon S3 buckets, etc…



```
[redacted]buster>gobuster dir -u https:/[redacted]/ -t 50 -w do.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     https:/[redacted]/
[+] Method:                  GET
[+] Threads:                 50
[+] Wordlist:                do.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/django-app-sentiment.do (Status: 302) [Size: 0] [--> htt[redacted]bGci
BvbGliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2  N50Lm
SI6ImVzc2UzLXBvbGliYS1wcm9kLTA0In0.t4ngrPFYEqTWTqUZ2HS02  F%2Fp
/acid.attacks.do       (Status: 302) [Size: 0] [--> https  iOiJ
GliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2luZ  ZXJBZ
b2xpYmEtcHJvZC0wMiJ9.8HFigvfoh-kVZjKgqoDFYqz-AXn4DWnK5YM  e3.ci
/cars.fish.popsci.do  (Status: 302) [Size: 0] [--> https  iOiJ
GliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2luZ  LCJ1c
ZTMtcG9saWJhLXByb2QtMDQifQ.JvG_2tVhaQpithWIurNsKDOx44wyBl  oa.es
/5b5573756172696f7342756e646c655c456e746974795c4461746f7  5d5d5
IsInN1YiI6Imd1ZXN0IiwiaWF0IjoxNzM4ODc4NjUzLCJleHAiOjE3Mz  m9kIi
DYxNzQ2ZjczMjQ2MTcwNjU2YzZjNjk2NDZmMzI0MDViNDE2ZTZlNmY3N  OiJnb
b2QtMDUifQ.VmKV_mGVDdma1qxN0Dfi71402_t3DLHeY3I7NYZHRuw&u  it%2F
/campbell.brown.transparency.do (Status: 302) [Size: 0]  g?t=e
udCI6InBvbGliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXN  duLnR
QxNywianZtUm91dGUiOiJlc3NlMy1wb2xpYmEtcHJvZC0wMSJ9.6ZgZ1  _GQ&u
/com.iskitz.do         (Status: 302) [Size: 0] [--> https  iOiJ
GliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2luZ  QWdlb
aWJhLXByb2QtMDIifQ.qetnRbTHfwn-2m9S0E1MyORR1itGyCiXfvkwx  .cine
/branch.do             (Status: 302) [Size: 0] [--> https  iOiJ
GliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2luZ  dCI6I
cHJvZC0wMiJ9.O9_WVy3IRvn8XRHKKx-WftfstcaauUrvKyKBHVnUuEU  a.it%
/farms_probe.1degree.do (Status: 302) [Size: 0] [--> htt  oGciO
vbGliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2l  JuZG8
oiZXNzZTMtcG9saWJhLXByb2QtMDMifQ.R-bTZnoMZaODNCh1dOszf8R  2Fpol
/sptimer_err02.do      (Status: 302) [Size: 0] [--> https  iOiJ
GliYS1wcm9kIiwidXJsIjoiaHR0cHM6Ly9wb2xpYmEuZXNzZTMuY2luZWNhLml0L3NwdGltZXJfZXJyMDIuZG8iLCJ1c2VyO
```

YOU TRYING EVERY COMBINATION KNOWN TO MANKIND
ME STOPPING YOU FROM BEING IP BLOCKED
imgflip.com

Server-Side Request Forgery

Server-Side Request Forgery

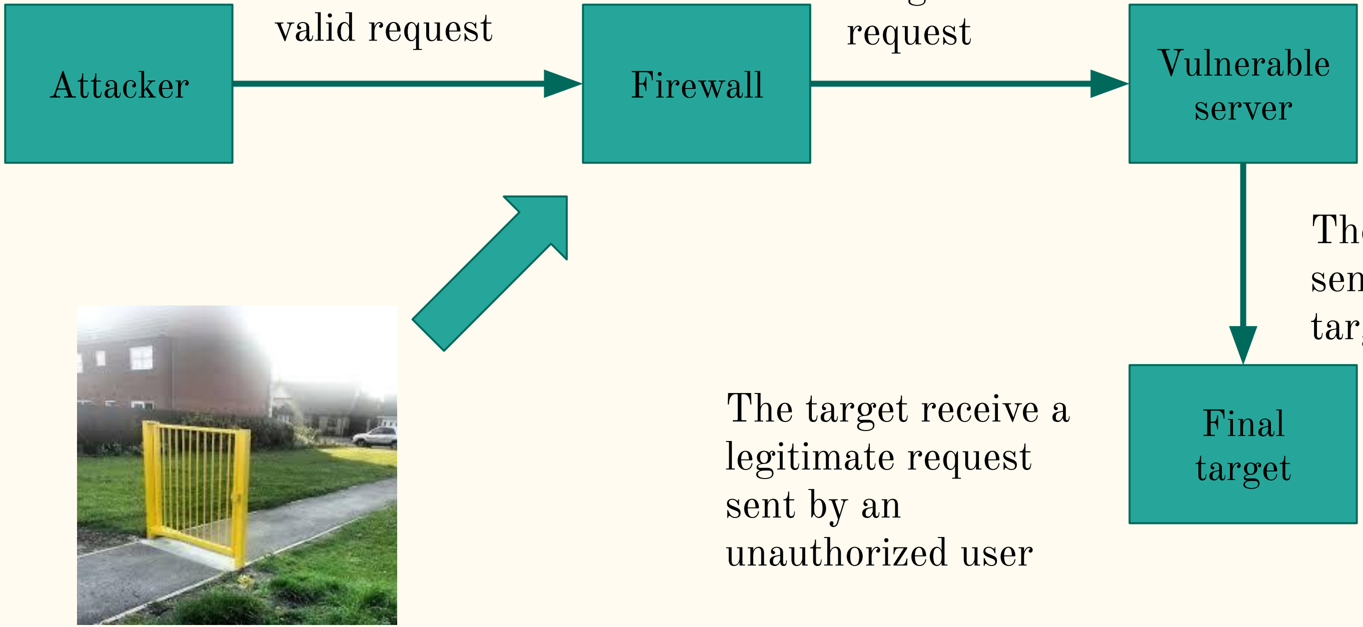
Source: https://portswigger.net/web-security/ssrf

A **Server-Side Request Forgery (SSRF)** is a vulnerability in which an attacker is able to send a request from a remote application.

The attacker might cause the target to make a connection to a service available only to them (i.e. intranet server, service that require authentication) or force a connection to arbitrary systems.

If successful, an SSRF can result in unauthorized access to sensible data or the execution of restricted actions.

Server-Side Request Forgery



Attacker

SSRF payload hidden inside a valid request

Firewall

The payload is hidden in a legitimate request

Vulnerable server

The vulnerable server while processing the SSRF payload execute an unwanted request

The vulnerable server sends a request to the target

Final target

The target receive a legitimate request sent by an unauthorized user

Server-Side Request Forgery

In order to find an SSRF, you should:

- Find a suspicious endpoint, a field where you can insert a custom URL controlled by you
- Check if you have pingback, if yes then probably you have an SSRF otherwise you may still have an SSRF but it will be more difficult
- Checking the response time based on the provided input could help

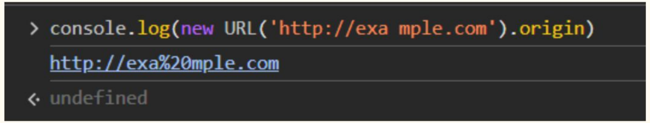
Try it: https://ctf.cyberchallenge.it/challenges#challenge-6

Server-Side Request Forgery - URL parsing

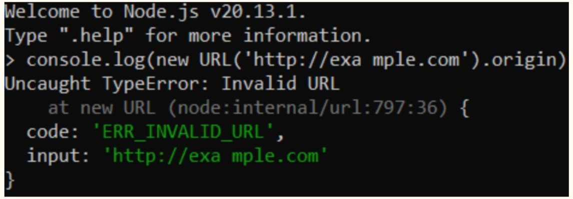
Remember this? Same input but different results...

HTTP Overview - URLencoding

There is a problem:



```
> console.log(new URL('http://exa mple.com').origin)
  http://exa%20mple.com
< undefined
```



```
Welcome to Node.js v20.13.1.
Type ".help" for more information.
> console.log(new URL('http://exa mple.com').origin)
Uncaught TypeError: Invalid URL
    at new URL (node:internal/url:797:36) {
  code: 'ERR_INVALID_URL',
  input: 'http://exa mple.com'
}
```

Foreshadowing alert!

Server-Side Request Forgery - URL parsing

https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf

Even though URL is a standard, it turns out the implementation of URL parsers is not the same in different languages and libraries.

Example:

```
http://1.1.1.1 &@2.2.2.2# @3.3.3.3/

urllib2 and httplib: http://1.1.1.1/

requests: http://2.2.2.2/

urllib: http://3.3.3.3/
```

Server-Side Request Forgery - URL parsing

Example:

```
http://127.0.0.1:11211:80/

readfile(PHP) and Perl LWP: http://127.0.0.1:11211/

parse_url(PHP) and Perl URI: http://127.0.0.1:80/
```

Server-Side Request Forgery - URL parsing

Example:

```
http://google.com#@evil.com/

parse_url(PHP): http://google.com/

readfile(PHP): http://evil.com/
```

The End