CSC 405
Introduction to Computer Security

Web Security

Alexandros Kapravelos
akaprav@ncsu.edu

(Derived from slides by Giovanni Vigna)

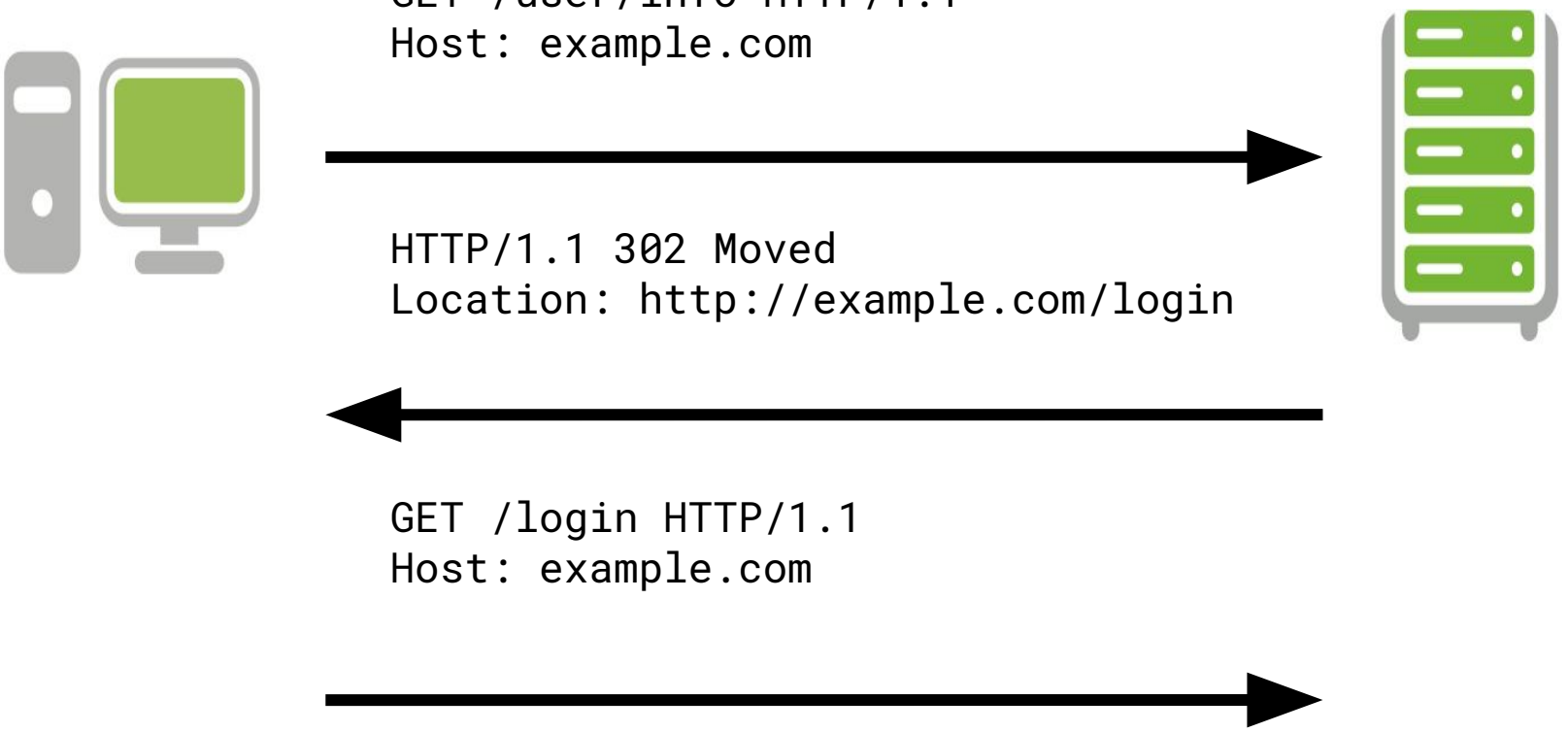
Logic Flaws

- Logic flaws come in many forms and are specific to the intended functionality and security policy of an application
- Received little attention
  - Are known to be hard to identify in automated analysis
  - Not much public information
- Are on the rise: "…as the number of common vulnerabilities such as SQL injection and cross-site scripting are reduced, the bad guys are increasing their attacks on business logic flaws" [J. Grossman, WhiteHat Security]

Fear the EAR

- Execution-After-Redirect vulnerabilities are introduced when code is executed after producing a redirect header
- The developer assumes that since a redirection occurred, code execution stopped
  - Redirect used as a goto
- Normally the behavior is invisible to the user, because the browser automatically load the page referenced by the redirection

HTTP Redirects



Execution After Redirect: Example

```
class TopicsController < ApplicationController
  def update
    @topic = Topic.find(params[:id])
    if not current_user.is_admin?
      redirect_to("/")
    end
    @topic.update_attributes(params[:topic])
    flash[:notice] = "Topic updated!"
  end
end
```

EAR History

- 17 Common Vulnerabilities and Exposures (CVE)
  - Starting in 2007
  - Difficult to find – no consistent category
- Blog post about Cake PHP 2006
  - Resulted in a bug filed and documentation changed
- Prior work on logic flaws
  - Found EAR in J2EE web application
- No one recognized it as a systemic logic flaw amongst web applications

Types of EARs

- Benign
  - The state of the web application does not change
  - No leak of sensitive information
- Vulnerable
  - Allows for the unauthorized modification of the application state or discloses unauthorized data

EAR: Information Leakage

```
<?php
$current_user = get_current_user();
if (!$current_user->is_admin())
{
  header("Location: /");
}
echo "457-55-5462";
?>
```

Prevention

- Secure design
  - Django, ASP.NET MVC
- Terminate process or thread
  - ASP.NET, CakePHP, Zend, CodeIgniter
- Patched Ruby on Rails
  - Exception handling

Attacking HTTP Protocol Implementations

- HTTP protocol heavily scrutinized
- HTTP protocol implementations might be erroneous
- Examples:
  - HTTP response splitting
  - HTTP request smuggling

HTTP Response Splitting

- HTTP response splitting exploits the fact that user provided data is included in the header of a reply
- See: “HTTP Response Splitting, Web Cache Poisoning Attacks, and Related Topics” by A. Klein

Redirection Example

- /redir_lang.jsp

```
<%
response.sendRedirect("/by_lang.jsp?lang="+
   request.getParameter("lang"));
%>
```

- For example calling with “lang” set to “English” will create the following redirect

```
HTTP/1.1 302 Moved Temporarily
Date: Tue, 17 Nov 2015 12:53:28 GMT
Location: http://10.1.1.1/by_lang.jsp?lang=English
...

<html>Error...</html>
```

Response Splitting

- What if we call:

```
/redir_lang.jsp?lang=foobar%0d%0aContent-
   Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-
   Type:%20text/html%0d%0aContent-
   Length:%2019%0d%0a%0d%0a<html>Shazam</html>
```

- If the server includes the value of the “lang” variable verbatim, the resulting answer may appear as two different replies
- If the attacker includes a request for /index.html, the second (fake) reply will be associated with it, possibly poisoning an intermediate cache

The Response Output

```
HTTP/1.1 302 Moved Temporarily
Date: Tue, 17 Nov 2015 15:26:41 GMT
Location: http://10.1.1.1/by_lang.jsp?lang=foobar
Content-Length: 0

HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 19

<html>Shazam</html>
...(ignored stuff here)
```

- From now on an intermediate proxy will associate the second reply with /index.html

HTTP Request Smuggling

- This attack exploits the difference in the parsing procedures performed by different components in the request delivery process
- For example, request is forged to confuse proxy and desynchronize its view from the view of the server
- See “HTTP Request Smuggling” by C. Linhart et al.

HTTP Request Smuggling

```
1   POST http://SITE/foobar.html HTTP/1.1
2   Host: SITE
3   Connection: Keep-Alive
4   Content-Type: application/x-www-form-urlencoded
5   Content-Length: 0
6   Content-Length: 44
7   [CRLF]
8   GET /poison.html HTTP/1.1
9   Host: SITE
10  Bla: [space after the "Bla:", but no CRLF]
11  GET http://SITE/page_to_poison.html HTTP/1.1
12  Host: SITE
13  Connection: Keep-Alive
14  [CRLF]
```

HTTP Request Smuggling

- Proxy
  - The proxy parses the POST request in lines 1-7, and encounters the two "Content-Length" headers
  - It decides to ignore the first header, so it assumes the request has a body of length 44 bytes
  - It treats the data in lines 8-10 as the first request's body. Then it parses lines 11-14, which it treats as the client's second request
- The server
  - Uses the first "Content-Length" header and believes that the first POST request has no body
  - The second request is the GET in line 8 (notice that the GET in line 11 is parsed by the server as the value of the "Bla" header in line 10)

HTTP Request Smuggling

- The server sends back the content of
  - foobar.html
  - poison.html
- The proxy associates these requests to
  - POST /foobar.html
  - GET /page_to_poison.html
- From now on, every client asking for page_to_poison.html will receive poison.html, instead

OWASP Top Ten Web Vulnerabilities

- **A1: Injection**
  - Injection flaws, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data
- **A2: Broken Authentication and Session Management**
  - Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, session tokens, or exploit implementation flaws to assume other users' identities

OWASP Top Ten Web Vulnerabilities

- **A3: Cross-Site Scripting (XSS)**
  - XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation and escaping. XSS allows attackers to execute script in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites
- **A4: Insecure Direct Object References**
  - A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data

OWASP Top Ten Web Vulnerabilities

- **A5: Security Misconfiguration**
  - Security depends on having a secure configuration defined for the application, framework, web server, application server, and platform. All these settings should be defined, implemented, and maintained as many are not shipped with secure defaults
- **A6: Sensitive Data Exposure**
  - Many web applications do not properly protect sensitive data, such as credit cards, tax IDs, and authentication credentials. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.

OWASP Top Ten Web Vulnerabilities

- **A7: Missing Function Level Access Control**
  - Most web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to forge requests in order to access functionality without proper authorization
- **A8: Cross-Site Request Forgery**
  - A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other authentication information, to a vulnerable web application. This allows the attacker to force the victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim

OWASP Top Ten Web Vulnerabilities

- **A9: Using Known Vulnerable Components**
  - Components, such as libraries, frameworks, and other software modules, almost always run with full privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts
- **A10: Unvalidated Redirects and Forwards**
  - Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages

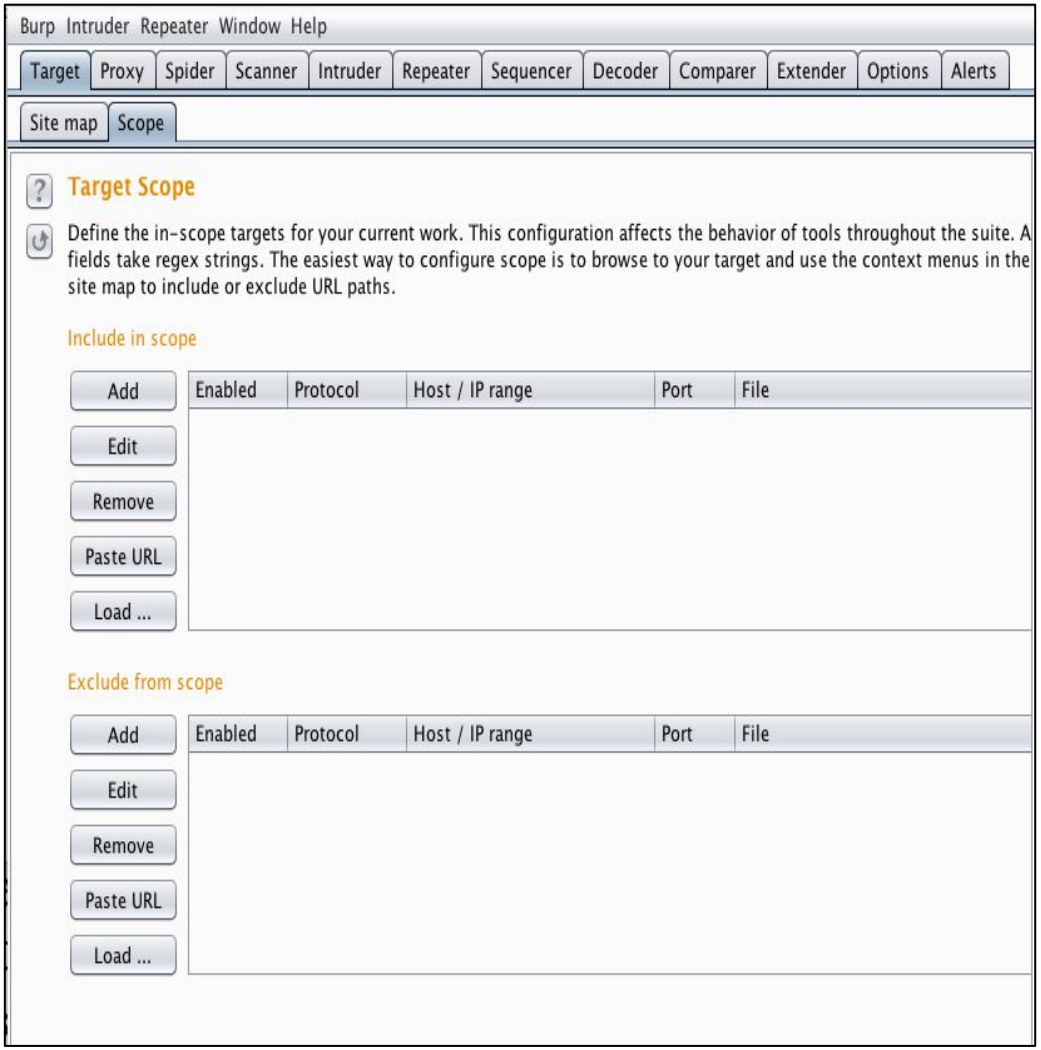
Burp

- Security tool for the interception, analysis, and modification of web requests
  - Free version: No save and restore capability
  - Commercial version ($299/year): adds vulnerability scanner
- Implemented in Java, runs on any platform
- The tool can be extended with custom modules

Target Specification

- Supports the description of the allowed hosts/URLs
- Important to avoid inadvertent attacks to hosts reached through links

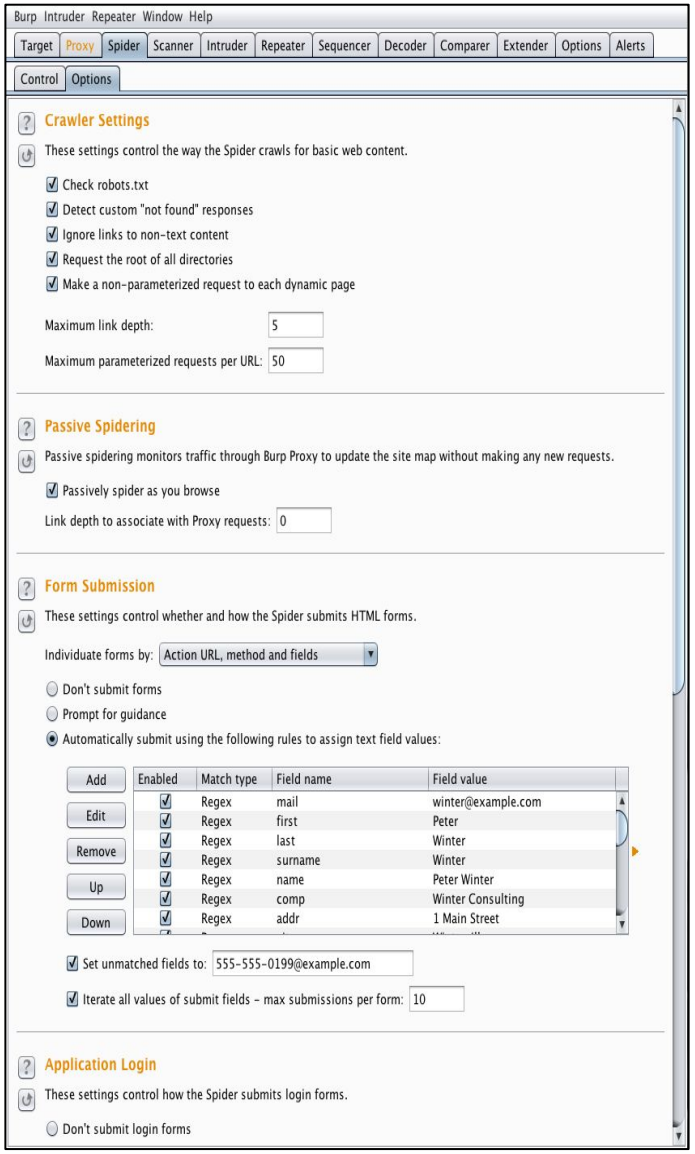
Target Specification



Crawling

- Burp provides a crawling component (spider)
  - Can be active or passive
- Allows for the discovery of pages automatically
  - Supports the submission of forms
    - Not perfect, but great value for the price
    - See: “Why Johnny Can't Pentest: An Analysis of Black-box Web Vulnerability Scanners,” Adam Doupe, Marco Cova, Giovanni Vigna, in Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA), 2010

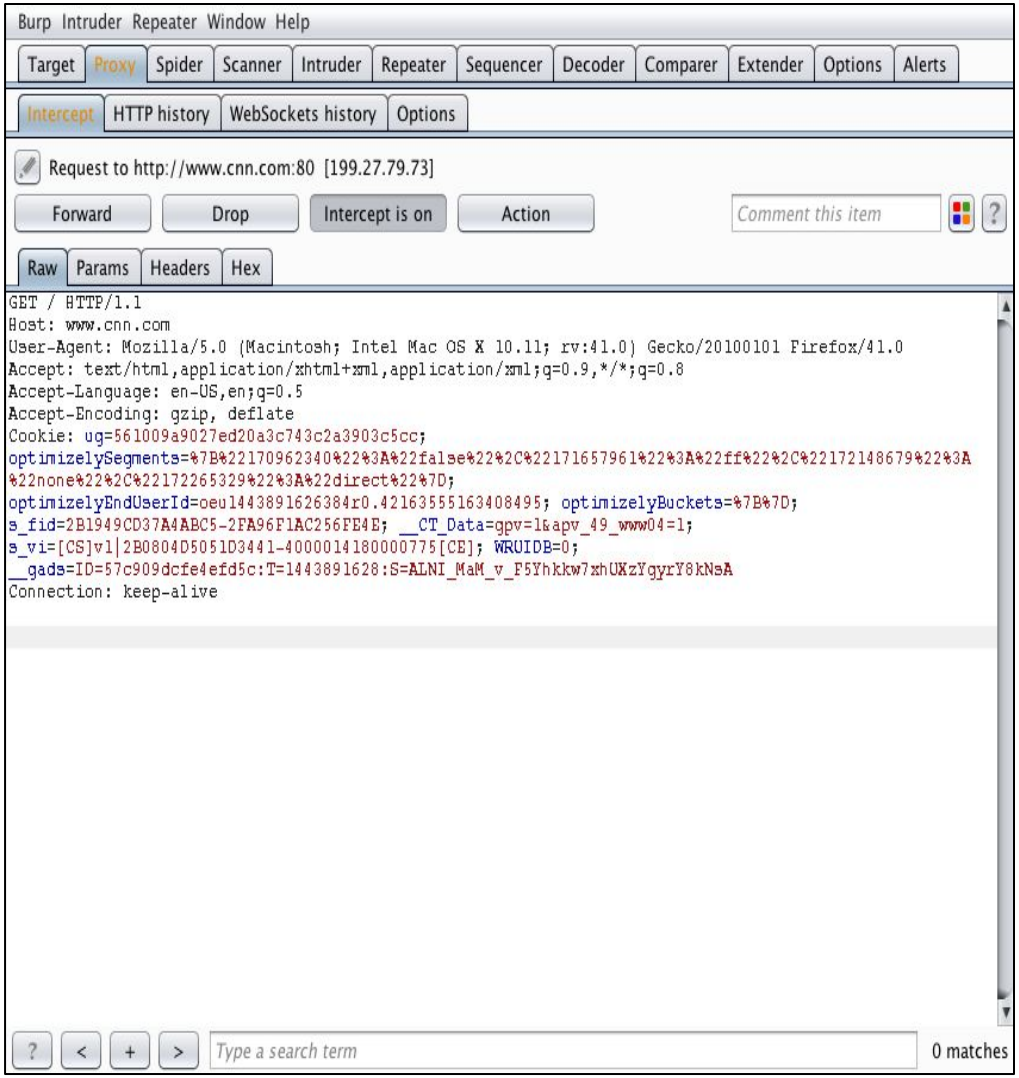
Crawling


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
Control Options
Crawler Settings
These settings control the way the Spider crawls for basic web content.
Check robots.txt
Detect custom "not found" responses
Ignore links to non-text content
Request the root of all directories
Make a non-parameterized request to each dynamic page
Maximum link depth: 5
Maximum parameterized requests per URL: 50
Passive Spidering
Passive spidering monitors traffic through Burp Proxy to update the site map without making any new requests.
Passively spider as you browse
Link depth to associate with Proxy requests: 0
Form Submission
These settings control whether and how the Spider submits HTML forms.
Individuate forms by: Action URL, method and fields
Don't submit forms
Prompt for guidance
Automatically submit using the following rules to assign text field values:
Add
Edit
Remove
Up
Down
Enabled Match type Field name Field value
Regex mail winter@example.com
Regex first Peter
Regex last Winter
Regex surname Winter
Regex name Peter Winter
Regex comp Winter Consulting
Regex addr 1 Main Street
Set unmatched fields to: 555-555-0199@example.com
Iterate all values of submit fields - max submissions per form: 10
Application Login
These settings control how the Spider submits login forms.
Don't submit login forms

Proxy

- The proxy functionality allows for the interception and modification of requests

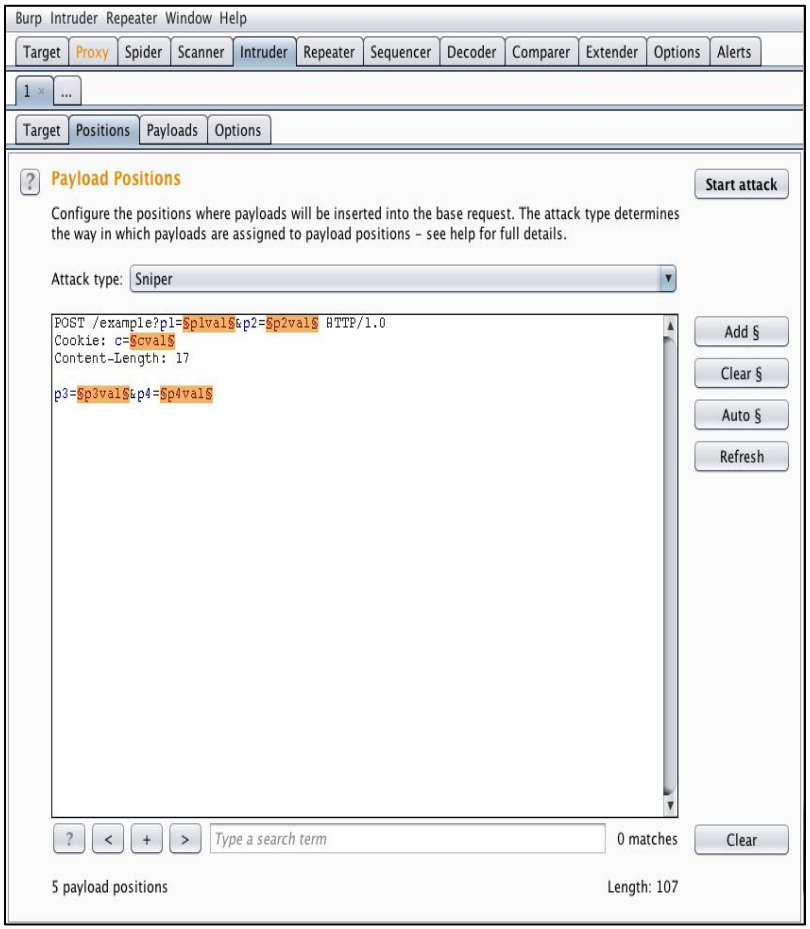
Proxy

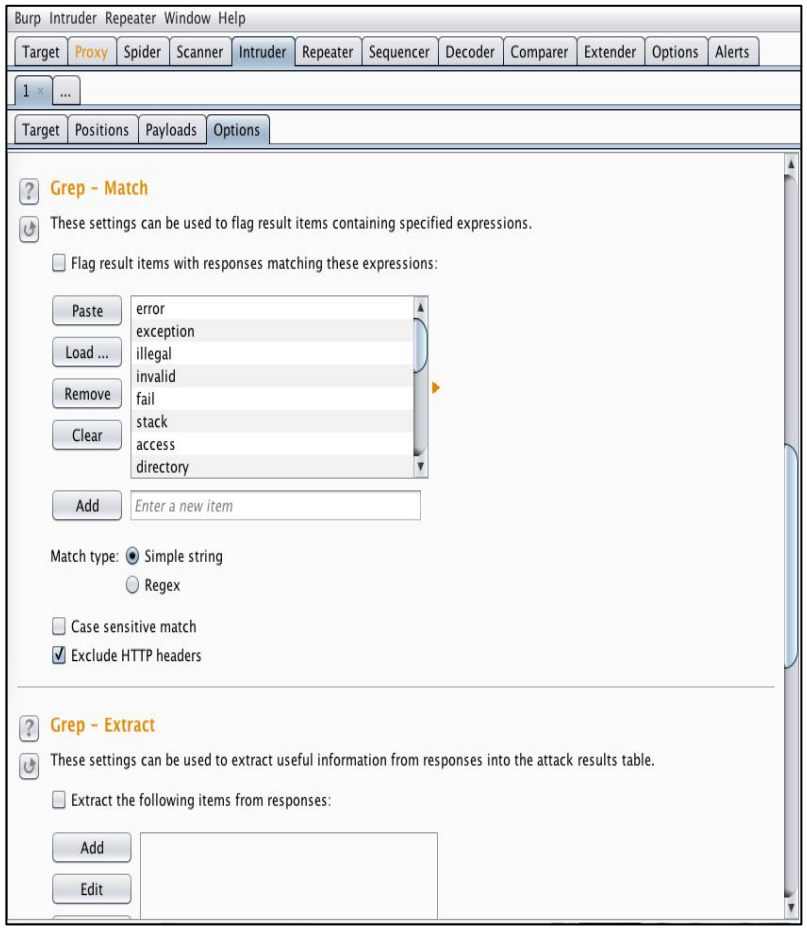

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
Intercept HTTP history WebSockets history Options
Request to http://www.cnn.com:80 [199.27.79.73]
Forward Drop Intercept is on Action
Comment this item
Raw Params Headers Hex
GET / HTTP/1.1
Host: www.cnn.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:41.0) Gecko/20100101 Firefox/41.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: ug=561009a9027ed20a3c743c2a3903c5cc;
optimizelySegments=%7B%221709623402%22%3A%22false%22%2C%221716579612%22%3A%22ff%22%2C%221721486792%22%3A
%22none%22%2C%221722653292%22%3A%22direct%22%7D;
optimizelyEndUserId=oeu1443891626384r0.42163555163408495; optimizelyBuckets=%7B%7D;
s_fid=2B1949CD37A4ABC5-2FA96F1AC256FE4E; __CT_Data=gpv=1&apv_49_www04=1;
s_vi=[CS]v1|2B0804D5051D3441-4000014180000775[CE]; WRUIDB=0;
__gads=ID=57c909dcfe4efd5c:T=1443891628:S=ALNI_MaM_v_F5Yhkkw7xhUXzYgyrY8kNsA
Connection: keep-alive
? < + >
Type a search term
0 matches

Intruder

- Intruder allows one to perform simple fuzzing of a request
- Request is selected from history and forwarded to the module
- The parameters are substituted with placeholders
- The user specifies regular expression to be matched against the result of the request
- When activated the intruder perform a large number of requests, looking for possible error conditions

Intruder

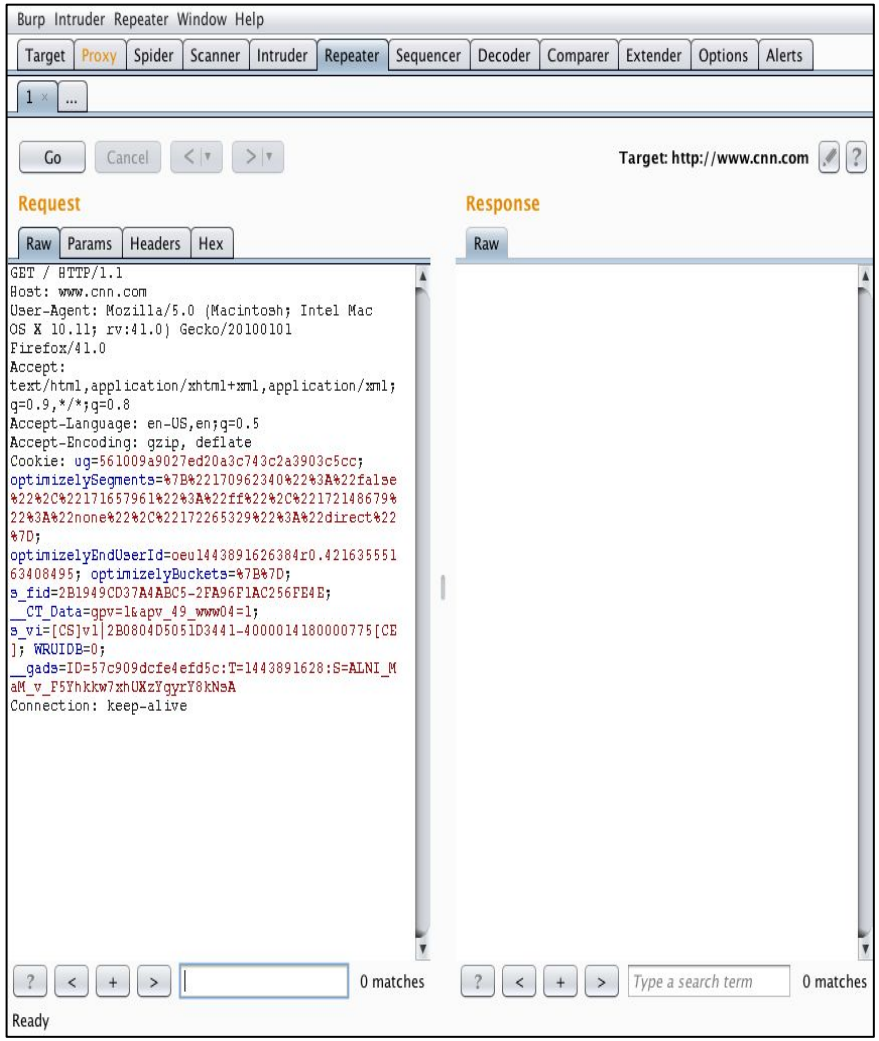

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
1 ...
Target Positions Payloads Options
Payload Positions
Start attack
Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.
Attack type: Sniper
POST /example?p1=$p1val$&p2=$p2val$ HTTP/1.0
Cookie: c=$cval$
Content-Length: 17
p3=$p3val$&p4=$p4val$
Add §
Clear §
Auto §
Refresh
Type a search term
0 matches
Clear
5 payload positions
Length: 107


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
1 ...
Target Positions Payloads Options
Grep - Match
These settings can be used to flag result items containing specified expressions.
Flag result items with responses matching these expressions:
Paste
Load ...
Remove
Clear
error
exception
illegal
invalid
fail
stack
access
directory
Add
Enter a new item
Match type: Simple string
Regex
Case sensitive match
Exclude HTTP headers
Grep - Extract
These settings can be used to extract useful information from responses into the attack results table.
Extract the following items from responses:
Add
Edit

Repeater

- Allows for manual exploration of requests
- Request in history is sent to the module
- User can modify request and observe immediately the effect on the response

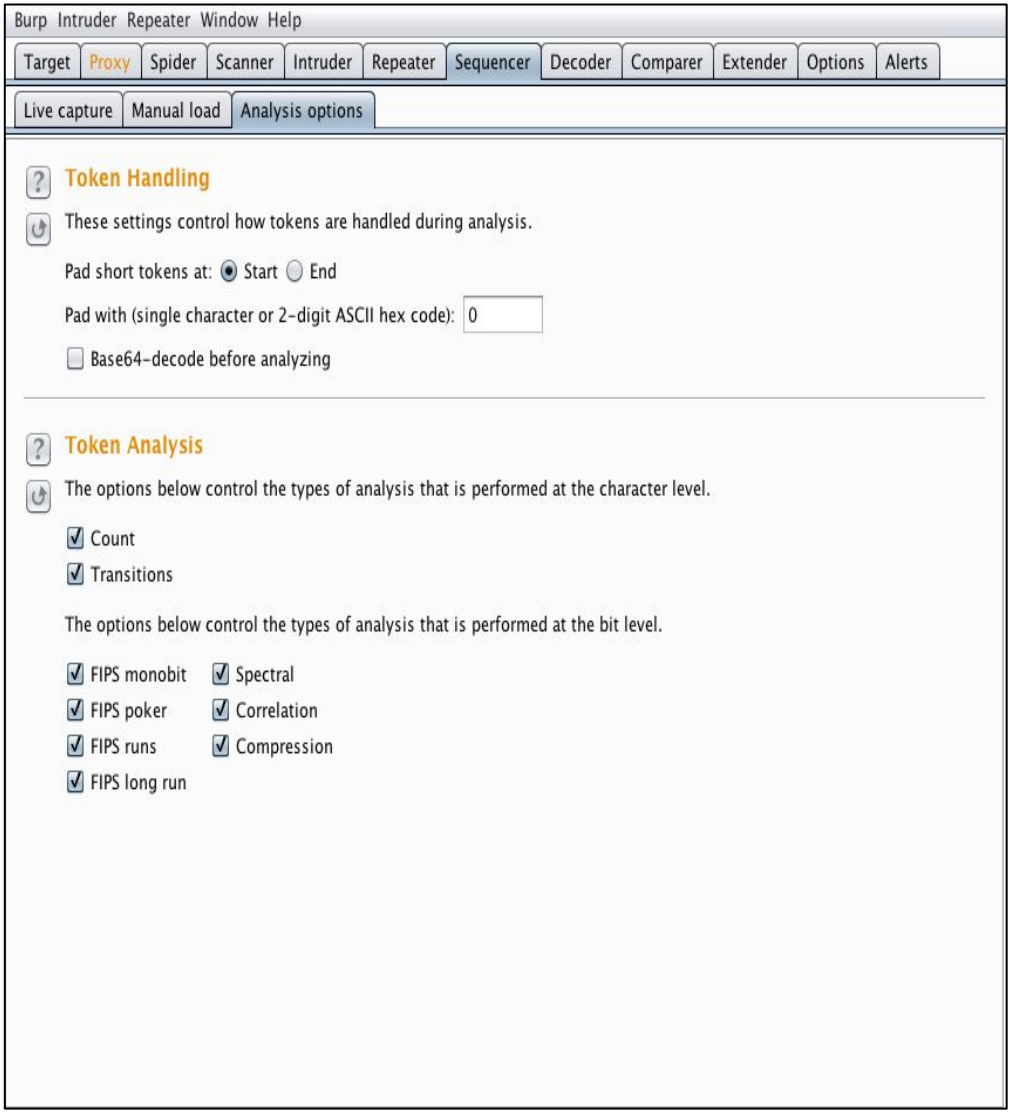
Repeater


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
1 ...
Go Cancel
Target: http://www.cnn.com
Request
Raw Params Headers Hex
GET / HTTP/1.1
Host: www.cnn.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:41.0) Gecko/20100101 Firefox/41.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: ug=561009a9027ed20a3c743c2a3903c5cc; optimizelySegments=%7B%22170962340%22%3A%22false%22%2C%221657961%22%3A%22ff%22%2C%22172148679%22%3A%22none%22%2C%2217226532 9%22%3A%22direct%22%7D; optimizelyEndUserId=oeu1443891626384r0.42163555163408495; optimizelyBuckets=%7B%7D; s_fid=2B1949CD37A4ABC5-2FA96F1AC256FE4E; __CT_Data=gpv=1&apv_49_www04=1; s_vi=[CS]v1|2B0804D5051D3441-4000014180000775[CE]; WRUIDB=0; __gads=ID=57c909dcfe4efd5c:T=1443891628:S=ALNI_MaM_v_F5Yhkkw7xhUXzYgyrY8kNsA
Connection: keep-alive
Response
Raw
0 matches
Type a search term
0 matches
Ready

Sequencer

- Module that allows for the statistical analysis of tokens used in web applications
- Useful to determine the “randomness” of fields (esp. session IDs)

Sequencer


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
Live capture Manual load Analysis options
Token Handling
These settings control how tokens are handled during analysis.
Pad short tokens at: Start End
Pad with (single character or 2-digit ASCII hex code): 0
Base64-decode before analyzing
Token Analysis
The options below control the types of analysis that is performed at the character level.
Count
Transitions
The options below control the types of analysis that is performed at the bit level.
FIPS monobit
FIPS poker
FIPS runs
FIPS long run
Spectral
Correlation
Compression

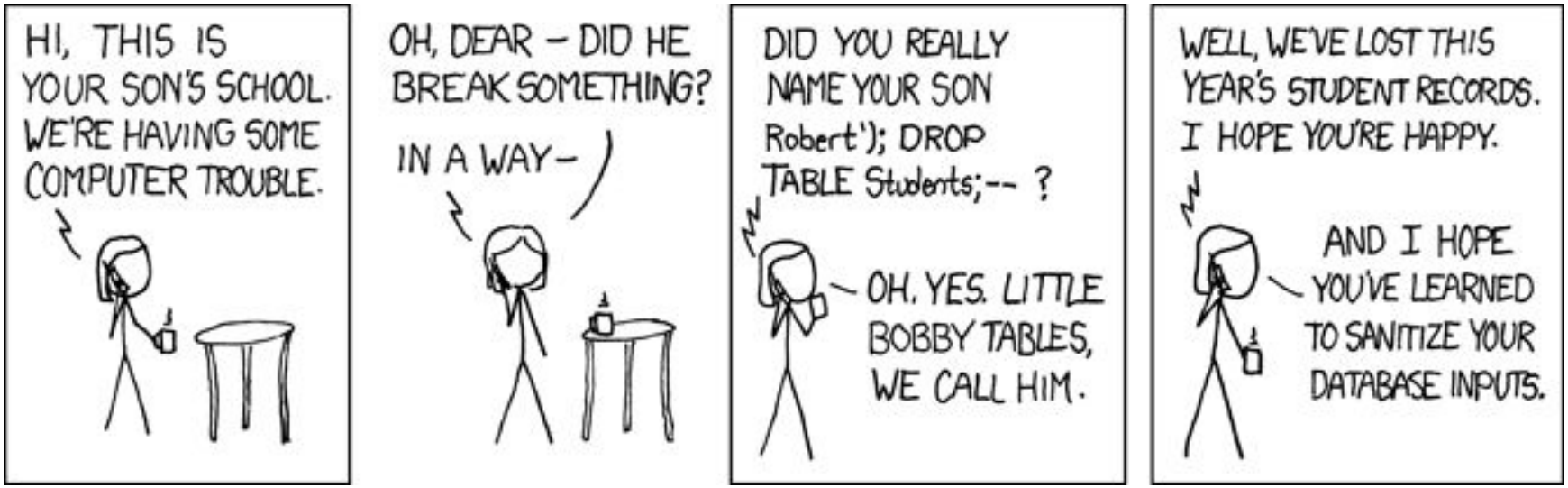
Decoding and Comparing

- The Decoder module allows for the encoding and decoding of content
- The Comparer modules allows for the diffing of content

Conclusions

- Web applications have become the way in which we store and manage sensitive information
- Web security is different from application security
  - Modules can be executed in any order
  - Modules can be invoked in parallel
- Often times the developers of traditional applications make erroneous assumptions when developing web applications

Questions?



Source: http://xkcd.com/327/

Your Security Zen

Microsoft Office 0-day in the wild

All Office and Windows versions are vulnerable

Even Windows 10…

Patch is coming today

Reports say it was first seen in July!

You can block the attack by changing:

Software\Microsoft\Office\15.0\Word\Security\FileBlock\RtfFiles to 2 and OpenInProtectedView to 0

Source: https://arstechnica.com/security/2017/04/booby-trapped-word-documents-in-the-wild-exploit-critical-microsoft-0day