



Industrial Control Systems

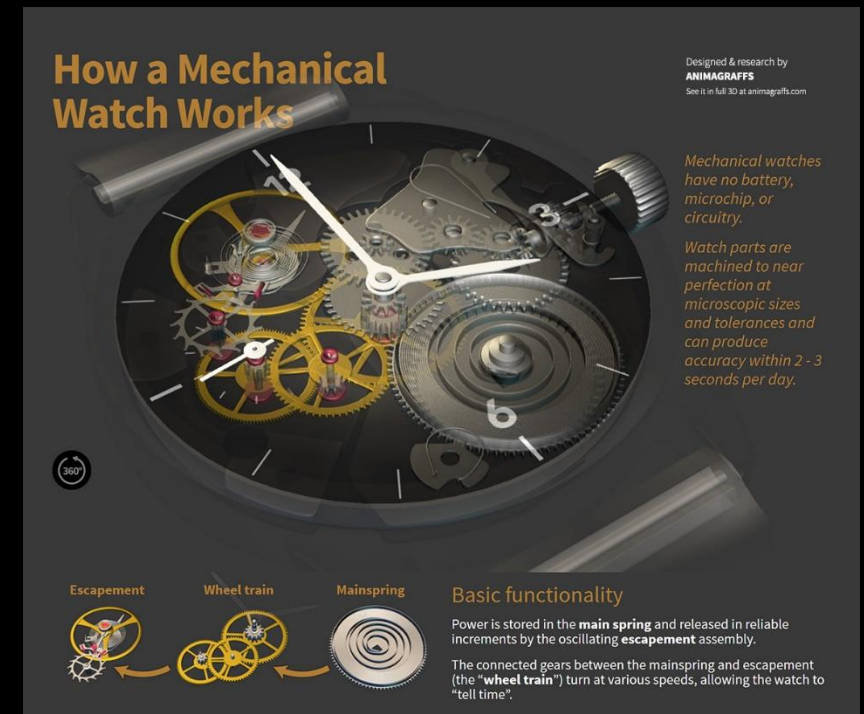
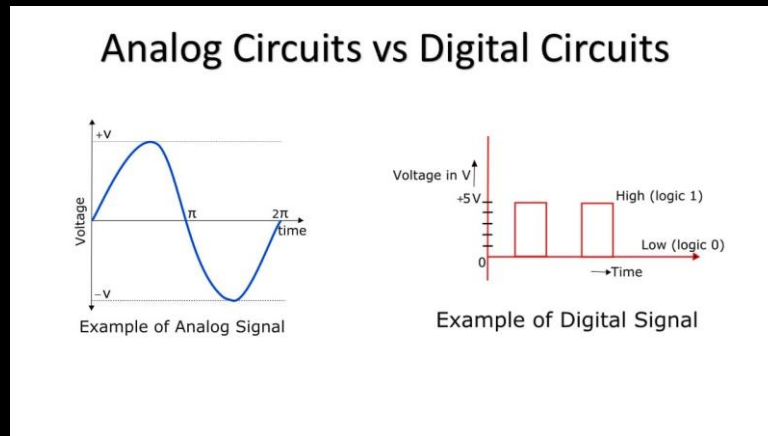
Internet of Things/Industrial Internet of Things (IoT/IIoT)

Thing1 and Thing2

- No, literally. Today, most electronics or electro-mechanical devices can connect to the “world” via the internet. While this may seem “Cool” – think again.
- What is the difference between Electrical, Electronic, & Electro-Mechanical?
- And What are the risks of connecting these devices to the internet after all?

Electronics Vs. Electrical vs. Mechanical

- Made with Semiconductor materials like Silicon
- You would hear things like Diodes, transistors, resistors etc.,
- Think “Applied Electricals”, Integrated chips that are smarter with the “juice” ;)
- Pertains to Electrons
- Made with Conducting materials alike Alloys/metals etc., Copper
- Hear things like Wires, Fuse, Switch, Motor, transformers
- Think Lightning rods and other cool stuff!
- Any physical body/thing
- The “juice” here can be electrical, electronics or just mechanical – like the clock



Internet of Things?

IoT and IIoT - Do we really need our personal coffee maker on the internet?

Industry 4.0

1st revolution



Mechanization, steam
and water power

2nd revolution



Mass production and
electricity

3rd revolution



Electronic and IT
systems, automation

4th revolution

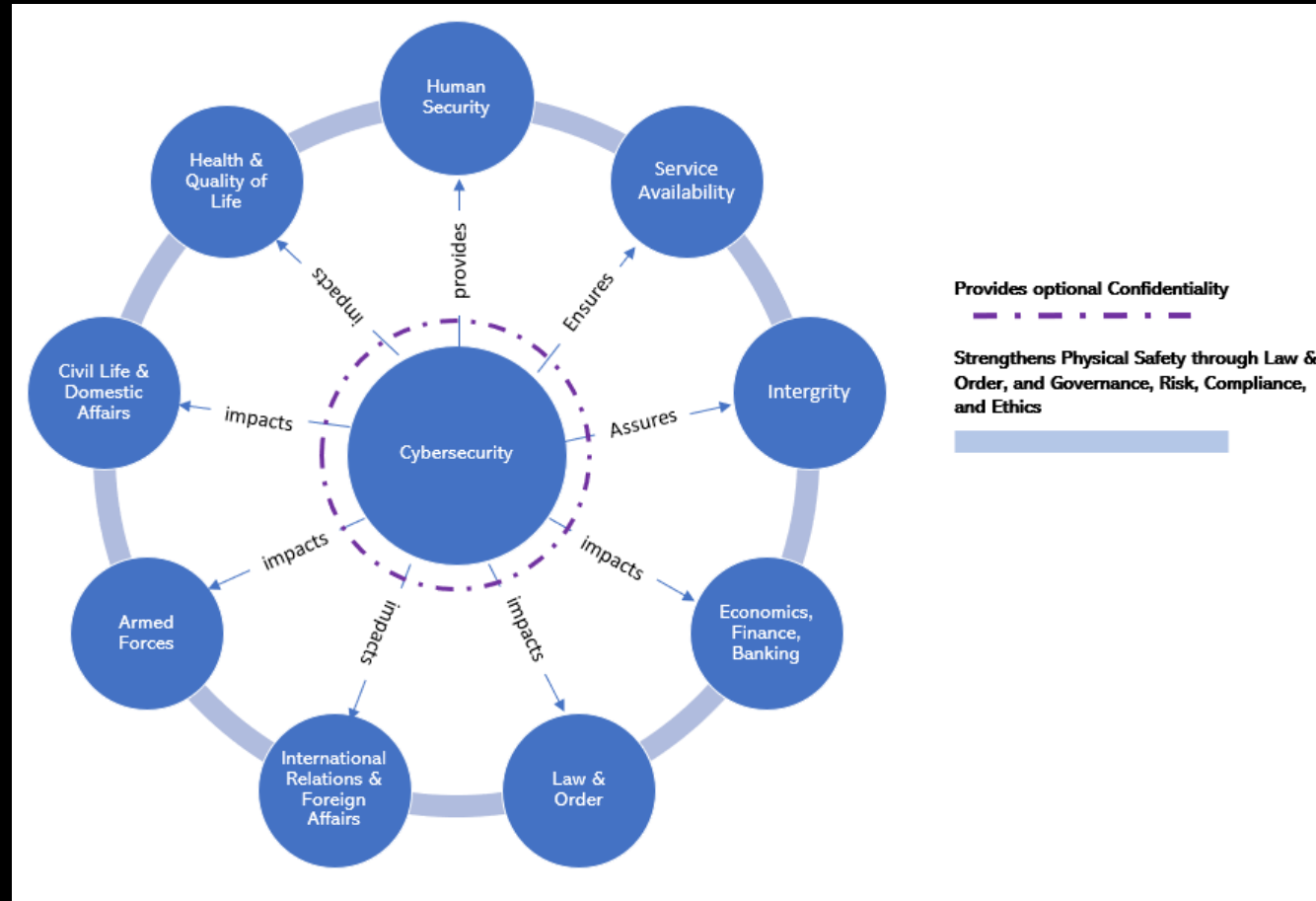


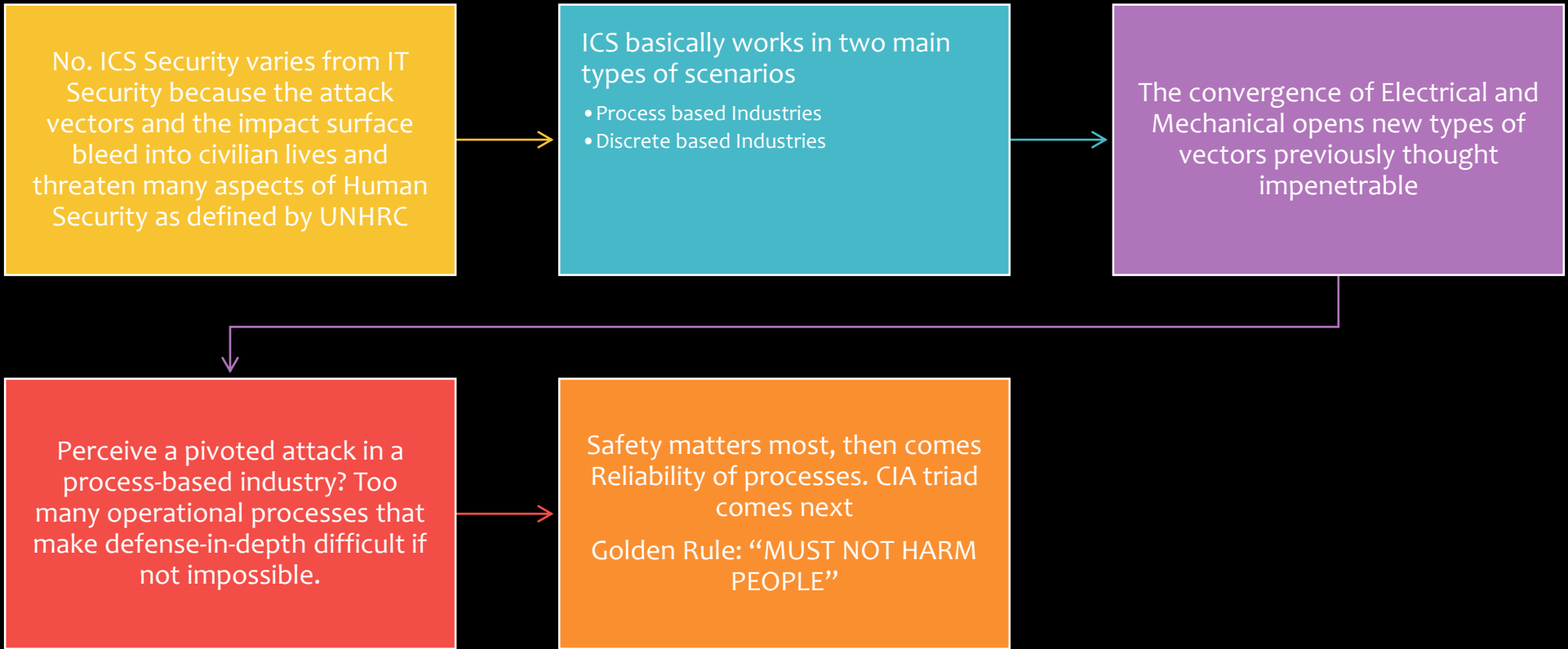
Cyber physical
systems

What are Industrial Control Systems?

What do they control?

Cybersecurity permeates many aspects of our lives





ICS Security, is it not the same as IT Security?

NIST Guide to Industrial Control Systems (ICS) Security

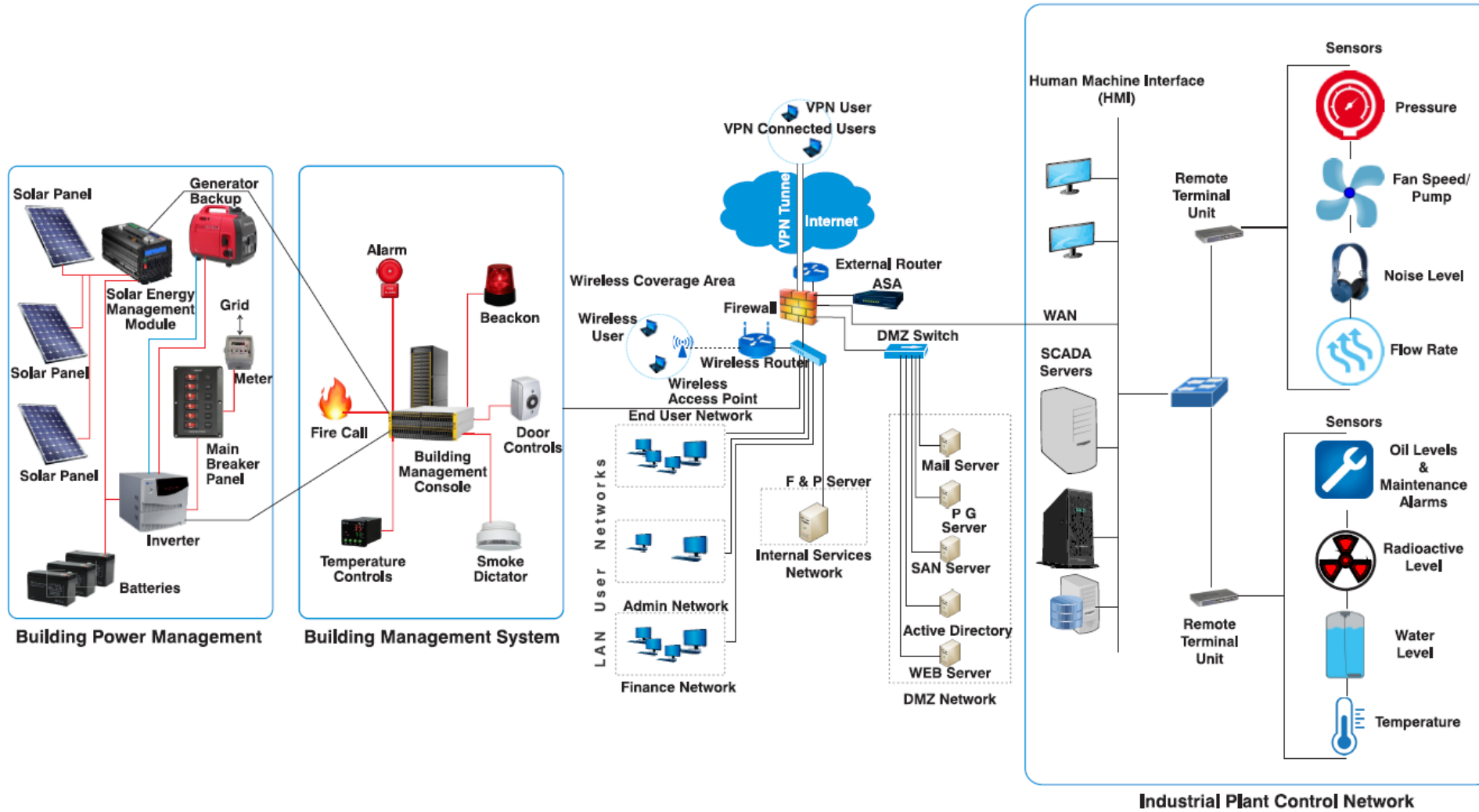


FIGURE 3. An example industrial IoT network.

Why A Smart City Framework?



Internet of Things blurs the line between Electrical and Mechanical



What were secure through obscurity are now deemed unsecure for the very same reasons



Engineering, Operational, Architecture, and Design professionals can no more detach themselves from the matters of security



As they embark on designing infrastructure for cities and industrial systems, thinking about safety, security, and privacy becomes essential

11 SUSTAINABLE CITIES
AND COMMUNITIES



17 PARTNERSHIPS
FOR THE GOALS



12 RESPONSIBLE
CONSUMPTION
AND PRODUCTION



9 INDUSTRY, INNOVATION
AND INFRASTRUCTURE



United Nations Sustainable Development Goals

<https://sustainabledevelopment.un.org/sdgs>

ICS close-up

- Process Control System (PCS)
- Distributed Control Systems (DCS)
- Programmable Logic Controllers (PLC)
- Supervisory Control and Data Acquisition (SCADA)
- Safety Instrumented Systems (SIS)
- Human Machine Interface (HMI)
- Remote Terminal Unit (RTU)

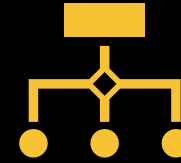
SCADA (Supervisory Control and Data Acquisition)



These systems are used in distribution systems such as water distribution and wastewater collection systems, oil and natural gas pipelines, electrical utility transmission and distribution systems, and rail and other public transportation systems.



SCADA systems integrate data acquisition systems with data transmission systems and HMI software to provide a centralized monitoring and control system for numerous process inputs and outputs.



SCADA systems are designed to collect field information, transfer it to a central computer facility, and display the information to the operator graphically or textually, thereby allowing the operator to monitor or control an entire system from a central location in near real time.



Based on the sophistication and setup of the individual system, control of any individual system, operation, or task can be automatic, or it can be performed by operator commands.

Distributed Control Systems (DCS)

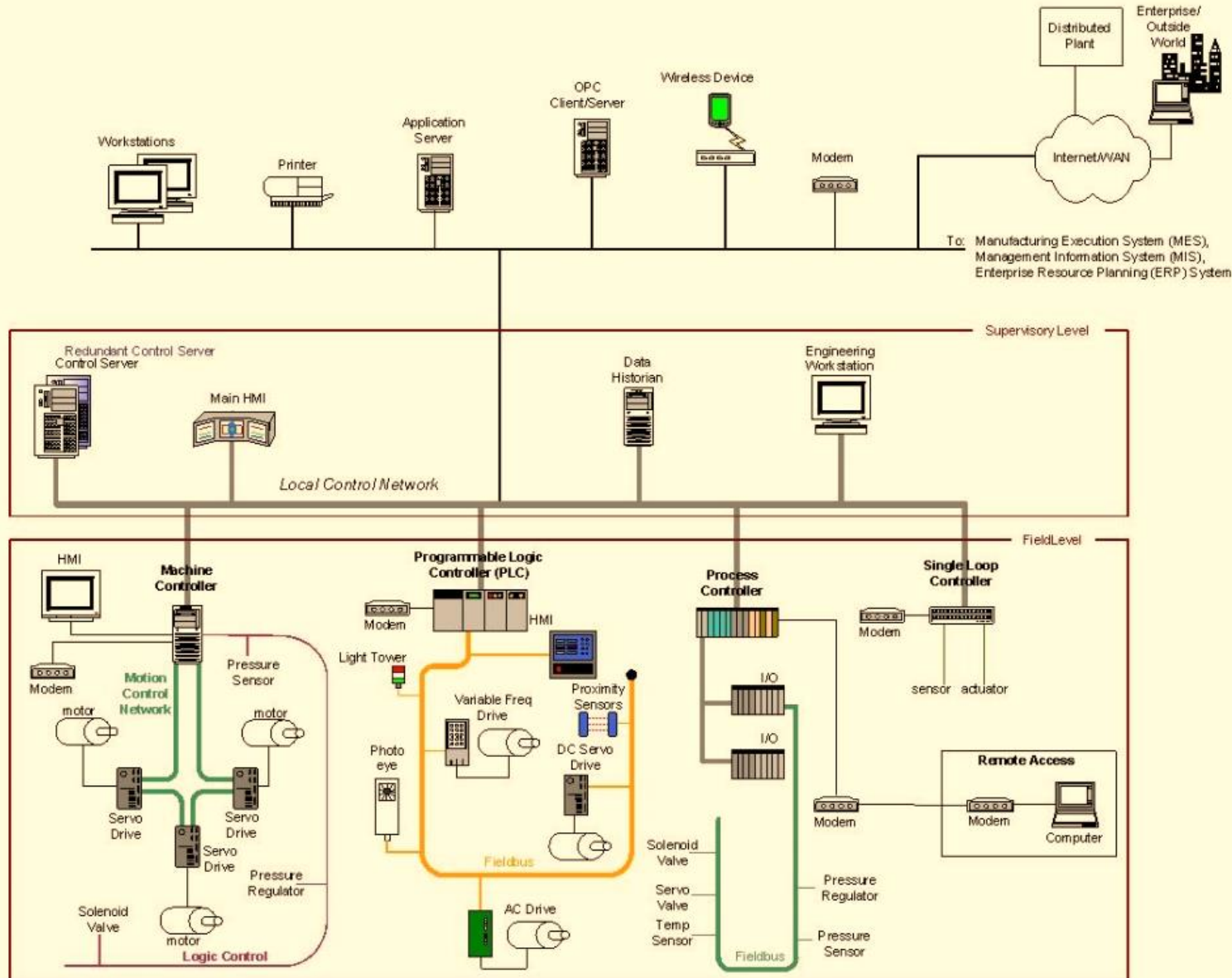


Figure 2-7. DCS Implementation Example

- DCS are used to control production systems within the same geographic location for industries such as oil refineries, water and wastewater treatment, electric power generation plants, chemical manufacturing plants, automotive production, and pharmaceutical processing facilities.
- Typical control devices include Programmable Logic Controller, a Process Controller, a loop controller, a machine controller

A great intro deck

https://www.msec.be/verboden/seminaries/ICS_ar_chs_and_sec_essentials/ICS_Overview.pdf

ICS Overview: Terms & Definitions

HMI: Human-Machine Interface

A human-machine interface is the user interface to the processes of an industrial control system. An HMI effectively translates the communication to and from PLCs, RTUs, and other industrial assets to a human-readable interface, which is used by control systems operators to manage and monitor processes. An HMI can range from a physical control panel with buttons to an industrial PC with a colour graphics display running dedicated HMI software.



Five terms to know well - Here's a great article on Electrical Engineering Portal (EEP) website



<https://electrical-engineering-portal.com/scada-dcs-plc-rtu-smart-instrument>

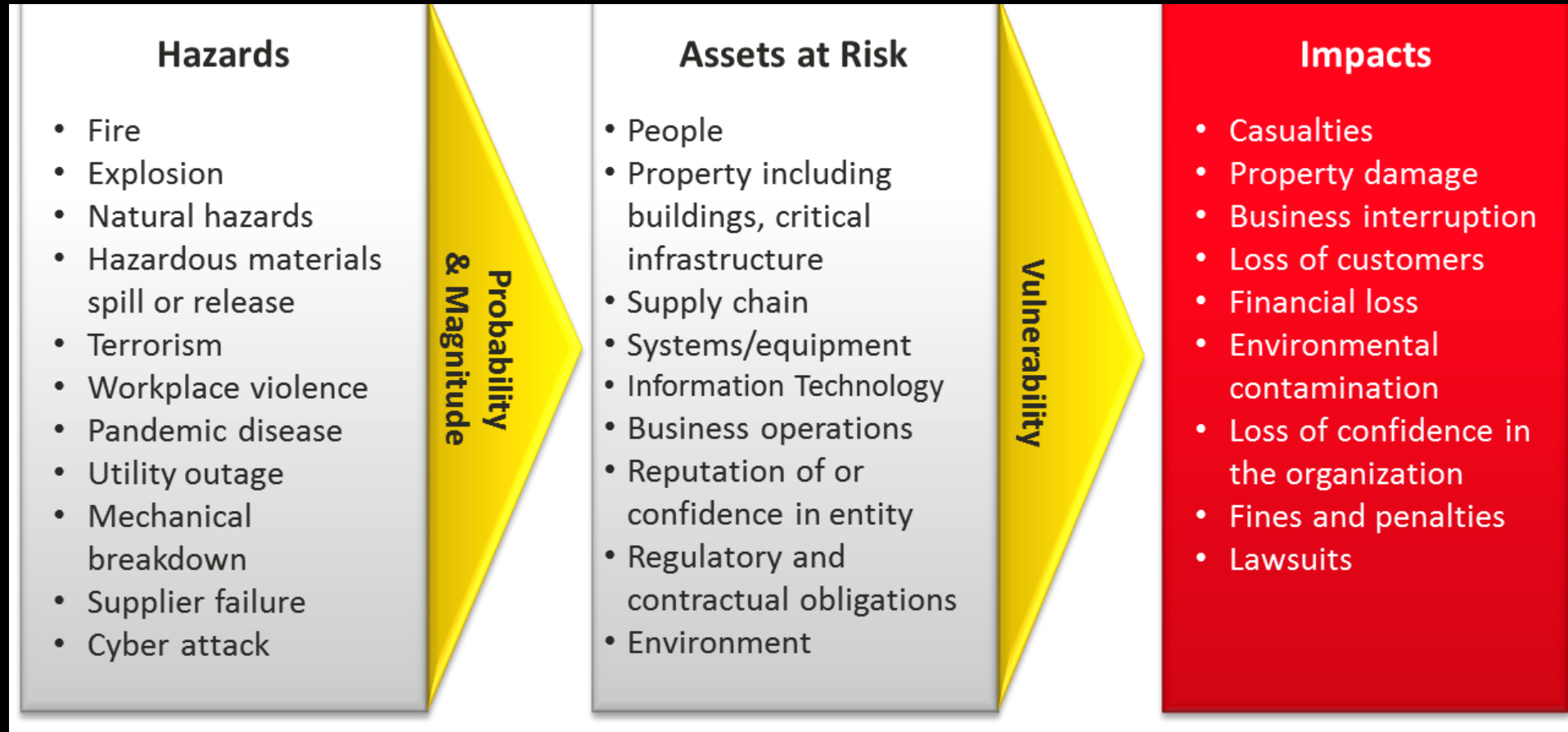
US ICS

| | | | |
|-------------------------------|--|------------------------------|--------------------------------|
| Chemical Sector | Commercial Facilities Sector | Communications Sector | Defense Industrial Base Sector |
| Critical Manufacturing Sector | Dams Sector | Emergency Services Sector | Energy Sector |
| Financial Services Sector | Food and Agriculture Sector | Government Facilities Sector | Healthcare and Public Sector |
| Information Technology Sector | Nuclear Reactors, Materials and Waste Sector | Transportation Systems | Water and Wastewater Sector |

Understanding Risks

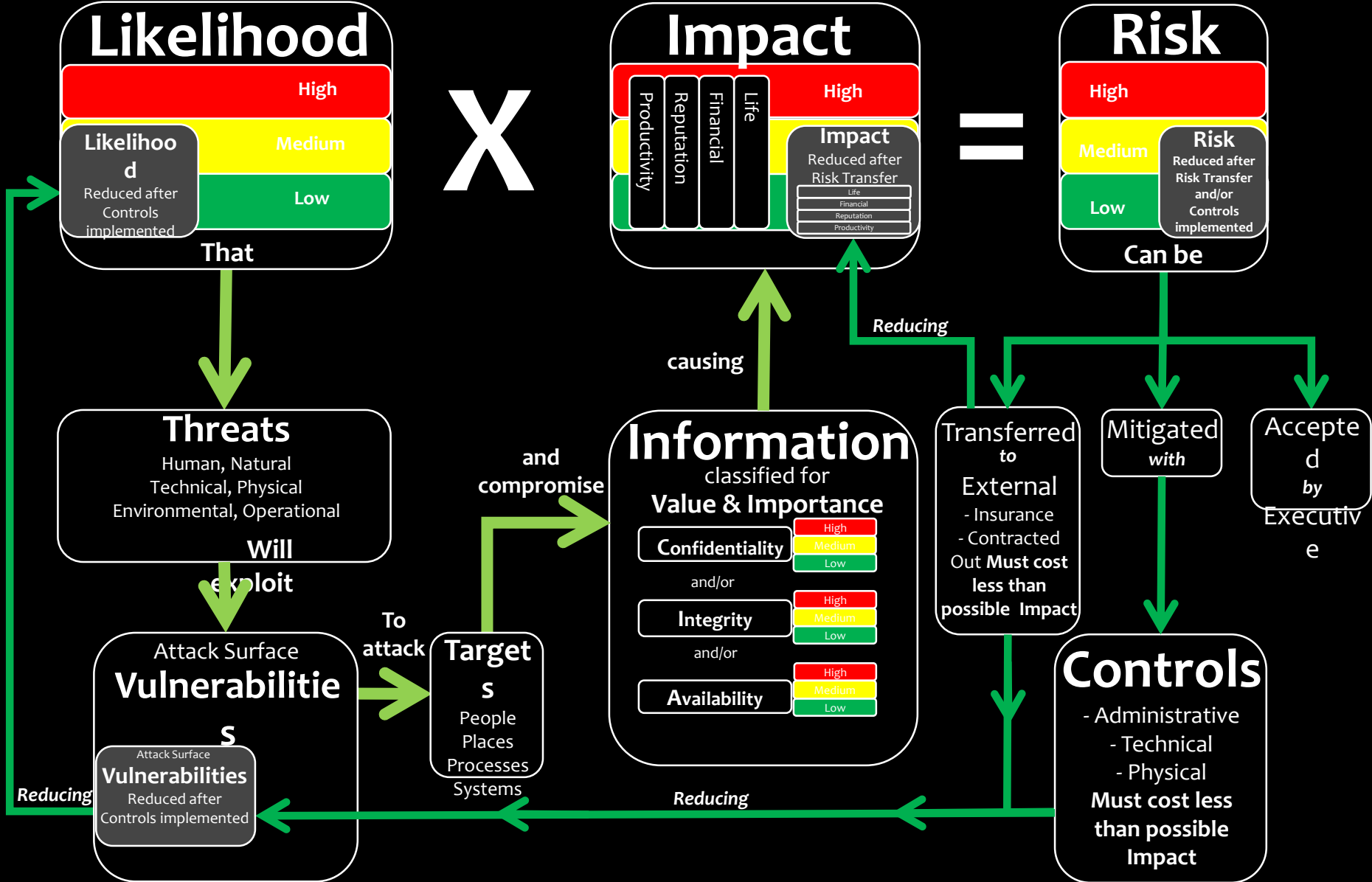
Risk, Threat, Vulnerability, Impact, Likelihood

Look at the landscape



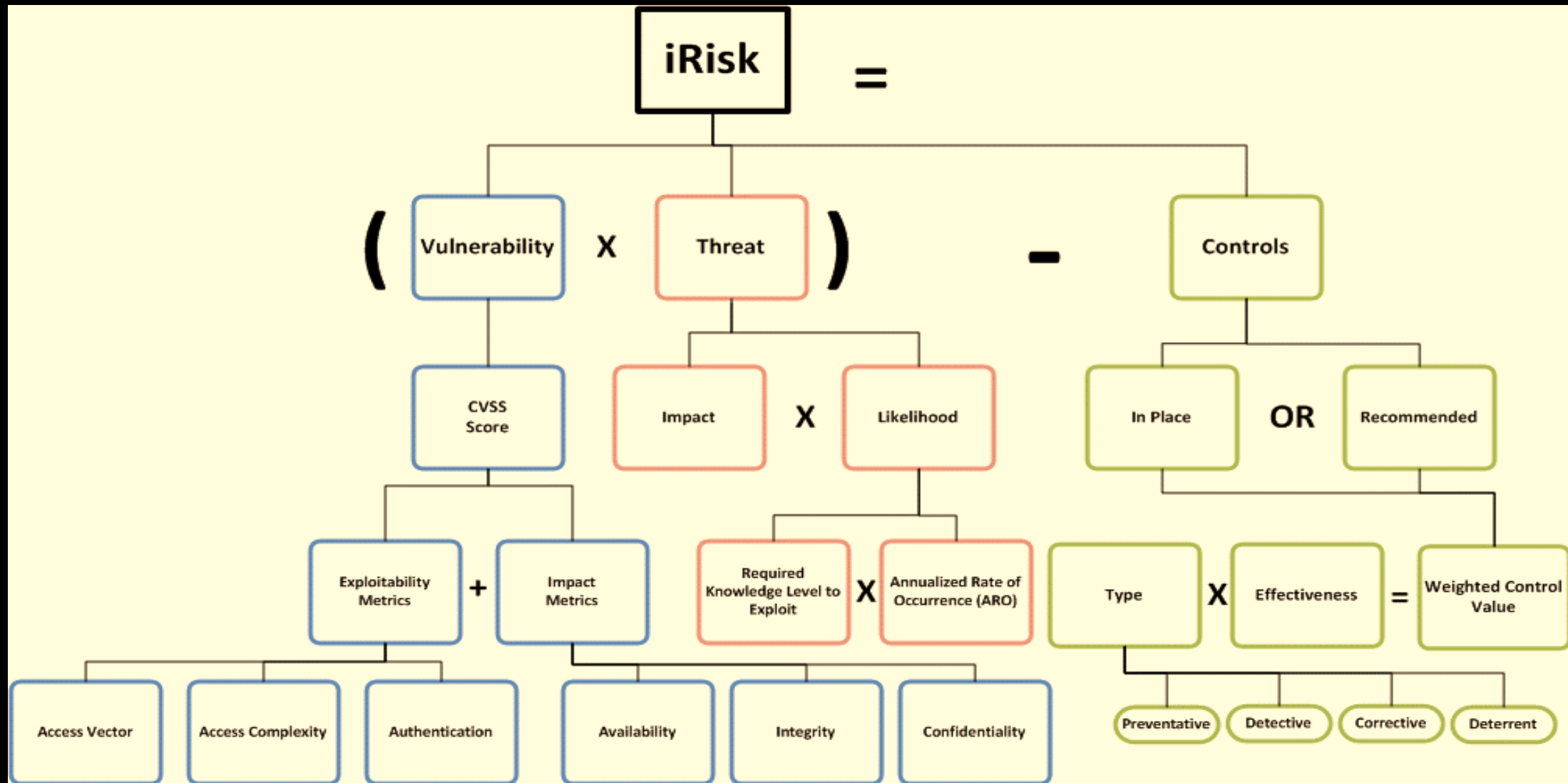
Risk Analysis

https://www.tru.ca/its/infosecurity/about/Risk_Analysis.html



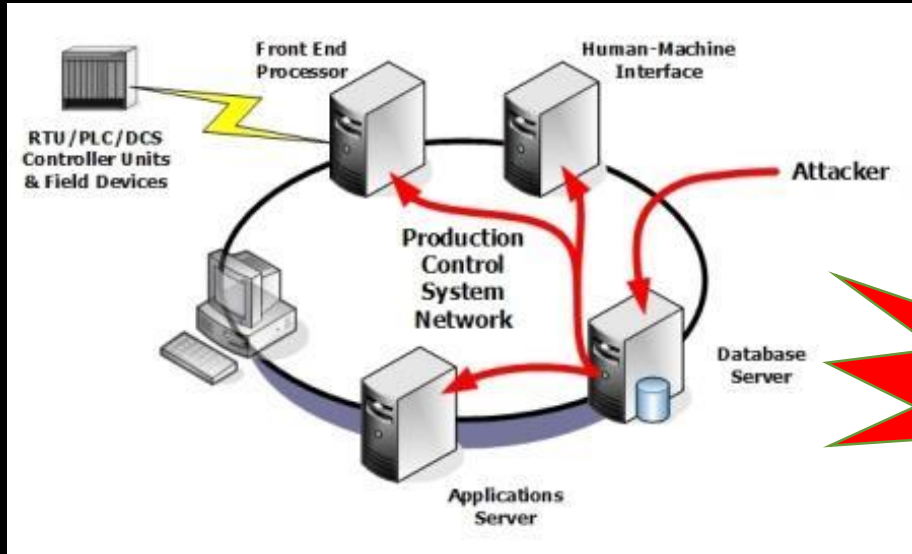
A bit more quantitative

https://www.carmelowalsh.com/wp-content/uploads/2014/03/Irisk_full_web.png

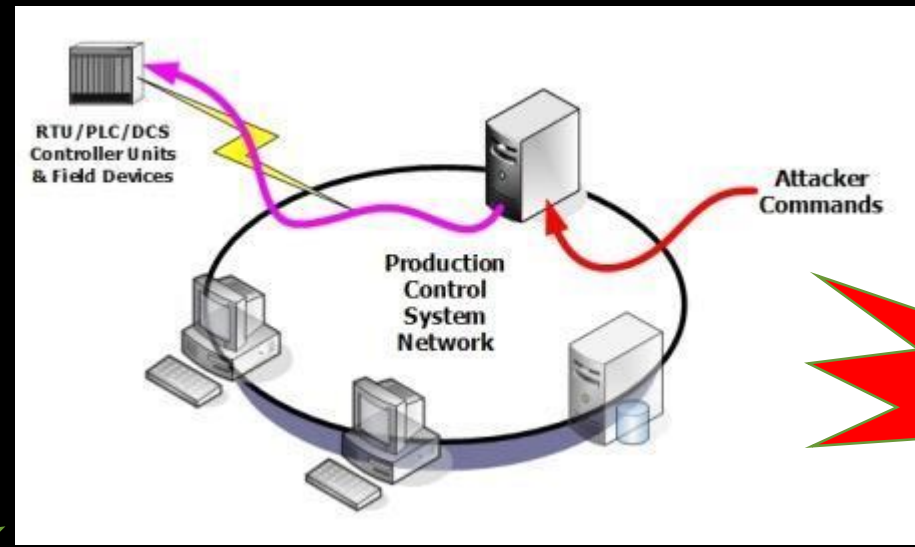


Cyber Vulnerabilities

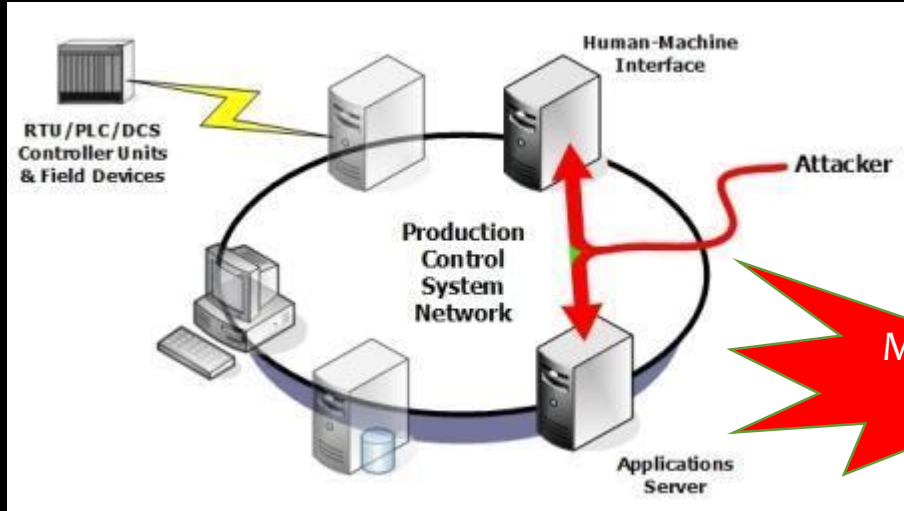
<https://www.us-cert.gov/ics/content/overview-cyber-vulnerabilities>



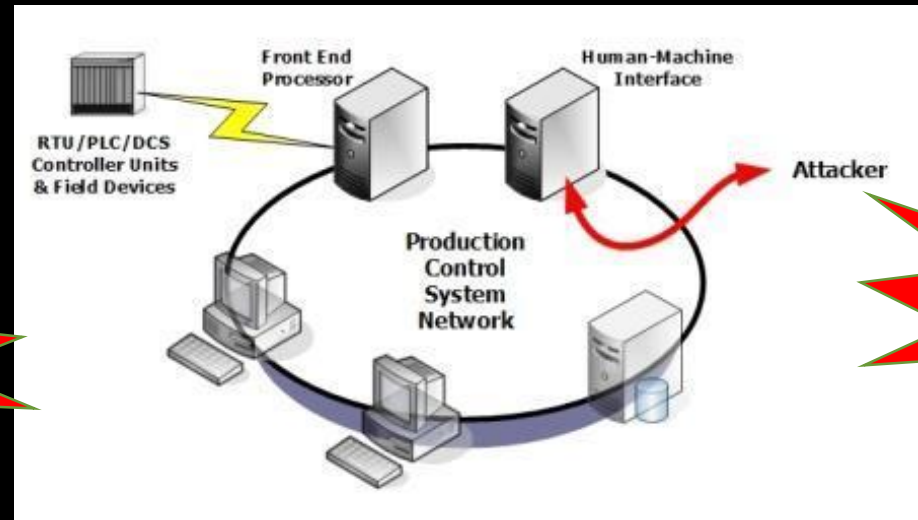
Changing the database



Sending commands directly



Man-in-the-middle



Exporting HMI Screen

Smart Grid Threat Landscape

| | Corporation | Cyber-criminals | Employees | Hacktivists | Nation States | Natural Disasters | Terrorists | Cyber fighters |
|--|-------------|-----------------|-----------|-------------|---------------|-------------------|------------|----------------|
| Physical attacks | | | | | √ | | √ | |
| Unintentional damage | | | √ | | | | | |
| Failures / Malfunction | | √ | √ | √ | √ | | | √ |
| Eavesdropping / Interception / Hacking | √ | √ | √ | √ | √ | | √ | √ |
| Legal | | | √ | | | | | |
| Nefarious activity / abuse | √ | √ | √ | √ | √ | | √ | √ |
| Outages | | | √ | | √ | √ | | |
| Damage / Loss (IT-Assets) | √ | √ | √ | √ | √ | | √ | √ |
| Disaster | | | | | | √ | √ | |

Table 3: Involvement of Threat Agents in the threats

<https://www.enisa.europa.eu/publications/smart-grid-threat-landscape-and-good-practice-guide>

Mobile Phones have their own problems!

Mobile Top 10 2016-Top 10

M1 - Improper Platform Usage

This category covers misuse of a platform feature or failure to use platform security controls. It might include Android intents, platform permissions, misuse of TouchID, the Keychain, or some other security control that is part of the mobile operating system. There are several ways that mobile apps can experience this risk.

M2 - Insecure Data Storage

This new category is a combination of M2 + M4 from Mobile Top Ten 2014. This covers insecure data storage and unintended data leakage.

M3 - Insecure Communication

This covers poor handshaking, incorrect SSL versions, weak negotiation, cleartext communication of sensitive assets, etc.

M4 - Insecure Authentication

This category captures notions of authenticating the end user or bad session management. This can include:

- Failing to identify the user at all when that should be required
- Failure to maintain the user's identity when it is required
- Weaknesses in session management

M5 - Insufficient Cryptography

The code applies cryptography to a sensitive information asset. However, the cryptography is insufficient in some way. Note that anything and everything related to TLS or SSL goes in M3. Also, if the app fails to use cryptography at all when it should, that probably belongs in M2. This category is for issues where cryptography was attempted, but it wasn't done correctly.

M6 - Insecure Authorization

This is a category to capture any failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.). It is distinct from authentication issues (e.g., device enrolment, user identification, etc.).

If the app does not authenticate users at all in a situation where it should (e.g., granting anonymous access to some resource or service when authenticated and authorized access is required), then that is an authentication failure not an authorization failure.

M7 - Client Code Quality

This was the "Security Decisions Via Untrusted Inputs", one of our lesser-used categories. This would be the catch-all for code-level implementation problems in the mobile client. That's distinct from server-side coding mistakes. This would capture things like buffer overflows, format string vulnerabilities, and various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device.

M8 - Code Tampering

This category covers binary patching, local resource modification, method hooking, method swizzling, and dynamic memory modification.

Once the application is delivered to the mobile device, the code and data resources are resident there. An attacker can either directly modify the code, change the contents of memory dynamically, change or replace the system APIs that the application uses, or modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.

M9 - Reverse Engineering

This category includes analysis of the final core binary to determine its source code, libraries, algorithms, and other assets. Software such as IDA Pro, Hopper, otool, and other binary inspection tools give the attacker insight into the inner workings of the application. This may be used to exploit other nascent vulnerabilities in the application, as well as revealing information about back end servers, cryptographic constants and ciphers, and intellectual property.

M10 - Extraneous Functionality

Often, developers include hidden backdoor functionality or other internal development security controls that are not intended to be released into a production environment. For example, a developer may accidentally include a password as a comment in a hybrid app. Another example includes disabling of 2-factor authentication during testing.

Understanding the value of what we want to protect

Asset Characterization

- What asset (information) needs to be protected?
- Why does the asset need to be protected?
- Who has the responsibility for managing and protecting the asset (what are the roles, responsibilities, accountabilities and authorities)?
- If the threat actor compromised the asset, what realistic worst-case scenarios would result?
- What is the value of the asset?
- What is the criticality of the process or information to the business mission?
- What are the protection levels for confidentiality, integrity, and availability?
- What interconnections are required for the systems to perform?
- What methods are currently available for user access?
- What dependencies are present for system functionality?
- How does the information flow through the system, and through what mechanisms?

MITRE Threat ATT&CK – Adversary tactics and techniques

ATT&CK Matrix for Enterprise

| Initial Access | Execution | Persistence | Privilege Escalation | Defense Evasion | Credential Access | Discovery | Lateral Movement | Collection | Command and Control | Exfiltration | Impact |
|-------------------------------------|--|---------------------------|--------------------------------|-----------------------------|------------------------------------|------------------------------|--|------------------------------------|---------------------------------------|---|----------------------------|
| Drive-by Compromise | AppleScript | .bash_profile and .bashrc | Access Token Manipulation | Access Token Manipulation | Account Manipulation | Account Discovery | AppleScript | Audio Capture | Commonly Used Port | Automated Exfiltration | Account Access Removal |
| Exploit Public-Facing Application | CMSTP | Accessibility Features | Accessibility Features | Binary Padding | Bash History | Application Window Discovery | Application Deployment Software | Automated Collection | Communication Through Removable Media | Data Compressed | Data Destruction |
| External Remote Services | Command-Line Interface | Account Manipulation | AppCert DLLs | BITS Jobs | Brute Force | Browser Bookmark Discovery | Component Object Model and Distributed COM | Clipboard Data | Connection Proxy | Data Encrypted | Data Encrypted for Impact |
| Hardware Additions | Compiled HTML File | AppCert DLLs | Applnit DLLs | Bypass User Account Control | Credential Dumping | Domain Trust Discovery | Exploitation of Remote Services | Data from Information Repositories | Custom Command and Control Protocol | Data Transfer Size Limits | Defacement |
| Replication Through Removable Media | Component Object Model and Distributed COM | Applnit DLLs | Application Shimming | Clear Command History | Credentials from Web Browsers | File and Directory Discovery | Internal Spearphishing | Data from Local System | Custom Cryptographic Protocol | Exfiltration Over Alternative Protocol | Disk Content Wipe |
| Spearphishing Attachment | Control Panel Items | Application Shimming | Bypass User Account Control | CMSTP | Credentials in Files | Network Service Scanning | Logon Scripts | Data from Network Shared Drive | Data Encoding | Exfiltration Over Command and Control Channel | Disk Structure Wipe |
| Spearphishing Link | Dynamic Data Exchange | Authentication Package | DLL Search Order Hijacking | Code Signing | Credentials in Registry | Network Share Discovery | Pass the Hash | Data from Removable Media | Data Obfuscation | Exfiltration Over Other Network Medium | Endpoint Denial of Service |
| Spearphishing via Service | Execution through API | BITS Jobs | Dylib Hijacking | Compile After Delivery | Exploitation for Credential Access | Network Sniffing | Pass the Ticket | Data Staged | Domain Fronting | Exfiltration Over Physical Medium | Firmware Corruption |
| Supply Chain Compromise | Execution through Module Load | Bootkit | Elevated Execution with Prompt | Compiled HTML File | Forced Authentication | Password Policy Discovery | Remote Desktop Protocol | Email Collection | Domain Generation Algorithms | Scheduled Transfer | Inhibit System Recovery |
| Trusted Relationship | Exploitation for Client Execution | Browser Extensions | Emond | Component Firmware | Hooking | Peripheral Device Discovery | Remote File Copy | Input Capture | Fallback Channels | | Network Denial of Service |

<https://attack.mitre.org/>

MITRE Threat Modeling

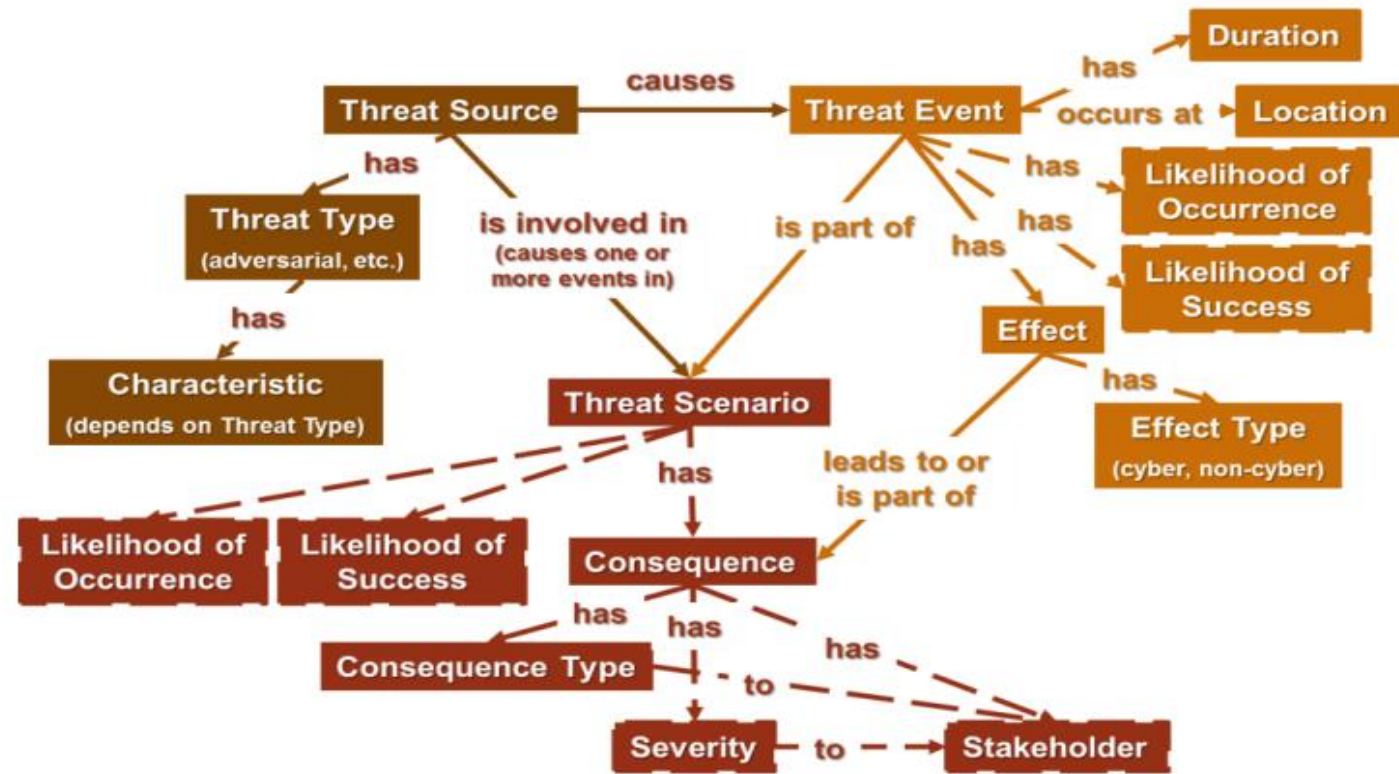


Figure 4. Key Constructs in Cyber Threat Modeling (Details for Adversarial Threats Not Shown)

System-of-Systems Threat

https://www.mitre.org/sites/default/files/publications/pr_18-1631-ngci-system-of-systems-threat-model.pdfModel

Some well-known cyber attacks on industrial systems

BRINGING INDUSTRIAL SYSTEMS ONLINE:

A HISTORY OF IIOT CYBER-ATTACKS & THE FUTURE OF SECURITY

The IIoT is poised to bring a new world of benefits to businesses operating industrial systems - optimized operations and supply chains, greater business agility, new revenue streams and services and more.

To fully capture these benefits, the systems are exploding in scope to greater internet connectivity and shifting further away from the historically closed systems that relied more heavily on physical security to ensure integrity.

Unfortunately, with this broader connectivity comes new attack vectors, vulnerabilities, and more opportunities for hackers.

WHAT IS THE IIoT?

The Industrial Internet of Things (IIoT), aka the Industrial Internet, is the integration of complex machinery with networked sensors and software. The machines are connected and talking to each other, and communicating back to centralized control systems. Example industries include:

- Manufacturing / factories
- Power plants
- Energy grids
- Semiconductors
- Automotive
- Aerospace
- Commercial Building Automation

<https://www.techrepublic.com/article/infographic-charts-history-and-potential-risks-of-the-industrial-internet-of-things/>



Exploits

An exploit is where a vulnerability was found and exposed by researchers in the media.

The Stuxnet Worm

Allegedly created by American-Israeli Governments in order to attack Iran's Nuclear Facilities. The systems compromised weren't connected to the internet at the time.

Centrifuges and valves were sabotaged and five companies related to the nuclear programme were also breached.

Nov 2007

SCADA System

Hackers destroyed a pump used by a US Water Utility Company after gaining remote access to their SCADA system by stealing usernames and passwords belonging to the manufacturer's customers.

Levels of chemicals in the treatment company were changed and 2.5 million customer's had their personal data exposed online.

Nov 2011

Smart Meters

Smart Meters were hacked in Puerto Rico to reduce power bills. The FBI was asked to investigate and found that these hacks did need a physical presence.

They also found that the Puerto Rico Utility Industry was losing an average of \$400million a year from Smart Meter hacking.

April 2012

Serial Port Servers

Rapid 7 found vulnerabilities in the configuration of serial ports or terminal servers, which could expose a range of critical assets such as POS terminals, ATM's and industrial control systems.

April 2013



Researchers were able to imitate BMW servers and send remote unlocking instructions to vehicles.

Jan 2015

Jeep

Wired reporter is shown how a Jeep can be controlled remotely by two security researchers. 1.5 million cars have been recalled since.

Jul 2015

Sniper Rifle

Security researchers at the Black Hat Hacker conference showed how you can hack into a TrackingPoint self-aiming rifle through vulnerabilities in its software.

Jul 2015

Power Quality Analyzers

Applied Risk released a report showing vulnerabilities in power quality analyzers used to monitor power quality and analyze electrical disturbances that can interfere with industrial equipment.

Oct 2015

TARGET

The company was breached when hackers used malware to penetrate a HVAC company working for them.

Personal data for over 70 million customers was stolen.

Nov 2015

Ukraine Power Grid

Hackers used stolen credentials to gain remote access to the Ukrainian power grid and cut power to 30 substations and 225,000 customers.

The attack included installation of custom firmware, deletion of files including master boot records, and shutting down of telephone communications.

Mar 2016

Cyber-Attacks

These were actual attacks by hackers!

Implement Security into Your IIoT Ecosystems Now

According to the Industrial Internet Consortium (IIC), only 25% of organizations have a clear IIoT security strategy. Leaders are struggling most with data security (51%) and privacy (39%).

Overcoming these barriers is essential to the success of the IIoT. The following are tips for implementing security in your IIoT ecosystem.



1. Security by Design

Build security into your IIoT systems as early as possible.



2. Information Security Principles

Leverage established standards covering these key information security principles: authentication, authorization, encryption and data integrity.



3. Use Proven Technologies and Standards

Combining secure hardware (such as Trusted Platform Modules - TPMs) with Digital Certificates (such as public key infrastructure - PKI) enables robust identity assumptions.



4. Leverage the Cloud

The SaaS model allows for high scale certificate deployments without changing infrastructure hardware and has built in mechanisms for audit-ability, access control and reporting.



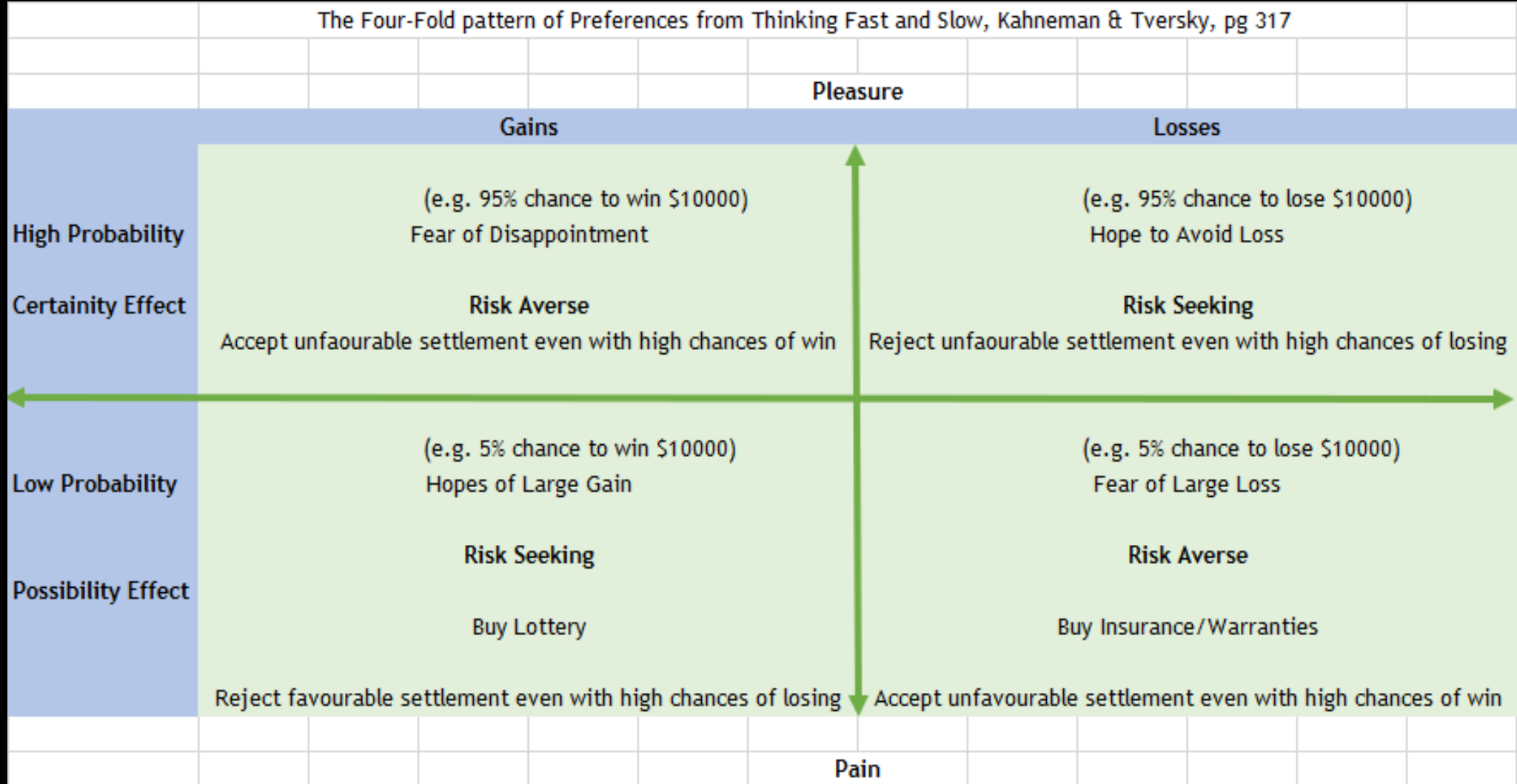
5. Don't Go It Alone

Whether you are an organization building your own IIoT products/solutions or a technology vendor, finding the right security partner to address the risk and needs of your ecosystem is the key to success.

With decades of experience as an identity services provider and proven Public Key Infrastructure (PKI) and Identity and Access Management (IAM) solutions, GlobalSign is uniquely positioned to help you build identity management and security into your IIoT ecosystem with minimal CAPEX and time to market.

<http://bit.ly/manage-iiot>

Humans are fallible!



But we also work under many limitations

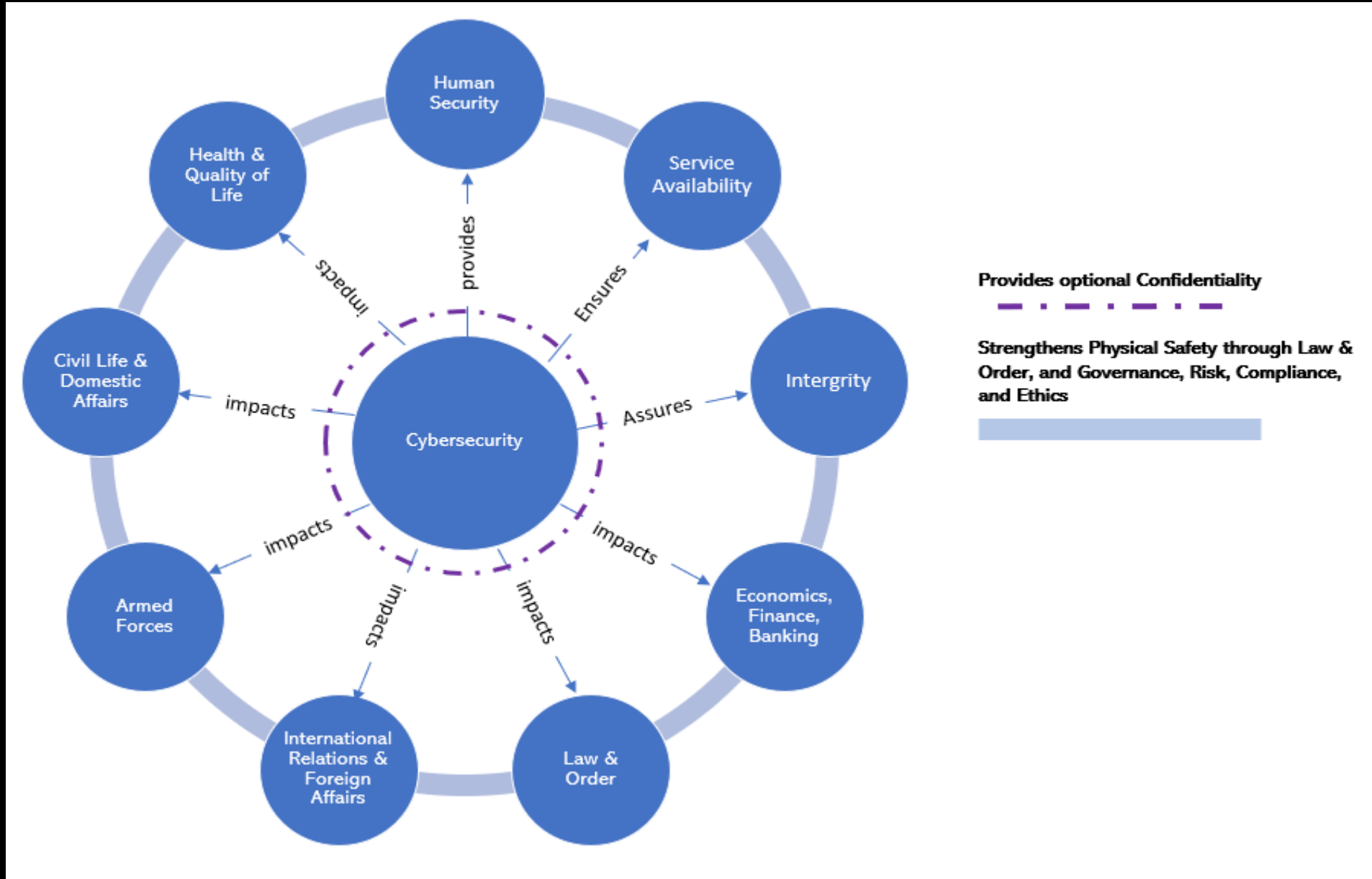
| Constraints (rings) within which businesses have to operate | | | | |
|---|-------------|-------------------------|-------------|-----------------|
| Law | Regulations | Contractual Obligations | Geopolitics | Social-cultural |

Godha, bapuji ©

Understanding security matters

Confidentiality, Integrity, Privacy, Availability,
Authentication, Authorisation, Access Control

Cybersecurity



Understanding Cultures and Geopolitics of the world

Nations do not engage in conflict for fun – there are reasons – find those intentions, motives, instigations, interests, their strong and weak points to better secure critical infrastructure

How do we protect these systems then?

Understanding defenses, Business Continuity
and Resilience Needs of each system

We must pick and choose

Prioritise what matters most; Think about what interests you most; Figure out what you can do the best in the available time

Can I help to protect our ICS?

Of Course! You must join in the workforce for the future! Skills required to defend our ICS

<https://dragos.com/blog/industry-news/a-dragos-industrial-control-system-security-reading-list/>

<https://www.state.gov/students/>

Understanding Network Protocols and how they might differ in Industrial systems

Understanding Policies and Safety Regulations in Industrial zones

Understanding how Electrical, Electronics and Mechanical devices work together in Industrial systems

Understanding Risks, Vulnerabilities, threats, and impact on communities due to an industrial system failure

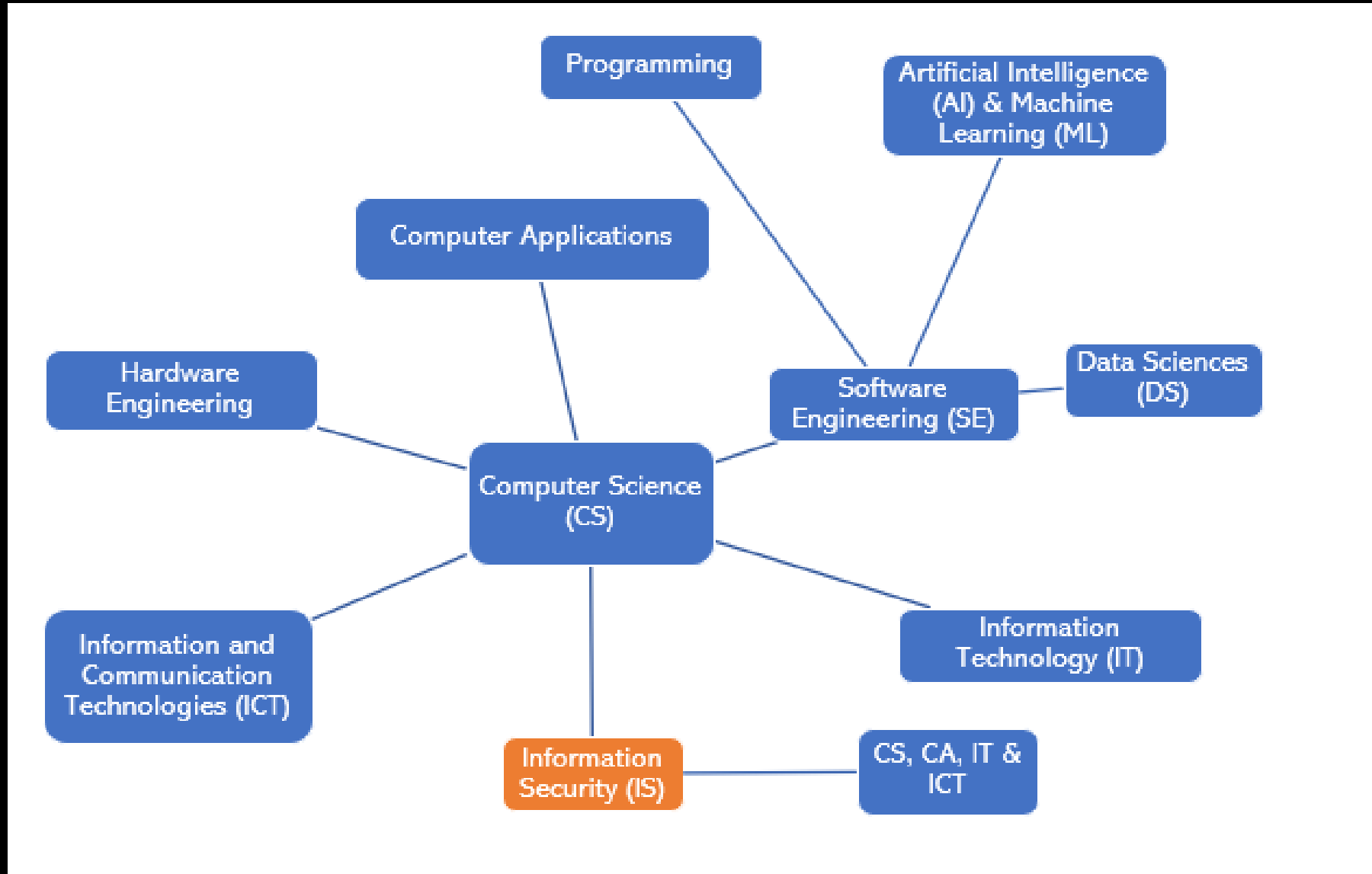
<https://ics-cert-training.inl.gov/learn>



Learn Cloud

AWS IoT Cloud Services:
<https://aws.amazon.com/iot/solutions/industrial-iot/>

Computer Sciences



An extremely simple and fun Case
Study with AWS I played around with
last year just for fun

AWS IoT with AWS Greengrass and Buttons

Raspberry pi 3 model B+



Through this PoC, I tried the following:

1. To use physical hardware to communicate and interact with AWS IoT
2. I used two physical hardware – AWS IoT button and a Raspberry Pi

This short framework consisted of AWS IoT and AWS Greengrass talking to each other securely via certificate-based authentication from Amazon's trust center's root certificate.

It also involves AWS Greengrass successfully communicating with AWS Lambda routines.

One of the interesting parts of this PoC was to use a physical hardware and in this case it was the IoT Button. We not only configured the IoT Button into the AWS IoT console but also onboarded the AWS IoT as part of the Greengrass Core devices through the device onboarding process for the groups.

The installation and setup of Raspberry Pi with AWS IoT and with Greengrass are very similar to EC2 instances with the only difference being the physical hardware setup involved in case of the Pi.

This PoC used the Pi as an example to see how easy or difficult it was to onboard a physical device into a Greengrass group and as a result of this PoC I conclude that except the hardware configuration part wherein you have to not only manage your hardware but also find the right API's that can be used with AWS IOT, once this step is done – AWS Greengrass and Lambda make it every straightforward to communicate with the devices. As per the documentation, new devices such as the such as sensors, motors can also be onboarded using the OPC-UA protocol mechanisms in Greengrass.

This project used MQTT and TLS for communication and security but Greengrass is capable of using other protocols such as MODBUS over TCP.

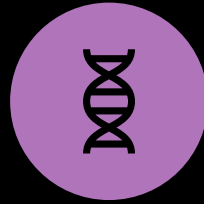
Workflow of my PoC



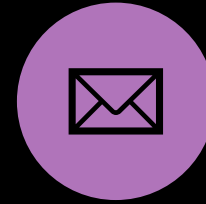
PRESS AMAZON IOT
BUTTON



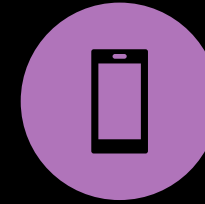
GREENGRASS
RECEIVES THIS



LAMBDA IS
TRIGGERED



AN SNS IS
EXECUTED TO SEND
SMS



MQTT SENDS SMS



MESSAGE IS
RECEIVED ON MY
PHONE

Useful information

Great presentation by Boaz Ziniman @Goto Conference Amsterdam, 2018

<https://www.youtube.com/watch?v=FrH-EQfQkRU>

onboard your iot button using this link

<https://docs.aws.amazon.com/iot/latest/developerguide/iot-console-signin.html>

onboard your raspberry pi using this link

<https://docs.aws.amazon.com/greengrass/latest/developerguide/module1.html>

The instructions are for Greengrass from Module 2 onwards

`recordmydesktop`

`sudo apt-get install recordmydesktop`

launch with `recordmydesktop - -no-sound` on the commandline

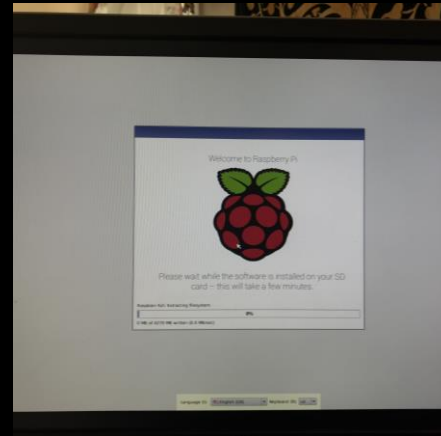
`ctrl+c` to quit

What's in my pi

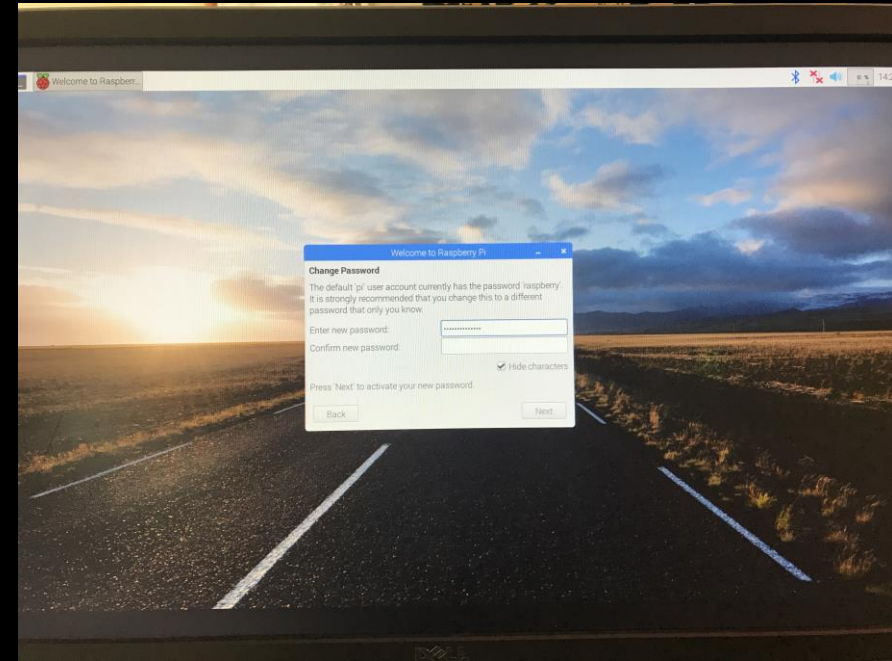
- 1.2 GHz 64-bit quad core CPU, ARM v7 Debian Stretch
- 1 GB RAM
- 4 USB ports
- 1 micro-USB port
- 1 HDMI slot
- 3.5 mm mini jack for AV
- 1 ethernet port
- And all other usual stuffing that go into a raspberry pi 😊
- Bring your own wires, cables, standard keyboard, a mouse and a HDMI or DVI compatible monitor and adaptor
- I also have a NOOBS SD card with Raspbian setup: I used the instructions on Raspberry Pi website to set this up. It took a while a few hurdles but worked in the end.
- I did not find the pre-installed NOOBS card useful as the memory will need expansion later and the process is quirky.
- Take usual care as you would with any delicate electronic chip or board!

Getting your Hardware ready – a minimalist version!

- Use `sudo raspi-config` to enable SSH and VNC on your Raspberry so you can remote login.
- I have the RealVNC software on my Windows to connect as client into my pi.
- Always remember to safely shutdown or reboot your pi either from console or from the Raspberry menu.



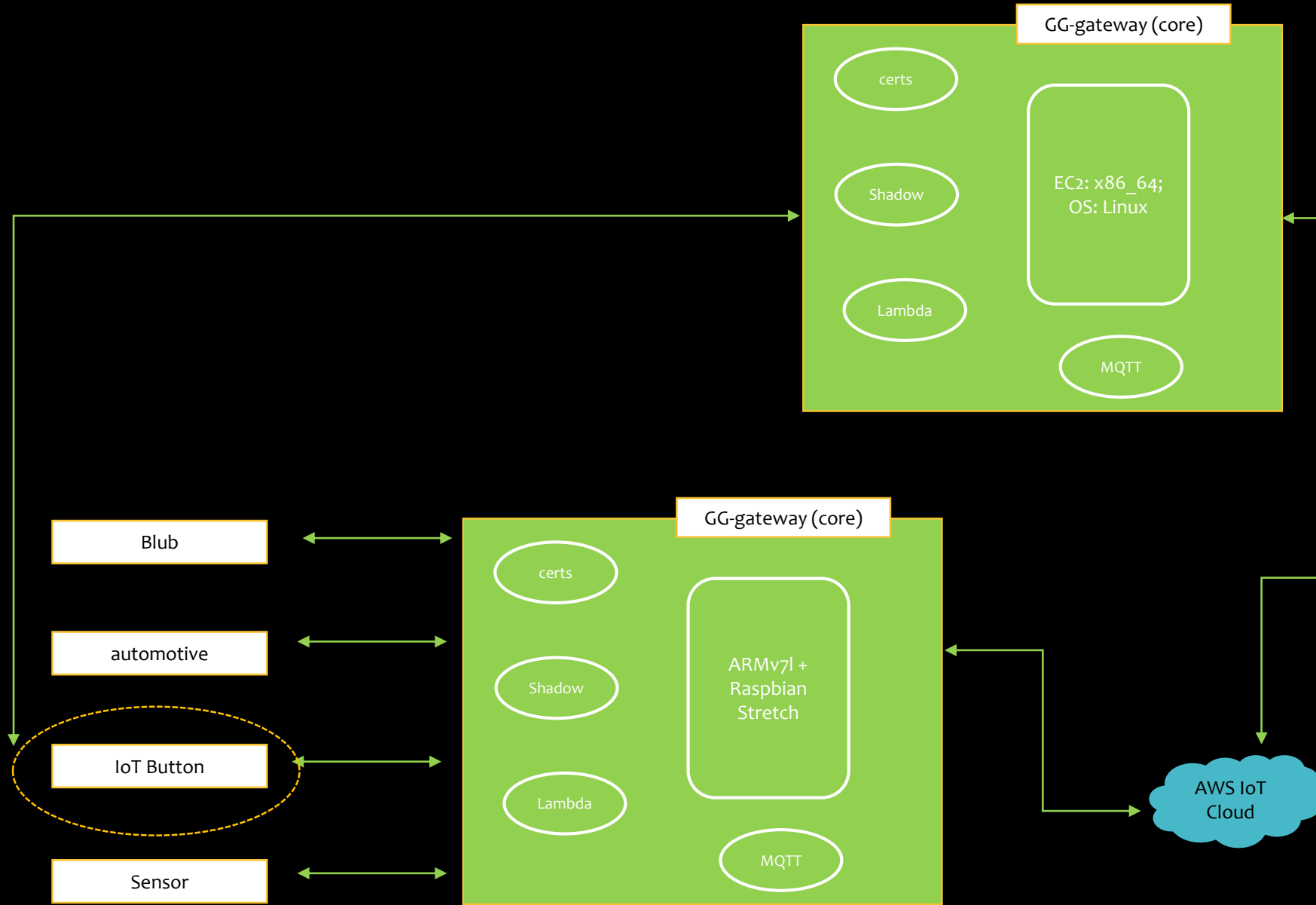
| System | Interfaces | Performance | Localisation |
|-----------------|---|--|--------------|
| Camera: | <input type="radio"/> Enable | <input checked="" type="radio"/> Disable | |
| SSH: | <input checked="" type="radio"/> Enable | <input type="radio"/> Disable | |
| VNC: | <input checked="" type="radio"/> Enable | <input type="radio"/> Disable | |
| SPI: | <input type="radio"/> Enable | <input checked="" type="radio"/> Disable | |
| I2C: | <input type="radio"/> Enable | <input checked="" type="radio"/> Disable | |
| Serial Port: | <input type="radio"/> Enable | <input checked="" type="radio"/> Disable | |
| Serial Console: | <input checked="" type="radio"/> Enable | <input type="radio"/> Disable | |
| 1-Wire: | <input type="radio"/> Enable | <input checked="" type="radio"/> Disable | |



Introduction and overview

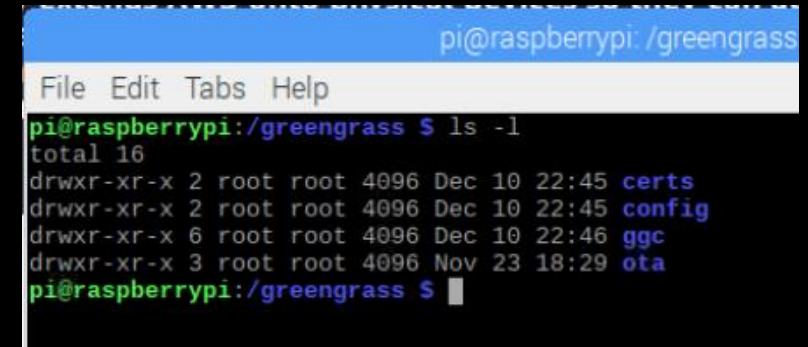
- AWS SDKs, AWS IoT and AWS IoT SDKs
- AWS Greengrass core and SDK
- AWS IoT Button
- AWS Lambda
- AWS SNS with SMS
- AWS EC2 Linux instance
- Python and respective AWS Greengrass Core SDK software
- My aim was to explore how Greengrass could be setup and how if at all make it work with AWS IoT and the buttons.
- Greengrass is found under AWS Internet of Things but has an ecosystem of its own on the console settings
- Greengrass is a software that sits in between your IoT and the edge IoT devices brokering connections across different platforms and protocols, a typical case in an industrial infrastructure.
- With greengrass your IoT device can now talk to IoT either using TCP or MQTT channels
- With Greengrass, your IoT devices need always not remain online and only connect to the cloud when they are due for an update.





Details

- The greengrass root directory is typically /greengrass and the config file is config.json found in the /greengrass/config folder.
- Default MQTT/TLS port is 8883 – there are instructions to change this if needed
- Create a ggc_user and a ggc_group for Greengrass
 - sudo adduser --system ggc_user
 - sudo groupadd --system ggc_group
- Make the greengrass directory read-only for security
- Make the ggc_user the owner of certs and lambda functions
- All steps clearly given in the developer guide – I used it to setup my greengrass. DO NOT USE STEP 10 – it corrupted my kernel and I had to redo everything on my Pi, from SD card formatting!
- MANDATORY STEPS 11 and 12 : to create protection for hardlinks and symbolic links and to enable memory limits for Lambda function.



```
pi@raspberrypi: /greengrass
File Edit Tabs Help
pi@raspberrypi:/greengrass $ ls -l
total 16
drwxr-xr-x 2 root root 4096 Dec 10 22:45 certs
drwxr-xr-x 2 root root 4096 Dec 10 22:45 config
drwxr-xr-x 6 root root 4096 Dec 10 22:46 ggc
drwxr-xr-x 3 root root 4096 Nov 23 18:29 ota
pi@raspberrypi:/greengrass $
```

```

ration:
ecture: x86_64
/bin/init
on: 4.14
NU libc
rsion: 2.17
/var/run: Present
Found
: Found
:: Found
-----Commands and software packages-----
sion: 2.7.14
0: Not found
ot found
ersion: 1.0.2
sent
Present
sent
: Present
: Present
: Present
Present
sent
Present
: Present
-----Platform security-----
inks_protection: Enabled
nks_protection: Enabled
-----User and group-----
user: Present
group: Present
-----Optional Greengrass container dependency check-----
-----Kernel configuration-----
rnel config file: /boot/config-4.14.77-70.59.amzn1.x86_64

mespace configs:
ONFIG_IPC_NS: Enabled
ONFIG_UTS_NS: Enabled
ONFIG_USER_NS: Enabled
ONFIG_PID_NS: Enabled

Cgroup configs:
CONFIG_CGROUP_DEVICE: Enabled
CONFIG_CGROUPS: Enabled
CONFIG_MEMCG: Enabled

Other required configs:
CONFIG_POSIX_MQUEUE: Enabled
CONFIG_OVERLAY_FS: Enabled
CONFIG_HAVE_ARCH_SECCOMP_FILTER: Enabled
CONFIG_SECCOMP_FILTER: Enabled
CONFIG_KEYS: Enabled
CONFIG_SECCOMP: Enabled

```

Before you run the daemon – it helps emotionally to check the dependencies 😊

- cd /home/pi/Downloads
- wget https://github.com/aws-samples/aws-greengrass-samples/raw/master/greengrass-dependency-checker-GGCv1.7.0.zip
- unzip greengrass-dependency-checker-GGCv1.7.0.zip
- cd greengrass-dependency-checker-GGCv1.7.0
- sudo modprobe configs
- sudo ./check_ggc_dependencies | more
- cd /greengrass/ggc/core
- sudo ./greengrassd start

Greengrass setup complete on my EC2 core & the daemon is running !!

```
ec2-user@ip-172-31-40-1:/greengrass/ggc/core
certs/dlc8911d03.private.key
certs/dlc8911d03.public.key
config/config.json
[ec2-user@ip-172-31-40-1 ~]$ cd /greengrass
[ec2-user@ip-172-31-40-1 greengrass]$ ls -l
total 16
drwxr-xr-x 2 root root 4096 Dec 13 15:47 certs
drwxr-xr-x 2 root root 4096 Dec 13 15:47 config
drwxr-xr-x 3 root root 4096 Nov 23 23:24 ggc
drwxr-xr-x 3 root root 4096 Nov 23 23:24 ota
[ec2-user@ip-172-31-40-1 greengrass]$ cd certs
[ec2-user@ip-172-31-40-1 certs]$ sudo wget -O root.ca.pem https://www.amazontrust.com/repository/AmazonRootCA1.pem
--2018-12-13 15:51:51-- https://www.amazontrust.com/repository/AmazonRootCA1.pem
Resolving www.amazontrust.com (www.amazontrust.com)... 13.32.254.63, 13.32.254.199, 13.32.254.21, ...
Connecting to www.amazontrust.com (www.amazontrust.com)|13.32.254.63|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1188 (1.2K) [text/plain]
Saving to: 'root.ca.pem'

root.ca.pem          100%[=====>]          1.16K  --.-KB/s    in 0s

2018-12-13 15:51:51 (216 MB/s) - 'root.ca.pem' saved [1188/1188]

[ec2-user@ip-172-31-40-1 certs]$ cd /greengrass/ggc/core/
[ec2-user@ip-172-31-40-1 core]$ sudo ./greengrassd start
Setting up greengrass daemon
Validating hardlink/softlink protection
Waiting for up to 40s for Daemon to start

Greengrass successfully started with PID: 3072
[ec2-user@ip-172-31-40-1 core]$
```

Now that Greengrass is up and running – I want to see if I can use Lambda to talk to my Greengrass. This is module 3 of the developer guide under the Getting started section.

```
ec2-user@ip-172-31-40-1:~  
[ec2-user@ip-172-31-40-1 ~]$ sudo tar -xvzf greengrass-core-python-sdk-1.3.0.tar.gz  
aws_greengrass_core_sdk/  
aws_greengrass_core_sdk/examples/  
aws_greengrass_core_sdk/examples/HelloWorld/  
aws_greengrass_core_sdk/examples/HelloWorld/greengrassHelloWorld.py  
aws_greengrass_core_sdk/examples/TES/  
aws_greengrass_core_sdk/examples/TES/README  
aws_greengrass_core_sdk/examples/TES/lambda_function.py  
aws_greengrass_core_sdk/examples/BinaryLambdaInvoke/  
aws_greengrass_core_sdk/examples/BinaryLambdaInvoke/invoker.py  
aws_greengrass_core_sdk/examples/BinaryLambdaInvoke/invoker.py  
aws_greengrass_core_sdk/manual/  
aws_greengrass_core_sdk/manual/lambda.html  
aws_greengrass_core_sdk/manual/secretsmanager.html  
aws_greengrass_core_sdk/manual/quickstart.html  
aws_greengrass_core_sdk/manual/index.html  
aws_greengrass_core_sdk/manual/iot-data.html  
aws_greengrass_core_sdk/sdk/  
aws_greengrass_core_sdk/sdk/python_sdk_1_3_0.zip  
aws_greengrass_core_sdk/NOTICE  
aws_greengrass_core_sdk/LICENSE  
[ec2-user@ip-172-31-40-1 ~]$ ls -l  
total 12480  
drwx----- 5 6319015 users      4096 Nov 26 07:03 aws_greengrass_core_sdk  
-rwxrwxr-x 1 ec2-user ec2-user  1275 Dec 13 15:22 cgroupfs-mount.sh  
-rw-rw-r-- 1 ec2-user ec2-user  2872 Dec 13 15:38 d1c8911d03-setup.tar.gz  
-rw-rw-r-- 1 ec2-user ec2-user 28061 Dec 13 16:01 greengrass-core-python-sdk-1.3.0.tar.gz  
drwxr-xr-x 2 ec2-user ec2-user   4096 Dec 13 15:24 greengrass-dependency-checker-GGCv1.7.0  
-rw-rw-r-- 1 ec2-user ec2-user 21072 Dec 13 15:23 greengrass-dependency-checker-GGCv1.7.0.zip  
-rw-rw-r-- 1 ec2-user ec2-user 12706252 Dec 13 15:38 greengrass-linux-x86-64-1.7.0.tar.gz  
[ec2-user@ip-172-31-40-1 ~]$
```

Next, I have downloaded the required AWS IoT Greengrass Core SDK software in this case

```
ec2-user@ip-172-31-40-1:~/aws_greengrass_core_sdk/examples/HelloWorld
[ec2-user@ip-172-31-40-1 sdk]$ ls -l
total 20
drwxr-xr-x 3 root root 4096 Nov 26 08:50 greengrasssdk
-rw-rw-rw- 1 ec2-user users 12949 Nov 26 10:13 python_sdk_1_3_0.zip
[ec2-user@ip-172-31-40-1 sdk]$ cd greengrasssdk/
[ec2-user@ip-172-31-40-1 greengrasssdk]$ ls -l
total 52
-rw-r--r-- 1 root root 469 Nov 14 01:25 client.py
-rw-r--r-- 1 root root 194 Nov 14 01:25 __init__.py
-rw-r--r-- 1 root root 5177 Nov 14 01:25 IoTDataPlane.py
-rw-r--r-- 1 root root 5623 Nov 14 01:25 Lambda.py
-rw-r--r-- 1 root root 10884 Nov 26 07:31 LICENSE
-rw-rw-r-- 1 root root 209 Nov 26 07:31 NOTICE
-rw-r--r-- 1 root root 7955 Nov 14 01:25 SecretsManager.py
drwxr-xr-x 2 root root 4096 Nov 14 01:25 utils
[ec2-user@ip-172-31-40-1 greengrasssdk]$ cd ../../
[ec2-user@ip-172-31-40-1 aws_greengrass_core_sdk]$ ls -l
total 28
drwxrwxrwx 5 ec2-user users 4096 Nov 22 00:27 examples
-rw-rw-rw- 1 ec2-user users 10884 Nov 26 06:43 LICENSE
drwxrwxrwx 2 ec2-user users 4096 Nov 26 21:08 manual
-rw-rw-rw- 1 ec2-user users 209 Nov 26 07:03 NOTICE
drwxrwxrwx 3 ec2-user users 4096 Dec 13 16:15 sdk
[ec2-user@ip-172-31-40-1 aws_greengrass_core_sdk]$ cd examples
[ec2-user@ip-172-31-40-1 examples]$ cd HE
-bash: cd: HE: No such file or directory
[ec2-user@ip-172-31-40-1 examples]$ cd HelloWorld/
[ec2-user@ip-172-31-40-1 HelloWorld]$ ls -l
total 4
-rwxrwxrwx 1 ec2-user users 1871 Nov 24 01:59 greengrassHelloWorld.py
[ec2-user@ip-172-31-40-1 HelloWorld]$
```

I zipped up the folder like the guide instructs.

```
sudo zip -r hello_world_python_lambda.zip greengrasssdk
greengrassHelloWorld.py
```

Management Co... pi@thegreatbritishba... Downloads

IAM Management x

console.aws.amazon.com/iam/home?region=us-west-2#/roles/ggServiceRole

Resource Groups Lambda IAM IoT Core

roles > ggServiceRole

Summary

| | |
|----------------------------------|---------------|
| Role ARN | arn:aws:iam:: |
| Role description | Allows G |
| Instance Profile ARNs | |
| Path | / |
| Creation time | 2018-1 |
| Maximum CLI/API session duration | 1 hou |

Permissions Trust relationships Tags Access Advi

Permissions policies (1 policy applied)

Attach policies

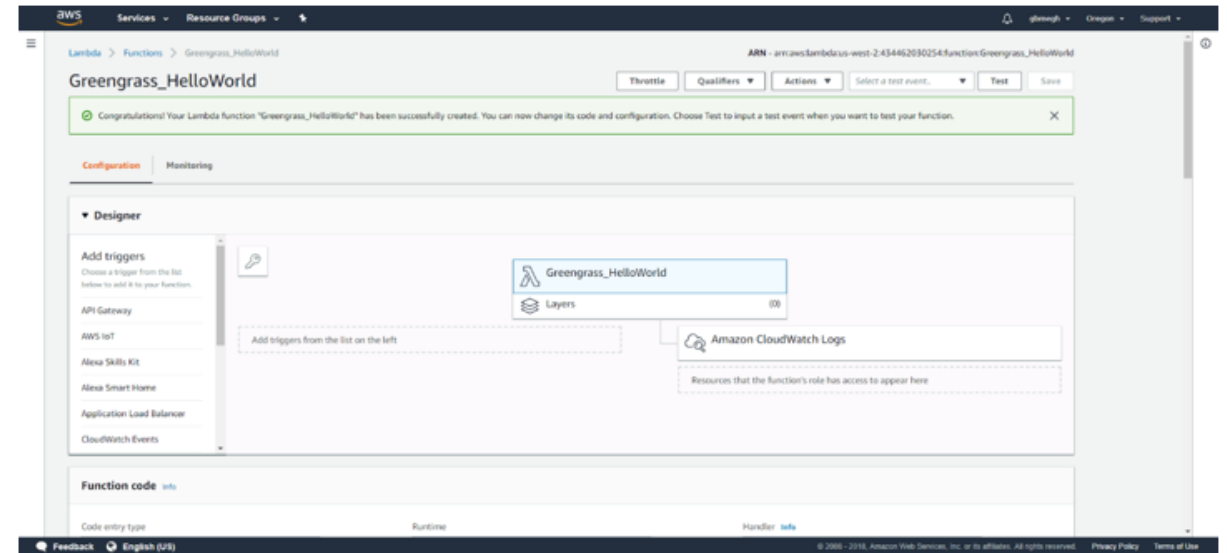
Policy name ▾

▶ AWSGreengrassResourceAccessRolePolicy

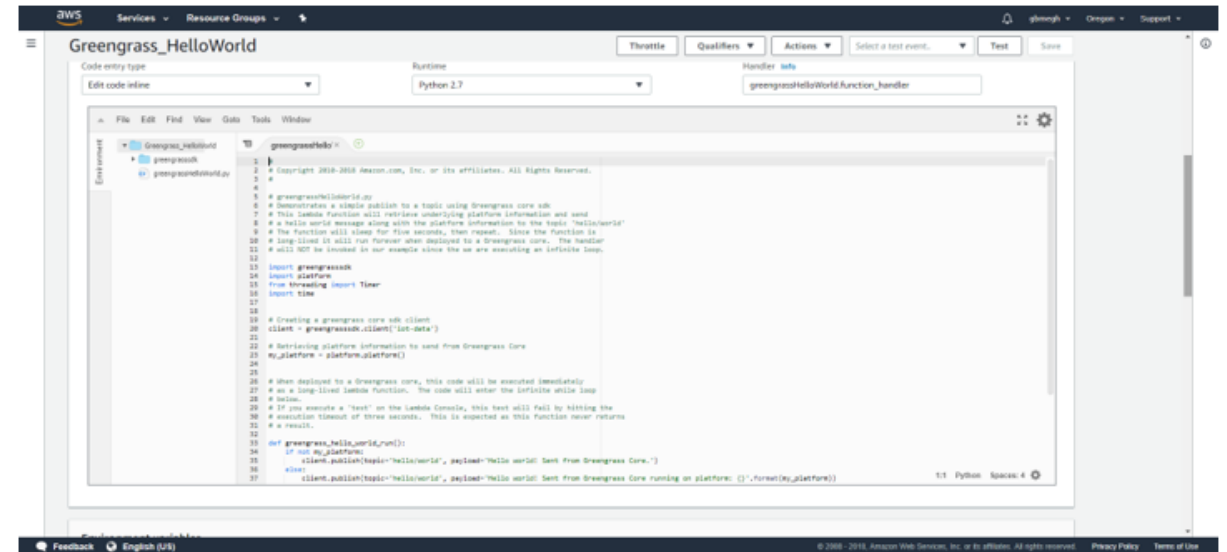
Servicerole permissions error with Greengrass!

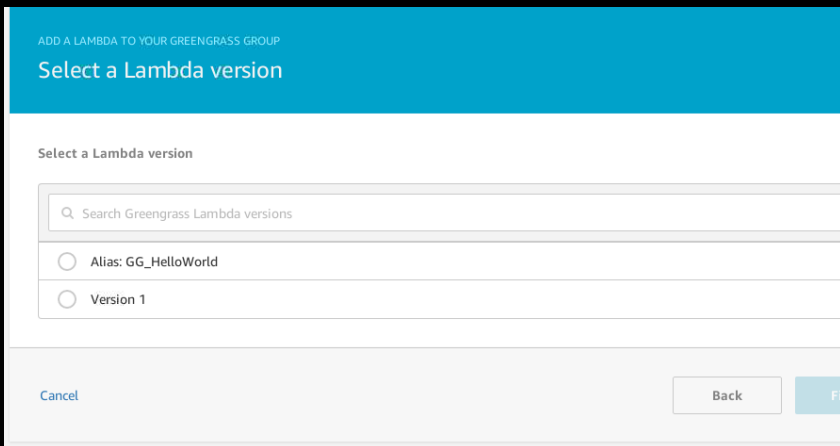
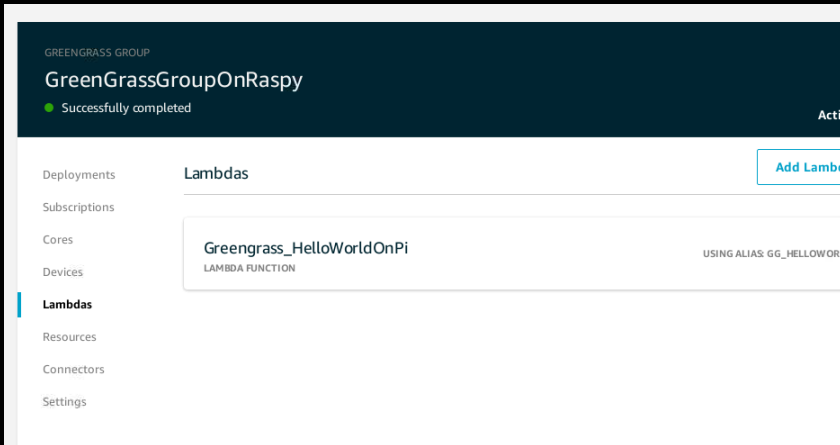
- This one took me a while to sort out, but it worked in the end!
- This role assignment is important for Greengrass and Lambda to work.

Next, I created the Lambda function and upload the deployment zip file.



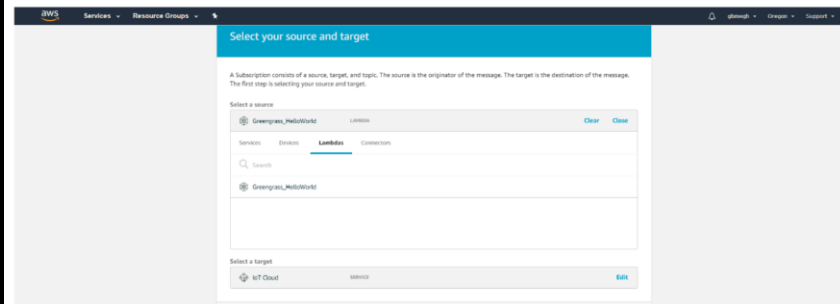
We have successfully imported our greengrass helloworld zip file into our Lambda function code.





and Pi nodes. The next step is to create the topic subscription so that MQTT can handle.

On the Subscriptions page for Greengrass – choose Lambda Tab and it should show the Lambdas that you created in previous steps and for Services tab, choose IoT Cloud. See the following image.



Here's one on my Raspy

- Choose the Greengrass group
- Choose Lambdas option on the left
- Choose the alias you need
- Create subscription


```
"message": "Hello from AWS IoT console"
```

```
nt from Greengrass Core running on platform: Linux-4.14.79-v7+-armv7l-with-debian-9.4
```

Dec 13, 2018 1:53:30 PM -0500

```
from Greengrass Core running on platform: Linux-4.14.79-v7+-armv7l-with-debian-9.4
```

Dec 13, 2018 1:53:25 PM -0500

```
Greengrass Core running on platform: Linux-4.14.79-v7+-armv7l-with-debian-9.4
```

Dec 13, 2018 1:53:20 PM -0500

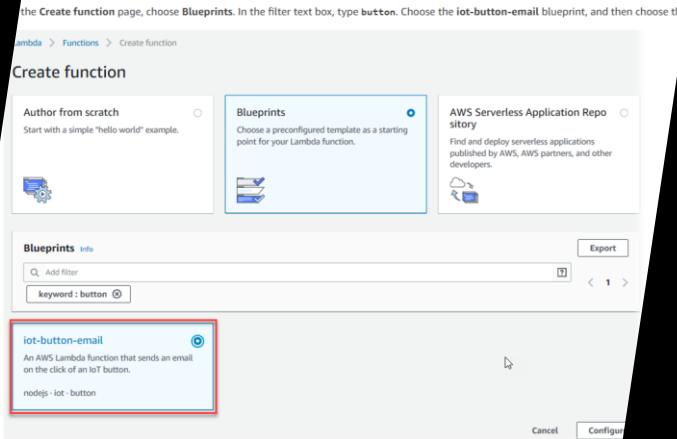
```
engrass Core running on platform: Linux-4.14.79-v7+-armv7l-with-debian-9.4
```

Dec 13, 2018 1:53:15 PM -0500

```
engrass Core running on platform: Linux-4.14.79-v7+-armv7l-with-debian-9.4
```

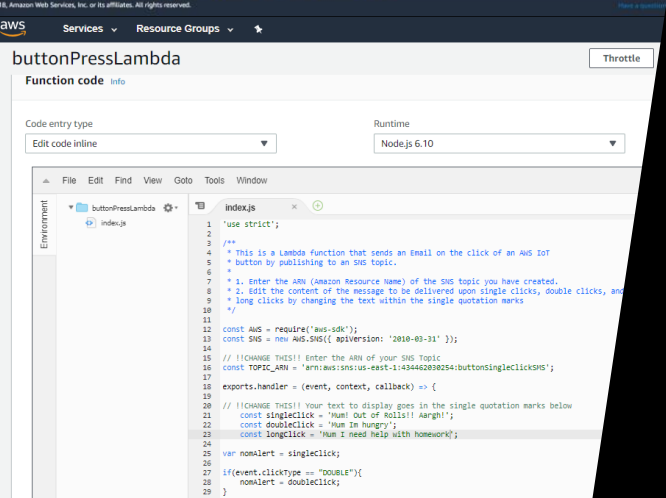
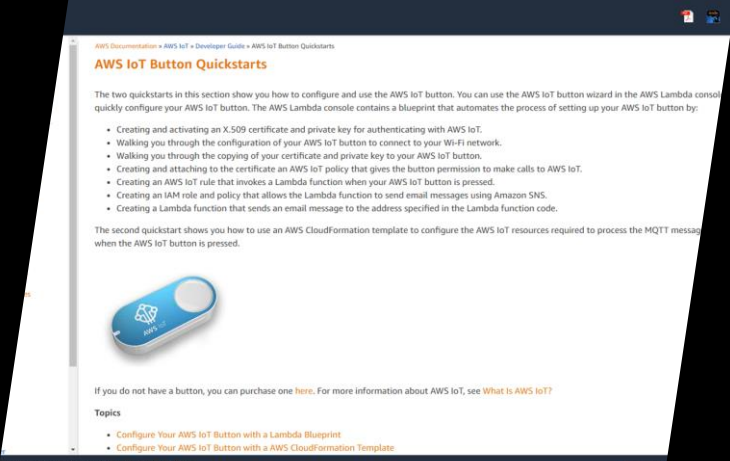
Dec 13, 2018 1:53:10 PM -0500

- Ah, the sweet sight of success, even with the hello/world!



IoT Buttons

- I used the AWS IoT Button QuickStart guide to setup my new button with the AWS Console.
- <https://docs.aws.amazon.com/iot/latest/developerguide/iot-button-quickstart.html>
- Rampant technical glitches – I could see the IoT blueprints once, and not now – wonder what's going on!
- But used the GitHub code to try it out.
- <https://github.com/aws-samples/aws-lambda-iot-button>



■ Configuring your amazon iot button is very simple!

Copy your device certificate and private key onto your AWS IoT button

To connect to AWS IoT, you must copy your device certificate and private key onto the AWS IoT button.

1. In a browser, navigate to <http://192.168.0.1/index.html>.

2. Complete the configuration form:

- Type your Wi-Fi SSID and password.
- Browse to and select your certificate and private key. For example, 2a540e2346-certificate.pem.crt and 2a540e2346-private.pem.key, respectively.
- Find your custom endpoint in the AWS IoT console. (From the dashboard, in the left navigation pane, choose **Manage**, and then choose **Things**. Select the box representing your button in the left navigation pane, choose **Interact** and look for the **HTTPS** section, near the top.) Your endpoint will look something like the following:

```
ABCDEFG1234567.iot.us-east-2.amazonaws.com
```

- where ABCDEFG1234567 is the subdomain and us-east-2 is the region.
- On the **Button ConfigureMe** page, type the subdomain, and then choose the region that matches the region in your AWS IoT endpoint.
- Select the **Terms and Conditions** check box. Your settings should now look like the following:

Button ConfigureMe

Enter the value for any field that you wish to change for device: G030JF055364XVRB

Wi-Fi Configuration:

SSID:

Security: Open Network(No Password)

Password:

AWS IoT Configuration:

Certificate:

Private Key:

Endpoint Subdomain:

Endpoint Region:

Final Endpoint:

By clicking this box, you agree to the [AWS IoT Button Terms and Conditions](#).

- Choose **Configure**. Your button should now connect to your Wi-Fi network.

the topic on which your thing publishes. In the case of the AWS IoT button, you can subscribe to iotbutton/+ (note that + is the wildcard character). In **Subscribe to a topic**, in the **Subscription topic** field, type iotbutton/+

MQTT client Connected as iotconsole-1509388246394-0

Subscriptions

[Subscribe to a topic](#)

[Publish to a topic](#)

Subscribe
Devices publish MQTT messages on topics. You can use this client to subscribe to a topic and receive these messages.

Subscription topic
 [Subscribe to topic](#)

Max message capture

Quality of Service

0 - This client will not acknowledge to the Device Gateway that messages are received

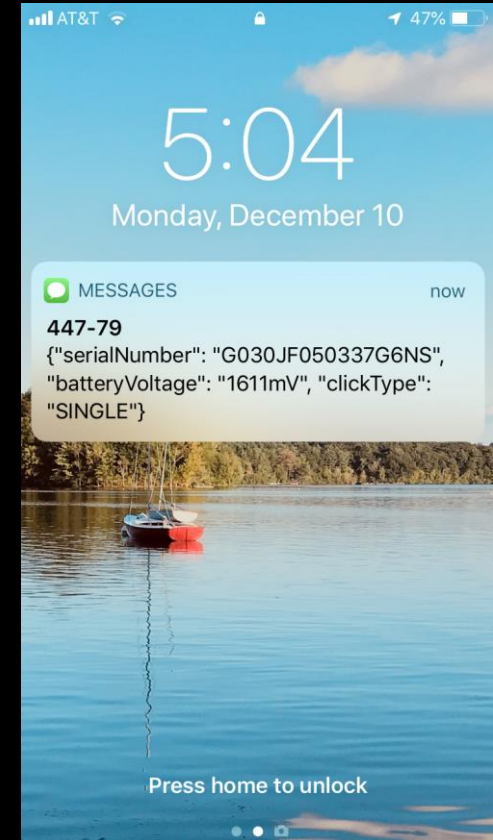
1 - This client will acknowledge to the Device Gateway that messages are received

MQTT payload display

Auto-format JSON payloads (improves readability)

Display payloads as strings (more accurate)

Display raw payloads (in hexadecimal)



Details on rules

The lambda captures 3 events:

- Single click
- Double click
- Long click

It has three different messages for each clickType as you see below in the test on my phone.

The screenshot shows the AWS IoT console configuration for a rule named "buttonPressRuleSMS". The rule is currently "ENABLED". The "Overview" tab is selected, showing the following details:

- Description:** No description. (Edit)
- Rule query statement:** `select * from 'iotbutton/+'`. (Edit)
- Using SQL version:** 2016-03-23
- Actions:** One action is configured: "Invoke a Lambda function passing the message ..." (buttonPressLambda). (Remove, Edit)
- Error action:** No error action is currently configured. (Add action)

The screenshot shows an MQTT client interface with a subscription to the topic "iotbutton/+". A message has been received from the topic "iotbutton/G030JF050337G6NS" at "Dec 12, 2018 7:39:35 PM -0500". The message payload is a JSON object:

```
{
  "serialNumber": "G030JF050337G6NS",
  "batteryVoltage": "1617mV",
  "clickType": "LONG"
}
```

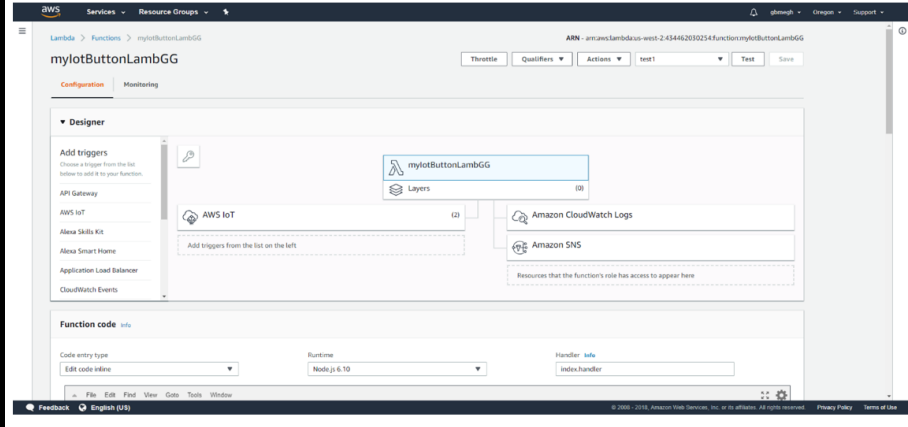
The screenshot shows an iPhone text message conversation. The messages are:

- Mum! Out of Rolls!! Aargh!
- buttonSingleClickSMS> Mum! Out of Rolls!! Aargh!
- buttonSingleClickSMS> Mum Im hungry
- buttonSingleClickSMS> Mum Im hungry
- buttonSingleClickSMS> Mum Im hungry
- buttonSingleClickSMS> Mum I need help with homework
- buttonSingleClickSMS> Mum I need help with homework

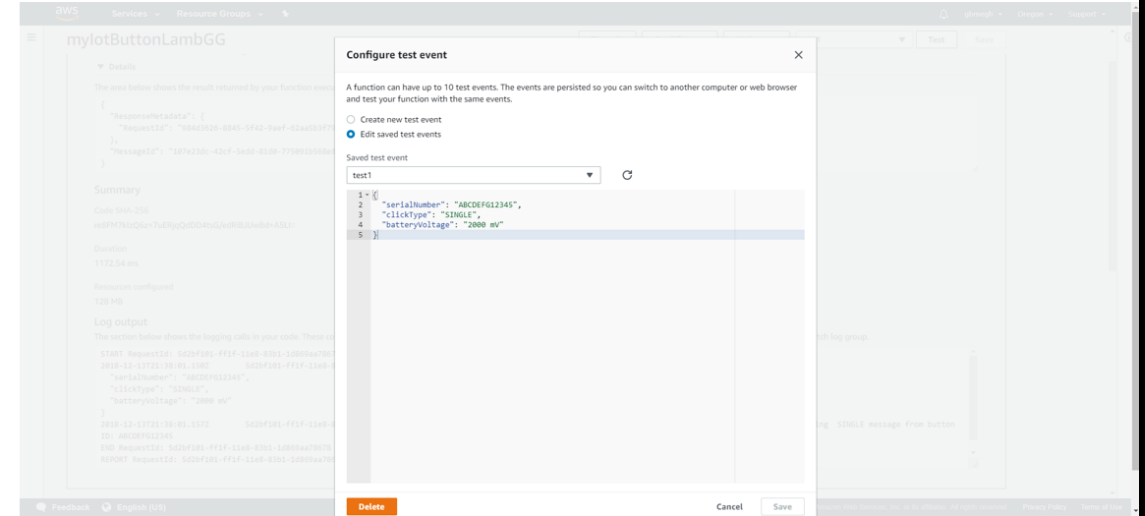
This link provides a really simple json script to send a quick text message via Lambda through SNS→SMS options to my phone.

<https://docs.aws.amazon.com/iot/latest/developerguide/iot-lambda-rule.html>

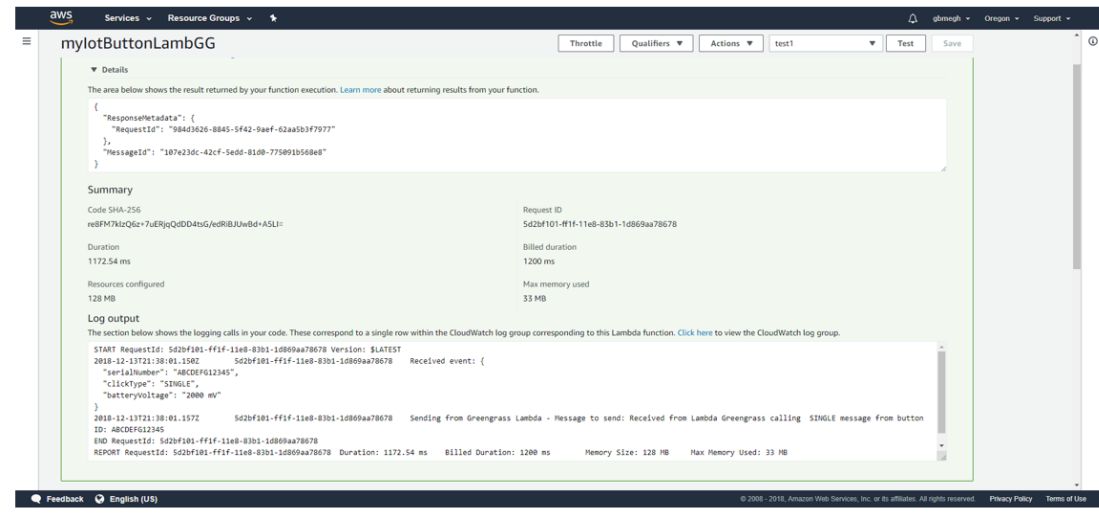
I followed the steps using the blueprint as a start and onboarded my IoT Button into the Lambda as per the AWS IoT setup instructions within this page.



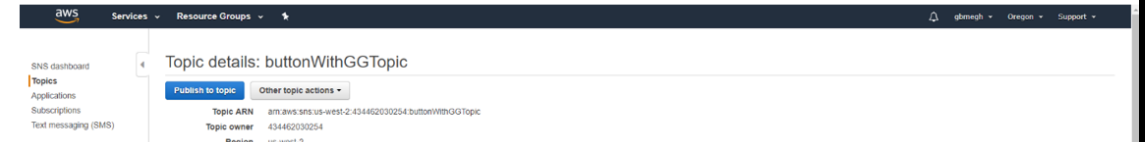
Using the following method



This completes my AWS IoT+Lambda communication as I have tested it within the console



Of course, I had to have my SNS topic created and setup for SMS.



GREENGRASS GROUP

GreengrassCoreSWOnEC2

● Successfully completed Actions ▾

Deployments Group history overview By deployment ▾

| Deployed | Version | Status |
|-------------------------------|--------------------------------------|-------------------------------|
| Dec 13, 2018 4:14:54 PM -0500 | 4a9d5dfa-6828-428c-9c68-3efc1ec63135 | ● Successfully complet... *** |
| Dec 13, 2018 3:28:16 PM -0500 | 2479534d-47fb-48a0-a24e-e479ec773bc8 | ● Successfully complet... *** |
| Dec 13, 2018 3:20:54 PM -0500 | bf809de4-be3e-421a-92c4-501ca3427454 | ● Successfully complet... *** |
| Dec 13, 2018 3:18:43 PM -0500 | bf809de4-be3e-421a-92c4-501ca3427454 | ● Successfully complet... *** |
| Dec 13, 2018 2:55:57 PM -0500 | 5f069fde-1d92-450a-9b4b-f8378e0d2f55 | ● Successfully complet... *** |
| Dec 13, 2018 2:15:54 PM -0500 | d6ec7790-fb26-46a2-b73c-f161319ebe6e | ● Successfully complet... *** |
| Dec 13, 2018 2:13:18 PM -0500 | a7dd68ce-4d1d-4c37-9d23-62d18c64c18b | ● Failed *** |
| Dec 13, 2018 2:13:00 PM -0500 | 9329f705-1576-4bdd-8d03-3719a5d5966e | ● Failed *** |
| Dec 13, 2018 2:06:05 PM -0500 | f8a5b977-a58c-48ea-a731-92b8ba1a4a87 | ● Successfully complet... *** |
| Dec 13, 2018 1:46:14 PM -0500 | 757b6621-8161-4c7d-8e6e-a55b8c99b937 | ● Successfully complet... *** |

n.com/iot/home?region=us-west-2#/greengrass/groups/51269e0f-9d21-468c-aa23-b52e512c4f1b

Lambda IAM IoT Core

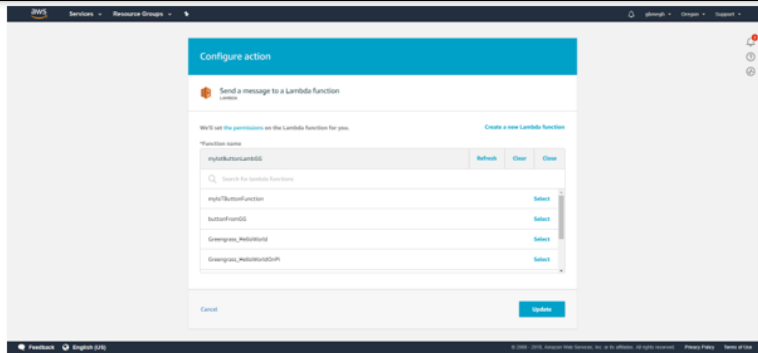
GREENGRASS GROUP

GreenGrassGroupOnRaspy

● Successfully completed

Deployments Group history overview By deployment

| Deployed | Version | Status |
|--------------------------------|--------------------------------------|------------|
| Dec 13, 2018 2:00:53 PM -0500 | 287caf15-42ef-4f53-a008-2b91696b6c6e | ● Successf |
| Dec 13, 2018 1:45:50 PM -0500 | 5ce9d3ee-c895-417e-b03c-4f75590a783d | ● Successf |
| Dec 13, 2018 12:16:19 PM -0500 | 5ce9d3ee-c895-417e-b03c-4f75590a783d | ● Failed |



In my case this was `"myIoTButtonLambGG"`

Once this is complete. Head over to AWS IoT → Test option and give your topic as `"iotbutton/<YOUR BUTTON's DSN found on the back of the hardware>"`

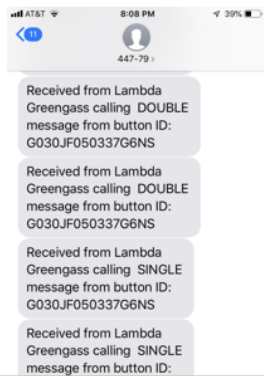
Now lets test it:

Press the button once.

I see my message as per the IoT+Lambda script!

Press the button twice in quick succession to see the logic trigger for `clickType=Double`

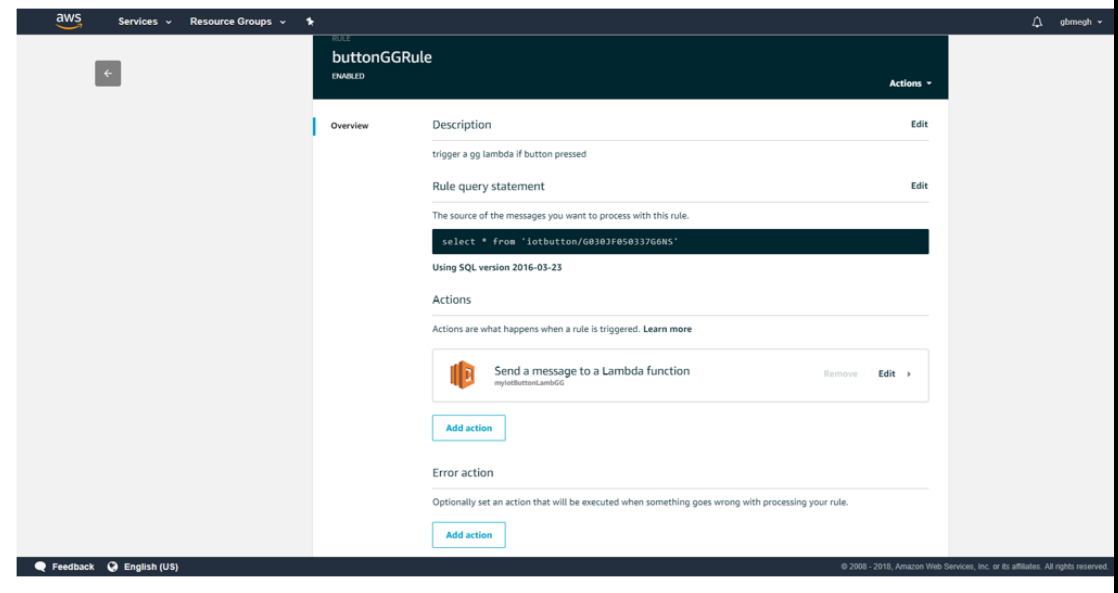
Press the button long enough for a few seconds to trigger the third logic! Viola! We have successfully controlled our Lambda with our little IoT Button and both of these are being run by our Greengrass core. Here's the sample of the messages I got on my phone.



Lets now create a rule in the IoT console to trigger this lambda o the press of the G friendly IoT Button!

Choose Act -> Create

In the Create a Rule dialog box – setup the rule as you would for any MQTT rule. I specified a simple query to fetch everything!



What else should I learn?

- Learn MITRE's threat modeling: a good intro is here:
- <https://digitalguardian.com/blog/what-mitre-attck-framework>
- Learn about Kill Chains - there are various - start here: <https://www.varonis.com/blog/mitre-attck-framework-complete-guide/> and here: <https://medium.com/datadriveninvestor/att-ck-model-c40a113aab4>

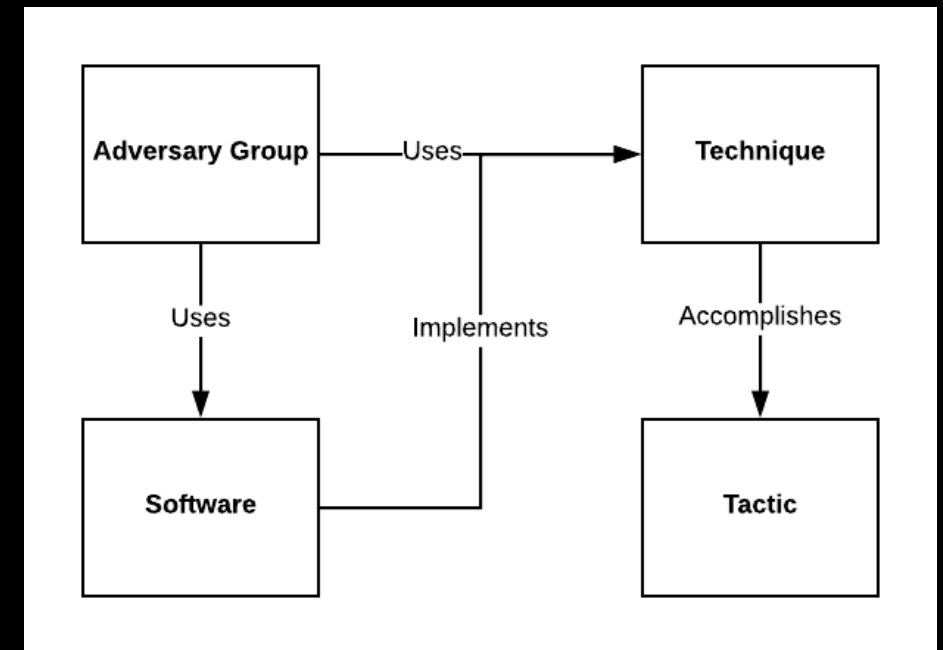
MITRE ATT&CK vs. CYBER KILL CHAIN

MITRE ATT&CK

- Initial Access
- Execution
- Persistence
- Privilege Escalation
- Defense Evasion
- Credential Access
- Discovery
- Lateral Movement
- Collection
- Exfiltration
- Command and Control

Cyber Kill Chain

- Reconnaissance
- Intrusion
- Exploitation
- Privilege Escalation
- Lateral Movement
- Obfuscation/
Anti-forensics
- Denial of Service
- Exfiltration

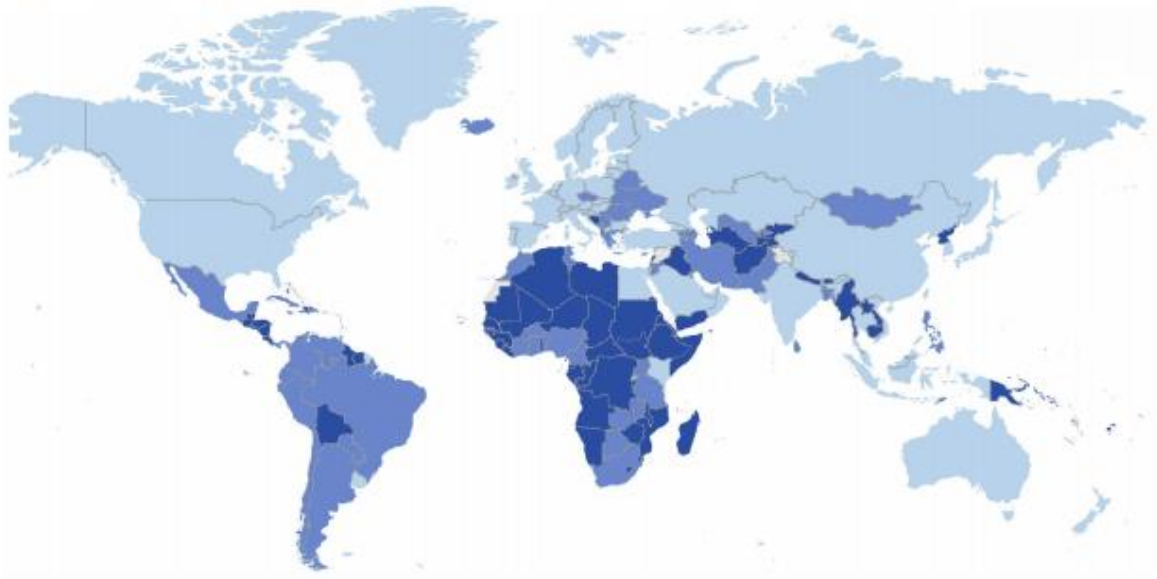


International Relations & International Security

- Take courses in International Relations
- Study how various nations perceive cybersecurity
- Take courses in International Security
- Study what international laws apply to the field of cyber security
- A good place to start would be the NATO website
- The Tallinn 2.0 manual is a great resource to understand cyber laws and other international laws that apply in a cyberspace conflict



Figure 4: Heat map showing geographical commitment around the world



Disclaimer: The designations employed and the presentation of this map do not imply the expression of any opinion whatsoever on the part of the ITU concerning the legal status of any country, state, territory or area and/or of its authorities, or concerning the delimitation of its boundaries or frontiers. Efforts were made to ensure this map is free of errors however there is no warranty the map or its features are either spatially or temporally accurate or fit for a particular use. This map is provided without any warranty of any kind whatsoever, either express or implied.

Source map: UN.org

The colours in the heat map above indicate differences in the level of commitment with high, medium, and low scores in a range of colours from light blue (peak commitment) to dark blue (low commitment). This is also reflected in the GCI groups in section 4.2.

4.2 GCI groups

Countries are classified according to their level of commitment: high, medium, and low.

1.  Countries that demonstrate high commitment in all five pillars of the index.
2.  Countries that have developed complex commitments and engage in cybersecurity programmes and initiatives.
3.  Countries that have started to initiate commitments in cybersecurity.

Global Cybersecurity Index

https://www.itu.int/dms_pub/itu-d/opb/str/D-STR-GCI.01-2018-PDF-E.pdf



FILTERS

Search within results...



VIEW: BOXES ▾

SORT: NEWEST TO OLDEST ▾



Cybersecurity Within IT and ICS Domains

English

FREE



Influence of IT Components on

English

FREE



Differences in Deployments of

English

FREE



Cybersecurity Practices for Industrial C...

English

FREE



210W-10 Cybersecurity for Industrial Con...

English

FREE



210W-09 Cybersecurity for Industrial Con...

English

FREE



210W-08 Cybersecurity for Industrial Con...

English

FREE



210W-07 Cybersecurity for Industrial Con...

English

FREE



210W-06 Cybersecurity for Industrial Con...

English

FREE



210W-05 Cybersecurity for Industrial Con...

English

FREE



210W-03 Cybersecurity for Industrial Con...

English

FREE

US DHS VLP provides resources to those interested to make a career in the field
<https://ics-cert-training.inl.gov/learn>

The Learning Map

