Intel (R) Regimented Potential Incident Examination Report: An information gathering windows framework

Steve Mancini
Joe Schwendt
2006 FIRST Conference
R.P.I.E.R.?

Regimented Potential Incident Examination Report
What’s in a Name?

RAPIER vs RPIER

Intel (R) RPIER is the name of the official GPL release of the tool.

So please erase the A from all your presentations when you get home so we don’t get fired. 😊
Introduction to RPIER

RPIER is..

RPIER is a modular incident response framework designed to acquire commonly requested information during an internal event, incident, or investigation in an easy, consistent manner.

RPIER is not..

RPIER is not a forensics tool.

It does not honor most industry guidelines for a proper forensics examination with regard to not affecting the image or files upon the system.

*RPIER was a way for a unix guy (Steve) to gather windows data in the environment.*
Attribution

Jesse Kornblum

FRED
First Responders
Evidence Disk
Purpose for RPIER

- The worst time to learn how to acquire information from a system is *during the incident*.
- Common reaction to an event is to patch, run AV scanners, spyware scanners, automatic OS updater, etc to get it working condition as soon as possible.
- Not everyone
  - (1) knows how to acquire the requested information nor
  - (2) do they acquire it in the same fashion
Incident Handling BKMs

- Introduce a limited number of decisions by the 1st responder that could result in differing results
- Automate where possible to free up incident handler’s focus for bigger event issues
- Provide a complete lifecycle for information gathering from start to delivery of data
- Expedite the acquisition of information since time is of the essence
- Comprehend all data that could be requested by analysts and gathers it during 1st execution of the tool
RPIER: Work Flow

Download → Update → Select → Execute → Upload → Notify → Analysis
RPIER Features

- Stand Alone
- Modular Design
- Fully configurable GUI
- SHA1 verification checksums
- Auto-update functionality
- Results can be auto-zipped
- Auto-uploaded to central repository
- Email Notification when results are received
- 2 Quick Scan Modes – Fast/Slow
- Separated output for faster analysis
- Pre/Post run changes report
- Fully automatable via command line and conf file
- Process affinity throttling
Command Line Arguments

- Data Bundling Options
- Program Execution Priority
- Email Header Information
- Path Definitions
- Webservice URLs
- Integrity Check options
- And a whole lot more...
Under the Hood:

RPIER Architecture
RPIER Requirements

- Windows NT* based Operating System
- Microsoft .NET* Framework 1.1+
- Microsoft WSH* (Windows Scripting Host) 5.6+
- Microsoft WMI* (Windows Management Interface) 1.5+
Engine Operational Flow - Launch

- Load RPIER.Conf file
- Interpret command line options
- Auto Update check (Optional)
- Auto Update if necessary (Optional)
- Restart EXE (if updated)
- Load Modules
- Display GUI (Optional)
Engine Operational Flow - Execute

- Pre-Run Forensics Checkpoint (Optional)
- Run Each Selected Module
- Compress results (Optional)
- Upload results (Optional)
- Post-Run Forensics Checkpoint and Differential Analysis (Optional)
- Send Email Notification (Optional)
RPIER Networking

- Uses the http (optionally https) protocol for all communication
- Port is configurable (non-port 80 is recommended)
- Webserver can be IIS or Apache on Windows
- Multiple servers can be setup for redundancy/load balancing
- Enables the following features:
  - RPIER distribution
  - Auto-update functionality
  - Auto-upload functionality
  - Central Results Repository
  - Central Documentation Resource (Manual/Training/FAQ)
  - Manual RPIER upload and non-RPIER upload
Gathering Information

RPIER Modules
RPIER Module Architecture

- Based on VBScript
- RPIER.vbi is a large library of VBScript functions to reference
- Modules can have individual conf files to allow for end user configuration
- Modules are stand alone
  - Can be added/removed at will
  - Allows for independent development/testing
# Feature Module Output

<table>
<thead>
<tr>
<th>Volatile Information</th>
<th>Static Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ complete list of running processes</td>
<td>□ System Name</td>
</tr>
<tr>
<td>□ locations of those processes on disk</td>
<td>□ System Startup Commands</td>
</tr>
<tr>
<td>□ ports those processes are using</td>
<td>□ Copies of application cache (temporary internet files)</td>
</tr>
<tr>
<td>□ Net (start/share/user/file/session)</td>
<td>□ Uptime</td>
</tr>
<tr>
<td>□ <em>Layer3 traffic samples</em></td>
<td>□ Local account and policy information</td>
</tr>
<tr>
<td>□ Output from nbtstat and netstat</td>
<td>□ List of all files with alternate data streams</td>
</tr>
<tr>
<td>□ <em>Dump memory for all running processes</em></td>
<td>□ Capture list of services installed on the system</td>
</tr>
<tr>
<td>□ Checksums for all running processes</td>
<td>□ Discover files marked as hidden</td>
</tr>
<tr>
<td>□ Capture last Modify/Access/Create times for designated areas</td>
<td>□ Export entire registry</td>
</tr>
<tr>
<td>□ Document all open shares/exports on system</td>
<td>□ Current patches installed on system</td>
</tr>
<tr>
<td>□ All files that are currently open</td>
<td>□ Current AV versions</td>
</tr>
<tr>
<td>□ Capture current routing tables</td>
<td>□ List of all installed software on system (known to registry)</td>
</tr>
<tr>
<td>□ <em>All DLLS currently loaded and their checksum</em></td>
<td>□ Capture all logs (system + application specific)</td>
</tr>
<tr>
<td>□ capture logged in users</td>
<td>□ MAC address</td>
</tr>
<tr>
<td>□ list of all network connections</td>
<td>□ Search/retrieve files based on search criteria.</td>
</tr>
</tbody>
</table>
Output

- Output is stored in directory path: SystemName\DATE\TIME\n
- Format: ASCII text
How to Interpret the Results

To teach you this would require several months (years?) of training and education in operating systems internals, hacking techniques, malware behavior, etc.

Ultimately, the results must be reviewed by people with sufficient knowledge of your environment to be able to discern the odd from the routine.
Start Demo Here
Over the Horizon

Where do we go from here?

- Validate on VISTA
- *NIX. Ask us after the talk...
- More Modules! (of course)
- Alternate output formats
- Program to parse output for interesting results
Release of the Tool

https://sourceforge.net/projects/riper/

Build Notes:

☐ Certain modules rely upon licensed software, or on tools we could not get permission to bundle with a GPL license.

☐ We’ve made it as easy as possible – acquire these on your own and drop into Module folders to get them working.
Contributions & Feedback

Have an idea for module?
Have code ready to drop into a module we don’t already have?
Have ideas how to improve it?

Contact us:

RPIER.securitytool@gmail.com
http://groups.google.com/group/rpiер
Questions?

Questions are guaranteed in life; Answers aren't.
Caveat

The opinions expressed in this presentation are those of the authors and may not reflect the opinions of our employer.