HARDWARE-ORIENTED SECURITY, CPS, AND IoT

MARILYN WOLF

CREATING THE NEXT®

OUTLINE

Why we need hardware-oriented security:

- Safety and security.
- The Internet, IoT, and CPS.

Threats and countermeasures.

Trends and approaches.

SAFETY AND SECURITY

Safety:

Guarantees on system physical characteristics and behavior.

Safe systems do not cause harm.

Critical services must provide high levels of reliability.

Security:

Guarantees on information.

Integrity, availability.

Privacy is closely related to security.

SAFETY AND SECURITY ARE INTERTWINED

Safety and security are no longer separate.

Security affects safety:

- Insecure devices can be made to perform unsafe operations.

Safety affects security:

- Many industrial SCADA systems are not easily updated, resulting in security holes.

IT AND OT

IT = information technology.

OT = operational technology.

IT and OT are often linked.

- Failures in one can result in failures of the other.

Examples:

- Airline dispatching.
- Manufacturing accounting, control.
- Utility billing, control.

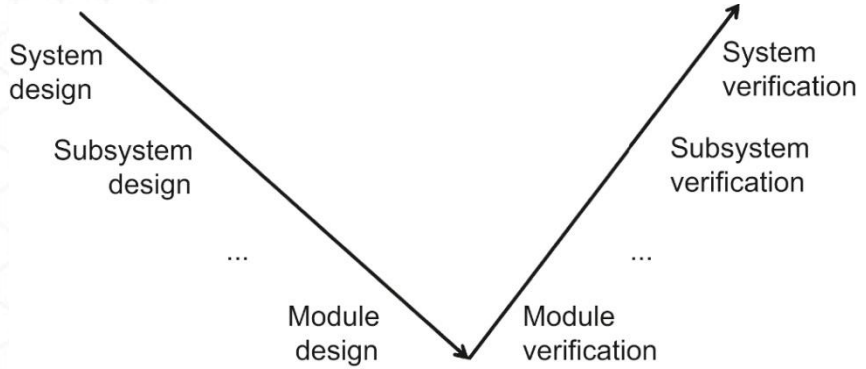
V METHODOLOGY IS INADEQUATE

Top-down design, bottom-up verification.

Assumes stable, well-understood specification.

Assumes that the system design comes to completion.

Not well-suited to Internet-enabled systems.



ATTACKS AND FAULTS

In many cases, users don’t care whether a problem is due to an attack or a fault.

Attacks, faults may need different diagnosis techniques, different remediations.

PRIVACY

Many aspects of behavior and intent can be inferred from limited information.

How do we provide useful operations while maintaining reasonable levels of privacy?

SIDE CHANNELS

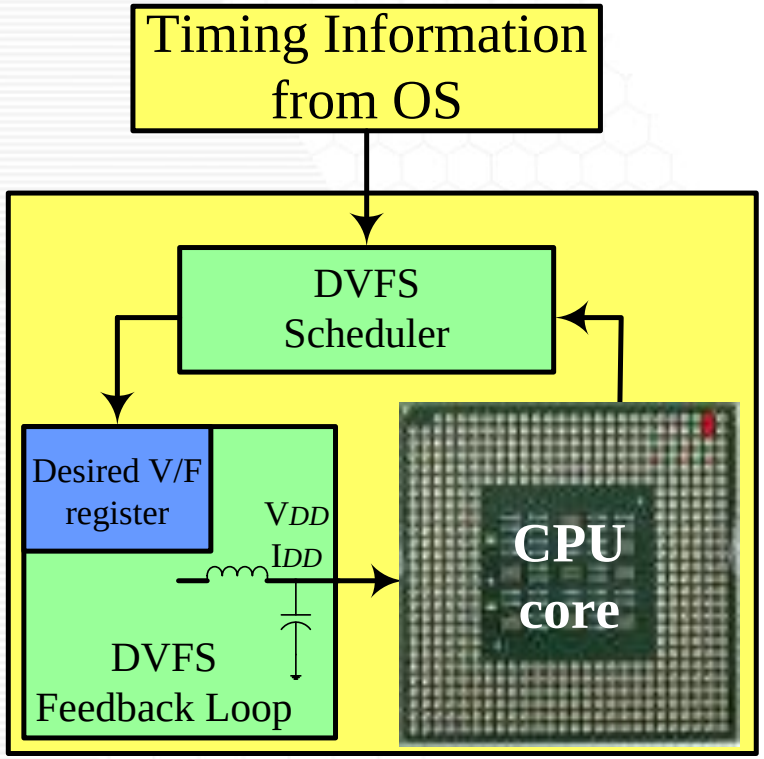
New side channel attacks appear regularly:

- Timing.
- Cache.
- Faults.
- Electromagnetic interference.
- Disk drive noise.

Equalizing side channel effects over decision cases can be expensive.

Plugging side channels one at a time is insufficient.

YANG ET AL: DVFS CRYPTOSYSTEM



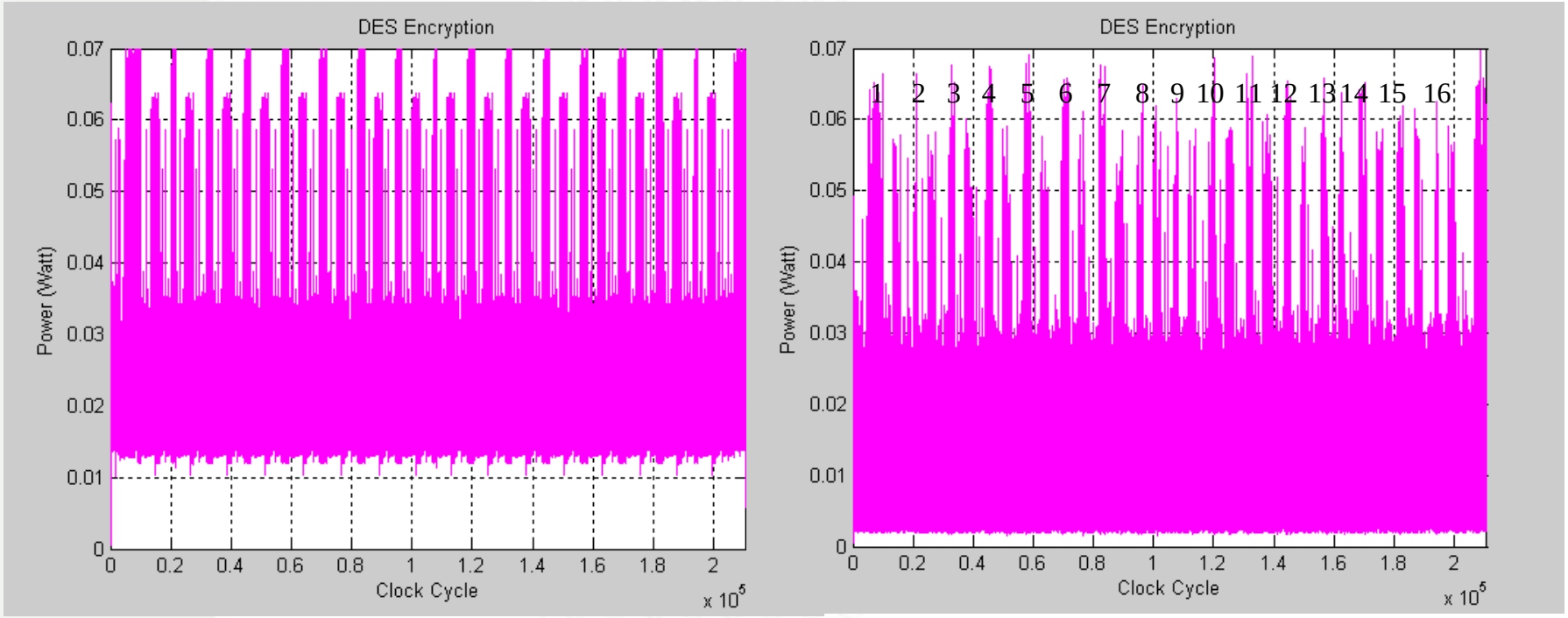
NAÏVE DESIGN

Input : V_{dd_normal}, F_{normal}, Signal_Done, RW_Enable
Output : v_{dd_scale}/f_{scale}

Algorithm :

1: **While** Signal_Done != 1 **Do**
2: generate random v_{dd_scale}/f_{scale} by using V_{dd_normal}, F_{normal}
3: **if** RW_Enable ==1 **then**
4: assign v_{dd_scale}/f_{scale} to register in DVFSFL
5: **End if**
6: **End While**

NAÏVE DESIGN (CONT'D)

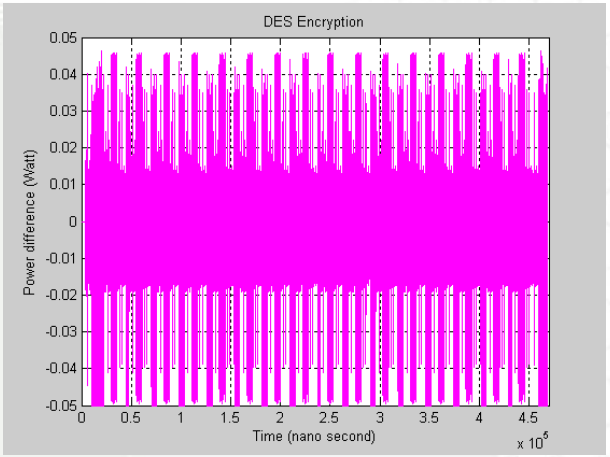


Input : V_{dd_normal}, F_{normal}, TB, ETCC, Signal_Done, RW_Enable
Output : v_{dd_scale}/f_{scale}

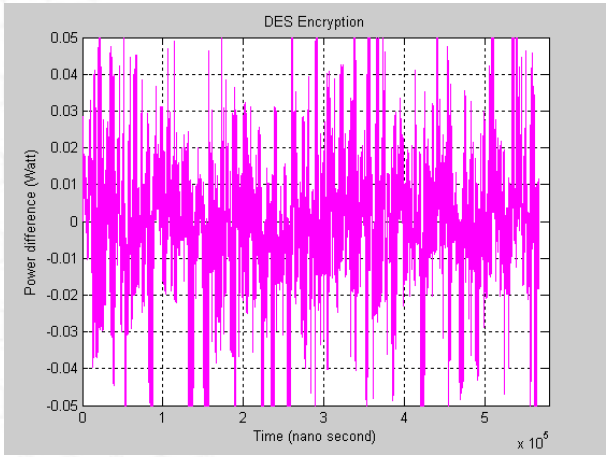
Algorithm :

1: *time_1*= **clock()**
2: generate random v_{dd_scale}/f_{scale} by using V_{dd_normal}, F_{normal}
3: assign v_{dd_scale}/f_{scale} to register in DVFSFL
4: initialize *timing*, *timing_space* to *zero*
5: determine High_limit_WT
6: **While** Signal_Done != 1 **Do**
7: *time_3*=**clock()**
8: generate a random number *NCC* < High_limit_WT
9: **for** i=1:1:*NCC*
10: NOP
11: **end for**
12: *time_2*= **clock()**;
13: *timing*= (*time_2*-*time_1*)/clocks_per_second_DVFSS
14: *timing_space* += *timing_space*
15: use *timing*, *timing_space*, TB, ETCC to determine the lowest value of $v_{dd\_scale}/f_{scale,}$
16: generate random v_{dd_scale}/f_{scale} > the lowest value
17: **if** RW_Enable==1
18: assign v_{dd_scale}/f_{scale} to register in DVFSFL
19: **end if**
20: *time_4*=**clock()**
21: *timing_space*=(*time_4*-*time_3*)
22:**End While**

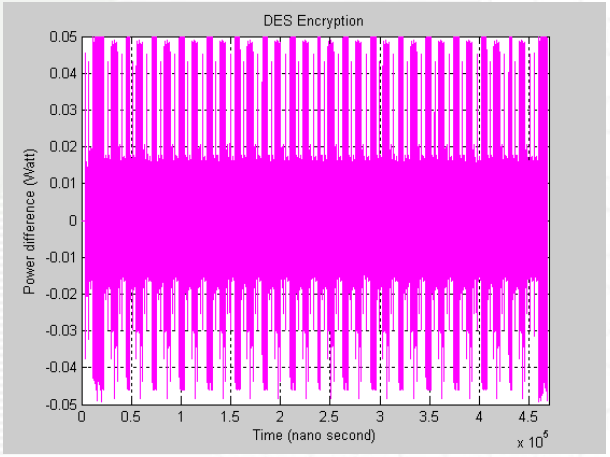
ADVANCED DESIGN (CONT'D)



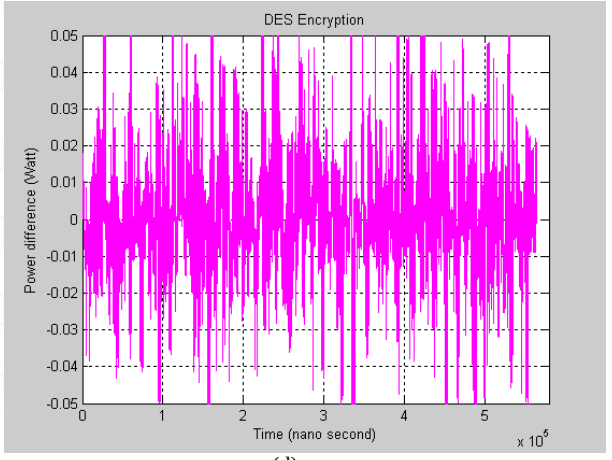
(a)



(b)



(c)



(d)

MORE RESULTS FOR ADVANCED DESIGN

DES	Energy Overhead (EO)	Time Overhead (TO)	Power Trace Entropy (PTE)	Time Trace Entropy (TTE)	Long Time Waiting Effect (LTWE)
Encryption	-27.32%	16.15%	5.42	6.02	NO
Decryption	-26.89%	16.01%	5.44	6.05	NO

BE AFRAID. BE VERY AFRAID.

January 2016: Cyber attack on Ukranian electric utilities.

October 2016: IoT-driven attack on DNS.

May 2015: In-flight hack into UA 737.

May 2015: Crash of Airbus A400M due to fuel system software bug.

2015: VW Dieselgate.

KEEP BEING AFRAID…

UK Nuclear Energy Institute: nuclear power plant technicians break air gaps.

Galaxy 7 Note fires.

787 battery fire.

2015: Database error involved in airplane crash.

May 2015: IT failures at United, Delta, Southwest ground flights.

Car attacks: UCSD, CMU, *etc*.

STUXNET

Designed to attack nuclear processing facility in Natanz, Iran.

- Centrifuges used in nuclear fuel refinement.
- Each centrifuge has valves.
- Centrifuges organized into cascades.

Infected Windows systems, leveraged known exploi8ts to steal data, hide itself.

Designed to attack PCs that run Siemens SIMAC Step 7 industrial control application.

First-known case of a virus designed to attack programmable logic controllers (PLCs).

STUXNET 0.5

May have been operational in November 2005, became known in November 2007.

Designed to stop compromising computers on July 4, 2009.

Designed to infect computers not on Internet.

- Infected USB key or other device sufficient.
- After first infection, spread through rest of the network.

Used external Web sites only for code downloads and updates.

PLC ATTACK

Replaced two dynamically linked libraries (DLLs) to attack PLC software:

- One DLL inserts malicious code onto PLC.
- Other DLL fingerprints target system and builds a PLC data block that can be used to attack the PLC.

Can identify particular pieces of code and data, as well as addresses.

Looks for symbols that identify target as either SIMATIC 400 or SIMATIC H-Station.

Looked specifically for CascadeModule IDs in range A21-A28, corresponding to units at Natanz.

PLC MODIFICATIONS

Records a snapshot of valve behavior, then attacks valves while replaying snapshopts.

If thos readings were normal, a secondary reading was obtained by opening a set of valves.

- Required waiting two hours for pressure to stabilize.

Code has several hard-coded constants related to the physical plant's behavior.

STUXNET TECHNIQUES

Georgia Tech

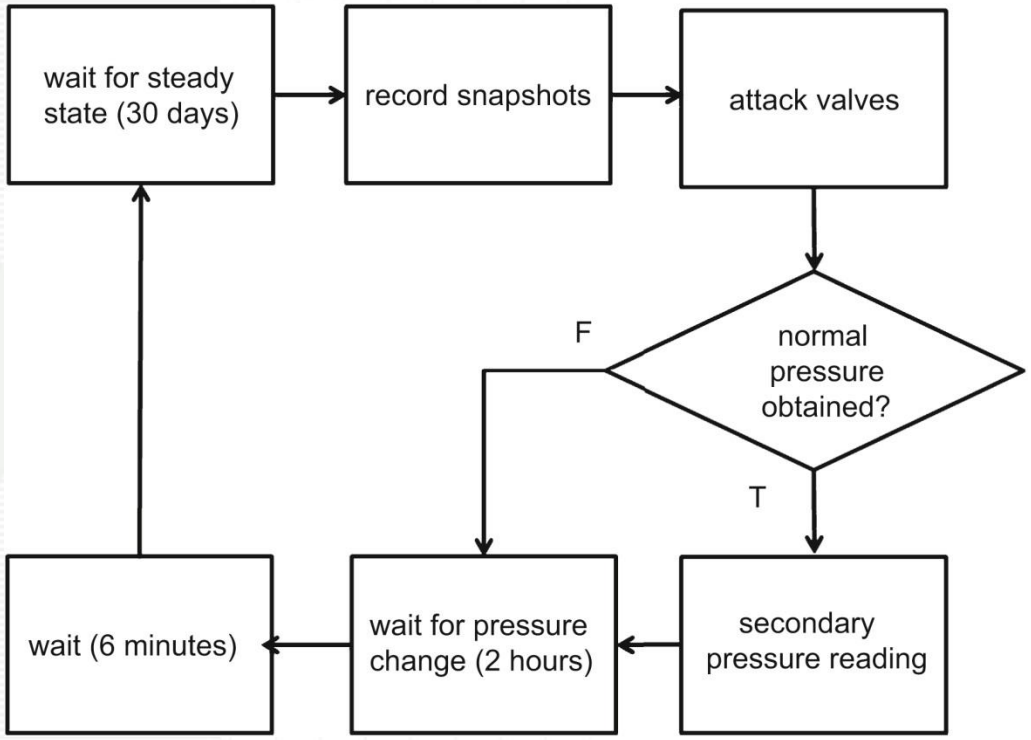
Can both inject malicious code onto PLC and hide the malicious code from a user.

- Intercepts requests for code and does not display modified code.
- Can customize code modifications depending on model of PLC being attacked.

Multiple methods of propagation

- Propagates peer-to-peer using RPC server.
- Propagates using removable drives.
- Exploits vulnerability of autorun.inf, Step 7 project files.

STUXNET ATTACK PROCEDURE



GOALS OF CYBER-PHYSICAL AND IOT ATTACKERS

Denial of service: Deny others use of the system without physically damaging the physical plant.

Large scale damage: Control the plant in a way that causes it to damage itself.

Theft of services: Make use of the plant without damaging it.

Spying: Watch the system in order to assess the activities of a legitimate user.

ATTACKS ON CPS

Static data.

Code replacement.

Data in transit.

Timing.

Power drain.

Design-time Trojan horses:

- Hardware.
- Software.

ATTACKS ON IOT

Code replacement.

Data in transit.

Network.

- Flooding.
- Blocking.

Energy drain.

- Battery-operated or scavenged-energy devices.

UNDETECTABLE AND UNIDENTIFIABLE ATTACKS

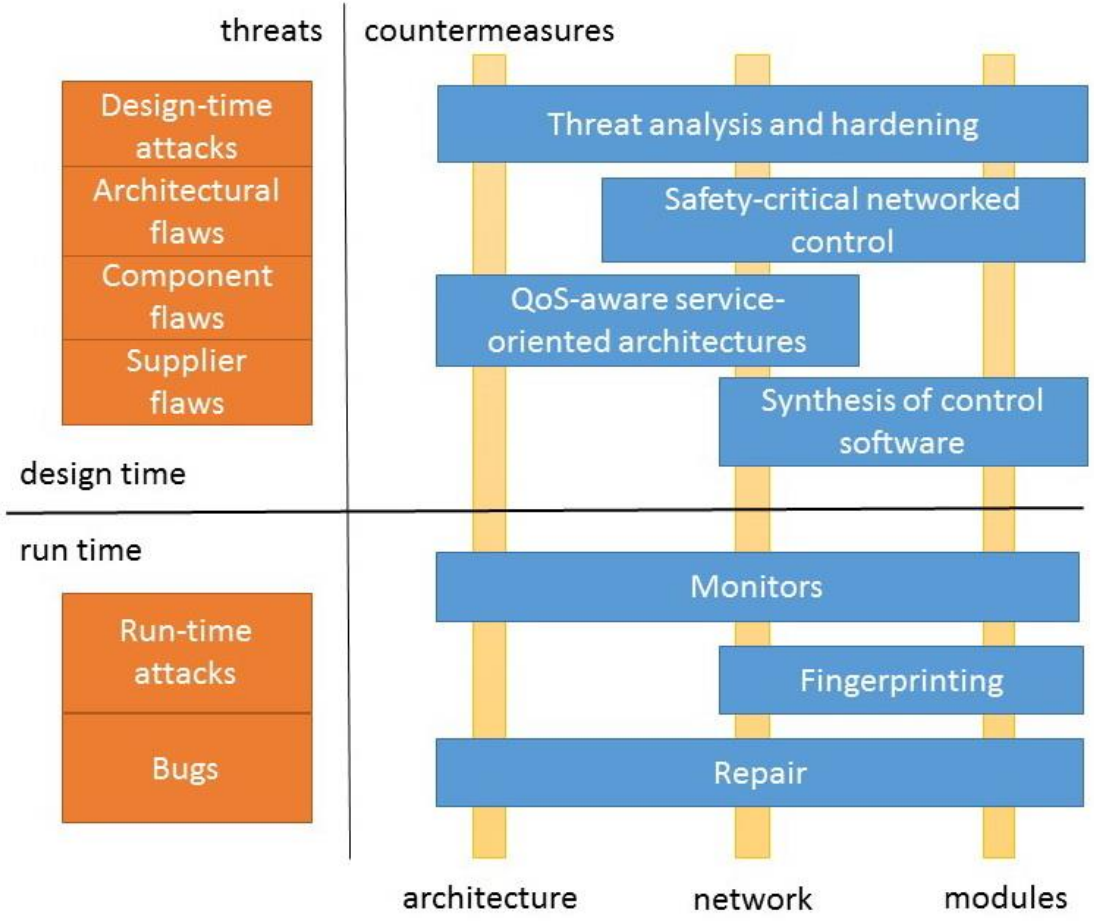
Liu et al. showed that attackers can make arbitrarily large changes to state of power system that cannot be detected.

- Take control of some power meters.
- Must know system topology.
- Showed how to construct an attack vector.

Qin et al. analyzed attacks for which operator can determine existence of attack but needs to localize attack.

- Successively estimate state and remove meter with largest residual.

THREATS AND COUNTERMEASURES



HARDWARE-ORIENTED SECURITY

Device identification and authentication.

Software authentication.

Cryptographic keys.

Communication:

- Secure communication in networked control systems.
- Secure communication in IoT networks.

Monitors.

- Architectures for secure monitoring.

Recovery.

LOW POWER/ENERGY

Many CPS and IoT systems provide limited energy, power.

We need to provide adequate encryption within platform constraints:

- Efficient design.
- New encryption algorithms.

BEST PRACTICES

Hardware root of trust.

Signed software.

Well-defined software development methodologies.

ADVANCED METHODS

Counterfeit-resistant tags.

QoS-oriented architectures.

Run-time monitoring.

- Model-based monitors.
- Distributed monitoring.
- Architectural support.

CPS CHALLENGES

Cyber-physical systems are networked control systems.

CPS control applications are sensitive to timing, particularly latency.

Architectures must preserve timing.

Design flaws, attacks cannot interfere with timing.

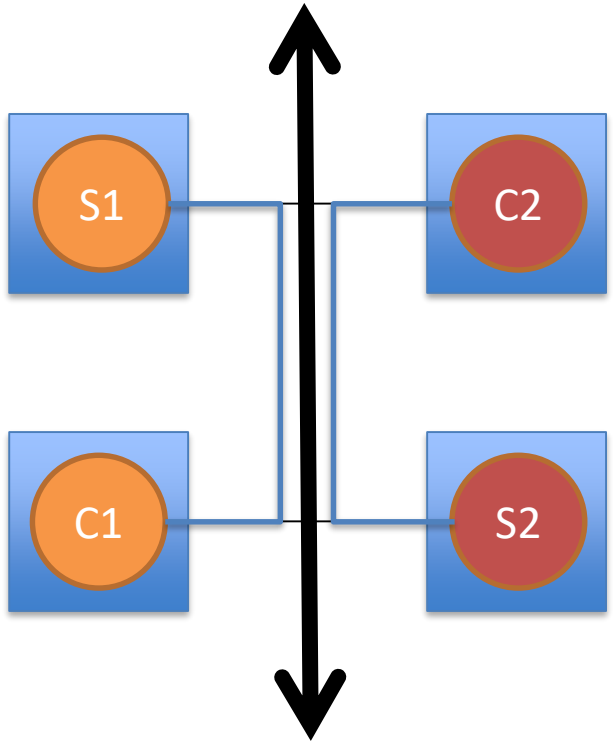
SERVICE-ORIENTED ARCHITECTURES AND CPS

Many Web services use service-oriented architectures (SOAs).

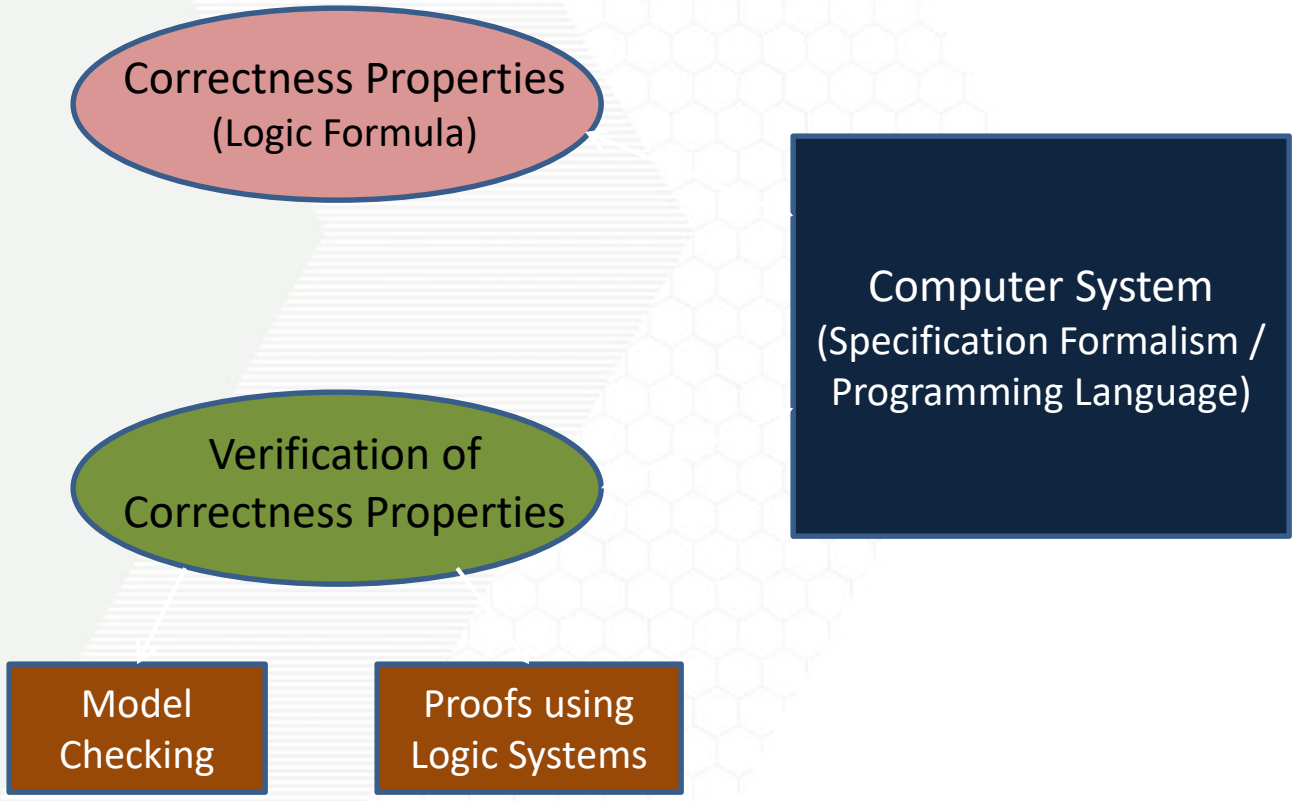
Allow services to be deployed along side existing services.

Existing patterns concentrate on functionality.

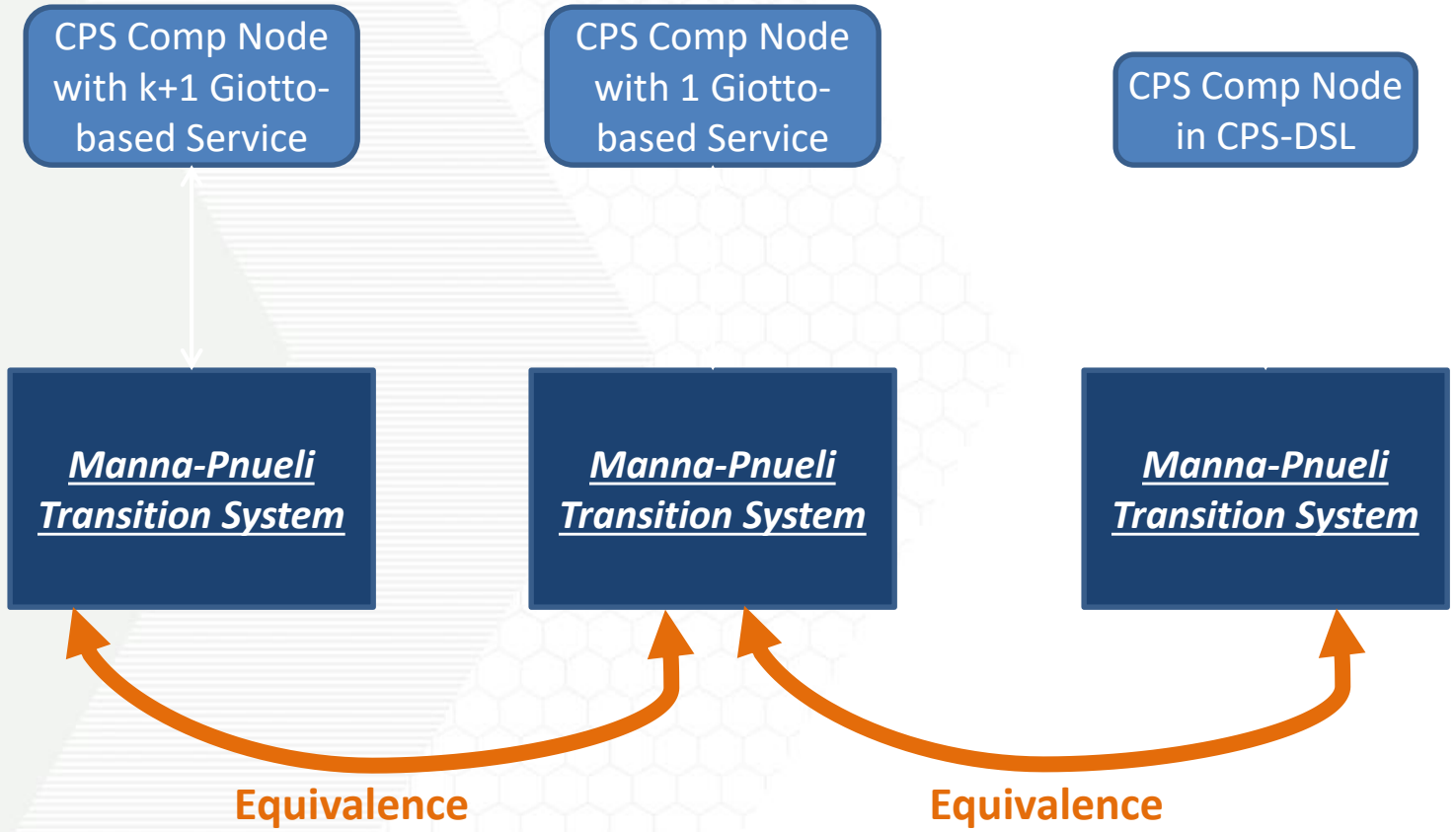
Cyber-physical systems require that newly-deployed services not interfere with the bandwidth of existing services.



Verify non-interference using Manna-Pnueli reactive system formalism.



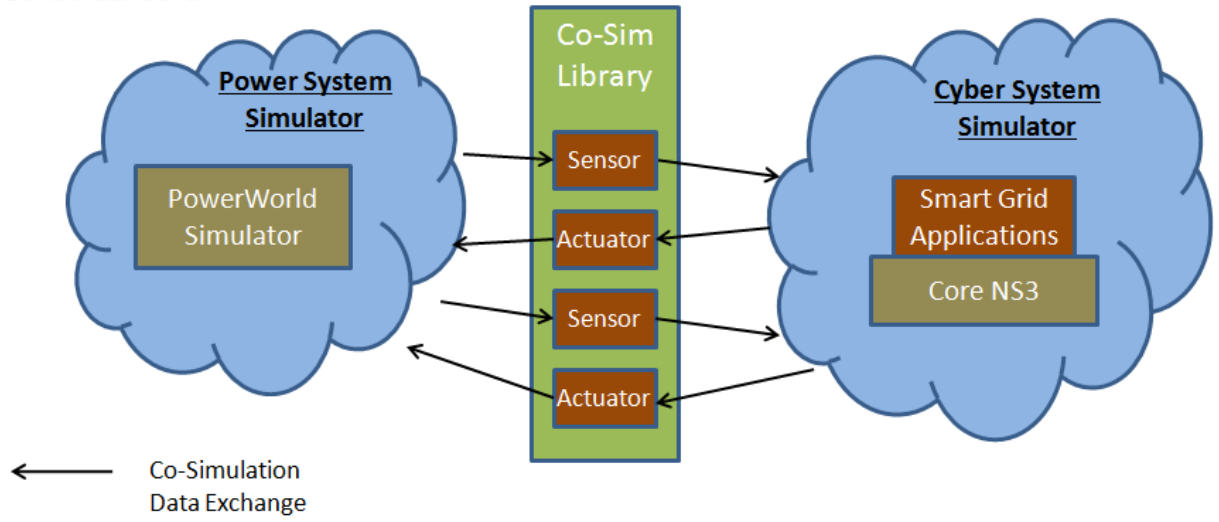
PROOF APPROACH



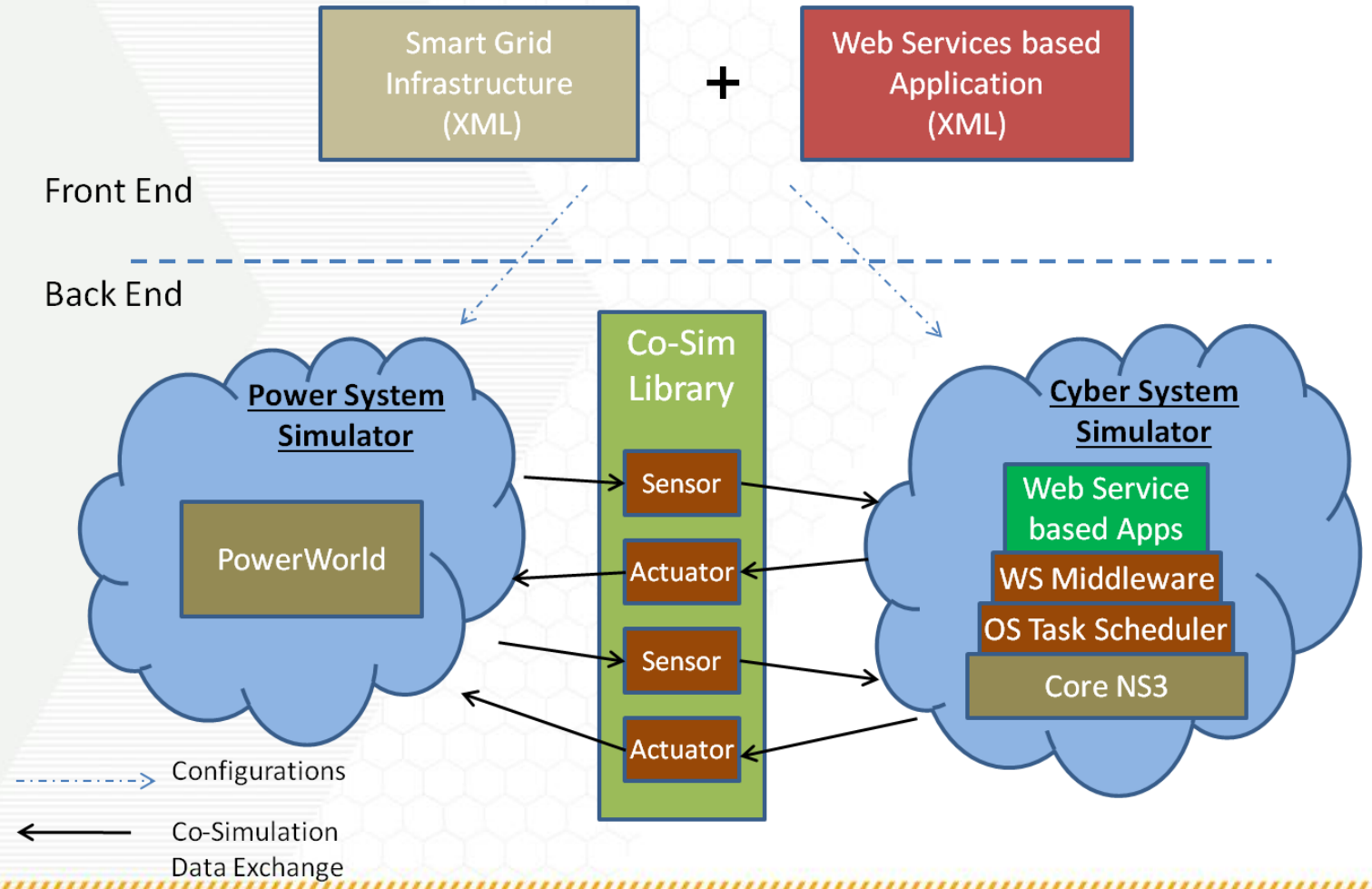
SMART GRID TEST-BED

Two versions:

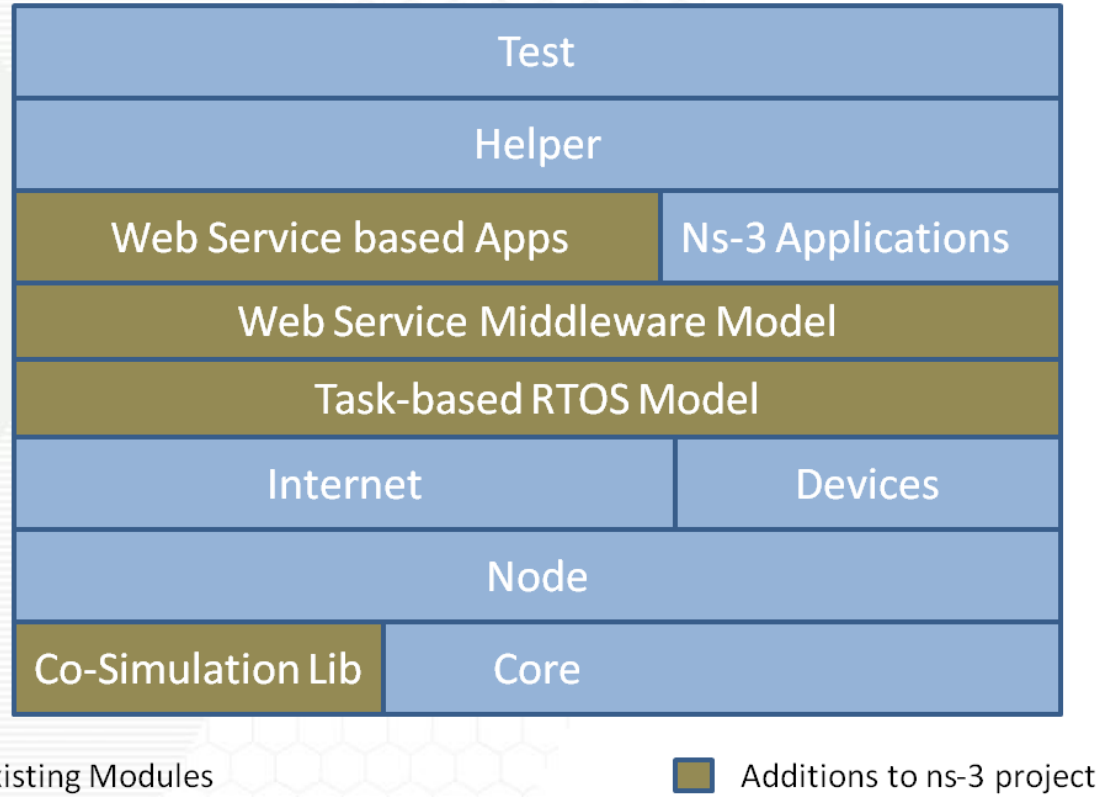
1. *Enterprise-Domain Service Oriented Computing*
2. *Proposed CPS-enabled Service Oriented Computing*



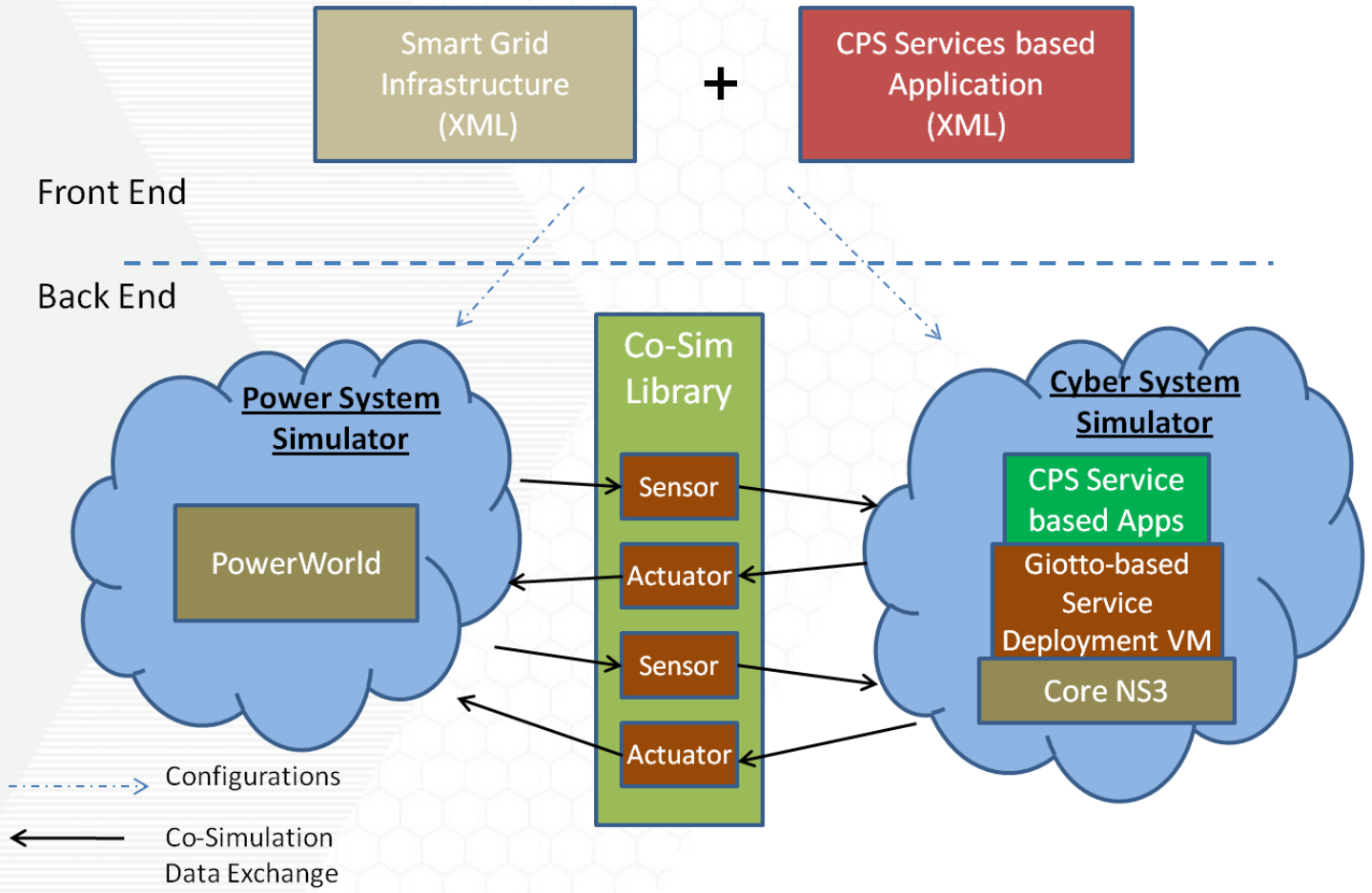
SMART GRID TEST-BED 1: ENTERPRISE-DOMAIN SOC



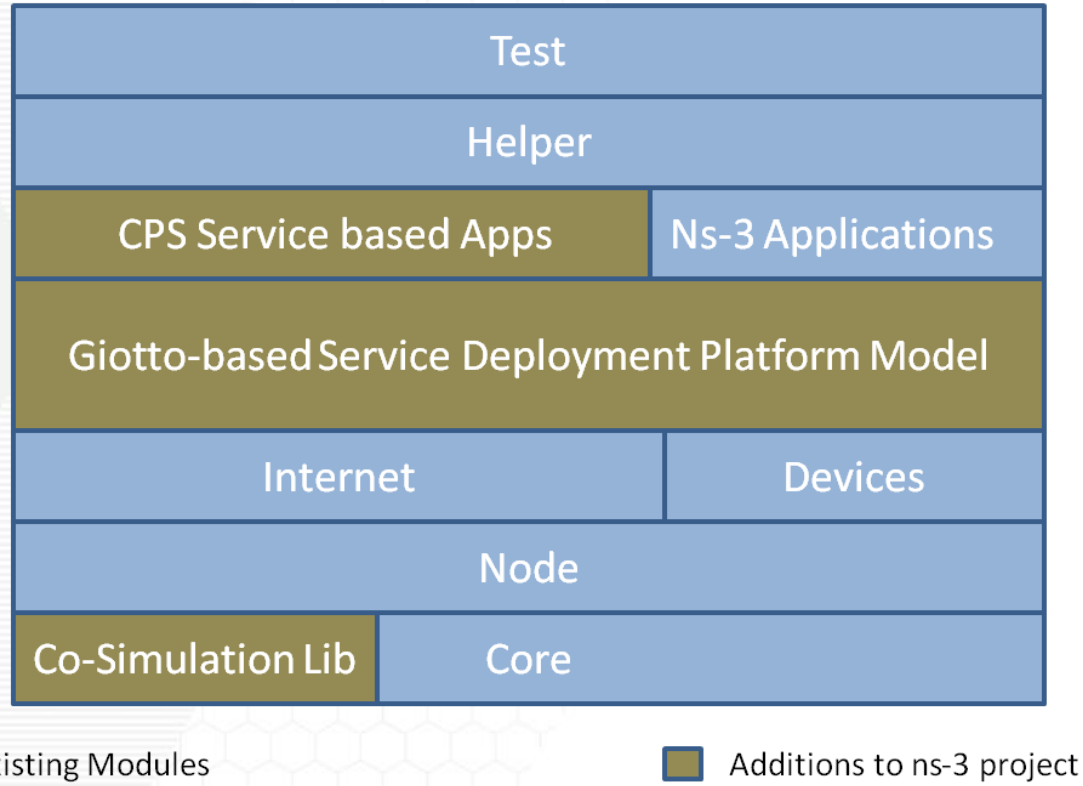
SMART GRID TEST-BED 1: ENTERPRISE-DOMAIN SOC



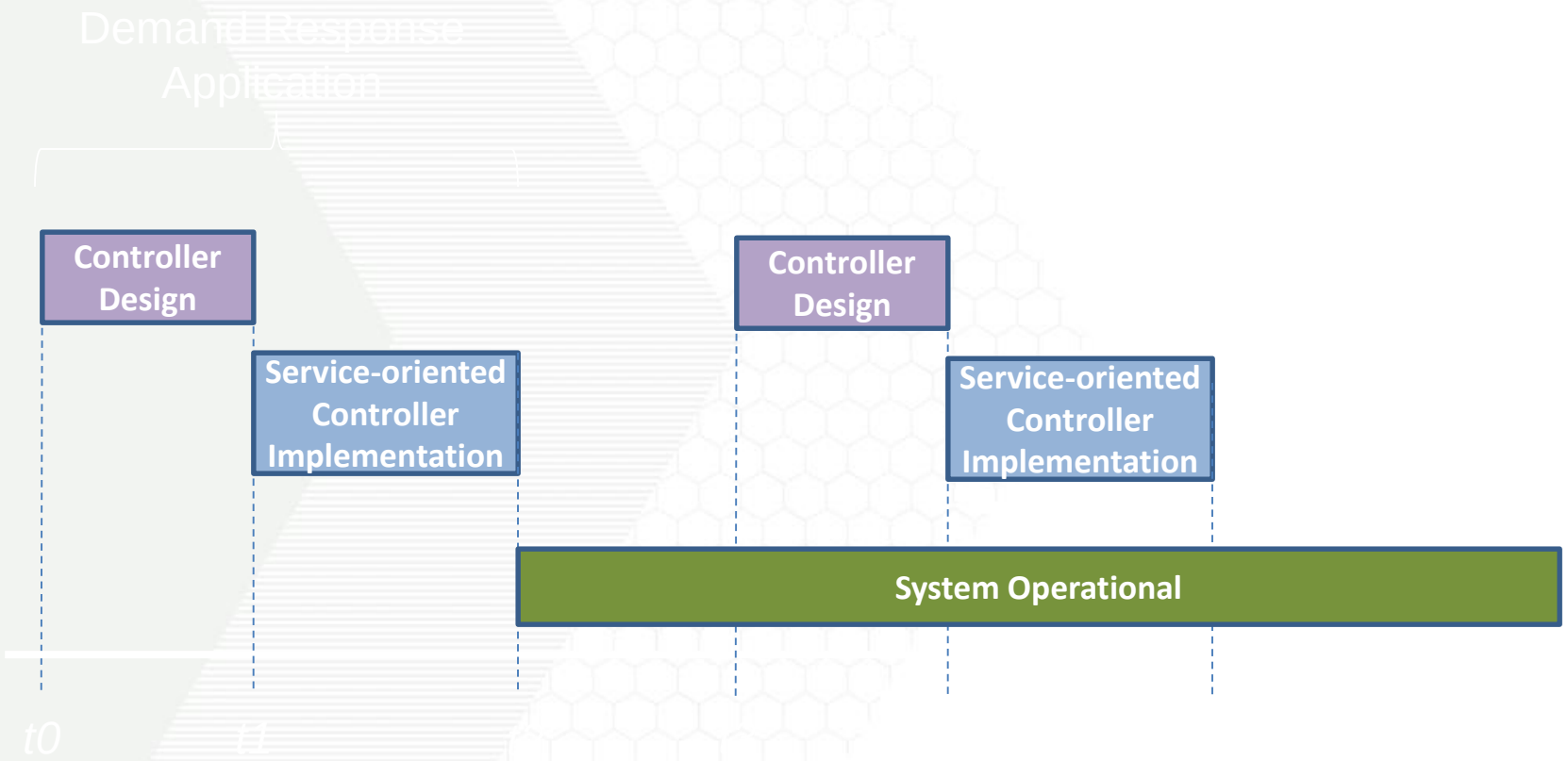
SMART GRID TEST-BED 2: CPS-ENABLED SOC



SMART GRID TEST-BED 2: CPS-ENABLED SOC

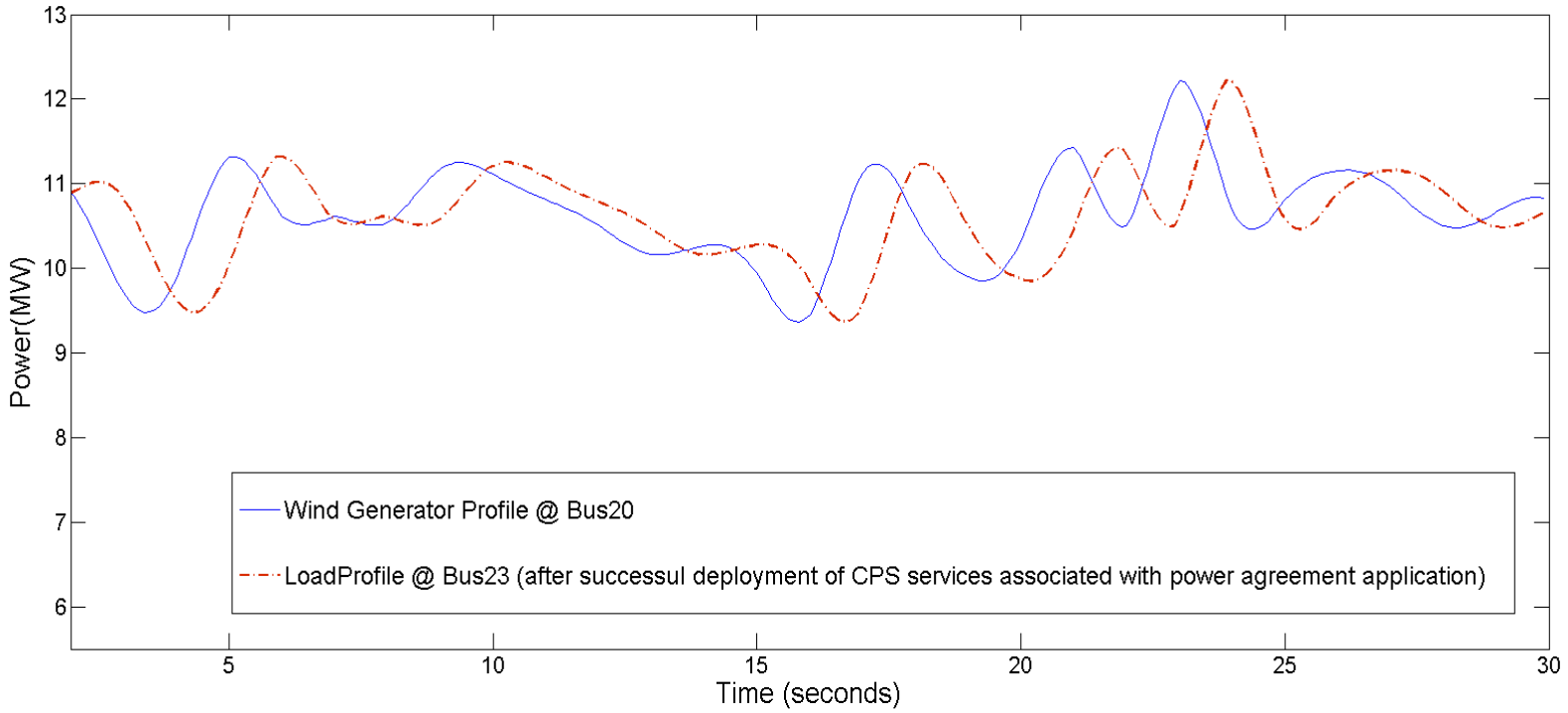


SMART GRID CASE STUDY



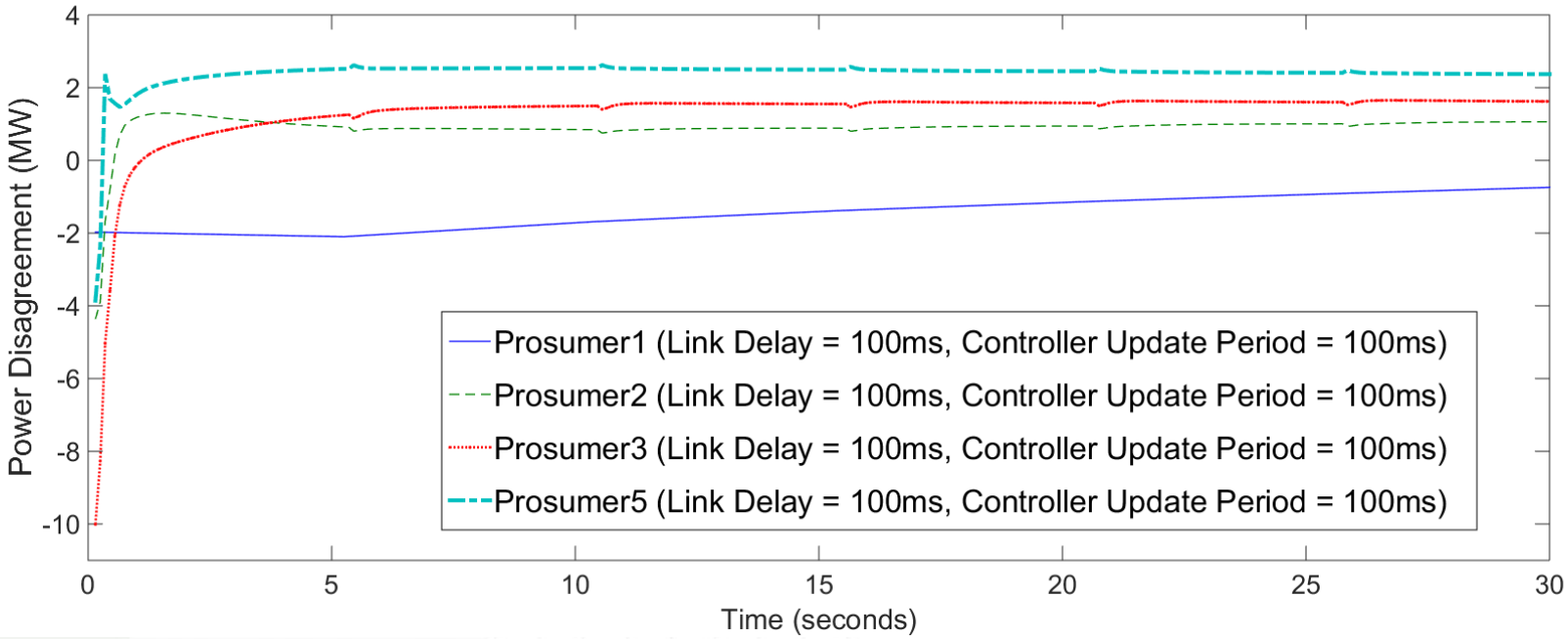
SMART GRID TEST-BED: DEMO

(DEMAND RESPONSE APPLICATION)



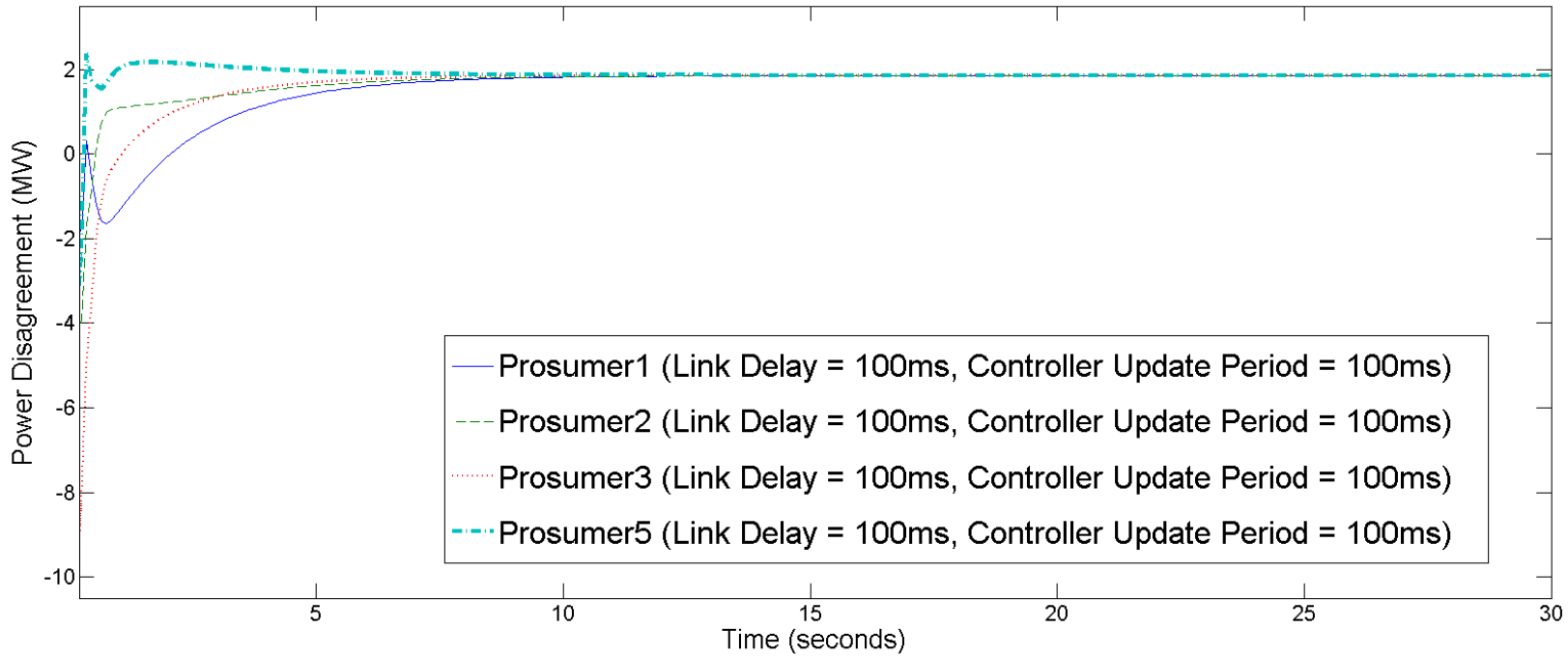
SMART GRID TEST-BED: DEMO

(POWER AGREEMENT APPLICATION: ENTERPRISE DOMAIN SOC)



SMART GRID TEST-BED: DEMO

(POWER AGREEMENT APPLICATION: CPS-ENABLED SOC)



CONCLUSIONS

Old boundaries have faded:

- IT vs. CPS.
- Safety vs. security.

Industry needs to apply best standards in all categories of systems, both consumer and industrial.

We need improved methods:

- Methodologies and tools.
- Architectures.
- Hardware and software components.