Attacking a Network Protected by Continuous Monitoring

Marcus J. Ranum
(mjr@tenable.com)
For you twits: @mjranum

Senior Strategist

Who Am I?

- Innovated in firewall design in the late 80s
- Innovated in VPN design in the mid 90s
- Innovated in IDS design in the late 90s
- “Idea Guy” and analyst at Trusecure in the early ‘00s
- CSO and Strategist at Tenable in the mid ‘00s till today

Continuous Monitoring

- All systems and network monitored with:
  - Systems hardened and vulnerabilities tracked
  - Real-time endpoint activity monitoring
  - Edge-of-network activity monitoring (firewalls)
  - Network trace data collected
  - Centralized analysis and tracking of changes in status of monitored systems
  - Automated analysis on combined data from all sensors

Why Continuous Monitoring?

- Proactive
  - Figure out what is going on within your network to detect and prevent attacks in progress
- Reactive
  - Analyse and automate enterprise-wide response to attacks
  - Determine spread, severity, and correct reaction
- Retroactive
  - Look back at past records and apply present knowledge

Continuous Monitoring As An Ideal

- We want to avoid the “no true scotsman” dilemma:
  - Did you have continuous monitoring and still get hacked?
  - Then, obviously you weren’t doing continuous monitoring!!
- This is a practical approach and embeds within it the idea that things still go wrong

Things Go Wrong, and How!

- Per previous slide, we embed the notion that “things will go wrong” what kind of attacks will evolve?
- Broadly:
  - Innovative (they are that already)
  - Localized
  - Transient
  - Meta-Infrastructure-Oriented
  - Transitive Trust
  - Deep Lifecycle

In Other Words…

… These are the problems I am working on figuring out what to do about ***now*** because our customers will be bleeding over them in 10 years*

- The ones on the cutting edge, in 5-7 years

* I am not worried 20 years out b/c I will be retired or dead by then

Innovative

- We're up against this today
  - The attackers continue to innovate
  - Mostly, it's along the "one more bug" axis
  - Attacks against cryptography stacks are interesting but ultimately manageable with existing techniques
  - Our concern is what about something ***new***?
    - Secondary: is there something new enough to fall outside of the existing techniques?

Localized

- Traffic hugs close to compromised system
  - Much of the game in detecting malware is searching for command and control
  - What if the command and control nearly vanishes?
  - If I were a bad guy, today, I'd be working on more autonomous malware
  - Or self-organizing malware command networks
    - This is basic "defeat your enemy's strategy" stuff

Transient

- No writes to disk, no attempt to achieve persistence, perhaps residence in device firmware
  - Imagine a semi-autonomous piece of malware that achieves persistence through pervasive transient infection
  - Imagine further that it assesses 2 or 3 places to achieve persistence in case it needs to go dormant

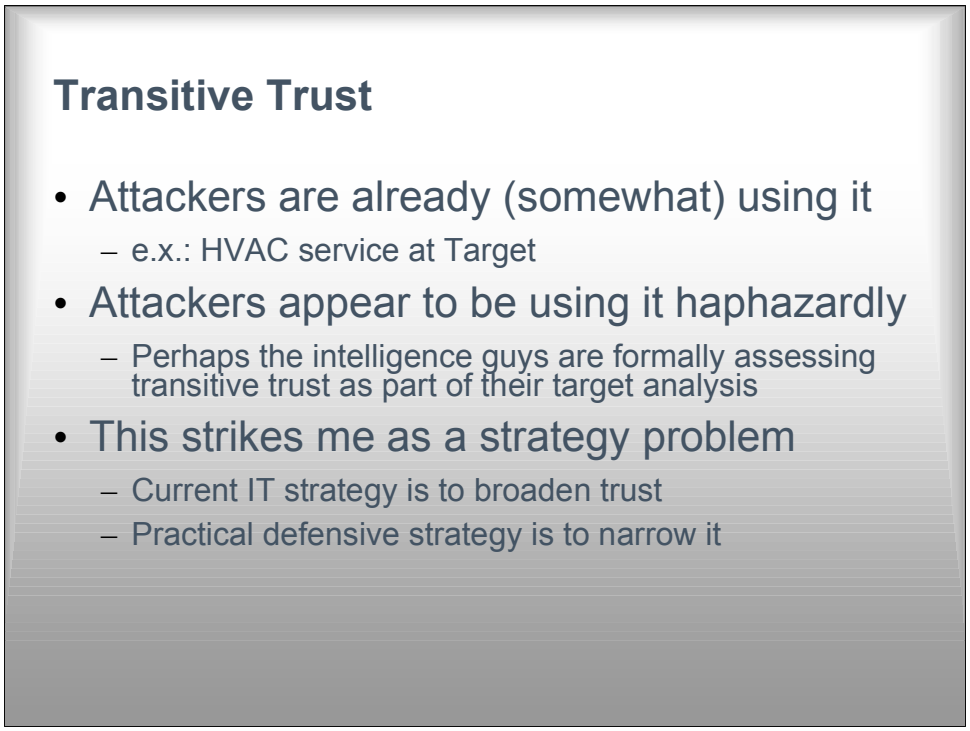
    e.x.: the antenna controller of your cell phone

Transient *(cont)*

- I am guessing that the authors of Stuxnet, Duqu, et al, are (deeply) regretting its persistence
  - As malware becomes more purpose-targeted we may see more transient attacks and reservoir-modal attacks
  - Attempting to persist gives too much away

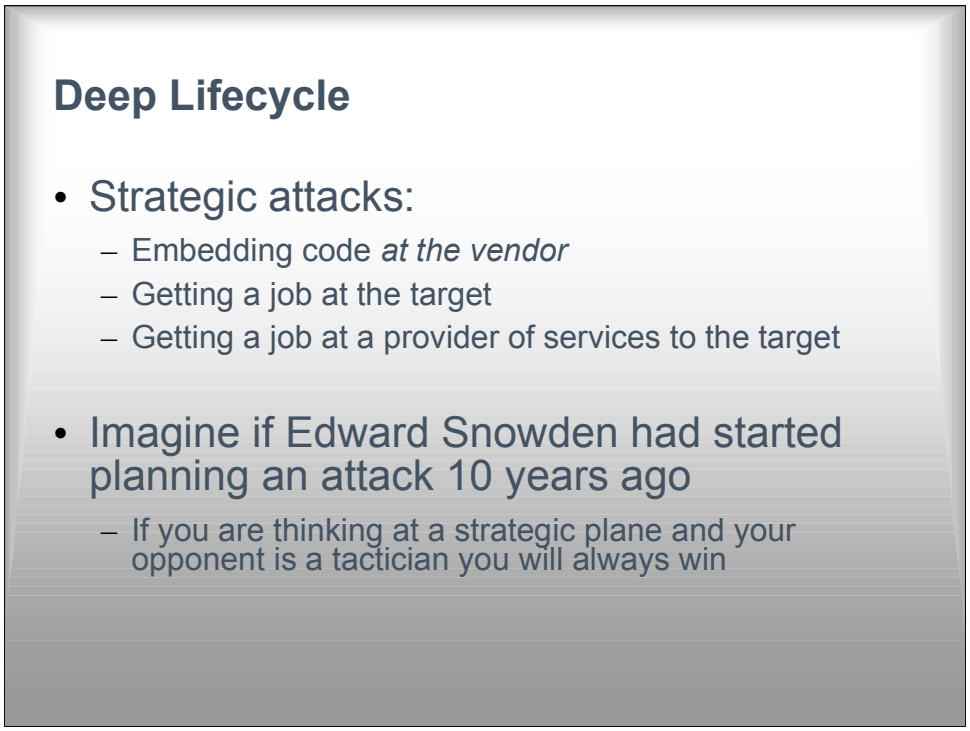
Meta-Infrastructure

- Going after CM, log servers, authentication servers, staging servers
  - Attempt to inject transient attacks into management infrastructure
- One odd and interesting thing: back in the day we *expected* attackers to try to shut off our log servers
  - They don't
  - My guess is: they haven't needed to



This is where I ran out of time.

Key point: transitive trust attacks have been incredibly successful in the rare instances where they have been used. What if the bad guys eventually level up their game and begin doing transitive trust analysis? The problem of trust boundaries has been largely ignored; what would most enterprises do if their providers were regularly being compromised in order to get at them? This is a huge potential problem for outsourcers (as Edward Snowden illustrates) -- imagine someone going after a system administrator at an outsourcer's home machine.

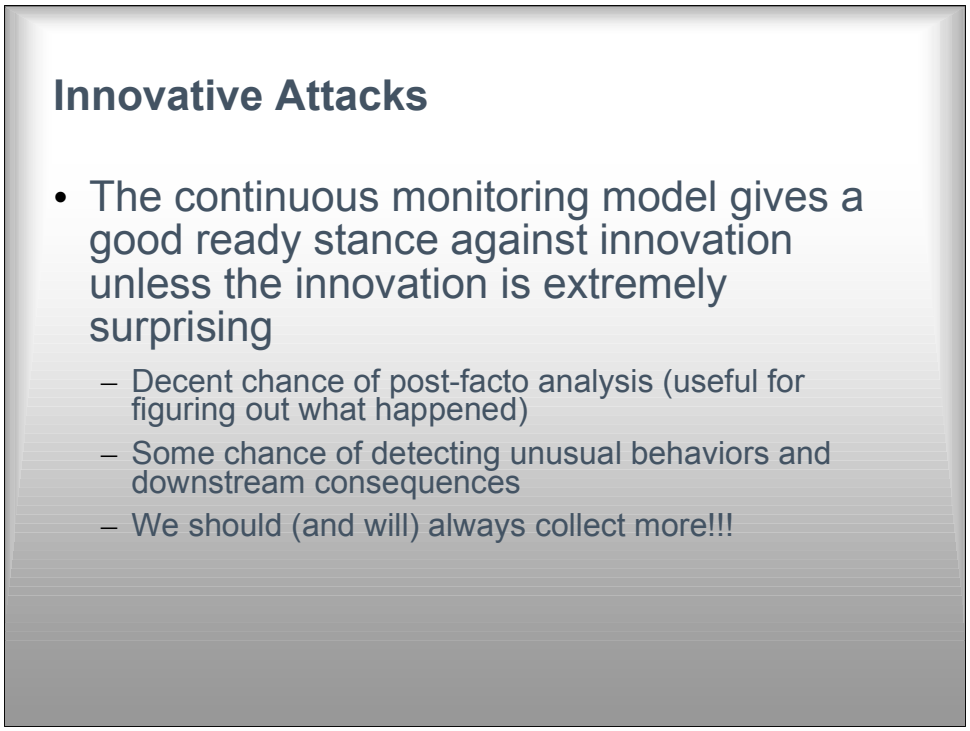


The KGB was devastatingly effective during the cold war, by embedding agents deep inside UK/USA agencies and letting them learn how the organizations worked, gain rank and access, and learn where the important information was prior to stealing it. Imagine if you had a software engineer who got a job working on the device driver for a graphics card or a network card used in a popular operating system! They could collect stock options and a decent salary while dropping a few mistakes in some code, then waiting 2-3 years for the code to propagate widely. Think about something like the merged Nvidia graphic card device driver pack - it contains code to talk to every kind of card, has kernel level access to memory, and the GPU, and is a great big hunk of code. A programmer who thought 5 years ahead could embed bugs that they could sell for $100,000 apiece eventually.

This same reasoning applies for cloud service providers, Google, Amazon, etc. What if someone got a job as an IT security auditor/pen tester with an eye toward dropping software that wouldn't activate for a year or two? What if someone got a job as a database administrator with plans to eventually sell the database and move to somewhere without an extradition treaty? Why steal credit cards and secrets when, ***if you plan ahead*** all you would have to do is collect them?

Responses

- So, how do we defeat these kind of threats with continuous monitoring?



Continuous Monitoring offers the potential for being able to go back and figure out what happened. Innovative attackers may still be able to get at you, but you'd be able to "burn" their attack technique and potentially attribute the attack, if you could go back through collected data and do historical analysis.

I am am huge fan of searching for downstream consequences of unauthorized activities. My favorite model is to ask myself what shouldn't happen, then "what would it look like if the thing that shouldn't happen, happened?" and then put something in place to detect that sequence of events. I call this a "burglar alarm" because it works the same way (and about as accurately!) as a burglar alarm. If I'm not home and the radio trigger in my gun cabinet signals that the cabinet has been opened, I don't know how someone got into my house but I know it's not me messing around in my gun cabinet when I'm not home and it's time to get very upset.

Localized Attacks

- Additional back-end analysis against data access and endpoint execution
  - Must defeat the attackers' strategy not their tools
- In the next 5 years we'll see more activity around file-level access analysis
  - The action will be where the data is being taken *from* not moved *to*

Transient Attacks

- This one is going to be hard!!
- Integrity monitoring of components (moving in that direction already)
- Close analysis of how data is moving
- Communications white-listing or never-before-seen detection
  - I am not sure what those would look like(!)
- Segregating networks into enclaves

Internal segmentation is going to be a big deal in the future. Being able to logically collect systems into groups (HR, Exec, office automation, software dev, etc) will allow more precise analysis of flows between groups. In an ideal world the segmentation would be using some kind of firewall - “current generation” (formerly known as “next generation”) firewalls like Fortinet and Palo Alto can map user activity to AD credentials. So why not have a firewall that logically isolates Exec Management desktops from software engineering and notifies about attempts to access their filesystems from a different enclave? This can be done purely at the level of analytics, or using the firewalls to actually perform segmentation. Guess which will work better.

Meta-Infrastructure

- Hysterisis
  - Detection of movement of state from established state (basically: change detection)
- Change control and configuration management
- Hardened internal critical infrastructures
- Hardened internal enclaves (e.g.: The Green Network)

Hysterisis is a very powerful analytic technique for network security. If you know how something is, assume its starting position is how you want it to be. Then notify someone if it changes from its starting position, and record the new position as the new 'correct position' and flag subsequent changes. Obviously, this will not work well on things like registry files that change constantly. But it works great on things like user group membership, file ownership, directory ownership, etc.

Change control/Configuration Management is a meta-hysterisis and compliance system. If you have a system that is supposed to be under CC/CM and suddenly it's the only system in your enterprise that has a different SHA-1 checksum for its keyboard driver, you *have* found malware. The great thing about CM systems is that a decent one will automatically stomp a machine back into compliance.

Consider the Sony breach: they had hundreds to thousands of desktops wiped. A reasonable implementation of a CM system (like Microsoft's SCCM) well-administered, can do remote operating system install; all the user would have to do is power-cycle their machine and bring it up on the network and 10 minutes later you have a shiny new Windows release. If your organization's approach to dealing with 1000+ downed desktops has the word "manual" anywhere in it, you're lunchmeat waiting to be sliced.

Transitive Trust

- Perhaps there is room for better analysis of trust in organizations
  - Mythical “Who do you love” charting tool would probably produce something that looked like a star map
  - Lumeta sort of tried this
- We can continue to look for second order effects

If you start to plot out the connections to your network, look for all the connections that are loosely firewalled (business partners, etc) versus tightly firewalled (internet) and add them to your map. Do this for two iterations. Your network will look like a gigantic puffball, or a bunch of plates of spaghetti that collided at high speed.

Deep Lifecycle

- This one scares me
- All we can do is detect second-order effects

You'll Notice Something...

- I say "look for second order effects" a lot
- That's because it's the most likely effective way of dealing with a future full of unknowns
  - To do this we will always be increasing the data we collect and the types of points we collect from
  - We will always be increasing the analysis we perform against the underlying data
  - Integrate in external data wherever we can ("why is Joe sysadmin accessing files our system has flagged as being created by HR?)"

The key point here is: "at least we've got the data" -- the people who have no data at all? What are they going to do?

Thank You