



They're Hacking Our Clients!

Why Are We Only Vuln Assessing Servers?

Jay Beale

Creator - Bastille UNIX

Co-Founder - Intelguardians, Inc.



A Curious Trend in Ethical Hacking

I'm working for Intelguardians, a security consulting firm. I get to do and lead a fair bit of network and application penetration testing.

A penetration test usually focuses on the compound question: could a "hacker" break in from the Internet and how far could he go?

The hard part is getting into the "internal" network.

Once you're inside, things get far, far easier.



A Curious Trend in Ethical Hacking

Over the last year, we're finding that breaching the perimeter and DMZ networks has gotten far more difficult under traditional attack models.

Increasingly, we've been getting to the internal network via client-side attack, hacking the Security or IT staff's workstations via vulnerabilities in their browsers, mail clients, Acrobat and Office programs.

These attacks have gotten easier for anyone with a copy of Core IMPACT, Metasploit, or hostile attacker toolkits.



Professional Hackers Started Years Ago

Real attackers moved to client-side attack years ago.

There's so much money in hacking the clients that it's become a great business for organized crime.

And it was so successful, the attackers' chief problem became creating a centrally-controlled, scalable means of controlling all the systems they've compromised.

And so they brought us the botnet.



Workstation Control is Powerful

Most botnet owner so many machines that they don't ever inventory them and figure out what companies and organizations they've compromised.

As highly-targeted attackers, penetration testing teams use these machines as a foothold to hit the internal organization. We get access to file shares, cached credentials, and applications that have never been designed or audited for security.

Further, even across their worldwide WAN, even the largest organizations have no filters.



Isn't this Social Engineering?

In the security community, we initially write off these attacks to social engineering. We blame the user.

Not all exploits require user interaction. And if they do, we'll always have some users get fooled. Even if that's 1/100 of 1%, it's bad.

But blaming the non-IT user isn't fair. Your grandmother shouldn't have to understand vulnerabilities to read e-mail. You can't expect her to unless you really make a driver's license for computing.

It's our responsibility as IT architects to train the user, but to protect them from attack anyway.

Think about your mortgage broker's computer? Or your dentists? High value target, no IT staff and little training. They've been owned.



Why is this difficult?

Most organizations' security has been focused primarily on the perimeter and on firewalls. That over-focus is decreasing, but only so fast.

Most security efforts are focused on the servers, particularly those accessible from the Internet.

This focus really has started to achieve its goal. Hacking organizations, from the Internet, through their servers, is finally getting difficult.

But the attackers have moved to attacking the workstation PCs. And few organizations have kept up with that change in focus. It's a difficult problem...



Why is this difficult?

First, the numbers are much tougher. As an attacker, I only need to find one workstation or laptop that has a vulnerable client out of the 10,000 you have.

And you thought protecting 150 servers was difficult!

Second, your users can stay disconnected from the network or have their machines powered off for extended periods, or even retire systems that get pulled out of retirement six months later when a position is filled again.

Patching has always been a race condition!



What about Patch Management?

The next thing we all think is...this is where patch management products should make the problem irrelevant.

But:

- 1) Not every organization has a commercial patch management tool.
- 2) Patch management tools may rely on a host inventory that isn't accurate. Here are some hosts commonly left off:
 - a) Old hosts that aren't part of the domain or inventory.
 - b) Dedicated scanning / machine control systems.
 - c) Hosts brought to the office from partner companies.
 - d) Legacy systems of any kind!
- 3) Patch mgmt tools rarely track patches for every third-party product.



The State of Internal Patching

Actually, most organizations don't patch consistently or frequently enough to avoid this threat.

Even if they can do consistent and frequent patching, they tend to only get there for Microsoft software.

Even those will have trouble keeping organization or user-installed browser plug-ins up to date.

Well if we're not solving this problem via patching, what about our regular vulnerability assessments?



Vulnerability Assessments

First, most organizations don't appear to do better than quarterly vulnerability assessments.

Second, the vulnerability assessments focus on the servers.

That's natural: servers actually answer you when you probe them and usually give you their version/patch level fairly easily.

Clients aren't quite so helpful...or are they?



Clients Identify Themselves Too

A whole lot of client-side software identifies itself often, if you only know to listen...or sniff...or read the logs...

First, web browsers tell every server they talk to what version they are:

```
HTTP_USER_AGENT = Mozilla/5.0  
(Macintosh; U; Intel Mac OS X;  
en-US; rv:1.8.1.4)  
Gecko/20070515 Firefox/2.0.0.4
```



Mail Clients Too...

Mail clients send their version string with every single message. I once had a friend e-mail me to tell me that my Thunderbird version was old and vulnerable.

Here's a string from an e-mail I got from another speaker here:

**User-Agent: Thunderbird 2.0.0.6
(Windows/20070728)**



Watching without Sniffing

If I stick to just watching all the browser user agent strings as people on my network browse, I could easily give you a list of vulnerable browsers.

But what if I don't want to sniff the network?

Many/most large organizations use transparent web proxies to decrease their Internet bandwidth costs - why download the same CNN graphics 2,000 times today?

Squid proxies, among others, easily log browser user agent strings. I can watch these for malware and vulnerable browsers.

Sendmail can be configured to log mail client user agent strings as well.



Sniffing

Alternatively, sniff the internal links to your outbound mail relays and outbound transparent web proxies.

But, either way, we're missing something in the browsers, aren't we?

Can anyone tell me what it is?



Browser Plugins!

Browser exploitability relies on third party code that may not even ship with the browser.

People add their own plug-ins, often automatically when they try to use a website that needed it.

They don't necessarily know they need to look for patches. They may not even know what site they got the plug-in from, since the site they were using sent them to the plug-in.

IT Departments can find it difficult to track these plug-ins, especially when they didn't install them!

Just to look at a couple examples...



Adobe Acrobat Reader

The Adobe Acrobat Reader Plugin hasn't been doing well lately:

Adobe Acrobat Reader Browser Plug-in for MSIE Malformed PDF
Request DoS Dec 27, 2006

Adobe Acrobat Reader Plugin for Microsoft IE Microsoft.XMLHTTP
ActiveX CLRF Injection Dec 27, 2006

Adobe Acrobat Reader Browser Plug-in PDF XSS Dec 27, 2006

Adobe Acrobat Reader Browser Plug-in PDF CSRF Dec 27,
2006

Adobe Acrobat Reader Browser Plug-in PDF Handling Memory
Corruption Dec 27, 2006



Macromedia Flash Plug-in

The Macromedia Flash Plugin hasn't had it easy either:

Adobe Macromedia Flash Player Plug-in Multiple Browser Remote Keystroke Disclosure
Apr 11, 2007

Macromedia Flash Flash8b.ocx Flash8b.AllowScriptAccess Method DoS Dec 29, 2006

Macromedia Flash Player swf Processing Multiple Unspecified Code Execution Mar 14, 2006

Macromedia Flash Player Flash.ocx ActionDefineFunction Function Arbitrary Code Execution
Nov 7, 2005

Macromedia Flash Player Flash.ocx Unspecified Function Arbitrary Code Execution
Nov 4, 2005



Detecting Plug-ins

Rsnake announced an excellent tool for this at Toorcon Seattle:

Master Reconnaissance Tool

Visit this URL to see what your browser's plug-ins are:

<http://ha.ckers.org/mr-t/>

Here are some of the highlights from my browser:

Java Embedding Plugin 0.9.6.2

Shockwave Flash 9.0 r28

QuickTime Plug-in 7.1.5

Move-Media-Player.plugin npmnqmp 07074032

JoostPlugin.plugin



Master Recon Tool

Rsnake announced an excellent tool for this at Toorcon Seattle:

Master Reconnaissance Tool

Visit this URL to see what your browser's plug-ins are:

<http://ha.ckers.org/mr-t/>

Here are some of the highlights from my browser:

Java Embedding Plugin 0.9.6.2

Shockwave Flash 9.0 r28

QuickTime Plug-in 7.1.5

Move-Media-Player.plugin npmngmp 07074032

JoostPlugin.plugin



What about Non-network Clients?

The applications most commonly targeted are browsers, mail clients, browser plug-ins, and Microsoft Office.

The PaulDotCom podcast recently highlighted the Metagoofill tool by Christian Martorella.

<http://www.edge-security.com/soft.php>

It searches a website in Google for public documents, including PDF, DOC, XLS, PPT, SDW, MDB, and SDC.

It then parses out metadata, including creator, creation time, and version of the client.



Metadata

If we can pull newly-saved/sent documents from file shares or sniff them on the wire, we can parse them for metadata.

If you just created this Word document five minutes ago, with a vulnerable version of Word, on a given system, that system's Word program is probably still vulnerable!



Looking Up Version Strings

So we've got version strings accessible to anyone who can read certain logs. You can get more if you sniff the network. And still more if you potentially inject a MR-T iframe in each person's browser once per day.

If you put it together with a nightly database update from OSVDB (<http://www.osvdb.org>), you've got client-side vulnerability assessment via a new tool we're building called ClientVA.

<http://www.clientva.org>

But we can go further than this - we can redirect some clients to a captive portal to patch.



Simple Client-side IPS?

If I see your browser ask the Squid cache for an external website while revealing that you have a vulnerable version, why not keep it off the Internet?

We're working on a Squid plug-in that can deny that single request right then and there, re-directing you to a captive portal with patches for your particular browser.

You can take this further. If you really want to win the race, deny the user access to their mailbox if their mail client is vulnerable. Send the next web request to a captive portal explaining what's going on.



Simple Client-side IPS?

If I see your browser ask the Squid cache for an external website while revealing that you have a vulnerable version, why not keep it off the Internet?

We're working on a Squid plug-in that can deny that single request right then and there, re-directing you to a captive portal with patches for your particular browser.

You can take this further. If you really want to win the race, deny the user access to their mailbox if their mail client is vulnerable. Send the next web request to a captive portal explaining what's going on.



About the Speaker

Jay Beale is a information security specialist, well known for his work on threat avoidance and mitigation technology. He's written two of the most popular security hardening tools: Bastille Unix, a system lockdown and audit tool that introduced a vital security-training component, and the Center for Internet Security's Unix Scoring Tool. Both are used worldwide throughout private industry and government. Through Bastille and his work with the Center, Jay has provided leadership in the Linux system hardening space, participating in efforts to set, audit, and implement standards for Linux/Unix security within industry and government. Jay also contributed to the OVAL project and the HoneyNet Project.

Jay has served as an invited speaker at a variety of conferences worldwide as well as government symposia. He's written for Information Security Magazine, SecurityFocus, and SecurityPortal. Jay has co-authored or edited nine books in the Information Security space. Six of these make up his Open Source Security Series, while two are technical works of fiction in the "Stealing the Network" series.

Jay makes his living as a security consultant with the firm Intelguardians, Inc, which he co-founded with other industry leaders Ed Skoudis, Mike Poor, Bob Hillery and Jim Alderson. His work in network and web application penetration testing, as well as security architecture review, allows him to maintain deep understanding of current threats and defenses. Prior to consulting, Jay served as the Security Team Director for MandrakeSoft, helping set company strategy, design security products, and pushing security into the then third largest retail Linux distribution.