Exploiting Network Printers

Jens Müller, Vladislav Mladenov,
Juraj Somorovsky, Jörg Schwenk

http://www.nds.rub.de/
Why printers?

PRINTERS

PRINTERS EVERYWHERE
Evolution

1987

2017
Yet another T in the IoT?
Contributions

- Systematization of printer attacks
- Evaluation of 20 printer models
- PRinter Exploitation Toolkit (PRET)
- Novel attacks beyond printers
- New research directions
Overview

1. Background
2. Attacks
3. Evaluation
4. PRET
5. Beyond printers
6. Countermeasures
How to print?

1. Printing channel (USB, network, ...)
2. Printer language (PJL, PostScript, ...)

6
What to attack?

Printer

Printing Unit

PJL Interpreter

PostScript Interpreter

Further Interpreter (PCL, PDF, ...)

USB

RAW

IPP

LPD

SMB
• Printer Job Language

• Manages settings like output tray or paper size

@PJL SET PAPER=A4
@PJL SET COPIES=10
@PJL ENTER LANGUAGE=POSTSCRIPT

• NOT limited to the current print job
PostScript

- Heavily used on laser printers
- Turing complete language
Overview

1. Background
2. Attacks
3. Evaluation
4. PRET
5. Beyond printers
6. Countermeasures
Attacker model: Physical access

- Is your copy room always locked?
Attacker model: Network access

• Who would connect a printer to the Internet?
Attacker model: Network access
Attacker model: Web attacker
Four classes of attacks

- Denial of service
- Protection bypass
- Print job manipulation
- Information disclosure
Denial of service

• Postscript infinite loop

{} loop
Next level DoS

PHYSICAL DAMAGE!
• NVRAM has limited # of write cycles
• Can be set in print jobs themselves!
• Continuously set long-term value for number of copies

@PJL DEFAULT COPIES=X
• Reset to factory defaults
• Can be done with a print job (HP)

@PJL DMCMD ASCIIHEX=
"040006020501010301040106"
Print job manipulation

• Redefinition of Postscript `showpage` operator
Information disclosure

- Access to memory
- Access to file system
- Capture print jobs
  - Save on file system or in memory
Attacker model: Web attacker
Same-origin policy

- evil.org
- internal.bank.com

Carrier
CORS spoofing

```
(evil.org) print
(evil.org) print ...
```

```
JavaScript (PS file)
```

```
(printer.bank.com:9100)
```

carrier
1. Background
2. Attacks
3. Evaluation
4. PRET
5. Beyond printers
6. Countermeasures
Obtaining printers

- How would you proceed?

Our approach: Contacted university system administrators
Printers. Lots of printers
### Evaluation results

<table>
<thead>
<tr>
<th>Attack Categories</th>
<th>Denial of Service</th>
<th>Protection Bypass</th>
<th>Print Job Manipulation</th>
<th>Information Disclosure</th>
<th>Print Vulnerabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>infinite loop</td>
<td>showpage redefinition</td>
<td>offline mode</td>
<td>physical damage</td>
<td>restoring factory defaults</td>
</tr>
<tr>
<td>Printers / Printer Languages</td>
<td>PS</td>
<td>PS</td>
<td>PJL</td>
<td>PJL</td>
<td>SNMP</td>
</tr>
<tr>
<td>HP</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Brother</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Lexmark</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Dell</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Kyocera</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Samsung</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Konica Minolta</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>OKI</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># Vulnerable Printers</td>
<td>20</td>
<td>14</td>
<td>8</td>
<td>8</td>
<td>11</td>
</tr>
</tbody>
</table>

**Legend:**
- **1**: Device vulnerable
- **1***: Vulnerability is limited
- **n/a**: Not tested – physically broken printing functionality
- **?**: No support for PostScript or PJL password protection
- **not vulnerable/PostScript feedback not available**

![Image of the table](image-url)
Overview

1. Background
2. Attacks
3. Evaluation
4. PRET
5. Beyond printers
6. Countermeasures
PRinter Exploitation Toolkit (PRET)

User command
```
$ ls
```

Result
```
- 834 .profile
- 1276 init
d - tmp
```

translator
```
PostScript Request
/str 256 string def (%*%../../../init) {==} str filenameforall

Postscript Response
(%disk0%../../../init) TYPE=FILE SIZE=1276
(%disk0%../../../profile) TYPE=FILE SIZE=834
(%disk0%../../../tmp) TYPE=DIR

PJL Request
@PJL FSDIRLIST NAME="0:\..\.." ENTRY=1 COUNT=3

PJL Response
init TYPE=FILE SIZE=1276
.profile TYPE=FILE SIZE=834
tmp TYPE=DIR
```
## PRET commands

<table>
<thead>
<tr>
<th>Command</th>
<th>PS</th>
<th>PJL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ls</td>
<td>✔</td>
<td>✔</td>
<td>List contents of remote directory.</td>
</tr>
<tr>
<td>get</td>
<td>✔</td>
<td>✔</td>
<td>Receive file: get &lt;file&gt;</td>
</tr>
<tr>
<td>put</td>
<td>✔</td>
<td>✔</td>
<td>Send file: put &lt;local file&gt;</td>
</tr>
<tr>
<td>append</td>
<td>✔</td>
<td>✔</td>
<td>Append to file: append &lt;file&gt; &lt;str&gt;</td>
</tr>
<tr>
<td>delete</td>
<td>✔</td>
<td>✔</td>
<td>Delete remote file: delete &lt;file&gt;</td>
</tr>
<tr>
<td>rename</td>
<td>✔</td>
<td>✔</td>
<td>Rename remote file: rename &lt;old&gt; &lt;new&gt;</td>
</tr>
<tr>
<td>find</td>
<td>✔</td>
<td>✔</td>
<td>Recursively list directory contents.</td>
</tr>
<tr>
<td>mirror</td>
<td>✔</td>
<td>✔</td>
<td>Mirror remote file system to local dir.</td>
</tr>
<tr>
<td>touch</td>
<td>✔</td>
<td>✔</td>
<td>Update file timestamps: touch &lt;file&gt;</td>
</tr>
<tr>
<td>mkdir</td>
<td>✔</td>
<td>✔</td>
<td>Create remote directory: mkdir &lt;path&gt;</td>
</tr>
<tr>
<td>cd</td>
<td>✔</td>
<td>✔</td>
<td>Change remote working directory.</td>
</tr>
<tr>
<td>pwd</td>
<td>✔</td>
<td>✔</td>
<td>Show working directory on device.</td>
</tr>
<tr>
<td>chvol</td>
<td>✔</td>
<td>✔</td>
<td>Change remote volume: chvol &lt;volume&gt;</td>
</tr>
<tr>
<td>format</td>
<td>✔</td>
<td>✔</td>
<td>Initialize printer's file system.</td>
</tr>
<tr>
<td>fuzz</td>
<td>✔</td>
<td>✔</td>
<td>File system fuzzing: fuzz &lt;category&gt;</td>
</tr>
<tr>
<td>df</td>
<td>✔</td>
<td>✔</td>
<td>Show volume information.</td>
</tr>
<tr>
<td>free</td>
<td>✔</td>
<td>✔</td>
<td>Show available memory.</td>
</tr>
</tbody>
</table>
Overview

1. Background
2. Attacks
3. Evaluation
4. PRET
5. Beyond printers
6. Countermeasures
Target: Google

Converting PostScript = interpreting PostScript

$ 3133.7
PostScript in the web?

- PS conversion websites
- Image conversion sites
- Thumbnail preview
Overview

1. Background
2. Attacks
3. Evaluation
4. PRET
5. Beyond printers
6. Countermeasures
Countermeasures
“Hacker Stackoverflowin made 160,000 printers spew out ASCII art around the world” -- theregister.co.uk
Countermeasures

- **Employees**: always lock the copy room
- **Administrators**: sandbox printers in a VLAN accessible only via print server
- **Printer vendors**: undo insecure design decisions (PostScript, proprietary PJL)
- **Browser vendors**: block port 9100
Christian Slater was right: Printers are insecure

- PostScript and PJL considered dangerous
- Exploitation through lots of channels (websites, even 😊)
- No real countermeasures yet
Thanks for your attention...

PRET („Printer Exploitation Toolkit“)
- https://github.com/RUB-NDS/PRET

Hacking Printers Wiki
- http://hacking-printers.net/

Questions?