eclipsecon 2015

The 5 Elements of IoT Security

Julien Vermillard - Sierra Wireless

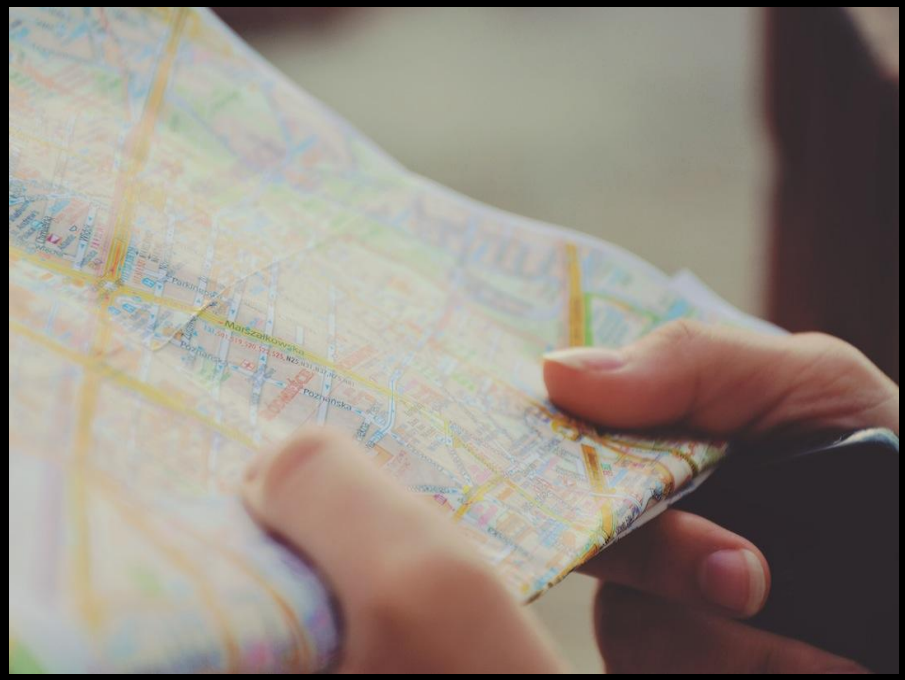
Who am I?

Software Engineer **Sierra Wireless**
AirVantage.net cloud service

Eclipse IoT:
Leshan project lead
Wakaama and **Californium** committer

Twitter: @vrmvrm
Email: jvermillard@sierrawireless.com

Agenda



In the news

Hardware

OTA Upgrades

Secure Communication

Key Distribution

Cloud Security

Open Source IoT Infrastructure

In the news

"The killer toaster"

"The nightmare on connected home street"

"What's wrong with connected devices"

HP Fortify 2014 IoT security report

Reviewed the most popular devices:

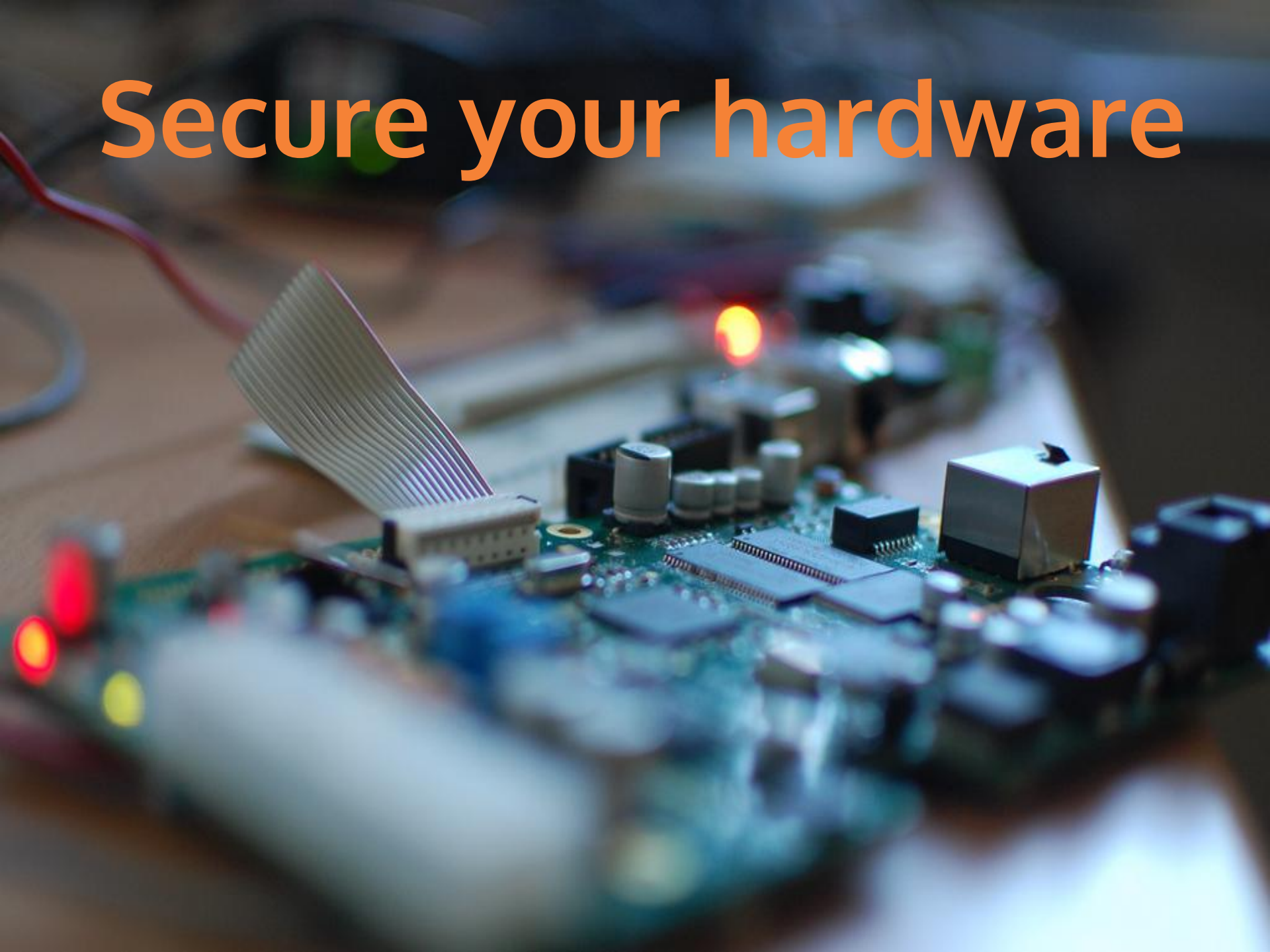
TVs, webcams, thermostats, power
outlets, sprinkler controllers,
hubs for controlling multiple devices,
door locks, home alarms,scales,
and garage door openers

90% collected personal data

70% used unencrypted network services

http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA5-4759ENW&cc=us&lc=en

Secure your hardware



Hardware security

Risks:

Rogue firmware

Invisible backdoor

Malicious certificate

Eavesdropping

Mitigation:

Secure storage

Secure boot

Drawbacks:

Vendor lock

Tivoization

Nest Example:

https://www.blackhat.com/docs/us-14/materials/us-14-Jin-Smart-Nest-Thermostat-A-Smart-Spy-In-Your-Home.pdf

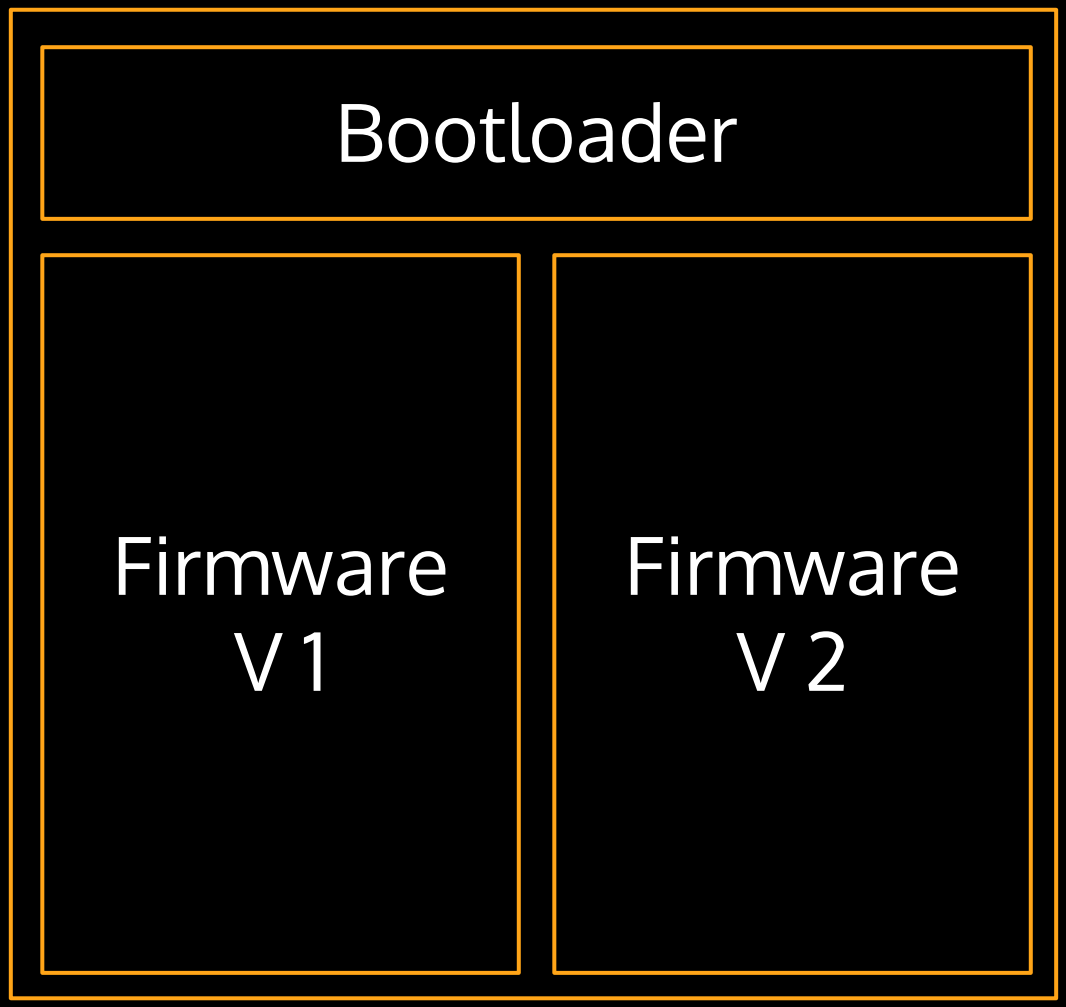

You can’t secure
what you can’t update

High engineering and BoM cost!

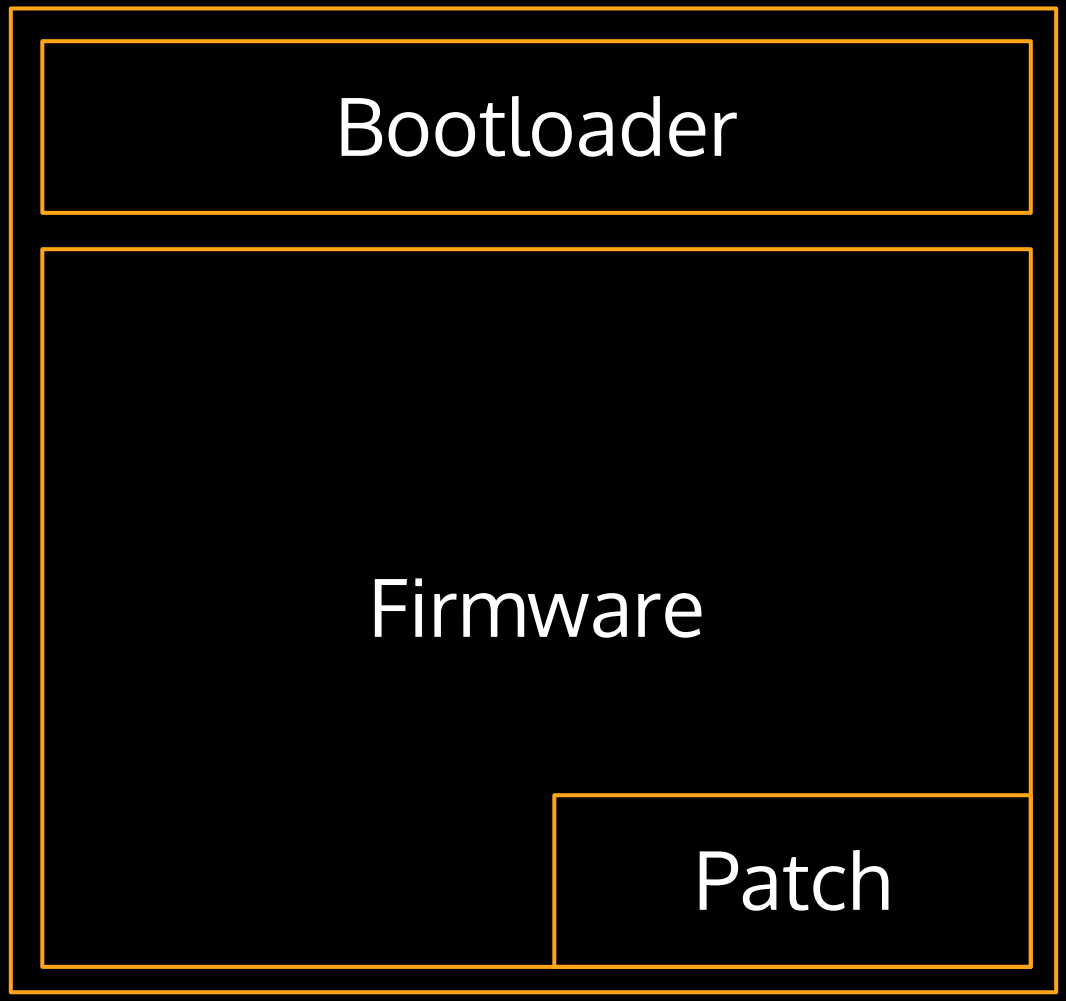
Custom bootloader

Flash size

Flash & switch update



Reboot & Patch update



Must be bulletproof

Upgrading is hard:

- NAND flash errors
- Unexpected power loss
- Network errors
- Unexpected incompatibilities
- Checksum, cryptographic signature

A 0.1% failure rate on a 1m fleet is 1000 bricked devices

Secure Communication

Cipher suite? Pre-shared key

TLS_PSK_WITH_AES_128_CCM_8

Client and server have a common secret

Symmetric cryptography

Tampering the device or the server give you access to all the future and past communications

Secure communication is not cheap

https://tools.ietf.org/html/draft-ietf-lwig-tls-minimal-01

	DTLS	
	ROM	RAM
State Machine	8.15	1.9
Cryptography	3.3	1.5
DTLS Record Layer	3.7	0.5
TOTAL	15.15	3.9

Table 1: Memory Requirements in KB

Cipher suite? Public Key

TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8

Server and client don’t share private keys, only **public keys**

Perfect forward secrecy: past communication can’t be decrypted after secret compromission

X.509 Certificate

Chain-of-trust for validating identity!

No more credential provisioning

Used for HTTPS

Certificate: revocation checks

Revocation checking is still an issue in 2015:

Validity date checking: RTC? NTP?

More and more complexity on the device side:

CRL, OCSP, stapling

Hard fail? Soft fail? Certificate pinning?

Pre-shared key vs X.509?

PSK is lighter, can run on very small target
X.509 crypto is heavier: (EC)DH,ECDSA/RSA

PSK Infrastructure is simpler but weaker
(Hello SIM card key files)

X.509 Public Key Infrastructure is complex, but can be outsourced

Key Distribution

Pre-shared key generation

Everything should be provisioned at factory?

Don’t move big plain text list of credentials

Don’t use stupid formulas:

```
password = MD5(IMEI + CARRIER_NOT_SO_SECRET)
```

https://www.blackhat.com/docs/us-14/materials/us-14-Solnik-Cellular-Exploitation-On-A-Global-Scale-The-Rise-And-Fall-Of-The-Control-Protocol.pdf

Secret rotation

Be sure to be ready to change them ASAP

Don’t wait the next Heartbleed for doing it 😄

Good practice:

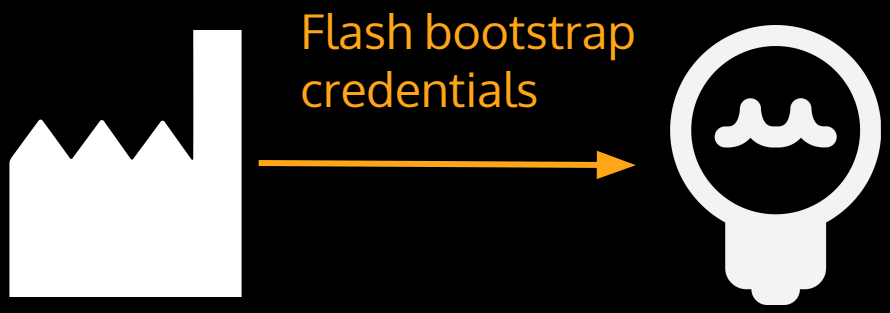
Changing the factory credential during the 1st communication

Key management protocols?

Enterprise PKI for X.509: CMP, OCSP

For PSK or X.509: Lightweight M2M bootstrap

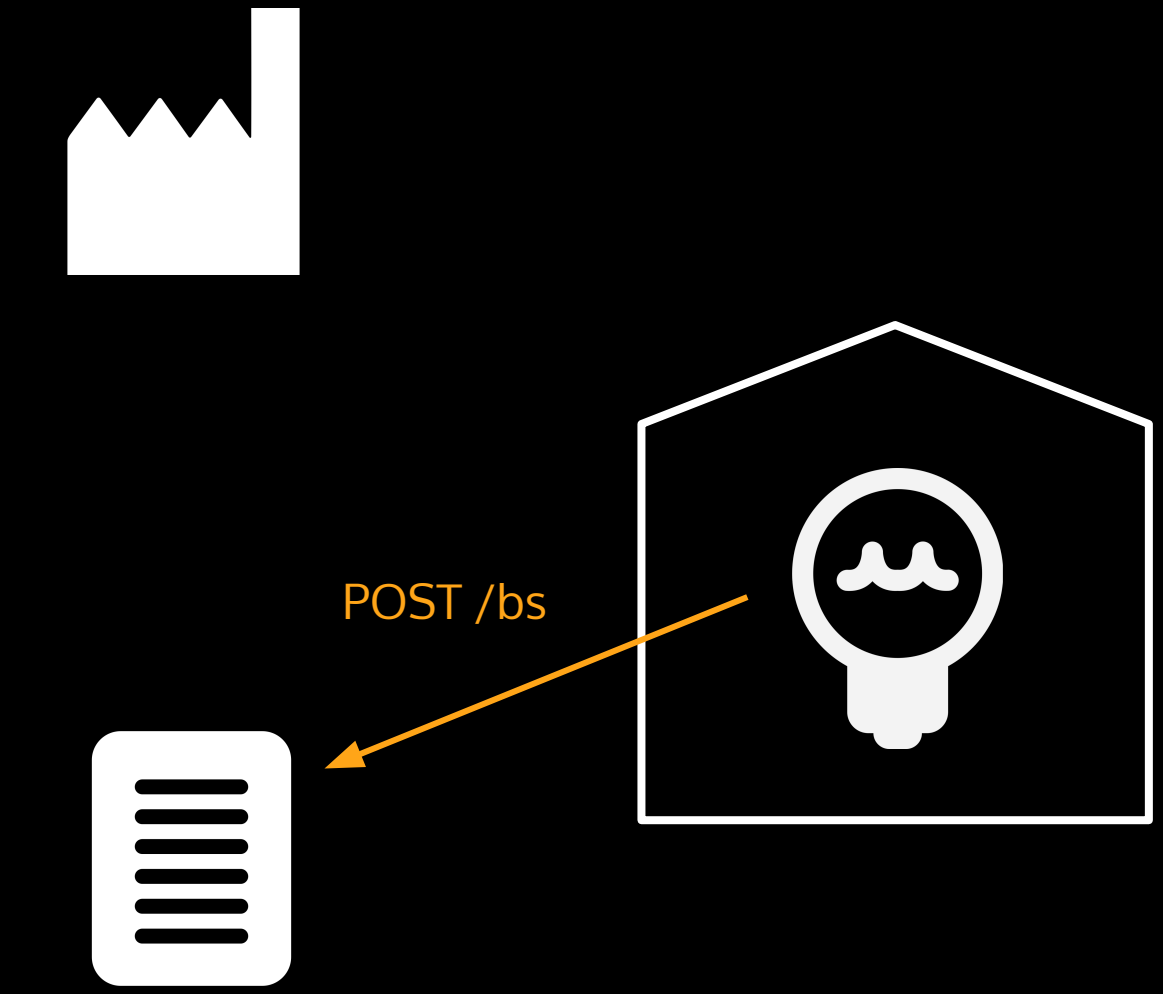
LwM2M bootstrap in a nutshell



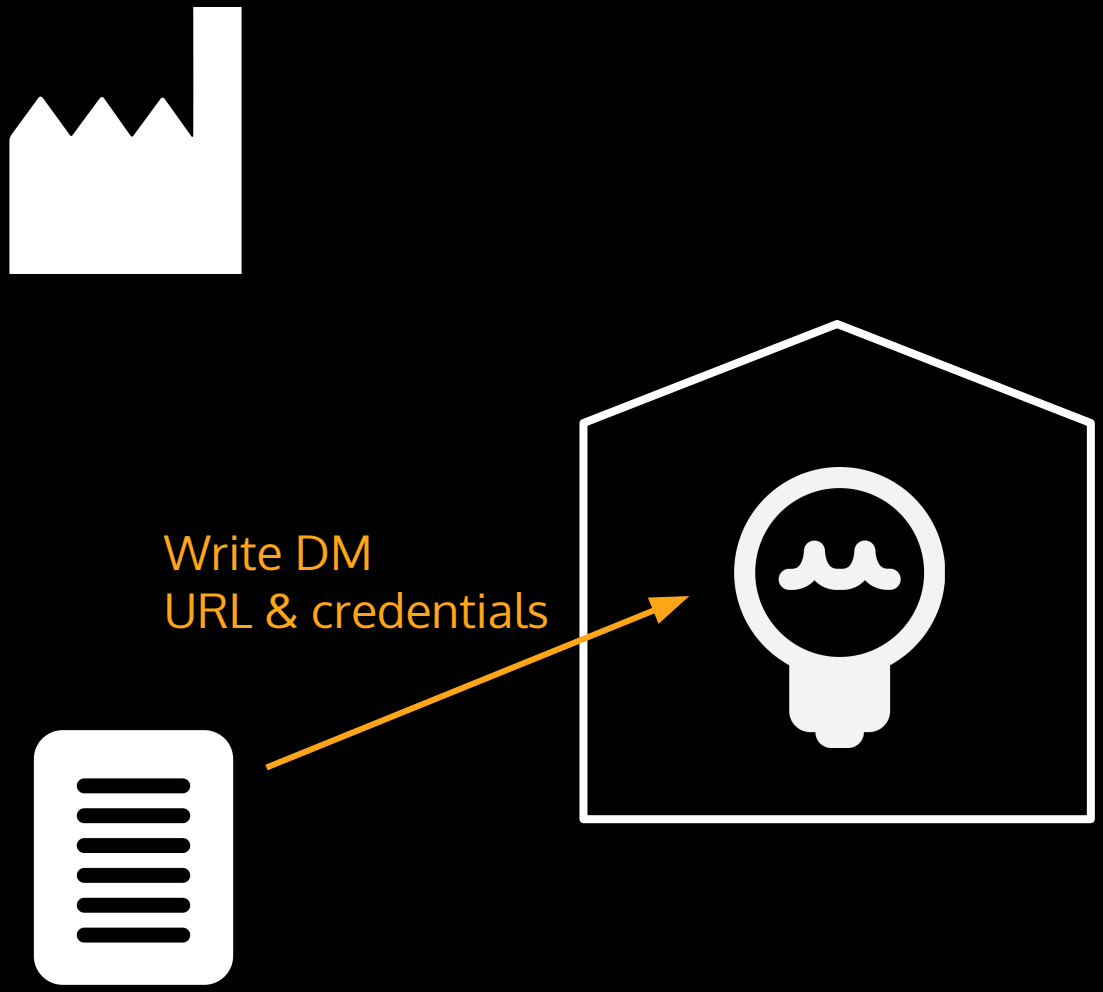
LwM2M bootstrap in a nutshell

I only have bootstrap credentials or I can’t reach final server

LwM2M bootstrap in a nutshell

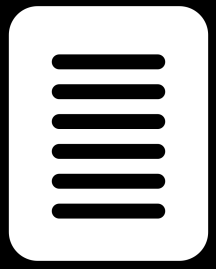


LwM2M bootstrap in a nutshell



LwM2M bootstrap in a nutshell

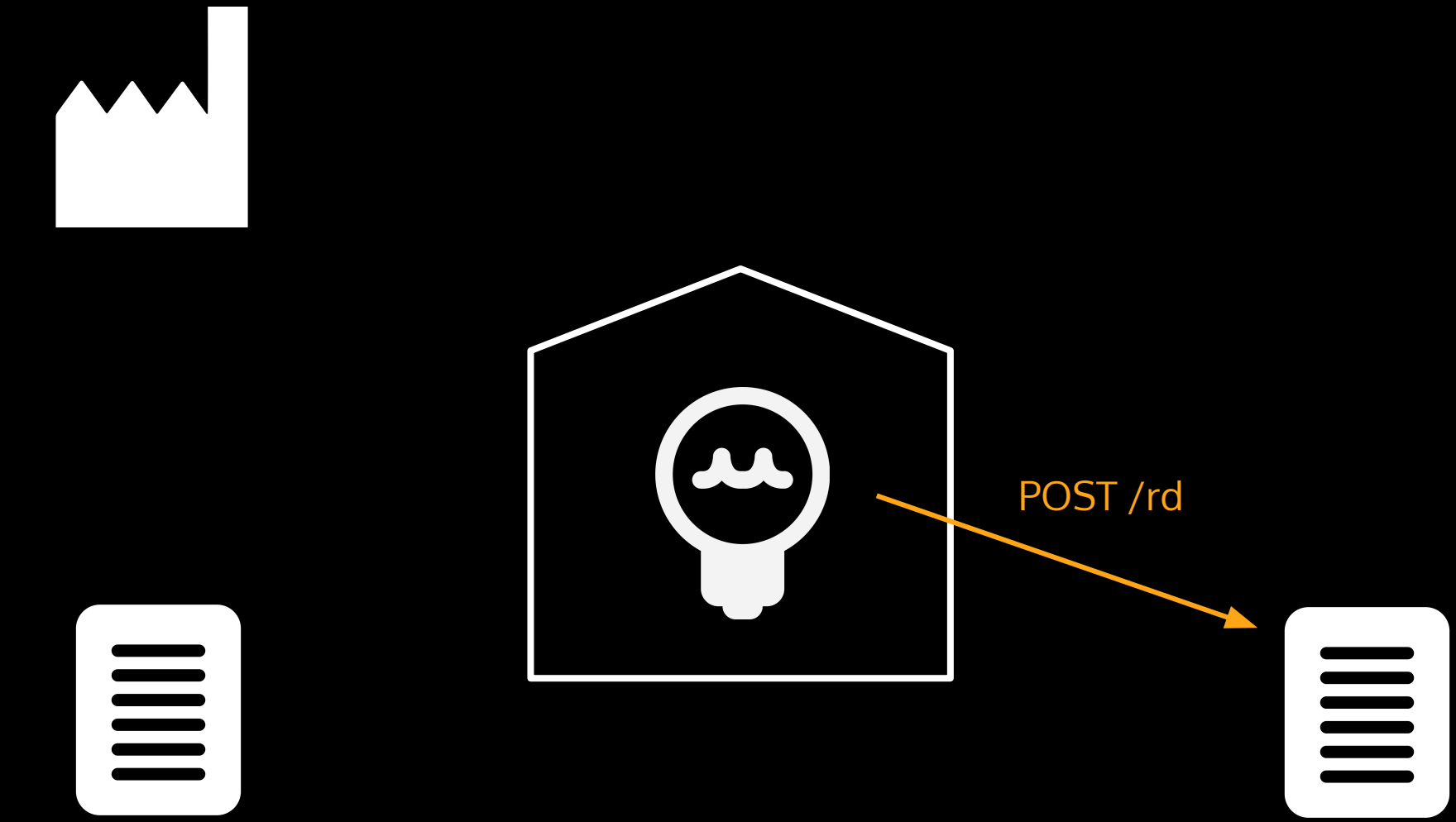
I have credential for the DM server



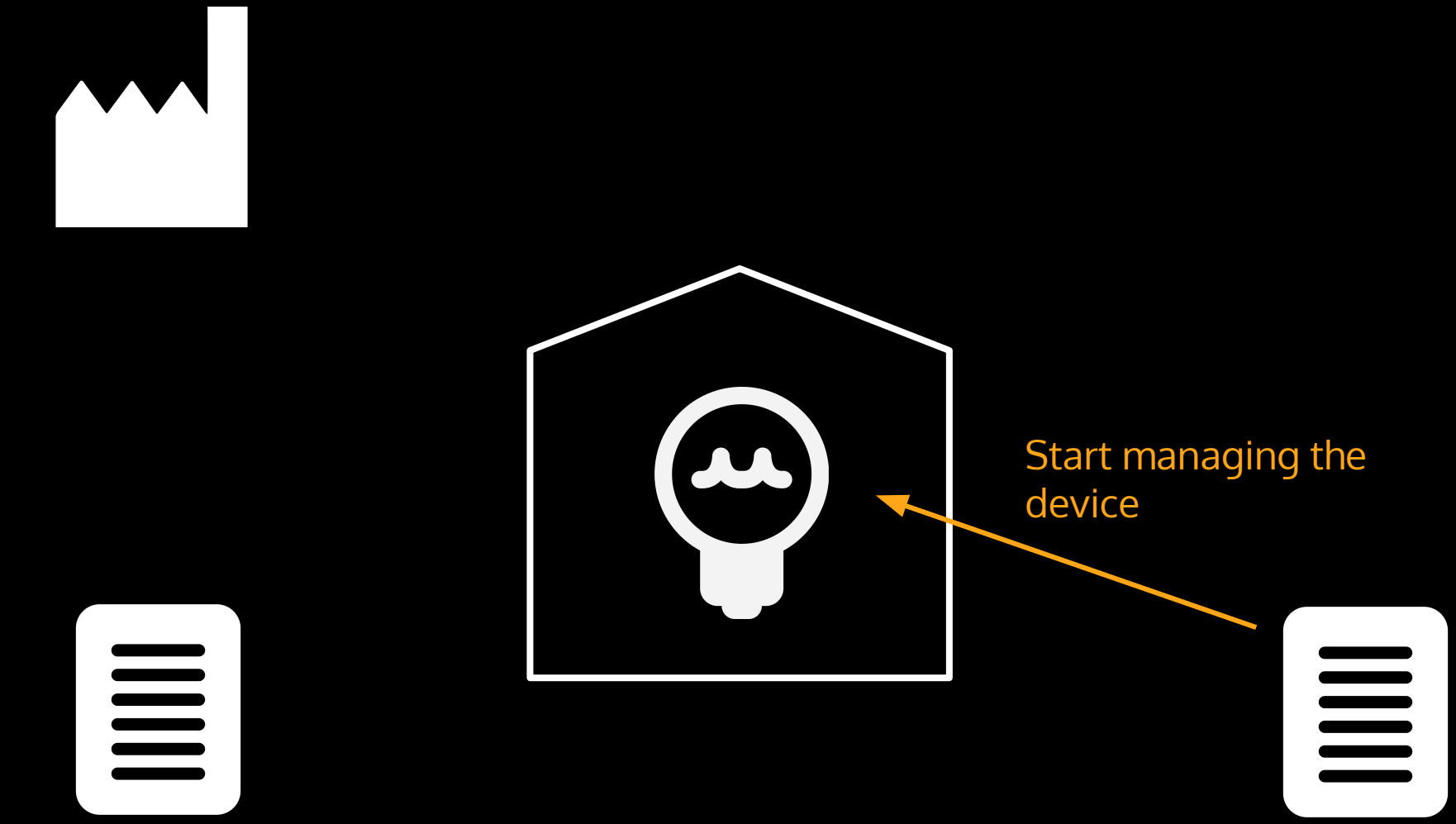
Bootstrap Server

DM Server

LwM2M bootstrap in a nutshell



LwM2M bootstrap in a nutshell




Server Security

Why it’s mattering?

Risk:

Takeover of your whole device fleet

You are a juicy target

Mitigations:

More security (ex. 2 factor auth) than classical web service

Collect only the necessary data

Isolate as much as possible web and devices

Now where I start?

Ask more time/budget?


DON'T CARE
Holland America Line
OOSTERDAM
SHIP IT!
memegenerator


DON'T CARE
Now you are part of the
70% unencrypted network services
Holland America Line
SHIP IT!
memegenerator

Open-source to the rescue!

Eclipse IoT - Leshan

Lightweight M2M implementation in Java

A library for building:

- bootstrap, and device management servers

Support DTLS PSK, RPK, (X.509 soon)

And also client for beefier devices or testing

Eclipse IoT - Leshan

Update firmware, software

Manage secrets (bootstrap)

Monitor and configure device

Can support custom object for applications

IPSO objects

Eclipse IoT - Wakaama

C implementation of Lightweight M2M

Focused on embedded

Bring your own IP stack

Bring your own DTLS implementation

Bootstrap supported

Eclipse IoT - Wakaama

You can receive packages for firmware/software update

But you need to implement live re-flashing on your platform

Known to be running on Linux, Arduino mega, ARM Cortex processors

TinyDTLS | https://tindydtls.sf.net

MIT License, Eclipse proposal!

“Support session multiplexing in single-threaded applications and thus targets specifically on embedded systems.”

Examples for Linux, or Contiki OS

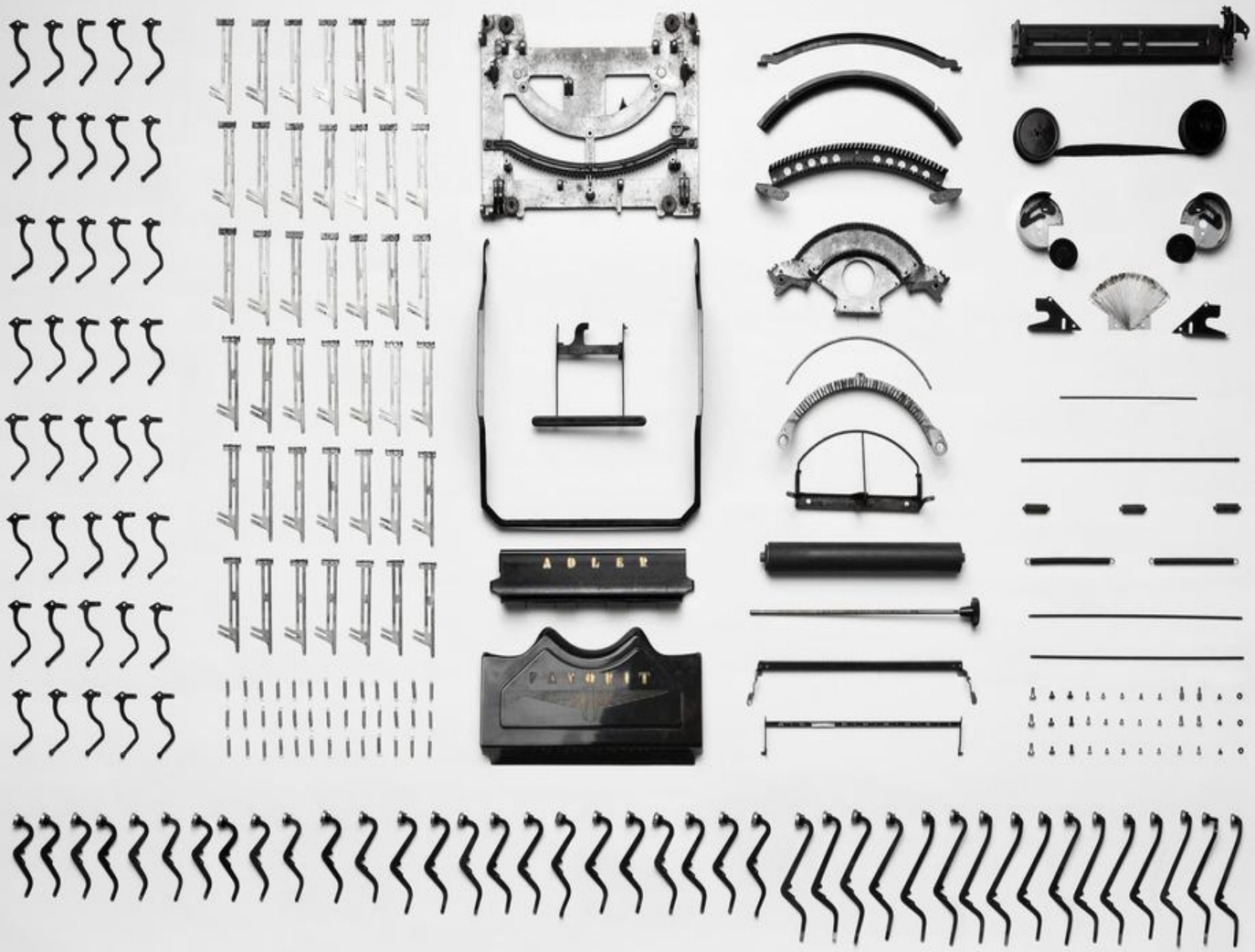
TinyDTLS

Supported ciphersuites:

TLS_PSK_WITH_AES_128_CCM_8

TLS_ECDHE_ECDSA_WITH_AES128_CCM_8

From Toolbox



To Jump start

Thanks!

Questions?

Contact me:
@vrmvrm
jvermillard@sierrawireless.com

Blog post

eclipsecon 2015

Evaluate the sessions

Sign in: www.eclipsecon.org