# Automating Web Application Security

## Getting the Most out of curl and Perl

**Paco Hope**
**Technical Manager**
**Cigital, Inc.**

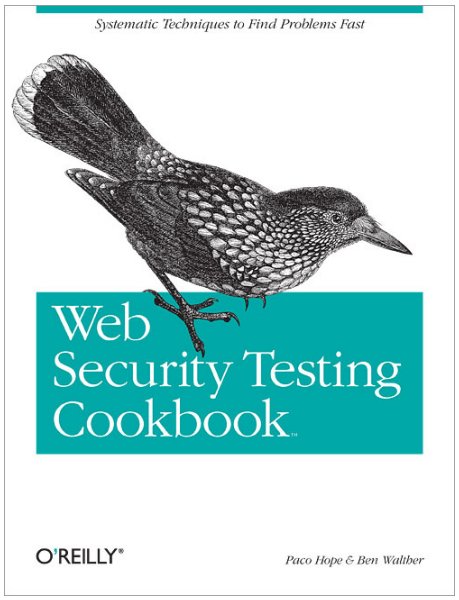
paco@cigital.com
+1.703.404.5769
http://www.cigital.com/

cigital

cigital

# Agenda

- Motivation
- Basis for automation: HTTP
- Blind automation: curl
- Thoughtful automation: Perl
- Automating security
- Thoughts for further application



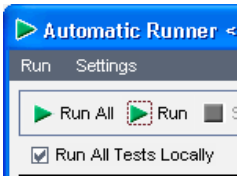

 © 2007 Cigital, Inc. All Rights Reserved. 

# Motivation

*"Phenomenal cosmic POWER!*
*...itty bitty living space"*



# Test Automation Vision




1

2

| Plan: Test Name | Plan: Type | Status |
|---|---|---|
| [1]Master-001 | VAPI-XP-TEST | N/A |
| [1]Child-001 | MANUAL | Failed |
| [1]Child-002 | MANUAL | Passed |
| [1]Child-003 | MANUAL | Passed |
| [1]Child-004 | MANUAL | Passed |
| [1]Child-005 | MANUAL | Failed |
| [1]Child-006 | MANUAL | Passed |

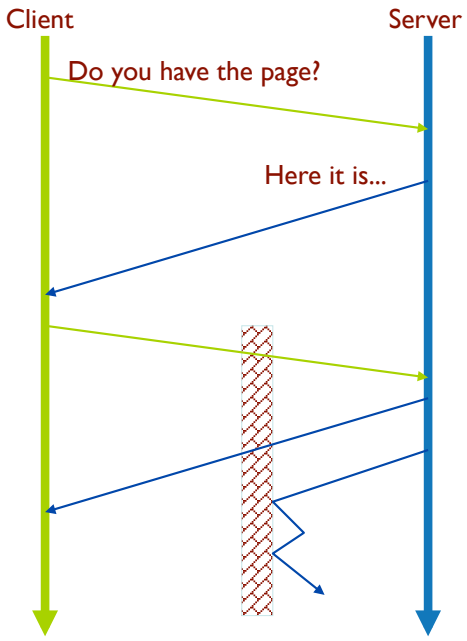
3

# Why Are Web Apps Special?

- Well-known interface (HTTP)
  - Automatically supports automation
- On the wild & woolly Internet
- Everyone is writing them
  - Professional developers
  - DBAs
  - IT people
  - Kids fresh out of school
- Built from parts that you didn't write
  - Web server
  - App server
  - Scripting languages (.NET, J2EE, PHP, etc.)
- Easy to have behavior you didn't write or intend
  - Demo code
  - Features you don't use



# HTTP: Client / Server




- Server sits around waiting for connections
- Clients initiate connections
  - There's no such thing as server "push"
  - There are ways to fake it
- Clients:
  - Browsers
  - Flash Player
  - Java Applets

# HTTP

```
GET /silverbullet/ HTTP/1.1
Host=www.cigital.com
User-Agent=Mozilla/5.0 (Macintosh;
   U; Intel Mac OS X; en-US; rv:
   1.8.0.6) Gecko/20060728 Firefox/
   1.5.0.6
Accept=text/xml,application/xml
Accept-Language=en-us,en;q=0.5
Accept-Encoding=gzip,deflate
Accept-Charset=ISO-8859-1,utf-8
Keep-Alive=300
Connection=keep-alive
```

```
HTTP/1.x 200 OK
Date=Tue, 29 Aug 2006 19:28:16 GMT
Server=Apache
X-Powered-By=PHP/4.3.10
Keep-Alive=timeout=15, max=100
Connection=Keep-Alive
Transfer-Encoding=chunked
Content-Type=text/html
Set-Cookie=
    SID=2951012237E410378D93B60D0FEE575E;
    path=/; domain=.cigital.com

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
    1.0 Transitional//EN" "http://
    www.w3.org/TR/xhtml1/DTD/xhtml1-
    transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    lang="en" xml:lang="en">
<head>
    <title>Cigital -- The Software Quality
    Company</title>
```



# Requests

**Getting http://www.cigital.com/silverbullet/**

```
GET /silverbullet/ HTTP/1.1
Host=www.cigital.com
User-Agent=Mozilla/5.0 (Macintosh; U;
   Intel Mac OS X; en-US; rv:1.8.0.6)
   Gecko/20060728 Firefox/1.5.0.6
Referer=http://www.cigital.com/
Accept=text/xml,application/xml
Accept-Language=en-us,en;q=0.5
Accept-Encoding=gzip,deflate
Accept-Charset=ISO-8859-1,utf-8
Keep-Alive=300
Connection=keep-alive
```

- Note path separated from host name
- User-agent is a courtesy
  - Might be a lie
- Referer
  - is a courtesy
  - not always there

# HTTP Methods

| | Advantages | Disadvantages |
|---|---|---|
| **GET** | • Params all in the URL<br>• Easy to bookmark<br>• Can work without server-side state (e.g. database) | • All params in server log in clear text<br>• Params show up in browser history on user's PC<br>• Limits to size and complexity of interactions |
| **POST** | • Data contained in the connection itself<br>• Allows complex and rich interactions<br>– Large reqs / resps<br>– File upload<br>– MIME<br>– Unlimited parameters | • A little harder to test<br>• Building MIME reqs<br>• Still have to account for GET possibilities |



# Paco's Rules of Web Security Testing

1. Throw away your web browser
   - Hackers don't use web browsers
   - Avoid Internet Explorer for *security* testing, (Use for UAT)
2. Throw away your mouse
   - Hackers don't click on things
   - Everything boils down to HTTP input that can be simulated
3. Divide & Conquer
   - Use boundary cases
   - Use equivalence classes
4. Automate, automate, automate
   - This is what your enemy does
   - Try variations programmatically

## cURL


In Ten Slides

Get it from **`http://curl.haxx.se/`**

cigital

cigital

# Summary

- Fetch URLs
- Save to files
- Lots of controls
- Easy to script

```
curl http://www.example.com/ -o example.html
```

 © 2007 Cigital, Inc. All Rights Reserved. 


# Fetching Ranges Automatically

## Expand range descriptions

```
curl http://www.example.com/category.asp?id=[0-9]
   -o category-#1.html
```

```
http://www.example.com/category.asp?id=0 → category-0.html
http://www.example.com/category.asp?id=1 → category-1.html
http://www.example.com/category.asp?id=2 → category-2.html
etc.
```

```
curl http://example.com/item.asp?id=[0-9]&style=[3-4]
   -o item#1-#2.html
```

```
http://example.com/item.asp?id=0&style=3 → item0-3.html
http://example.com/item.asp?id=0&style=4 → item0-4.html
http://example.com/item.asp?id=1&style=3 → item1-3.html
http://example.com/item.asp?id=1&style=4 → item1-4.html
```




But How Is This Security Testing?

# DEMO

# Fetch Lists

{specific,instances} [ranges]

```
curl 'http://example.com/{item,details,review}.asp?id=[0-2]' -o '#1-
   #2.html'

[1/12]:  http://example.com/item.asp?id=0    → item-0.html
[2/12]:  http://example.com/item.asp?id=1    → item-1.html
[3/12]:  http://example.com/item.asp?id=2    → item-2.html
[5/12]:  http://example.com/details.asp?id=0 → details-0.html
[6/12]:  http://example.com/details.asp?id=1 → details-1.html
[7/12]:  http://example.com/details.asp?id=2 → details-2.html
[9/12]:  http://example.com/review.asp?id=0  → review-0.html
[10/12]: http://example.com/review.asp?id=1  → review-1.html
[11/12]: http://example.com/review.asp?id=2  → review-2.html
```

- See the potential for automation?
  - Programmatically issue requests
  - Save results to files automatically



# Tracking cookies

- Create a cookie jar automatically (-c)
- Use the jar automatically (-b)

```
curl -c cookies.txt -b cookies.txt http://
   www.example.com/secure.asp -o secure.html
```

- See the potential for automation?
  - Jar files with test cookies for regression tests
  - Cookie jar files under version control!

# Posting Form Data


Assume we have a form that looks like this:

```
<form method="POST" action="http://www.example.com/
   servlet/login.do">
<p>User Name: <input type="text" name="userid"></p>
<p>Password:  <input type="text" name="passwd"></p>
<p><input type="submit" value="Login"></p></form>
```

POST using curl:

```
curl -d "userid=root" -d "passwd=fluffy"
   -d "submit=Login" -o output.html
   http://www.example.com/servlet/login.do
```



# Complex Script (Login to eBay)

```
curl -s -L -c cookies.txt -b cookies.txt -e ';auto'
   -o step-1.html http://www.ebay.com/

curl -s -L -c cookies.txt -b cookies.txt -e ';auto'
   -o step-2.html 'http://signin.ebay.com/ws/eBayISAPI.dll?SignIn'

curl -s -L -c cookies.txt -b cookies.txt -e ';auto'
   -o step-3.html
   -d MfcISAPICommand=SignInWelcome -d siteid=0 -d co_partnerId=2
   -d UsingSSL=1 -d ru= -d pp= -d pa1= -d pa2= -d pa3= -d i1=-1
   -d pageType=-1 -d rtmData= -d userid=MYUSER -d pass=MYPASS
'https://signin.ebay.com/ws/eBayISAPI.dll?
   co_partnerid=2&siteid=0&UsingSSL=1'

curl -s -L -c cookies.txt -b cookies.txt -e ';auto'
   -o step-4.html 'http://my.ebay.com/ws/eBayISAPI.dll?MyEbay'

grep MYUSER step-4.html
```

# Script Output

```
step [1 OK] [2 OK] [3 OK] [4 OK]
PASS: MYUSER appears 5 times in step-4.html
```

- My script does more than just make requests.
- See the potential for automation?
  - Smoke Tests
  - Test Setup for additional tests requiring logged in state



# Automated Security Test

Reflected Cross-Site Scripting

**Method**

- Read a list of XSS strings from a file
- Read a list of URLs to attack from a file
- Append each attack string to each URL
- Submit and record the output to a file
- Grep for attack string in output file

**xss-strings.txt**

```
<script>alert('xss');</script>
abc<xyz
abc'xyz
```

**urls.txt**

```
http://example.com/login.jsp?user=
http://example.com/cart.php?id=
```


# Feel the Automation

- Put scripts in the hands of developers
  - Not successful? Can't submit to QA!
- Put scripts into regression
  - Spot regression failures easily
- Hook into test frameworks
  - Use standardized output in your own scripts



Perl


In Six Slides

Get it from http://www.perl.com/

or http://www.activestate.com/

# Getting Started


Get Perl

- Active State (Windows)
- CPAN (http://www.cpan.org/)

Get a book or two

- *Learning Perl*
- *Programming Perl*
- special topics



# Basic Script to Fetch a Page

```
#!/usr/bin/perl
use LWP::UserAgent;
use HTTP::Request::Common qw(POST);

$UA   = LWP::UserAgent->new();
$req  = HTTP::Request->new( GET => "http://www.nova.org/" );
$resp = $UA->request($req);

# check for error. Print page if it's OK
if ( ( $resp->code() >= 200 ) && ( $resp->code() < 400 ) ) {
    print $resp->decoded_content;
} else {
    print "Error: " . $resp->status_line . "\n";
}
```

# Why Perl?

- Seems complicated
  - Could have been simpler
  - Not as simple as curl
  - Powerful, Flexible
- Make requests, think, make more requests



# Parse a Page

```
$UA   = LWP::UserAgent->new();
$req  = HTTP::Request->new( GET => "http://www.nova.org/" );
$resp = $UA->request($req);

my $p = HTML::Parser->new(api_version => 3,
    start_h => [\&viewstate_finder, "self,tagname,attr"],
    report_tags => [qw(input)] );
$p->parse($resp->content);
$p->eof;

print $main::viewstate . "\n" if $main::viewstate;

sub viewstate_finder {
    my($self, $tag, $attr) = @_;

    if (         $attr->{name} eq "__VIEWSTATE" )
        $main::viewstate = $attr->{value};
}
```

← Gimme a parser
← Call my func with name, value
← Only act on <input> tags
← Do it
← If <input>'s name is VIEWSTATE
← Set our global variable to the value

# POST a Request

```
#!/usr/bin/perl
use LWP::UserAgent;
use HTTP::Request::Common qw(POST);

$UA   = LWP::UserAgent->new();

$req = HTTP::Request::Common::POST( "$page",
   Content_Type => 'form-data',
   Content => [
    myFile => [
        "myfile.pdf",
       "myfile.pdf",
        "Content-Type" => "application/
    pdf" ],
    Submit => 'Upload File',
    FDesc  => 'My Test File',
    __VIEWSTATE => $main::viewstate;
    ]
   );

$resp = $UA->request($req);
```

- Add all the attributes in a map
- Post to the web site
- Read the response

 © 2007 Cigital, Inc. All Rights Reserved. 

# Upload a Malicious File

What does your application do when it receives a virus in a file upload?

This script uploads a file named Virus.jpg which is guaranteed to be considered a virus by your anti-virus software. It **ISN'T** a virus. It is a standard test file.

What does your app do when one minute the file is there, and the next minute it's gone (to the AV quarantine)?

```
#!/usr/bin/perl
use LWP::UserAgent;
use HTTP::Request::Common qw(POST);

$UA    = LWP::UserAgent->new();
$page  = "http://www.example.com/upload.aspx";
$EICAR = 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-".
   "STANDARD-ANTIVIRUS-TEST-FILE!$H+H*';

$req   = HTTP::Request::Common::POST( "$page",
   Content_Type => 'form-data',
   Content => [ myFile => [ undef,"Virus.jpg",
     "Content-Type" => "image/jpeg",
     "Content"      => $EICAR,
     ],
   Submit => 'Upload File',
   ] );

$resp = $UA->request($req);
```

 © 2007 Cigital, Inc. All Rights Reserved. 


# Examples of Perl's Strengths

- Read URLs and fetch variations
- Read pages for links and follow them
- Read dynamic content (e.g. sessions) and vary them
- Robust
  - Error handling
  - Pattern matching
  - File handling

Remember Curl eBay?

- VIEWSTATE would eliminate curl
- Lots of extra effort to
  - Store cookies
  - Follow redirects
  - Record intermediate pages




# Security Test Automation

- Security is about lots of different cases
- Use automation to get coverage
- Use programs to automate

# Further Information

- http://websecuritytesting.com/
- cUrl: http://curl.haxx.se/
- Perl: http://www.perl.com/
- O'Reilly titles:
  - Perl & LWP
  - Programming Perl (the Camel book)
  - Free: http://www.oreilly.com/openbook/webclient/




# About Security Testing

*The best time to plant an oak tree was twenty years ago. The next best time is now.*

—Ancient Proverb

Send me email!

paco@cigital.com

# Software Quality Group of New England

SQGNE is made possible by the support of our sponsors:






## Welcome to SQGNE's 15th season!

- An all-volunteer group with no membership dues!
- Supported entirely by our sponsors…
- Over 700+ members
- Monthly meetings - Sept to July on 2nd Wed of month
- E-mail list - contact John Pustaver **pustaver@ieee.org**
- SQGNE Web site: **www.swqual.com/sqgne/main.html**



## Volunteers / Hosts / Mission

| Volunteers | Our gracious Hosts |
|---|---|
| John Pustaver - Founder and Director | Paul Ratty (Sun) - room, copies, cookies |
| Steve Rakitin – Programs and web site | Tom Arakel (Sun) - room, copies, cookies |
| Gene Freyberger – Annual Survey | Margaret Shinkle (Sun) - room, copies, cookies |
| **We need more volunteers!!!** | Jack Guilderson (Sun) – A/V equipment |

SQGNE Mission

- To promote use of engineering and management techniques that lead to delivery of high quality software
- To disseminate concepts and techniques related to software quality engineering and software engineering process
- To provide a forum for discussion of concepts and techniques related to software quality engineering and the software engineering process
- To provide networking opportunities for software quality professionals



## ASQ Software Division

- Software Quality Live - for ASQ SW Div members…
- Software Quality Professional Journal www.asq.org/pub/sqp/
- CSQE Certification info at www.asq.org/software/getcertified
- SW Div info at www.asq.org/software




## SQGNE 2008-09 Schedule

| Speaker | Affiliation | Date | Topic |
|---|---|---|---|
| 1. Lou Cohen | None | 9/10/08 | Introduction to using Quality Function Deployment on Software Projects |
| 2. Brian LeSuer | Star Quality | 10/8/08 | A Survey of Test Automation Projects |
| 3. Howie Dow and Steve Rakitin | None | 11/12/08 | Estimating using Wideband Delphi Method - An interactive exercise |
| 4. Russ Ohanian | Tizor Systems | 12/10/08 | Integrating Agile into the Development Process |
| 5. Johanna Rothman | Rothman & Assoc. | 1/14/09 | Schedule Games |
| 6. Carol Perletz | None | 2/11/09 | The Nitty Gritty of QA Project Management |
| 7. Robin Goldsmith | GoPro Management | 3/11/09 | Testing the Untestable |
| 8. Paco Hope | Cigital Networks | 4/8/09 | Automating security testing of web apps using cURL and Perl |
| 9. Derek Kozikowski | None | 5/13/09 | Automated Functional Test Design |
| 10. Stan Wrobel | CSC | 6/10/09 | Test Tool - Make or Buy? |
| 11. Everyone | | 7/9/09 | Annual Hot Topics Night… |



## Tonight's Speaker…

**Automating security testing of web apps using cURL and Perl**
**Paco Hope, Cigital, Inc.**

Web applications are everywhere. Since many of them are exposed to un trusted networks, we must take security seriously and include it in our standard testing regimen. Fortunately, web applications submit readily to this sort of testing and we can follow a few straightforward techniques to test for the most common vulnerabilities. In this session you will learn about two tools: cURL and Perl and how you can use them to test for common security vulnerabilities.

cURL is a free program that helps us automate basic requests. Perl is a well-known programming language ideally suited for writing scripts that test web applications. We'll look at the basics of automating tests in both ways, and also explore some of the more complicated concerns that arise during automation: authentication, session state and parsing responses. We will see how a few simple command lines can generate hundreds or thousands of test cases. Given malicious inputs, we'll see how to use these tools to automate testing throughout an application. The techniques in this session apply regardless of whether your Web platform is Java EE, .NET or something custom. The techniques are also independent of whether your test platform is Windows, Mac OS X, Linux or Unix. You'll leave with an understanding of the basics and a long list of resources you can turn to for learning more about Web security test automation.

**Paco Hope** is a Technical Director with Cigital, Inc. and has 13 years of experience in web application security, software security and operating system security. He has focused on analyzing the security of web-based applications and embedded systems (lottery systems, cell phones, casino gaming devices, smart cards). He is a frequent speaker on software security, security testing, and web application security. He is co-author of two security books and is also co-chair of VERIFY, an international conference on software testing.