Owning the Users
With "The Middler"

Jay Beale

Co-Founder, Intelguardians
Author, Bastille UNIX

Talk Agenda

I’m releasing The Middler now, an attack proxy tool to automate attacks on browsers and everything else using HTTP.

Here’s the Talk Agenda:

- The Attack Vector: Shared Networks
- Automatically exploiting mixed HTTP/HTTPS sites, including Gmail, LinkedIn and LiveJournal
- Launching non-interactive CSRF attacks on Online Banks
- Trojaning software installation and update
- Injecting browser-exploits and adding root certificates
- Protecting yourself on hostile LANs

HTTP and Shared Networks Don’t Mix

Most users use a tremendous number of shared networks when they leave their homes and offices.

- Hotels
- Non-security Conferences
- Coffee Shops / Bookstores
- Airplanes

Whether those are wireless or wired networks, they open themselves up to application-level monitoring and attack constantly.

Proxy Attacks

If we share a LAN, I can view and modify all of your traffic:

I can replace the real DHCP server on the network, setting my laptop up as your DNS server, DHCP server, and router.

OR

I can ARP spoof the real router and any local DNS servers.

Mixed HTTPS/HTTP Sites are a Menace

Many companies misunderstand that encrypting only their application's password form leaves their users very vulnerable to man in the middle attacks.

Before we demo an attack on this, let's look at how LinkedIn.com works.

If you start up your browser with https://www.linkedin.com, click on "Sign In," you'll be taken to this URL:

```
https://www.linkedin.com/secure/login?trk=hb_signin
```

Following sign-in, you're taken to this one:

```
http://www.linkedin.com/home
```

What if I change the URL?

You can change the URL to:

https://www.linkedin.com/home

…but clicking on any link will just take you right back to an HTTP URL!

Unless you modify your browser or surf with a special defensive proxy, you’ll constantly be pulling down cleartext links.

And all I have to do as an attacker is inject my own Javascript into a single one of those.

How Do I Attack This?

First, direct the client to my host with DNS, DHCP or ARP spoofing.

Second, pass the HTTPS traffic through unmodified, but:

1) Inject Javascript into the cleartext traffic.
2) Store session keys and send my own parallel requests.
3) Intercept logout requests.
4) Replace HTTPS links in any proxied pages with HTTP links.

Best of all, I'm releasing a tool right here to do this. It features a rich plug-in architecture to let other people add on handling for sites we're not including in this release.

DEMO: The Middler

Interactively, we can

- Clone session for the attacker by transparently using the same cookies and form parameters as the user.
- Inject Javascript into every HTML page
- Log the valid user’s session.

Let’s demonstrate these.

But the real power is in site-specific features, which we’ll demonstrate with:

- Gmail
- LiveJournal
- LinkedIn

Demo: Gmail

Once the Gmail session moves back into cleartext, we can:

- Read the user’s e-mail
- Read past GoogleTalk conversations
- Harvest the address book
- Send our own e-mails
- Profile the user in other Google applications
- Prevent a real logout, presenting the user with an actual logout

Demo: LiveJournal

Once the LiveJournal session moves back into cleartext, we can:

- Read the user’s private and friends-only journal entries
- Make the user’s private/friends-only entries public
- Harvest the friends list and those friends’ private profiles
- Add our own user as a friend

Demo: LinkedIn

Once the LinkedIn session moves back into cleartext, we can:

- Read the user’s full contact information
- Gather full contact information for their entire Network
- Read the user’s Inbox
- Add ourselves to their Network
- Place the user in our Network

Start with a CSRF Attack

Imagine a non-security friend on a hotel network…

He types the name of his online banking site into his browser:

http://www.bankofamerica.com

He's used to the bank protecting him from himself. The site reloads the page with an HTTPS version:

https://www.bankofamerica.com.

It's already too late. It's a race condition and he lost.

Race Condition

I have already served him my own index.html file for the HTTP site, which accomplishes the reload, but not before inserting its persistent window.

```
window.open("http://www.bankofamerica.com/mitm","mitm",'width=1,height=1,sc
   rollbars=0,menubar=0,toolbar=0,location=0,status=0');
window.blur("mitm");
document.location.href="https://www.bankofamerica.com";
```

While his primary browser window is no longer under my control, I can continue to serve my own version of the bank's website. From there, I'll wait for the user to log in to the main site, then begin CSRF attacks.

How do I know when he logs in?

Knowing When the User Logs In

First, remember that I'm proxying the user's traffic.

Even if I wasn't, from my persistent HTTP-provided window, I can read the browser history to see what links the user has visited.

If the victim has pop-blocking in place, I can even just inject Javascript into any HTTP-carried pages the user has open.

Knowing When the User Logs In

First, remember that I'm proxying the user's traffic.

Even if I wasn't, from my persistent HTTP-provided window, I can read the browser history to see what links the user has visited.

If the victim has pop-blocking in place, I can even just inject Javascript into any HTTP-carried pages the user has open.

Trojaning Software Installation

So far we’ve kept our eyes on web applications. But there’s more that happens over HTTP than that.

Several non-giant software vendors do software installation and update over HTTP, with no public key verification of the packages you’ll install.

Your system pulls down a page over HTTP that includes available update names, versions, locations, and sometimes MD5sums.

DEMO: Free Software Installation

While the large operating system vendors generally get this right, packaging their own PGP public keys with the original operating system, not everyone does.

The Middler has plug-ins to automate:

- Installer.app for the iPhone
- MacPorts (formerly DarwinPorts)

Let’s demo a trojan horse insertion on both my iPhone and my MacbookPro.

Exploiting Vulnerable Browsers

The Middler has one more attack feature.

As long as we’re able to inject HTML into a users’ browser windows, we can also serve up client-side attacks from Metasploit.

The Middler can insert Javascript to refresh the current page or a pop-under to an exploit that it serves. We could just take the exploit and serve it ourselves, but that’s not as easy to maintain.

DEMO: Exploiting a Browser

Let's demo this.

The user surfs to a page, then gets redirected to the captive portal and pop-ups.

We'll inject Javascript into the portal, which will redirect the browser to an exploit page.

Protecting Yourself at a Conference

What can you do to protect yourself at a conference?

You could try bringing your own Internet connection. EVDO/CDMA and HSPDA/GSM modems make this very difficult or at least reduce the attacker pool to people with the equipment and know-how.

If this isn’t an option, and even when it is, here’s what I like to do.

Recipe for a Safer LAN Experience

Here’s what I do on a hostile conference network:

1. Set up a dynamic port forwarding SSH tunnel
2. Ask for the DHCP server’s and router’s MAC address and IP addresses
3. Set my DNS servers to localhost or tunnel over SSH
4. Configure my firewall to allow outbound IP traffic only to the SSH tunnel host and the DHCP server.
5. Configure static MAC address (ARP table) entries for the DHCP server and router.

Here’s how that works.

Step 1: Dynamic SSH Port Forward

Thanks to Dan Kaminsky, who added this feature to OpenSSH years ago, your SSH client can function as a SOCKS proxy. Most network clients are SOCKS-aware, particularly browsers, mail clients, and instant messenger applications.

Just run this command and configure your network clients to use 127.0.0.1:8000 as a SOCKS5 proxy:

```
ssh -C -D8000 user@server
```

The network clients will forward SOCKS5 requests across the encrypted SSH tunnel to your SSH server. The default configuration of most SSH servers allows this automatically. You just need an account. You should put the server name in your /etc/hosts file or use an IP address.

Steps 2-5

Set your system’s firewall to only allow only this outbound traffic:

- Outbound DHCP traffic to udp/67 only to the known DHCP server IP and MAC address and only from udp/68.
- Outbound SSH traffic only to your SSH server and only on the port you’re running the SSH server on.
- Allow your machine to communicate only with the known MAC address of the router and DHCP server.
  - Use a static ARP mapping if your firewall doesn’t support this.

The Middler

The Middler attack proxy is Open Source, hosted at:

https://www.TheMiddler.com

Help us add more plug-ins!

Questions and Speaker Bio

Jay Beale created two well-known security tools, Bastille UNIX and the CIS Unix Scoring Tool, both of which are used throughout industry and government, and has served as an invited speaker at many industry and government conferences, a columnist for Information Security Magazine, SecurityPortal and SecurityFocus, and an author/editor on nine books, including those in his Open Source Security Series and the "Stealing the Network" series. Jay is a security consultant and managing partner at Intelguardians, where he gets to work with brilliant people on topics ranging from application penetration to virtual machine escape.