Protecting Critical Infrastructure From Cyber Attacks

Dave Thaler

Microsoft

Newer threats target critical infrastructure

Theft and ransom
Source: Organized Crime

Los Angeles Times
Hollywood hospital pays hackers $17k in bitcoin
Feb. 18, 2016

The Register
Ransomware worm melts down UK hospital
May 12, 2017

IndyStar.
Hospital paid $50k ransom for patient data
Jan. 17, 2018

Attack vectors: Wannacrypt, WannaCry, SamSam

Damage and disruption
Sources: Nations, Terror Groups, Rogue Admins

CBS EVENING NEWS
Russian hacks into Ukraine power
Dec. 21, 2016

REUTERS
Merck says cyberattack halted production
June 27, 2017

MarketWatch
Hack at Saudi petrochemical plant on safety system
Jan. 18, 2018

Attack vectors: Industroyer, NotPetya, Triton

Security goal to strive for

The device owner/operator is in complete control of critical systems

Minimizing set of trusted entities

Each entity that could affect critical systems is a point of potential vulnerability

Always have to trust:

- Your own admins

In practice, also have to trust:

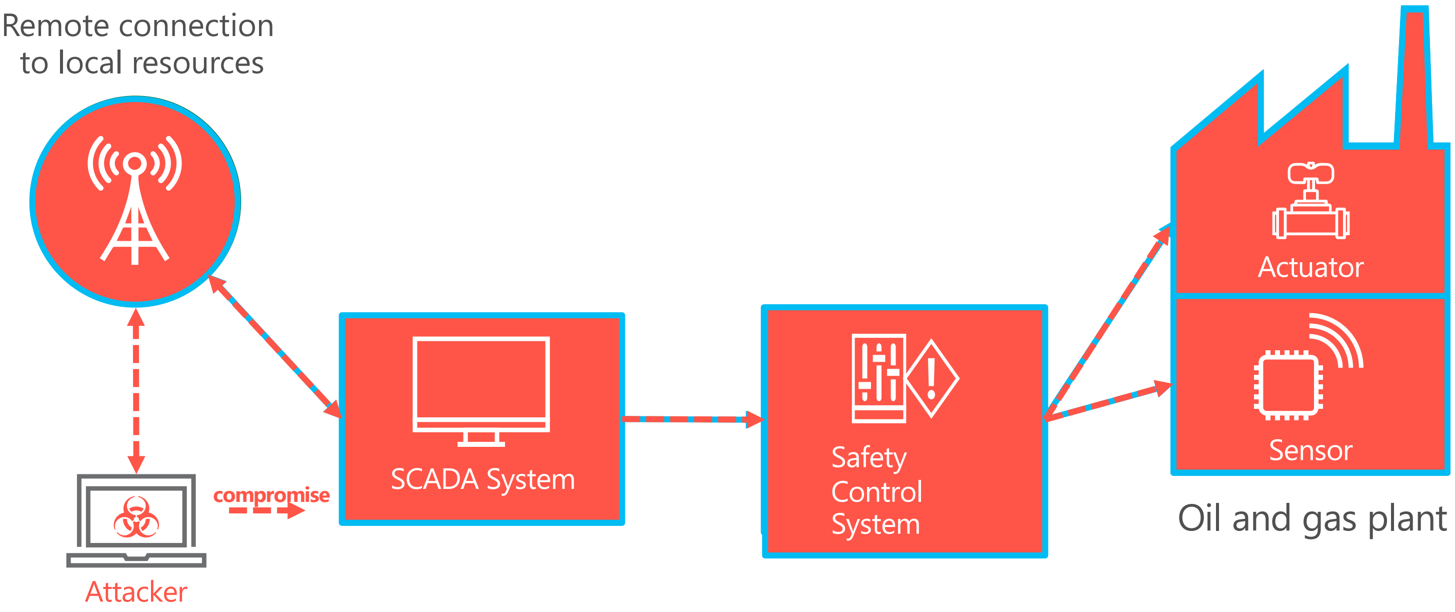
- Security chip manufacturer (and their government)

*For global products, it's important to allow **choice***
of security chip manufacturer (and jurisdiction)

Others can be excluded from implicit trust:

- OS maintainer
- App maintainer
- Tools maintainer
- Cloud service provider

Why was Triton successful?



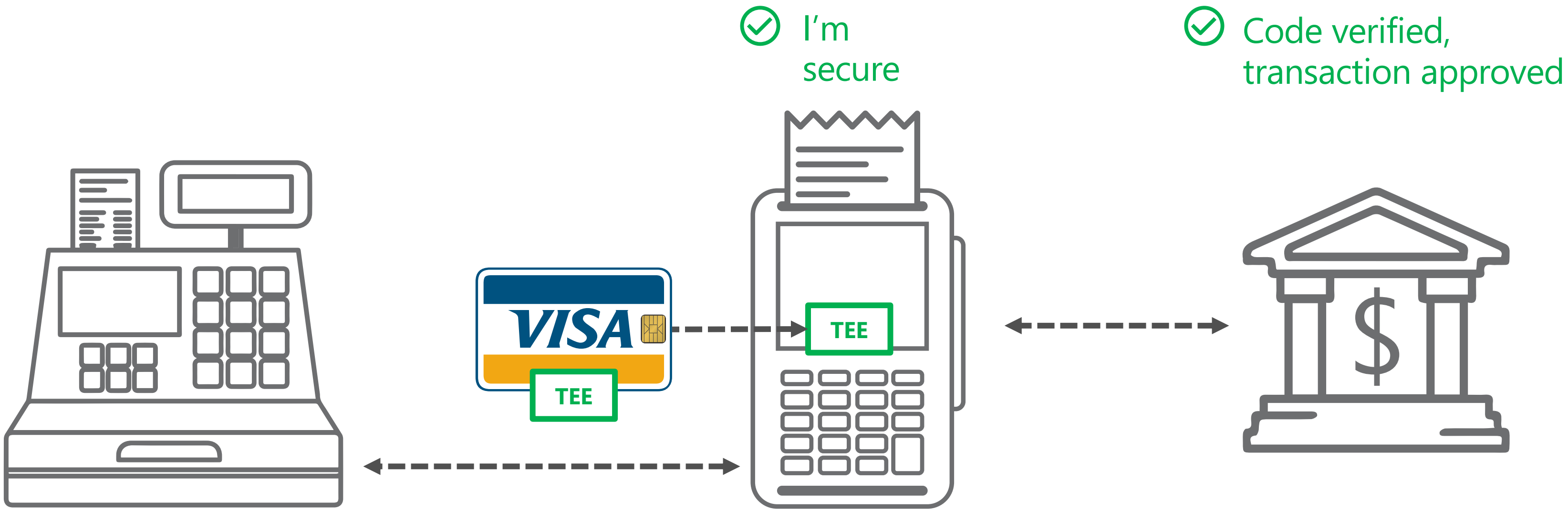
*"...hackers have infiltrated the **critical safety systems** for industrial control units used in **nuclear, oil and gas plants**, halting operations at at least one facility."*

*"The hackers used sophisticated malware, dubbed 'Triton', to **take remote control** of a safety control workstation..."*

*"Some controllers entered a failsafe mode as the hackers **attempted to reprogram** them..."*

Source: *The Guardian*, Dec. 15, 2017

Payments already use relevant secure execution technology



Cash register is vulnerable and can be hacked

Generates unique authentication code for each transaction

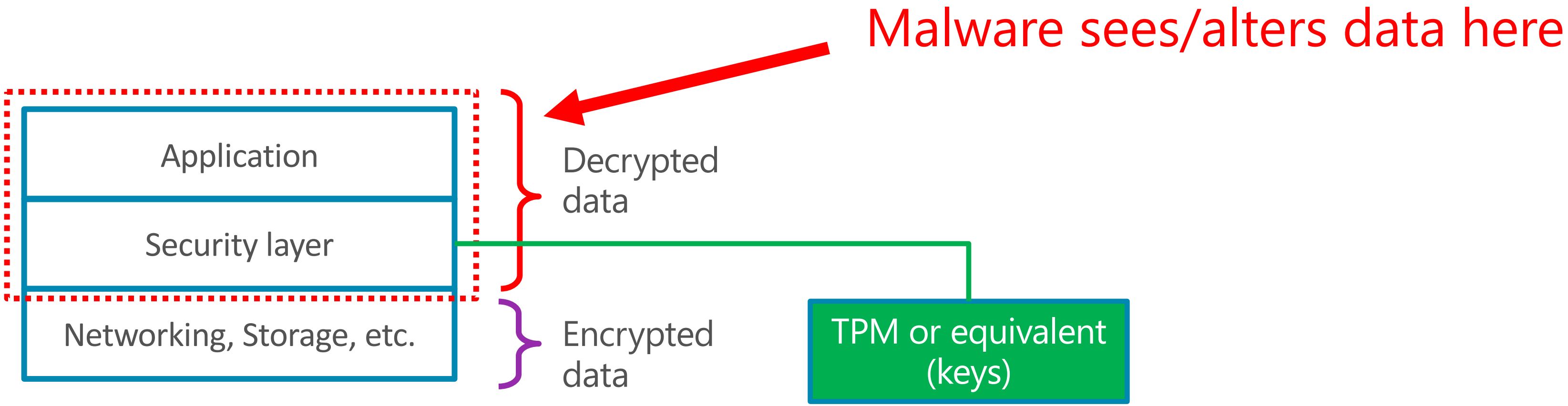
Banks saw 70% drop in fraudulent charges in 2 years with TEE-based solutions

Any malware on the cash register can't make payments without user authorization.

Source: Visa data, Feb. 2018

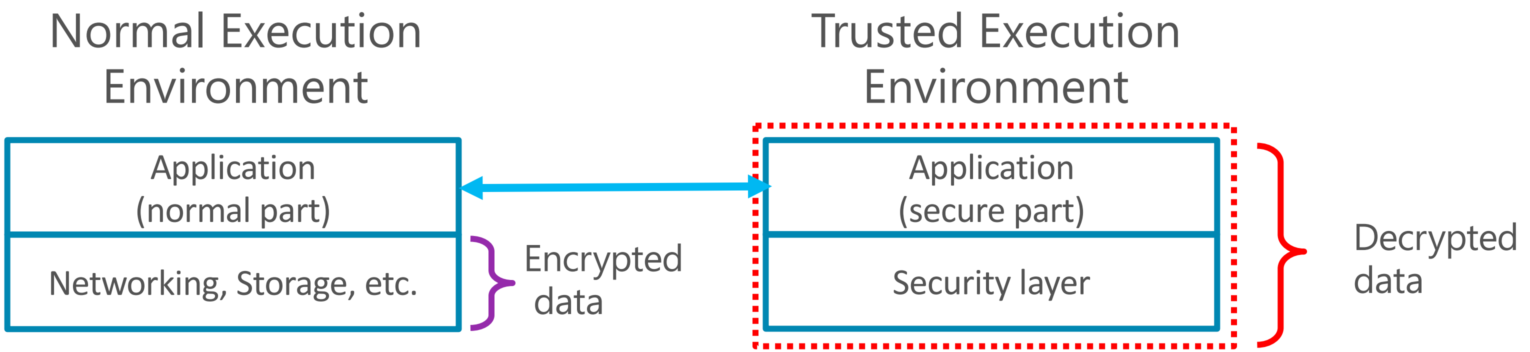
Protecting data at rest and data in flight is not sufficient

Microsoft



- ✓ Data at rest
- ✓ Data in flight
- ⊘ Data in execution

Protecting data in execution



- Data at rest
- Data in flight
- Data in execution

TEE provides hardware-enforcement that:

1) any code inside the TEE is operator-authorized code
2) any data inside the TEE cannot be read or modified from outside the TEE

Examples: Secure Elements, ARM TrustZone, Intel SGX, Azure Sphere

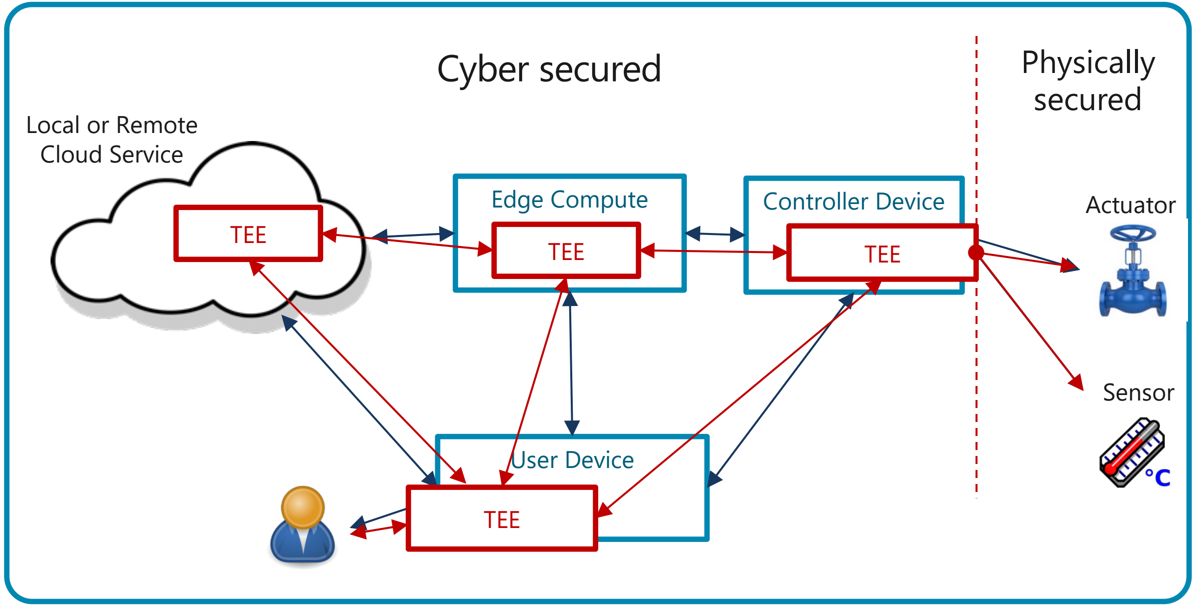
Trusting an End-to-End System

Every component that has keys to critical operations needs a TEE

Cloud and Edge Compute components can use **any TEE** and secure protocol

User Device and Controller Device components also need **Trusted I/O,** where physical connections are *only* accessible from within a TEE, isolated from any malware

Without Trusted I/O, malware on a device could directly access secure peripherals



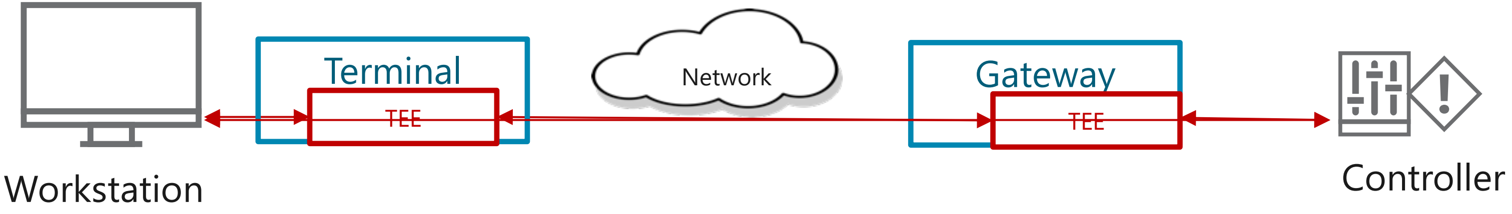
Incremental Deployability

Microsoft

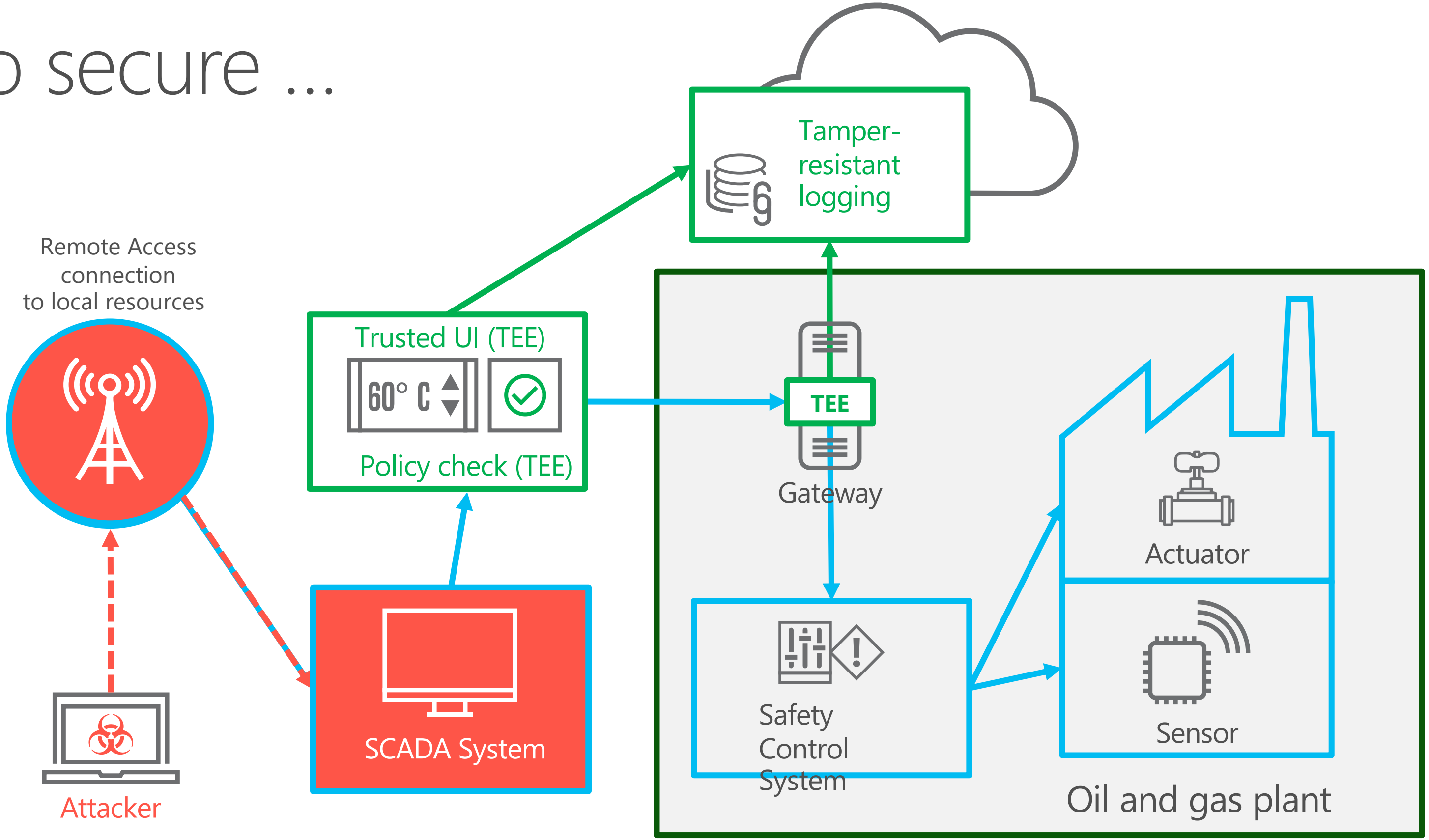
Can't require waiting to replace equipment with 10+ year lifespan

Put a gateway with Trusted I/O in front of existing equipment

Put a secure confirmation terminal with Trusted I/O in front of existing client



How to secure …



✓ **Critical operations are secured** in Trusted Execution Environments (TEE)

✓ **Tamper-resistant log entry** of every command

TEEs are applicable to services involved in critical operations

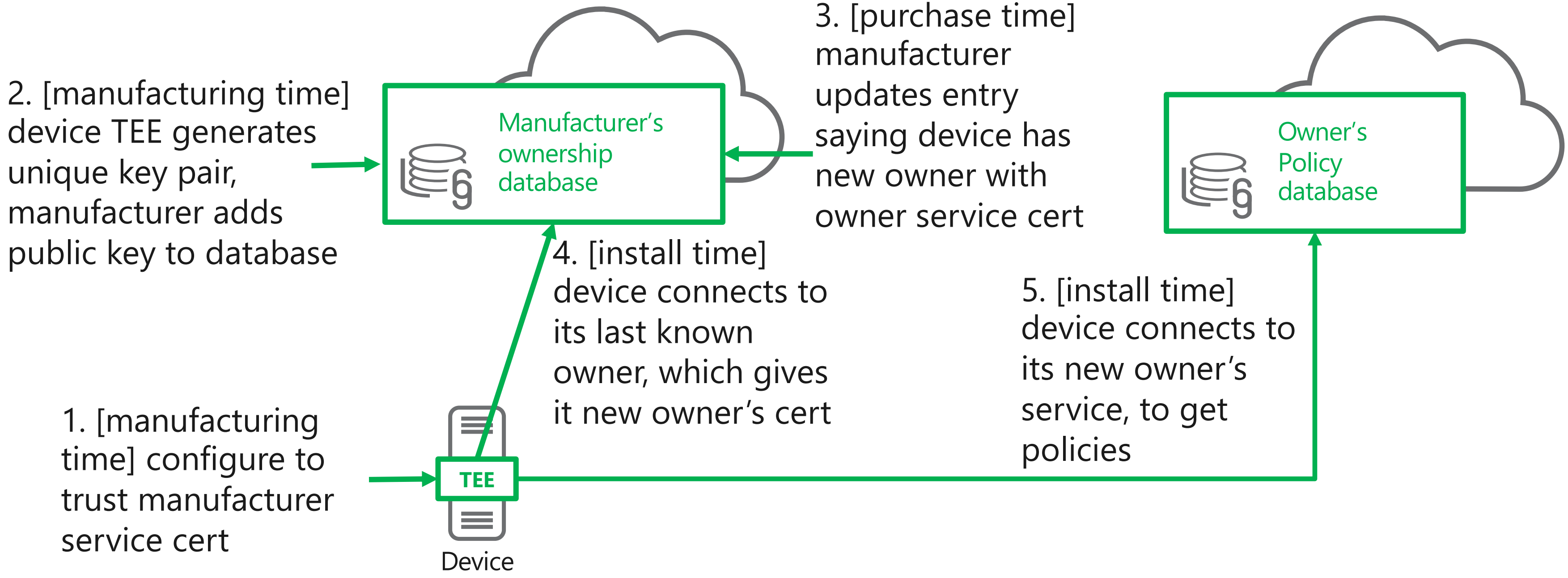
Microsoft

- Certificate Authorities
- Provisioning Services
- Operations Log Services
- Patch Management Services
- Key Management and Escrow Services
- Policy Decision Points

How can you bootstrap configuration of a device?

Microsoft

Example of one Provisioning Services solution (other variations exist):



Trusting TEE Code

- TEE must only run code the owner/operator trusts
- Code might be:
  a) Written by the owner/operator's organization
  b) Vetted by the owner/operator's organization
  c) Vetted by security analysts they trust
- Simply vetting source code is not sufficient
  a) Vet binary itself (may be impractical)
  b) Use a trusted compiler chain

Standards & compliance challenges today

- Technology exists, but is not widely deployed today nor widely known in IoT

- Challenges:
  1. Security Levels defined by standards today do not cover securing Data in Execution
     - Triton compromised systems certified at the highest IEC 62443 level
  2. Certification/compliance for critical infrastructure can't easily require securing Data in Execution as a result

Key Aspects

Microsoft

- Security:
  - Hardware-enforced integrity of critical code and data
  - All TEE code is available and vettable by operator or their security analysts
  - Components are commonly available already, ready for equipment vendors

- Incremental deployability:
  1. Place a TEE app-layer gateway in front of legacy equipment or apps, and
  2. Physically protect communication between the gateway and the equipment

Resources

- Flyer: https://aka.ms/TCPS_TwoPager_HMI2018
- Technical whitepaper: https://aka.ms/TCPS_Whitepaper

Microsoft