Intelguardians, Inc.

Cold-Boot Attacks for Penetration Testers*

*Spies, Disgruntled Employees, Suspicious Spouses, Left-Handed People, or even that Jared dude from the Subway commercials…

SANSFire 2008

Ed Skoudis and Tom Liston - Intelguardians, Inc.

Beware… Jared is after you!

- What if Jared, that dude from the Subway commercials, decided one day that he was really, ***really*** ticked off at your company
  - Note: Ed was fine with just ignoring the question of what your company might have done to have earned Jared's enmity…
  - Tom, however, believes it should be put to the floor for suggestions…

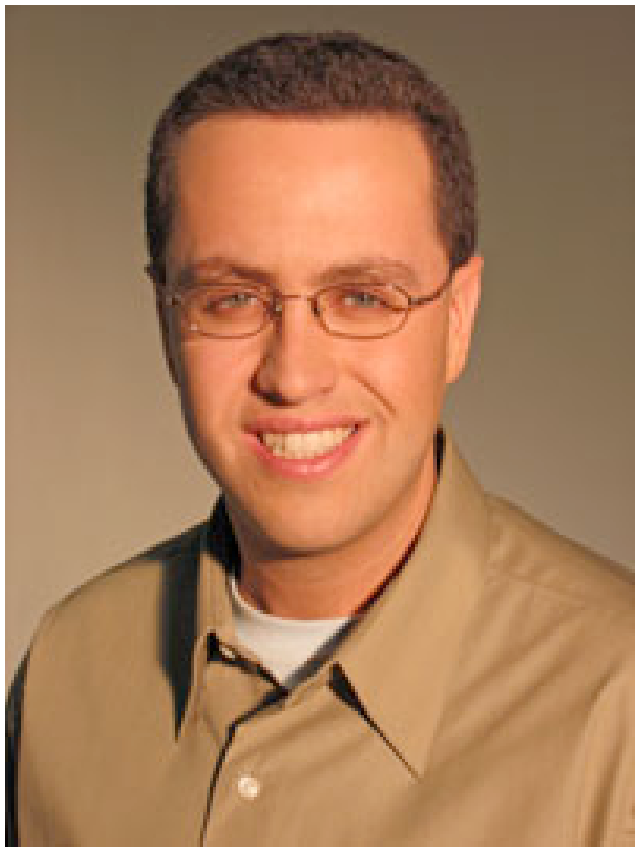
The Problem:

(And the obligatory Matrix reference… this is, after all, an *Ed Skoudis Presentation*™)

- Jared wants to mount some sort of technological attack against your company
- But, Jared has no 'leet skillz
  - Apparently his only truly marketable talent is that he's not overweight anymore…
- He does, however, have a whole "Matrix" thing goin' on in this photo, dontcha think?
  - "You eat the six inch sub and the story ends. You wake in your bed and believe whatever you want to believe. You eat the twelve inch sub and you stay in Wonderland and I show you how deep the rabbit-hole goes…"

Gaze into the face of Evil and be afraid…

- But Jared ain't givin' up…
  - He's read about this thing called "Memory Remanence" and its ugly stepchild, the "Cold-Boot" attack
  - No 'leet skillz required…
    - Or, at least, very few…



And so…

…armed only with a 4GB USB token and a size 60 pair of Levis, Jared begins his journey into the deep, dark, and transfat-free realm of the Cold Boot attack…

The Transfat-free Cold Boot Attack

- Jared's journey begins with a ground-breaking paper published on February 21, 2008
  - Researchers from Princeton University, EFF and Wind River Systems released a paper titled "Lest We Remember: Cold Boot Attacks on Encryption Keys"
  - Able to dump RAM from a system seconds to minutes after reboot
  - USB, network dumpers, specialized hardware
  - Temperature effects, etc…
  - Recovered encryption keys and broke full-disk encryption

Unfortunately…

- The paper is chock full of terms like:
  - Capacitive decay
  - MOSFET
  - Hamming distance
  - Liquid nitrogen
  - …and some **deep** mathematics and statistics…

- The paper is a little bit… uh… *"weighty"* for Jared.

But…

- Jared has that "heroic" (pun intended) experimenter's spirit that drove him to try eating only Subway sandwiches to lose weight…

- And so, despite all the big scary words in the original paper, Jared decides to just go ahead, try it, and see what happens…

Stuff I'm gonna need…

- Learn about memory "remanence"
- Get something to dump memory with
- Someplace to store the dumped memory
- Something to search through the dumped memory

- A 6" Turkey, Bacon, and Avocado on Whole Wheat
- What could I find?

Memory Remanence Essentials

- "Cold boot" – Is the term used to describe starting computer from a powered-down state
- Memory (a VERY simplified overview)
  - Acts much like a capacitor
  - If that capacitor is "charged", then bit it a 1, otherwise, a 0
    - Or, the other way around…
    - Don't ask… It isn't important… I just put that in there because I know, if I don't put it in, someone will complain…
  - Over time, the charge will "decay" therefore the charge must be "refreshed" every so often
  - Modern memory specifications set the refresh time on the order of a few milliseconds
  - BUT REMEMBER: That specified refresh time is designed to ensure 100% reliability…
- What do you do to your specifications when you need to ensure 100% reliability?

Overcompensatin'…

- Even if you remove power from memory for several seconds, it's probably going to continue to hold MOST of its information
  - **Remember:** the memory chip manufacturer's specs are aimed at ensuring *100%* reliability
  - So… a machine can be quickly rebooted while maintaining much of the contents of RAM
  - There is a statistical likelihood of memory corruption that increases with "un-powered" time
- In tests:
  - Machines powered off 5-10 seconds
  - Still recovered useful data from memory

But…

- …machines always ***CLEAR*** RAM on boot, don't they?
  - Uh… no…
  - Whatever gave you that idea?
  - They can (sometimes) be configured to do that…
    - But, it's a HUGE hit to boot time…

- …my OS will clear RAM when it boots, right?
  - Maybe… But who said ***YOUR*** OS was going to get to boot?

Stuff I'm gonna need…

- ✓ Learn about memory "remanence"
- Get something to dump memory with
  - Someplace to store the dumped memory
  - Something to search through the dumped memory
  - A 6" Turkey, Bacon, and Avocado on Whole Wheat
  - What could I find?

Takin' a dump…
The Definitive "How-to"

- Creating a memory dumper
  - It's not as easy as you might think!
- USB is the obvious choice
  - Size generally approximates memory
  - Small, portable
  - Bootable on most modern systems
  - Easy to use
  - Easy to understand
- Other options:
  - PXE boot, PCI add-in card, etc…

Takin' a dump…
The Definitive "How-to" (continued)

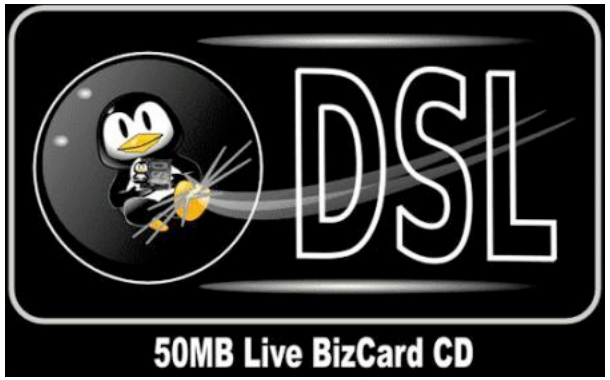
- The idea:
  - We're going to boot to different OS off of our USB key
  - Then, we'll simply run through RAM, and copy it out to our USB key
    - "Simply"
    - Ha!
  - Using a USB key kills two of Jared's problems with one stone: Dumping ***and*** Storing

Takin' a dump…
The Definitive "How-to" (continued)

- OS Requirements
  - We want to affect memory as little as possible
  - Cross platform (i.e. should work on most machines)
  - As *fast* as possible
    - Problem: USB 1.0 (BIOS supported) is SLOOOOW
    - USB 2.0 requires drivers
      - Platform specific issues…
  - We would like whatever we use to have decent development tools
  - Open Source would be really nice

Solution #1 - Damn Small Linux

- How:
  - Remastered standard distro
  - Shrunk ramdisk
  - Created an autodump script
  - Dump memory to a 2nd partition
- Pros:
  - Fast! Easy! Workable!
  - 32-bit programming environment
  - Flat memory
- Cons:
  - Huge memory hit (ok, maybe not "huge" but still…)

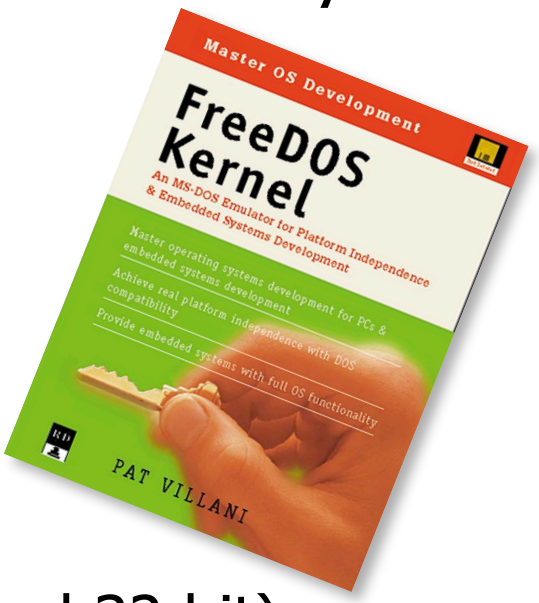


Solution #2 - msramdmp

- Based on EFF/Princeton paper
  - Written as a Syslinux "com" file
    - Somewhat like a DOS "com" file
- Written by Wesley McGrew - McGrew Security
  - **http://www.mcgrewsecurity.com**
- Pros:
  - Available!
  - Small memory hit (~500K)
- Cons:
  - Syslinux isn't an OS
    - No direct filesystem access
    - Writes out memory using raw writes to disk
    - No real development environment
  - USB 1.0 speeds

Solution #3 - FreeDOS

- *Check it out:* DOS only ***SEES*** 640k of memory!
- Wait… then how do we read it all?
- UNREAL mode
- Pros:
  - Absolutely ***minimal*** memory hit
  - Tools, tools, tools!!
- Cons
  - Difficult to code a dumper (mixed 16 and 32 bit)
    - UNREAL mode is cool, but coding in it SUCKS!
  - Painfully slow dumping
    - USB 1.0 dumping via the BIOS



An interlude…

For those of you in the audience who find it difficult to attribute malice to one of America's most beloved marketing mascots, we present the following slide, titled "Separated at Birth?"

Separated at Birth?

Ok… maybe not "brothers," but at least "cousins"…

Stuff I'm gonna need…

- ✓ Learn about memory "remanence"
- ✓ Get something to dump memory with
- ✓ Someplace to store the dumped memory
- Something to search through the dumped memory

- A 6" Turkey, Bacon, and Avocado on Whole Wheat
- What could I find?

Let's try this out…

- Jared considers all three solutions and decides on #2
  - McGrew Security's "msramdmp"
- He downloads the files, follows the instructions, and creates a bootable USB key
- He sets up a test machine at home and does a trial run
- "WTF! This thing takes 20 minutes to run!?!?!"

WTF! 20 Minutes!?!

- No disrespect to Wesley McGrew is intended here…
  - He did the best he could…
  - Think about it: How long does it normally take you to copy a couple of GBs to a hard drive?
  - Shoving a GB of data onto ANY storage medium is a SLOOOW process
    - Worse still, most BIOSs only support USB 1.0
  - There ain't no way around it…
  - Or… is there?

What if we combine some steps…?

- The next thing needed on Jared's checklist was: "Something to search through the dumped memory"
- Why are we insisting on making this a two-step process?
  1. Grab memory
  2. Search memory
- Jared isn't necessarily interested in grabbing memory
  - He's interested in actionable "stuff"

Physical Pentesting

- If you've ever been involved in physical security pentesting, the idea of sitting around for 20 minutes, waiting for a memory dump to finish sounds crazy
  - Physical pentests tend to be: get in, get your stuff, get out…
  - Any wasted time tends to get you caught
- Intelguardians has performed a number of physical security pentests, and we wanted to add Cold-Boot memory dumps to our "toolkit"
- But if it takes 20 minutes to dump RAM, it won't work… for us, or for Jared

A different approach…

- Thus far, most of the people investigating this technique have focused on dumping memory and then using "offline" tools on the dumps
- We're more interested in "operationalizing" Cold-Boot memory dumping

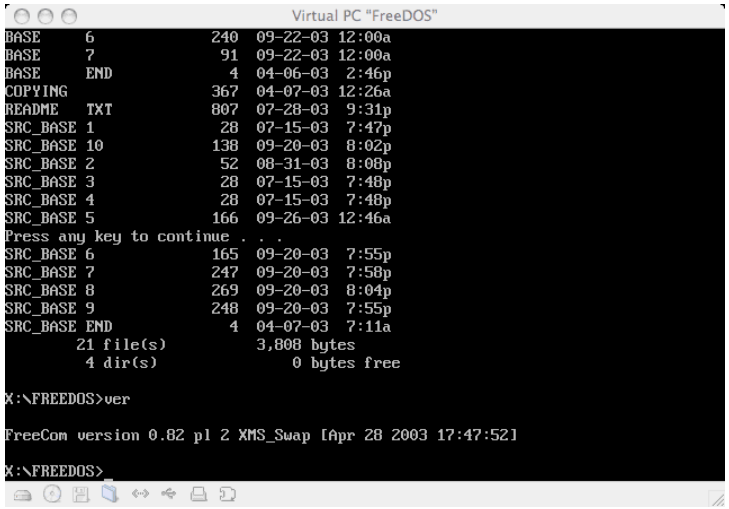
- So, we want to go back to our original three dumper "solutions," and reinvestigate them in light of this new approach
  - Realizing, of course, that "different approaches" don't ***always*** work…

A re-examined life…

- If we combine our "searching" tool with our "dumping" tool, we'll be writing much less data to the USB…
  - In theory, we'll only be writing out the data that we actually want
- That should makes "dump speed" less important
  - But it places FAR more emphasis on the ease of development for our combined dump/search tool

So, now the winner is…

- FreeDOS
  - While the syslinux approach used by msramdmp is workable and has a slightly smaller footprint, syslinux "com" programs have too many limitations



  - UNREAL mode ***is*** a pain to work in, but once you've got dumping working, everything else becomes much more straightforward
  - There are, literally, hundreds of programming tools available for DOS
  - Programming in DOS is kinda cool and nostalgic…

Therefore…

- Intelguardians is currently working on a FreeDOS based memory dumping / searching tool
  - The "dumping" portion, written in assembly language, interfaces with the search portion, written in C
- The goal:
  - Create an easy means of specifying "target data" to search for in memory
  - A "search language"

So… what are we looking for?

- This really gets us to the heart of the last item on Jared's list
  - The one after the sandwich…
- What CAN we find, using this technique?
  - So, let's see what we can find…
- Once we know what's available, that might point us in a good direction for the design of our "search language"

Another "interlude"

That last one was so much fun, how about if we try another?

Let's take a look at another, notable security person and see if we can find his "twin"

Other notable security people

and their twins…

"Stuff" we find in memory…

- Passwords!
  - Linux:
    - Truecrypt, GPG, sudo, ssh
      - Some appear to be associated with the shell
      - When we close the shell, they go away…
  - Windows:
    - Putty, IE stored passwords, AIM, Outlook
- Scraps of documents
  - Word, Excel, graphics, text, web-pages
- In short:
  - If you recently used it, viewed it, or typed it in, it probably left evidence behind in RAM

Wait a minute… you're finding PASSWORDS?!?!?

- Who's to blame!?!
  - Those pesky programmers!
    - They're always messing things up…
  - Yes, but it might not be the programmers you're thinking of…

Bitten…

- Sure, sometimes programmers just plain do the wrong thing…
- But sometimes they do the right thing and… well… their tools do this:

Optimize this…

```
void DoSecureOperation(void) {
  char pwd[64];
  if (GetPassword(pwd, sizeof(pwd))) {
    /* checking of password, secure operations, etc */
  }
  memset(pwd, 0, sizeof(pwd));
}
```

- This is a good example of how you *should* deal with passwords in code
- The problem:
  - Some compilers will look at this code and *OPTIMIZE OUT* the call to memset();
    - Hey… "pwd" isn't being referenced again… so that memset isn't needed….
  - BEWARE: Both GCC and MSVC will do stuff like this…

What about Jared?

- He wants more out of life than just unlimited opportunities to show his really big pants to small children…
- He WANTS to 0wn you!
- Getting an AIM password from your CEO's PA might be fun, but it ain't enough
- Jared wants more… Oh, so much more…

So… is there other, perhaps MORE interesting, "stuff" in memory?

- LSASS.EXE holds the current user's password in memory…
  - Normally, playing around with LSASS's memory would require Admin level access…
  - …but not if Windows has gone bye-bye and you're just looking at a memory dump…
  - Note: Even on low memory machines, LSASS.EXE is unlikely to ***EVER*** be swapped out of memory…

How do you find it?

- Well, once again, it's not as easy as you might think…
- It's not like you can just go looking through memory and hope to find it…
  - There are TONS of candidate strings in memory…
- So, where do we start?

Finding LSASS in memory

- We need to get some important information from the Kernel Process Control Region
  - This is found at a known location in virtual memory (0xffdff000), so we need to figure out where that is…
- This allows us to locate a pointer value called *KdVersionBlock* (offset 0x34)
- From this, we can locate another value called *PsActiveProcessHead'*
  - This points to a linked list of active processes in memory
  - We then run through that list, looking for a process with the name "lsass.exe"

LSASS Found!

- Once we've located information on the LSASS process itself, we can then dump specific portions of the process' virtual memory
- LSASS stores the currently logged in user's password near a fixed virtual memory address (OS version dependent…)
- So… we simply search through dumped memory near that location for some specific cues that point to the actual password location
  - It sounds complicated, but it can be automated

Tempis fugit

- The really cool thing is, this type of targeted search can be performed very, VERY quickly
  - Especially when compared to the time required to dump *ALL* memory
- *Targeting* our search turns these cold boot attacks into a tactical weapon for physical penetration testing
- Additionally, with a targeted attack like this, it's easy to judge success or failure
  - How do you know what you'll find in a full memory dump?
- Finally, the speed of this attack allows the results to be used *immediately*

Demo Time!

- Before we began tonight, we had a member of the audience create a password for an account on a WinXP virtual machine and log into that account
  - We don't know the password they set…
- We also invited them to play around a bit after logging in…
  - Start a few programs, play some Solitaire, you know… what people normally do under Windows…
- We're going to run our proof-of-concept LSASS password grabber on a memdump of that machine

Meanwhile, back at the ranch…

- Jared, still more than a bit ticked-off at your company, hatches an Evil Plan®©™
- It all begins with a visit to a branch location…

The Set Up…

Hello, youthful office manager, it is I, Jared, the Subway dude.

I'd like to speak with someone at your corporate headquarters about being a spokesman for your company.

The Hook

Oh my, a celebrity! And he's even sexier in person than on TV!

Yes sir, Mr. Jared, sir. I'll set up a conference call with headquarters. I'll be right back… Here, sit at my desk… just please don't touch my stapler…

For those of us who aren't Jared…

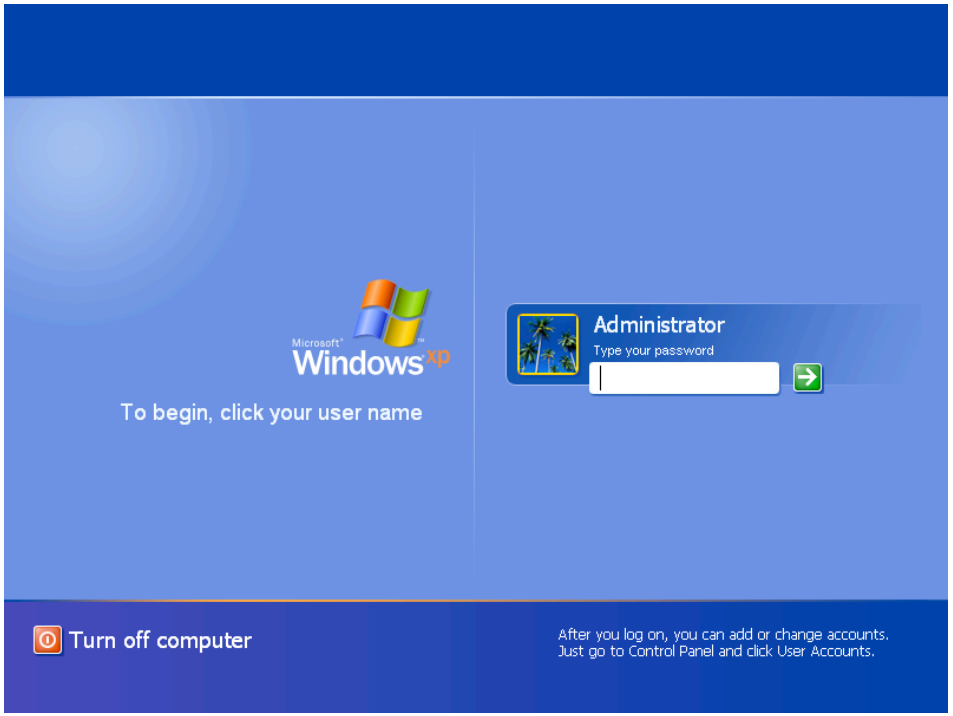
- This is a tactic that we use all the time when doing physical penetration tests
  - No, not "being Jared"…
  - Essentially, you walk in the door and make a request to get the "mark" working for you…
    - Pose as someone from another office, a delivery person, claim to have arranged a meeting with someone who isn't there…
    - In other words, create an "issue"
  - While they're off trying to straighten out the mess, we could:
    - Install a physical key logger on their machine
    - Plug in a wireless access point
    - Or, now, grab their username and password using a cold boot attack

The Sting

- While Milton is away, we momentarily drop power to the machine
  - In testing, we've found this works better and more reliably than "power buttons"
    - Most power buttons are "soft" switches anymore
    - Send a shut-down signal to the computer
- Plug in the USB device (you could plug it in before the reboot, but that might leave a registry entry in Windows… so, be quick!)
- Watch the boot screen for messages to press "ESC", "F10", or "Swing a dead cat over your head while pressing Ctrl-Shift-Alt-Esc-F10" to change the boot order if necessary…

The Big Con

- Even if Milton "screen-locked" his machine, he'll probably not notice he's been 0wned
- In this case, a screen locking policy actually works ***for*** the attacker…
- And if you're really worried about him noticing, log him back in and kick on the screen lock…



Yet another interlude…

Yet again, we find ourselves
pondering the mysteries
that caused another security
professional and his twin
to be separated at birth

Other notable security people

and their twins… (continued)

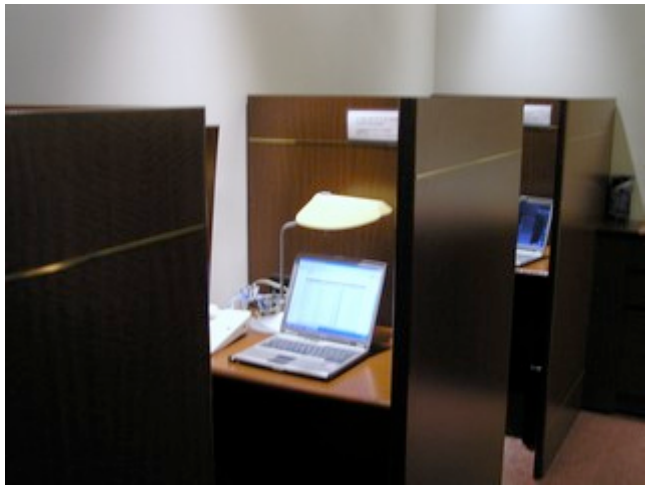
TRAVELOCITY

Jared Attack : Take 2

Jared stops by your corporate headquarters and, momentarily distracting your CEO with his size 60 Levis, grabs a laptop and walks out of the building

For those of us who aren't Jared…

- Laptops are a *huge* target
- Laptop lock usage is spotty at best
  - Don't believe me? Walk your offices…
  - In the physical pentests we've done, we've *ALWAYS* scored *at least* one laptop
    - More than a few desktop machines too…



- Carry a briefcase or a backpack (depending on the target) and a big, thick sponge
  - Shove the sponge between the screen and the keyboard to keep the laptop from closing completely and hibernating

For those of us who aren't Jared…

(continued)

- If it starts off screen locked, or screen locks during the interim
  - Cold boot the machine to grab the username / password
- MANY companies still have their laptop users (or, depressingly, *ALL* of their users) running as local Administrators
- Log back in, and have at it… the machine is all yours
- Better still, if you have a few extra moments on-site, simply install some fun "utilities" and put it back

But, remember, we said…

- …that we were designing the tool to allow it to search for arbitrary "stuff" in memory…
  - Not just the current user password…
  - That's just an example
  - Also, as we've seen, by giving our tool a way to target the memory of specific processes, we can further refine the types of information we target
- Ruminate a bit on that…
  - What else might be fun to search for?

Here's something fun

- Using a memdump from a WinXPSP2 system, we were able to extract the following:

Local Address	Remote Address	Pid
192.168.1.6:1035	209.85.201.109:995	2536
192.168.1.6:1044	213.251.133.91:80	3484
192.168.1.6:1040	64.233.189.99:443	3140
192.168.1.6:1043	4.23.51.123:80	1160

- The kernel keeps track of connection information

Something else that's kinda' fun…

- We can list open files for each process and if they're mapped into memory, we can dump them…

```
Process: wordpad.exe
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.10.0_x-ww_f7fb5805
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.10.0_x-ww_f7fb5805
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.10.0_x-ww_f7fb5805
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.10.0_x-ww_f7fb5805
File    \lsarpc
File    \ntsvcs
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.10.0_x-ww_f7fb5805
File    \srvsvc
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.10.0_x-ww_f7fb5805
File    \wkssvc
File    \svcctl
File    \DAV RPC SERVICE
File    \wkssvc
File    \samr
File    \Documents and Settings\Tom\My Documents\Secret.txt
```

What other stuff is in memory?

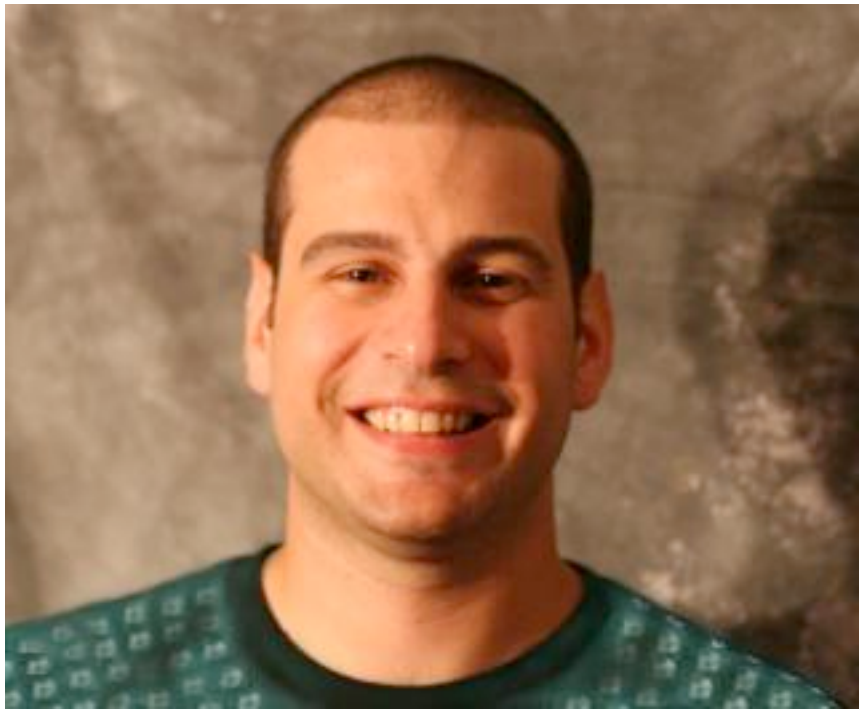
- While it's inconvenient as heck to spend 20+ minutes during a physical pentest doing a full memdump….
  - We would strongly suggest looking through a "test dump" of your own sometime to see what you can find…
  - You'll be *AMAZED*
  - It will also generate some interesting ideas for other uses of cold-boot attacks

One final interlude

The most uncanny resemblance of them all…

Other notable security people

and their twins… (continued)



Note: *Payback*, Mr. Asadoorian… - TL

Thank you!

- *Moderately difficult questions, niggles about spelling or grammar, harshly worded complaints:*

  ed@intelguardians.com

- *Effusive praise, offers of money, high-tech electronics or sexual favors:*

  tom@intelguardians.com

- *Difficult questions, hate mail, subpoenas, or death threats:*

  psw@pauldotcom.com

And, of course, thank you Jared

Every generation has it's pop culture phenomena…