Defeating Extrusion Detection

matasano
About matasano

• An Indie Product and Services Security Firm: Founded Q1’05, Chicago and NYC.

• Research:
  – Hardware Virtualized Root-Kits
  – Endpoint Agent vulnerabilities
  – Windows Vista (on contract to msft)
  – Firefox (on contract to Mozilla)
  – Storage Area Networks (broke Netapp)
  – A Protocol debugger
  – 40+ pending advisories
The Problem of Info Leaks

- Privacy Rights Clearinghouse** cites more than 150 million personal records leaked in incidents between 2005-2007.

- Unintentional leakage
  - Boston.com employees wrap newspapers with paper found in recycling bin. Papers contained customer records.

- Data theft
  - July 5th 2007: A senior database administrator at payment firm Certegy Check Services secretly copies 2.3 million records containing bank-account and credit-card information and sold it to marketing firms.

- ** Much more at:
  - http://www.privacyrights.org/ar/ChronDataBreaches.htm
Goals of Extrusion Detection

- Identify sensitive data and stop it from leaving the enterprise.
- Implement monitors between enterprise workstations and the “outside world”.
- Gather forensic data associated with alerts.
- May block illegal transactions based on alerts to achieve “prevention”.
- Be secure and resistant to attack, evasion, and tampering.
Types of E-D Solutions

- **Network Based Solutions**
  - Think NIDS in reverse
  - Worst case: tcpdump | strings | grep
  - Best case: Wireshark | file_format_decoder | grep
  - Force Multiplier
  - Not effective against workstation -> external storage

- **Agent Based**
  - Think HIDS in reverse
  - Monitoring agents on each workstation
  - Some products wear the policy enforcement hat
  - Local I/O as well as network traffic

- **Hybrids**
  - Combines elements of Network and Agent based solutions to “leverage the strengths of each” (and may expose you to problems of both).
Why We’re Here

- We reversed and audited numerous DLP products
  - “4-8” Products
  - Commercially released
  - Mainstream, market-leading
  - Mostly endpoint-based

- Several vulnerabilities found in all of them
  - No product emerged completely unscathed
  - Serious problems in every one
What We Found

• **Not evasion attacks**
  – Take evasion as a given. All of these systems can be evaded
  – Like the IDS problem, but the target is you

• **Real Vulnerabilities:**
  – Compromise of sensitive information
  – Agent takeover attacks
  – Management console takeover

• **Installing a bad ED product can be like:**
  – Installing a latent botnet on your network
  – Creating an open file share with your most sensitive data on it
What We Can Tell You

- [www.matasano.com/log/mtso/ethics](http://www.matasano.com/log/mtso/ethics)
  - Can’t disclose vulnerabilities that don’t have patches
  - Can’t violate NDAs

- Rationalize: you don’t care about the specifics
  - You haven’t operationalized these products yet
  - The individual vulnerabilities will get fixed

- We want you to know what questions to ask your vendor before you deploy a data loss prevention botnet file share

- So we did something a little different:
Introducing: PlugBoy

- PlugBoy 0.6.6.6
- “Cutting-Edge” imaginary Extrusion Detection from the minds at Matasano
- Agent-Based Extrusion Detection Solution
- Plug your leaky information dyke…. TODAY
- Combining the best (read: worst) of all the market leaders out there
PlugBoy: Our Made Up ED System
PlugBoy Agent

- Agents installed on every workstation
  - Spreads privileges of components between Kernel (ring-0), SYSTEM (ring-1), and desktop agents running as user.

- Responsible for:
  - Data interception
  - Extrusion Detection
  - Reporting.
  - Can wear the IPS hat - blocking extrusion

- Catches forensic data included in alerts
  - Actual copies of IP and confidential data that trigger alerts
  - Useful for investigation, prosecution, and/or disciplinary action
Agent Security Issues

- **Agents Are Scary.**
  - Common-codebase/common-binary
  - Homogenous installs on thousands of machines
  - Complex communication patterns
    - Agent-server
    - Server-agent
    - “Push” v “Pull”
  - Sensitive functionality
    - Software update
    - OS queries

- **ED Agents Are Scarier**
  - You can’t ask Windows to feed you credit card numbers; you have to hack the kernel to do it or run something in user-land with extensive privileges.
  - Many bugs in ED Agent code are ring-0 kernel or LOCAL_SYSTEM. Game over. Worse than losing “Administrator”.
Agent Questions

- How much of the agent is in-kernel?
- How much runs with escalated privileges?
- How does the server talk to the agent?
- Can the server update the agent’s software?
- Do the agents broadcast their presence?
PlugBoy Data Intercept Engine

- Monitors and intercepts I/O
  - Network, USB, peripherals, files, clipboard, screenshots, etc.

- Decodes file formats and network protocols.

- Passes content to Detection Engine

- May also block extrusion based on Detection Engine
  - Think IPS vs. IDS
PlugBoy Data Intercept Vulnerability

- Decodes AIM/OSCAR protocol in kernel
- FLAP/SNAC headers with bogus length: integer overflow.
- Anyone who can create a direct IM session with a machine running the agent owns the kernel.
- Any software installed on the machine can bust the kernel by making fake IM connections.
Data Intercept Questions

- What file formats do you handle?
  - To what depth?
  - Just regexing streams? Trivially evadable even by uninitiated.
  - Full parse? Good luck with integer overflows.

- Are file formats parsed in-kernel? Which ones and how?

- What archive formats do you unpack?
  - What are the specific version numbers of the unpacking libraries you use: extremely common vulnerability!

- Do you install browser “helpers” that can monitor data inside SSL sessions?
  - Does your chain of custody from that point on comply with HIPAA?

- What protocols do you parse?
  - To what depth?

- Where do you intercept network traffic?
Data Intercept Evasion

- **Encryption**
  - ED may even want you to hobble your enterprise encryption standards. (hint: Don’t tell your SOX/PCI/COBIT auditors)

- **Conversion, compression, archiving**
  - UUENCODE, Base64, EBCDIC, ZIP, ARC, LHARC, DMG
  - Roll your own format with extra sneaky sauce.

- **Format mangling**
  - What will the parser do with a mangled word doc?

- **Combine and Nest**
  - “Something” is bound to break.
Data Intercept Questions

- **E-D and Encryption Are At Odds**
  - There is no good way for E-D systems to “look inside” of PGP.
  - If not, how does PlugBoy handle keys, pass-phrases, and cleartext?

- **What file formats the PlugBoy engine understand?**
  - Can it handle N-number nested formats?
    - Mixed?
  - How well tested are PlugBoy’s parsing routines?
PlugBoy Detection Engine

- Receives data from the Intercept Engine

- Scans data against known REGEX patterns for sensitive data.
  - Example: SSN’s look like…
    \d{3}[- ]?\d{2}[- ]?\d{4}
  - … which matches …
    “123-45-6789”,
    “321 54 9876”,
    or “987654321”

- On match, sends Extrusion alert with forensic data
  - False positives are a big problem. Patterns must balance coverage and specificity.
PlugBoy Detection Evasion

- **Evasion is trivial**
  - Attacker controls both origination and destination.
  - Possibilities are endless. Unlike IDS evasion, your target is you!

- **Use encryption**
  - Probably don’t even need “good” encryption.

- **Or just absurdly simple obfuscation.**
  - Search and replace every digit uniquely. Reverse on the receiver.

- **Add stego to really mess with ED.**
  - How many SSNs can you fit in a GIF?

- **Add fragmentation if you wear tinfoil**
  - (or just for kicks).
Detection Questions

• How customizable is the pattern matching?
  – Can you at least see the rules under the hood?
  – Can you add rules?

• What pattern matching engine is used? (EBNF, PCRE, GLOB, etc.)
  – Does your pattern matching syntax offer you enough granularity and flexibility (like PCRE)?
  – EBNF and Turing complete detection languages imply high overhead.
    • Haven’t yet seen any in R/L but vendors are beginning to tout them.
  – Will the engine crack under high load?
PlugBoy Server

- **Server initiates to agent**
  - Heartbeat monitoring

- **Pushes configuration changes**
  - Detection Policies
  - Blocking Policies
  - Detection filters

- **Agent software push**
  - New versions of software!
PlugBoy Reporting Protocol
How PlugBoy Reporting Works

- Agent initiates to Server
  - Not authenticated

- Uses a proprietary message protocol
  - Binary format with alert/event information fields
  - Consists of header, then data segment
  - Data Segment Compressed with ZLib
  - Base64 Encoded

- Event types include:
  - Heartbeats
  - Administrative activity logs
  - Extrusion Incident
  - Extrusion Forensic Updates
Protocol Reversing

- Sniffing and a hex editor reveals all!
- 90% Educated guessing/Trial and error
- Scripting language of choice for protocol implementation and attacks
- Blackbag for prototyping and attacks at the network
PlugBoy’s Raw Reporting Frames

- Base 64 decoded first
- **Msg Header:**
  - PBOY msg name
  - Msg type: 2 (0x000002)
  - Version 0.6.6.6
  - Data length: 129 (0x000081)

- **Message data ???**
  - ZLib header and adler32 tail
  - Some quick ruby to decompress
  - Or use blackbag’s “deezee”

```ruby
#!/usr/bin/env ruby
require 'zlib'
zs = Zlib::Inflate.new
buf = STDIN.read
STDOUT.write( zs.inflate( buf ) )
```
Reporting Frame In the Clear

- Decompressed message to reverse
- Back to protocol decoding until we can make sense of fields
- Transmit forged alerts with BlackBag (or socat/netcat/...
PlugBoy Reporting Vulnerabilities

- No authentication
- No encryption

“Being an agent” lets you:
  - Generate arbitrary events (malicious ones)
  - Ends up in SQL without authentication: Injection
• **How is the protocol authenticated?**
  – None by design?
  – Windows Domain Credentials?
  – Windows MACHINE Credentials?
  – Public Key
  – SSL

• **Is the protocol encrypted?**
  – Yes?
    • How are keys handled?
      • Hard-coded keys?
    – Or just obfuscated?

• **What operations does the protocol support?**
PlugBoy Forensics Storage
How PlugBoy Forensics Works

- Detailed logs associated with alerts by ID.

- Individual alerts can have “secondary” alerts that convey more information.

- Information can include inferred username, OS information, network location, along with full file snapshots.

- Administrators get access to forensics through the web interface and through SQL.
PlugBoy Forensics Vulnerabilities

- **Follow-on alerts can alter or manipulate forensics!**
  - Violates chain of custody; anybody who can spoof an alert can erase previous events.
  - Forge malicious logs (in conjunction with event spoofing)

- **The server is a store of nothing but confidential data**
  - Read access == tons of juicy data from past alerts.

- **Forensic data is vulnerable to tampering or destruction while in agent’s queue.**
  - Endpoint agents are on “the honor system”.
Forensics Storage Questions

• How is the agent authenticated for forensics pushing?

• Can forensics be “updated” (read: overwritten)?
  – Is forensics “quarantined” and revision controlled as soon as it arrives?

• Is forensics queued by the agent if the server is unavailable?
  – If so: What mechanisms are used to protect queued forensics?
  – If Not: What happens to alerts when server’s down?

• Complies with PCI, COBIT, SOX, etc., organization encryption policies?
PlugBoy Admin Interface

- Web based management interface
- Reports back ended by SQL
- Uses windows-integrated authentication.
- Allows admin to open and view forensics files associated with events
Plugboy UI Vulnerabilities

• Alerts include forensic detail, such as snapshots of files with credit card info.

• This detail is rendered as HTML in the admin interface.

• Because the input didn’t come through an HTML form, nobody thought to scrub it for tags.

• Attackers can seize control of admin logins through XSS “submarined” in spoofed data loss alerts.
Admin UI Questions

- Has the web interface been audited?
  - Who did the audit?
- At what points in the UI is input filtered?
  - Alerts
  - Logs
  - Form fields
  - OS version information (common!)
- What classes of information are output filtered?
- Does the UI launch file viewers?
  - Are they hardcoded into the program?
  - How does the vendor deal with malicious files?
- All the classic web app questions
  - Authentication, traversal, SQL Injection, XSS, CSRF, and on...
  - Even if the UI isn’t a browser, does it use HTTP?
Conclusions

- Extrusion Detection products tackle the wrong problem
  - Trying to hold onto sensitive info after it's already in the wrong hands.

- Worst case ED vulnerabilities can undermine other security controls

- Evasion is often trivial. The simplest attacks are the most likely to succeed

- The answer to leakage is definitely not just monitoring
Not a Complete Loss?

• Seriously... the most effective ways to prevent info leaks are still:
  – Well designed access controls
  – Sane information gathering and retention policies
  – Strong encryption!

• But... ED is still not a complete loss:
  – It’s really good at catching accidents, lazy hackers, and stupidity

• If and when vendors fix their most gruesome holes:
  – Buy Extrusion Detection
  – Cite your use of it in your PR on privacy protections
  – Maybe actually catch some accidental leakage