Stay out of the kitchen: A DLP Security Bake-off

BlackHat USA 2014
Introduction

Zach Lanier
Sr. Security Researcher

Kelly Lum
Security Engineer
Introduction

• Our research is on-going and results herein are not exhaustive

• Note the “security” qualifier before “bake-off” — this isn’t just a feature comparison

• Read: we totally went down the rabbit hole of bug hunting, not bypass hunting
Agenda

- DLP overview
- Targets/products in scope
- Components breakdown
- Assessment criteria/Methodology
- Findings (by target)
- Conclusion / Q&A
DLP Overview
What is DLP?

- “Data Loss/Leakage Prevention”
- Identify “sensitive stuff”, keep it from leaving the company
- Various approaches:
  - Network monitoring/sniffing
  - Endpoint agent
    - Real-time monitor
    - Filesystem/DB/CMS/etc. crawler
Why DLP?

- Used to be a hot-button topic
- Panacea to solve all data leakage woes
- “Keeps honest people from doing dumb things”
- Data breaches and “files falling off the back of a digital truck” spurred DLP
Why WE chose to look at DLP

- Curious about attack surface, reliability, etc.
  - Like other security products, DLP agents/appliances often have high privileges or are “ideally” situated (i.e. see all the traffic)
- Testing the “security of security products” is always interesting
  - Big vendor buys small vendor, integrates then shelves them... meaning security is often overlooked
Previous Research

- A bunch of blog posts and whitepapers by Securosis
- “Defeating DLP”, Matasano, BlackHat USA 2007
- “Gone in 60 Minutes: Stealing Sensitive Data from Thousands of Systems Simultaneously with OpenDLP”, Andrew Gavin, DEFCON 19
- Many others...
DLP workflow example - Trend Micro

Step 1: Define Data Identifiers
- Create Fingerprints
- Create Keywords
- Create Expressions
- Select File Attributes

Step 2: Create Template
- Define Rules from Data Identifiers

Step 3: (A) Create Policy
- Identify Target
- Select Channels
- Select Template
- Specify Actions

or

Step 3: (B) Data Discovery
- Data Discovery
- Schedule Scan

Step 4: Enforce Policy
- Deploy Policy

Step 5: Monitor
- Logs
- Reports
Rule creation example - Trend Micro DLP
Targets
<table>
<thead>
<tr>
<th>Vendor</th>
<th>Product</th>
<th>Version</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trend Micro</td>
<td>DLP Management Appliance</td>
<td>5.6</td>
<td>Linux</td>
</tr>
<tr>
<td></td>
<td>DLP Endpoint Agent</td>
<td>5.6</td>
<td>Windows</td>
</tr>
<tr>
<td>Sophos</td>
<td>Astaro UTM Appliance</td>
<td>9.201</td>
<td>Linux</td>
</tr>
<tr>
<td></td>
<td>Sophos Enterprise Console</td>
<td>5.2.1r2</td>
<td>Windows</td>
</tr>
<tr>
<td></td>
<td>Sophos Endpoint Security</td>
<td>N/A</td>
<td>Windows</td>
</tr>
<tr>
<td>Websense</td>
<td>TRITON Management Server</td>
<td>7.8.3</td>
<td>Windows</td>
</tr>
<tr>
<td></td>
<td>Data Protector Endpoint Agent</td>
<td>7.8.3</td>
<td>Windows, Linux, OS X</td>
</tr>
<tr>
<td></td>
<td>Data Security Protector Appliance</td>
<td>7.8.3</td>
<td>Linux</td>
</tr>
<tr>
<td>OpenDLP</td>
<td>OpenDLP</td>
<td>0.5.1</td>
<td>Linux</td>
</tr>
</tbody>
</table>
Components Breakdown
Trend Micro

- Windows endpoint agent - monitoring and policy enforcement on client machines
  - Acts like a “legitimate” rootkit and hides itself
- Network agent - virtual appliance; monitors network traffic
- Remote crawler - for digital assets on machines not on corporate network
- Management server - Linux-based virtual appliance
Websense

- TRITON management server - unified management console; Apache Coyote on Windows, backed by MSSQL DB
- Windows, OS X, Linux endpoint agents
  - File and network drivers
  - Can also monitor clipboard operations
- Linux-based “Protector” appliance
  - Restricted “admin” shell
- Crawler agents can index/identify sensitive documents
Sophos

- Enterprise Management Console - fat/native, Windows-based unified management console
  - Whole lotta .NET...
- Sophos endpoint security - antivirus + DLP + ... (Windows, OS X, Linux)
OpenDLP

- Typically Linux virtual appliance
- Apache + a lot of Perl
- Windows agent
- File system crawler and document parser (PCRE-based)
- SSHFS-based crawler
- And some Metasploit modules (wtf?)
On the ubiquity of KeyView...

- “kvoop” binary (“KeyView OOP”) showed up a lot
  - Part of KeyView Filter SDK, used for parsing and normalizing various data and document formats
- Used in numerous DLP products, messaging servers, and "big data" platforms
- e.g. “EPClassifier” in Websense spawns kvoop processes to handle documents
Assessment Criteria/Methodology
<table>
<thead>
<tr>
<th>Target</th>
<th>Component</th>
<th>Test(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Appliance</td>
<td>Parsers (docs and configuration)</td>
<td>Invalid/mangled files</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol analysis; crypto/signing</td>
</tr>
<tr>
<td></td>
<td>Update/Deployment mechanism</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operating System</td>
<td>Configuration auditing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hardening practices</td>
</tr>
<tr>
<td>Endpoints/Agents</td>
<td>Parsers (docs and configuration)</td>
<td>Invalid/mangled files</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol analysis; crypto/signing</td>
</tr>
<tr>
<td></td>
<td>Update/Deployment mechanism</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Drivers and Services</td>
<td>Hardening practices/config</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fizzing (i.e. IOCTLs, network, etc.)</td>
</tr>
<tr>
<td>Management Server</td>
<td>Web Server/Web App</td>
<td>LOL OWASP TOP 10</td>
</tr>
<tr>
<td></td>
<td>Database</td>
<td>Configuration auditing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sensitive data storage</td>
</tr>
<tr>
<td></td>
<td>Operating System</td>
<td>Configuration auditing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hardening practices</td>
</tr>
</tbody>
</table>
Findings
• Little to no hardening on (Linux) appliances
  ▪ Many services run as root
  ▪ Lack of exploit mitigations
• Highly privileged endpoint agent software out of the box (root, LOCALSYSTEM)
• General absence of security best practices
  ▪ Comms encryption, webappsec101, etc.
• Occasional bug inheritance (e.g. OpenSSL!)
Findings - Trend Micro
Trend Micro - XSS
Trend Micro - CSRF
Encryption would have been a good idea
Findings - Sophos
Sophos: What we didn’t find

- Majority of code implemented in .NET
- Utilizes most of the MS core libraries, which means:
  - DB best practices
  - Contextualized Input/Output
  - Standardized Encryption Libraries
Findings - Sophos Astaro UTM
Not a whole lot...

- Most services chroot’ed (eh...), drop privs
- Web app fairly clean (just a few really low impact “issues”)
- Tight network- and login-access control restrictions
Findings - OpenDLP
OpenDLP - CSRF

Manage existing scan profiles

- Scan profile name
- Edit
- Delete

Options:
- Create New Profile
- Manage Profiles
- Regular Expressions
- Scans
- Metasploit
- False Positives
- Logs
- OpenDLP Homepage
Findings - Websense
Websense Protector & Endpoint Agent - RCE + Privesc

- Websense DLP policy objects include keywords, regexes, etc.
- Regex entries are actually Python pickled objects
- TRITON management server encrypts, bundles policies/files, pushes to agents and appliances
- Local admin on TRITON server could replace “.pic” file with custom pickled objects...

```
s.sh-3.2# cat fileTypes.pic
(dp1
I0
S'Unknown_FMT'
p2
sI1
S'AES_Multiplus_Comm_FMT'
p3
sI2
S'ASCII_Text_FMT'
p4
sI3
S'MSDOS_Batch_File_FMT'
p5
sI4
S'Applix_Alis_FMT'
```
Because my video didn’t work out...

Our crappy pickle POC; after overwriting a “legitimate” policy file

Reverse shell from Protector after policy update
A note on DLP bypasses
“Is your objective to improve security, or make your quarterly targets?”
-@snowcrashmike

- Defenses add weaknesses
- Caveat emptor
- Every new piece of infrastructure is additional attack surface
- Security companies should know better
- If a scanner can find it, what’s your excuse?
- Know what/who you’re defending against
- An advanced insider probably has own abilities
Questions?

zach@duosecurity.com  
@quine

george.sims@jukt-micronics.com  
@aloria