How To Catch A Hacker

An introduction to anomalous event
driven intrusion detection and prevention

SELL. SERVICE. MARKET. SUCCEED.

About Me

- Currently a Security Researcher at Salesforce.com on the Detection & Response team
  - Responsible for forensic analysis of incidents in our SF offices
  - Reversing malware and exploit kits
  - Developing analysis tools
  - Developing automated anomaly detection solutions
- Graduated from UAB in May of 2013 with a BS in CS
  - Specialized in Computer Forensics and Security
- As an Undergrad, worked in the UAB Computer Forensics Research Lab as a student researcher and malware analyst
  - Worked on spam analysis and clustering
  - Worked on automated phishing detection
  - Worked on the DARPA Malware Cyber Genome project

Roadmap

- Introduction To the OSI Stack
- The Hacker Mindset
- Attacks on the Stack
- Current Capabilities of IDS/IPS, AV, and other stuff
- Introduction To Statistics
- Statistical Anomaly Detection Techniques
- Statistical Applications on the Stack
- Scenarios and Demonstrations
- Advice
- Conclusions
- Q &A

The 7 Layers

- Layer 1 – Physical Layer
  - Cables, hubs, switches, etc.
  - Concerned with the transmission of raw bit streams from one medium to another
- Layer 2 – Data Link Layer
  - Frames (like an envelope)
  - Frame management – traffic control, sequencing, acknowledgment, delimiting, error checking, MAC addressing
  - Establishes and terminates the logical link between nodes

The 7 Layers cont.

- Layer 3 – The Network Layer
  - Packets (like the letter in the envelope)
  - Routing, Subnet traffic control, frame fragmentation, L to P mapping, subnet accounting
  - Controls the operations of the subnet, deciding what physical path the data takes
- Layer 4 – The Transport Layer
  - TCP, UDP, etc. (Like the mailman)
  - Message segmentation, message ack., message traffic control, session multiplexing
  - Ensures the messages are delivered error free, in sequence, and with no losses or duplication (Ideally)

The 7 Layers cont.

- Layer 5 – The Session Layer
  - Logical ports
  - Session establishment, maintenance, and termination
  - Allows session establishment between processes running on different stations
- Layer 6 – The Presentation Layer
  - Syntax
  - Character code translation, data conversion, compression, encryption
  - Formats the data to be presented at the application layer (layer 7).

The 7 Layers cont.

- Layer 7 – The Application Layer
  - End user layer
  - Resource sharing, remote file access, remote device access, directory services
  - Servers as the window for users and application processes to access network services

The Hacker Mindset

- Recon, Research, Exploit, Exfiltrate, Persist
- Given enough time and resources, anything becomes susceptible to exploitation
  - Nothing is “unhackable”
  - As long as a human interaction element exists in a security mechanism, it can be broken (social engineering, human error)
- Motivation is a big factor in deciding who to attack
  - What is gained by a successful breach?
    - Fame, money, knowledge…

Recon

- The goal is to gather as much information as possible
  - The more the merrier
- Information can consist of all kinds of things you wouldn't expect to be very useful
  - OS version and patch level (like Windows Vista SP1)
  - Running applications and their versions
  - Domain registration information
  - People that work in the target environment
- All of this information shortens the amount of time it takes to research the environment components to find a vuln.

Research

- Any serious attacker always does their homework
  - Reconstruct the target network for testing exploitation
- It helps to know all of the vulnerable components of a network
  - Re-infiltration might be needed later
- Future attacks on different networks become easier because of vulns in similar applications

Exploit

- Once some vulns are discovered, working exploits are developed to attack with
  - Buffer overflows, phishing attacks, SQL injection, etc.
- Thanks to tools like Metasploit, things like exploit dev become really easy
  - Shell code already written
  - Obfuscation tools built in
  - Reverse connection handling built in

Exfiltration

- The act of extracting information from a controlled environment
- Can be done in all kinds of creative ways
  - FTP, SCP, DNS, HTTP(S), etc.
- Data can be obfuscated so it isn't recognized while in transit

Persistence

- Once an attacker has what they are after, the last thing to do is set up persistence for future re-access of the network
  - For more information as it comes along
  - DDoS resource
  - Malware distribution
  - Phishing
- Can be in the form of malware, compromised user accounts, new user accounts, etc.

Attacks on the Stack

- Physical layer - Susceptible to malicious hardware
  - Pwn plugs, pineapple wifi, wire taps, etc.
- Current solutions
  - NAC via MAC
    - What if I change the MAC…
  - Physical access countermeasures (locks, cameras, etc.)
    - Lock picking and social engineering

Attacks cont.

- Data link layer – Susceptible to device driver exploitation
  - Buffer overflows, DoS attacks
- The Data link layer is primarily driven by device drivers that are responsible for making the hardware usable to software
- Current solutions
  - Better coding practices?...
    - Writing a device driver is hard enough already…so, not likely.
- On the bright side, device driver exploitation to gain remote access is an extremely hard task to accomplish
  - Don’t forget about the hacker motivation factor though!

Attacks cont.

- Network layer, Transport layer, and Session layer
  - Attack surface on these three is pretty broad
  - Similar attacks can be applied to each of them as well
    - APR poisoning, MITM, packet sniffing, etc.
- Current solutions
  - Firewalls, MAC filtering, IDS systems, etc.
    - A good firewall that is *properly* configured is hard to beat, but not impossible (If there's an open port, it could be used in unexpected ways)
    - MAC filtering…macchanger ftw
    - IDS systems – mostly signature based, not very smart though. (CRC checks, packet ordering)

Attacks cont.

- Presentation Layer and Application layer
  - These two layers go hand-in-hand, especially in attack surface
  - Attack surface here is very broad as well
    - Encoding error attacks
      - XSS, CSRF, etc.
    - Injection attacks
      - RFI, LFI, SQL, etc.
- Current solutions
  - Application filtering, proper character escaping, secure coding frameworks, etc.
  - This is a very field as well

Defense Mechanisms

- Most of the systems that are employed for security are signature-centric systems
  - AV is especially dependant on signatures which is why they can't detect new malware
- IDSs aren't too bad. They can do a good job of protecting from mediocre threats, but they are still signature dependent
  - Based on blacklists of URLs, User-Agent strings, and packet content signatures
- Signature dependencies mean that if you can hide the attack behind a dynamic obfuscation technique, you won't be detected

Defense Mechanisms cont.

- IPSs are progressing but not fast enough
- An IPS can be implemented at the Host level or the Network level
  - Host level protects the host from things executing on a machine
  - Network level protects users from badness on the internet by killing the connection
  - IDS can be implemented at the Host level too
- The main concern with implementing a good IPS is False Positive rate
  - A detection in IPS land means that something is going to be blocked from doing what it needs to
  - This can be really bad if it's an FP
  - Signatures have a very low FP and can be easily remediated if detected as an FP - however, it's hard to redesign an algorithm to accurately allow what it thought was bad
- FPs on IDSs can have bad side effects as well
  - Less trust in the system (like the boy who cried wolf)
  - Consumes Incident Response Team's time too much to deal with real problems

A New Perspective

- Catching a hacker in a network is like looking for a needle in a haystack
  - That proverbial haystack can be on the order of a few terabytes of data as well
- Rather than looking through all that mess for a something that matches some picture of a needle, let's take a step back and look at it from a different angle

A New Perspective cont.

- You know you are looking for a needle in a haystack, but what if you don't know what a needle looks like?
  - Pretend you don't know what hay looks like either
- This will allow for the problem to almost sort itself out
  - Essentially group everything that looks the same into piles and look at the end result
  - There will be a huge heap of hay, and a few things that you don't recognize
- This is what the power of statistics can do for us

Intro to Statistics

- The most commonly used statistical measurement is mean (or average)
- The mean of some set of data tells us what that data mostly consists of
  - In our thought experiment with hay and needles, it would tell us that most of the data is hay
- The measurement of the variation from the mean is known as standard deviation
  - This would tell us that, among all this hay, you can expect ± x number of needles (or things that aren't hay)

Statistics cont.

- The next step that can be taken on the needle/hay example is by using those statistical measurements on the actual characteristics of the hay itself
- Since we (presumably) have a giant heap of hay, analyze the characteristics of each piece of hay
  - The weight, length, RBG of it's color, etc.
- Next, take those values from each piece and generate the averages and std. devs of those values
  - For example, the average weight of a piece of hay is 5 grams ± 2 grams, it's 6 inches long ± 1.5 inches on average, and it has an average RGB value of 16,113,284 (0xF5DE84) ± 4096 (0x1000)
- These features can now be used as a classifier for hay
  - With the classifier, you can reprocess all the hay and find anything that isn't hay

Applying Statistics to the Stack

- The same idea can be extrapolated to the various layers of the OSI Stack
- To monitor any particular layer of the stack, just collect data at that point and build a “haystack”
- Once enough data has been collected, it can be analyzed to see what kind of data is present and, of that data, what can be safely ignored
  - For example, assume you have a giant log of frame data, and you want to know if there are any frames that are trying to cause a buffer overflow on some device driver
    - Survey the average frame sizes and look for anything that falls outside the scope of “proper” frame sizes

More Application Examples

- Port scanning
  - Survey the network and see how many internal systems make connections other internal systems
  - Also see how many ports a system connects to
  - If you see the average is that an internal system connects to only one or two other internal systems on only a couple of ports, then the machine that is connecting to 100 other internal systems and hitting thousands of ports is very suspicious
- This idea also points to another aspect to look at in monitoring activity – request rates and session lengths
  - Scanning could be observed through very fast connections to other systems in very short intervals
  - So could fuzzing attempts at web applications
- The possibilities for these techniques are only limited by how many different angles one can make when looking at the data

Profiling

- Profiling can be done on many different levels
  - Machine, User, Application, Network, etc.
- Each Model developed through profiling has it's own pros and cons
- One of the hardest problems in profiling is implementation
  - Should you profile a user's network activity from their machine, or from a network monitoring tool like Moloch
    - With Moloch, you have to track the IP through DHCP logs to make sure you profile that particular user
    - Host based solutions can be hard to develop and it gives a user the chance to circumvent the monitoring

Advantages of Profiling

- Suppose you have a host based solution for monitoring the behavior of applications that commonly run on the machine
  - Internet browser, Word, Adobe, etc.
- If you detect that one of your applications (like Adobe) just started allocating ridiculous amounts of memory, it could indicate that the user just opened a malicious pdf and it is trying to exploit a heap vuln.
  - This would give you the power to raise an alert on the suspicious activity – or even block the execution by killing the process

Demonstration

What to do when you're hacked

- Don't panic
  - If you panic, you lose objectivity on the severity of the compromise
- Identify all the machines that have been compromised
  - Use the IOCs that caused the initial alert to check the rest of the environment
- The appropriate actions for clean up have a lot of variables – there's no such thing as one size fits all in incident response
  - When do machines get removed from the network
  - When should upper management be informed
  - Should the press or customers be notified
  - etc.

What to do cont.

- It’s not a great idea to try and lure the hackers in your network to a honeypot to gather more intel
  - You get distracted from the IR
  - Errors can open up more vulns
- I personally recommend the passive intel gathering on threats by getting them out of your networks first and then looking at the IOCs
  - Use the IOCs they leave behind to profile their skill level
  - Track them with the IOCs they leave behind to see who they are targeting
  - Use OSInt on the IOCs to find out who they are

Conclusions

- As verbose as technologies have gotten, there is more attack surface on the stack than ever
  - But also more features for data mining and learning behaviors
- Signature-based detection is growing more obsolete by the day
  - Fun fact – < 30% of AVs can detect a new malware sample that is a variant of an older sample. 0% can detect something totally new like the Black POS malware
- You can't prevent what you can't detect, so get better at detecting and prevention will follow

Questions??

Thank You

Contact Info

Tommy Stallings

Tommy.stallings@salesforce.com

@The_An0maly