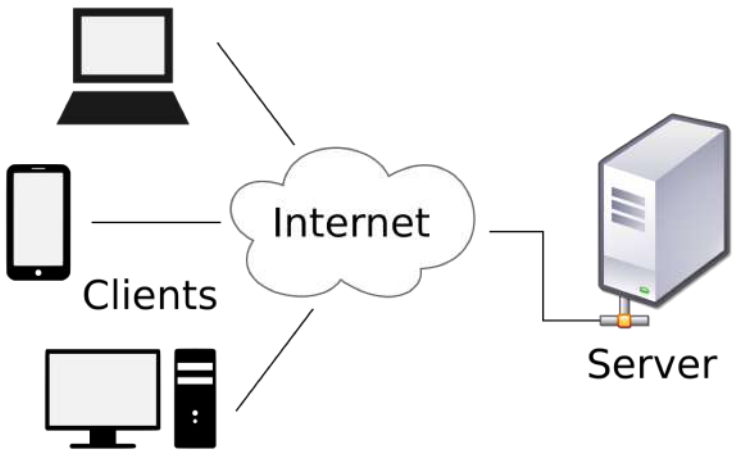
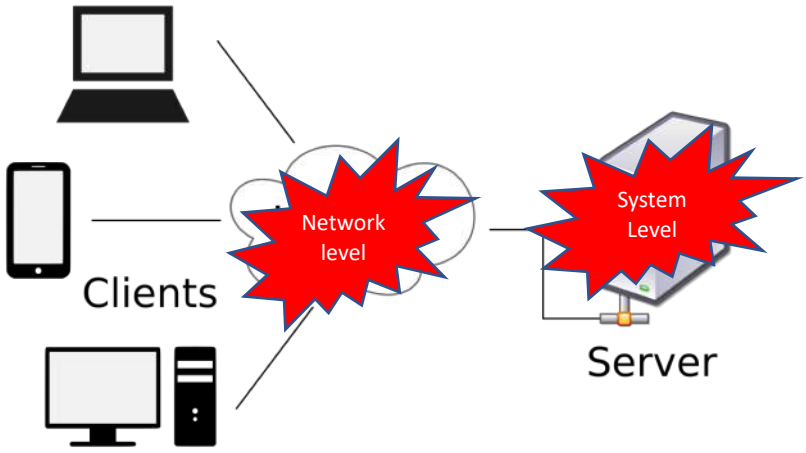
Resilient System Design with Examples

Sahiti Bommareddy

High Overview of General System



Failure Scenarios



Topic 1: Addressing System Level Failure

I want to build a resilient system for my clients such that clients can be masked from failures

Primary – Backup Approach

To mask the failure of 1 server, another replica will be employed

Protocol:

- Client Requests are handled by Primary only
- If primary fails, backup takes over
- Backup is kept up-to-date with periodic checkpointing/ per update

Issues with Primary – Backup Approach

To mask the failure of 1 server, another replica will be employed

Protocol:

- Client Requests are handled by Primary only
- If primary fails, backup takes over
- Backup is kept up-to-date with periodic checkpointing/ per update

What are the potential issue?

Issues with Primary – Backup Approach

To mask the failure of 1 server, another replica will be employed

Protocol:

- Client Requests are handled by Primary only
- If primary fails, backup takes over
- Backup is kept up-to-date with periodic checkpointing/ per update

What are the potential issue?

- Client is exposed to primary failures
- Lost updates

Outsource

What would you do ?

Scope the Problem

What will be the questions you would ask ?

Scope the Problem

What will be the questions you would ask ?

- System Requirements
  - Types of fault
  - Number of faults
  - Performance Requirements

Scope the Problem: Option 1

What will be the questions you would ask ?

- System Requirements
  - Types of fault : Fail-Stop Faults
  - Number of faults : f=1
  - Performance Requirements

Scope the Problem: Option 1

What will be the questions you would ask ?

- System Requirements
  - Types of fault : Failstop faults
  - Number of faults : f=1
  - Performance Requirements

SMR in which agreement is needed from f+1 servers out of 2f+1 total servers

Scope the Problem: Option 2

What will be the questions you would ask ?

- System Requirements
  - Types of fault : Byzantine faults
  - Number of faults : f=1
  - Performance Requirements

Scope the Problem: Option 2

What will be the questions you would ask ?

- System Requirements
  - Types of fault : Byzantine faults
  - Number of faults : f=1
  - Performance Requirements

SMR in which agreement is needed from 2f+1 servers out of 3f+1 total servers

Scope the Problem: Option 2.1

What will be the questions you would ask ?

- System Requirements
  - Types of fault : Byzantine faults
  - Number of faults : f=1
  - Additional Requirement : Long system life with robustness
  - Performance Requirements

SMR with diversity and proactive recovery in which agreement is needed from (2f+k+1) servers out of (3f+2k+1) total servers

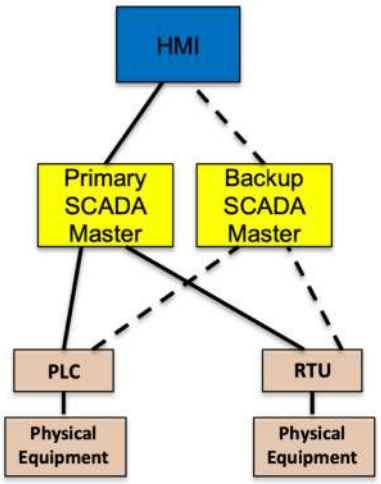
Use Cases

- Cloud / IT Services
  - File Systems
  - SDN Controllers
  - Schedulers
- Critical OT Services
  - ICS (Industrial Control Systems)

Basic Blocks in a SCADA system

HMI : Human Machine Interface
SCADA : Supervisory Control and Data Acquisition Systems
PLC : Programmable Logical Controller
RTU : Remote Terminal Unit

Power System Protocols:
Modbus
DNP3
IEC61850

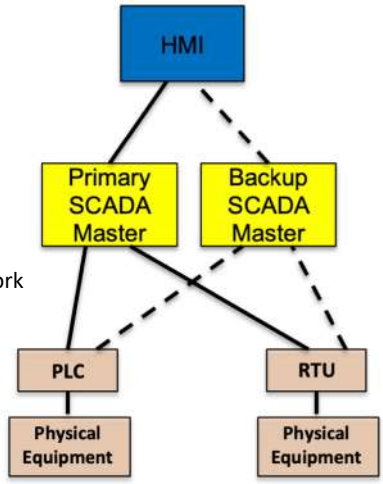


Vulnerabilities in SCADA system

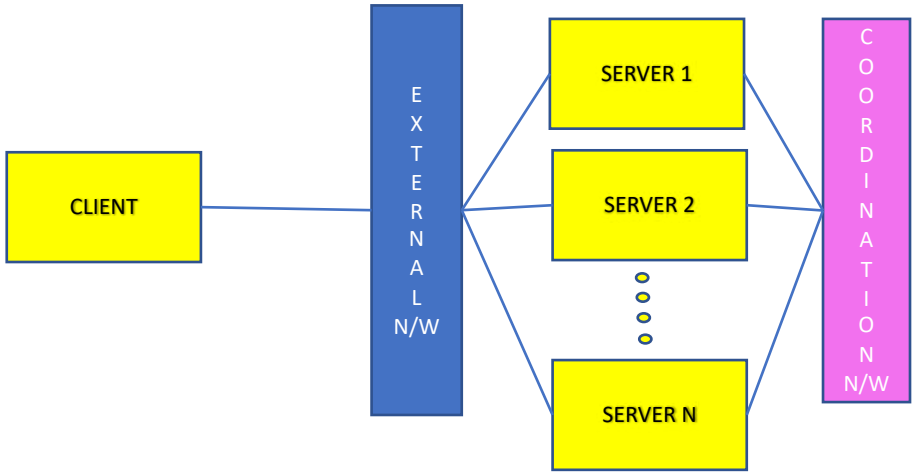
SCADA systems are vulnerable on several fronts:

SCADA system compromises (Fail–Stop and Byzantine)
- SCADA Master : system-wide damage
- RTUs , PLCs : limited local effects
- HMIs

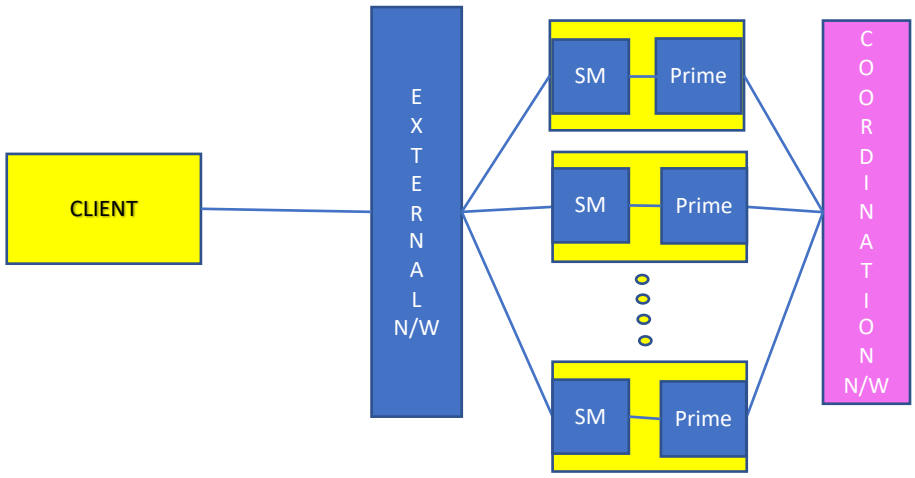
Network level attacks
- Routing attacks that disrupt or delay communication
- Isolating critical components from the rest of the network



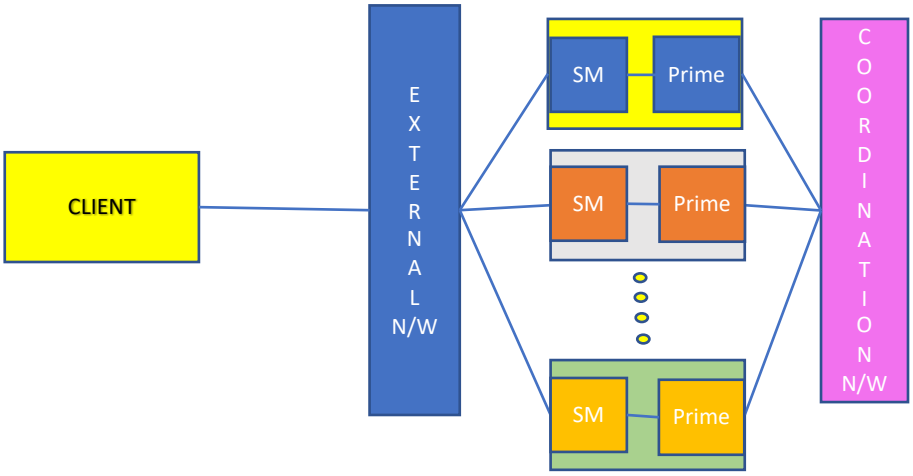
Byzantine fault Tolerant System



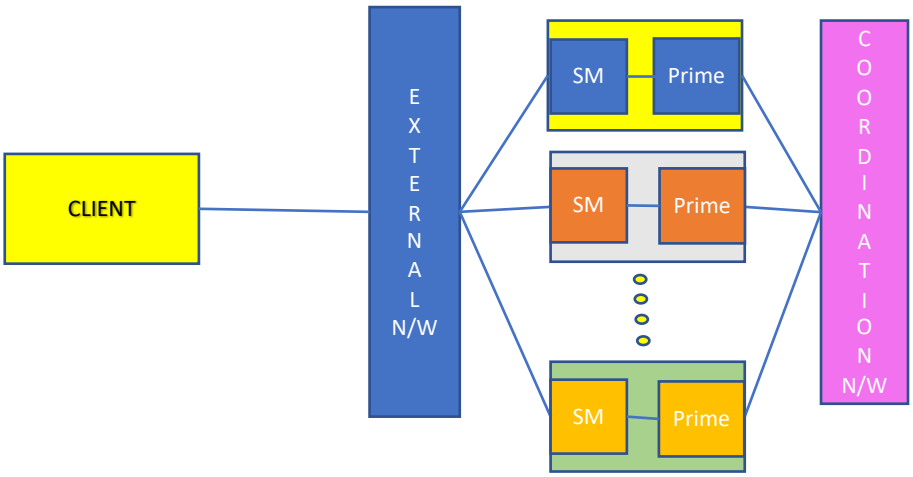
Byzantine fault Tolerant System



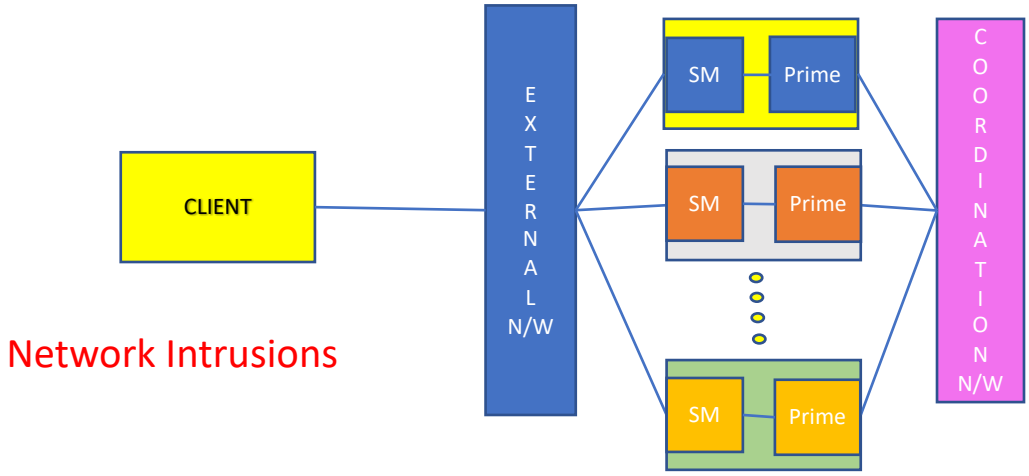
Byzantine fault Tolerant System



What is missing?



What is missing?



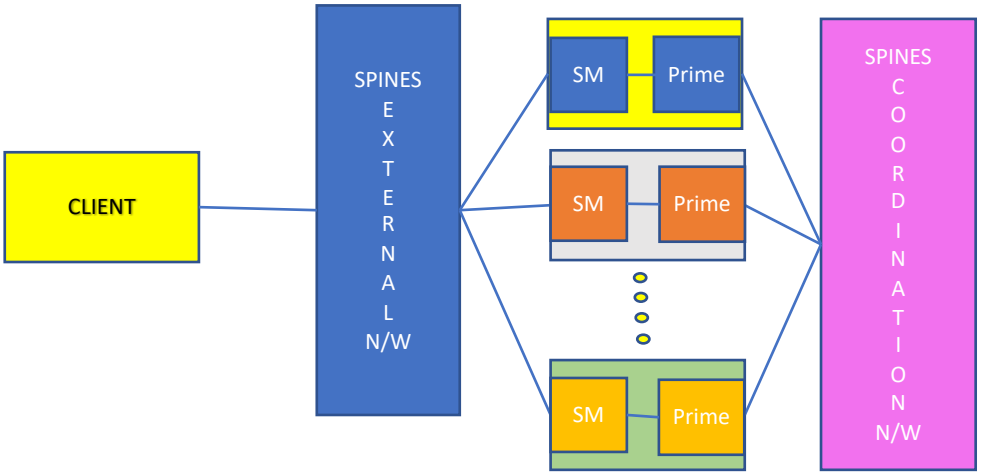
Network Intrusions

Designing Intrusion Tolerant Networks

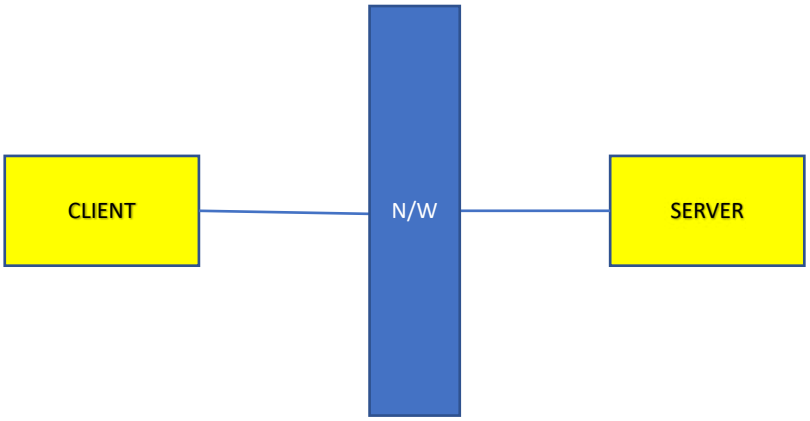
Requirements of such networks:

- Protection against –
  - Link Level Tampering
  - Single ISP Meltdown
  - DDoS attacks
  - BGP Hijacking
  - Byzantine Node failures (forwarders and sources)
  - Multiple QoS (Reliable and Priority based)

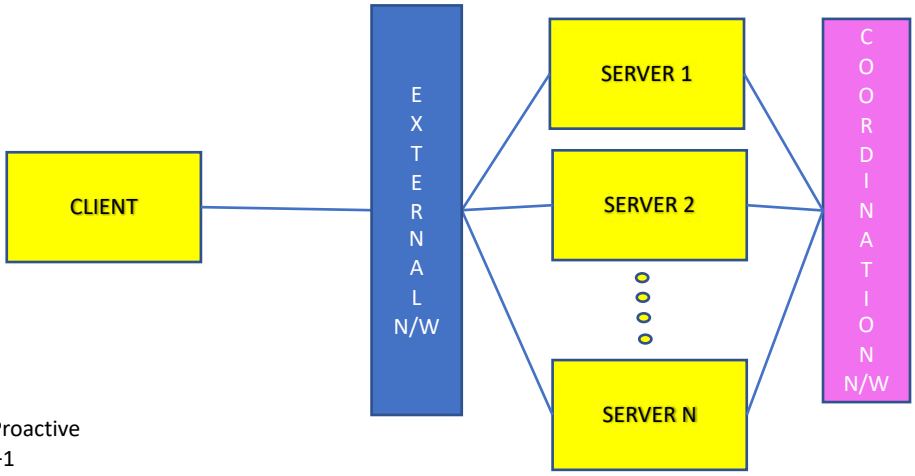
Byzantine fault Tolerant + Network Intrusion Tolerant System



Recap : System



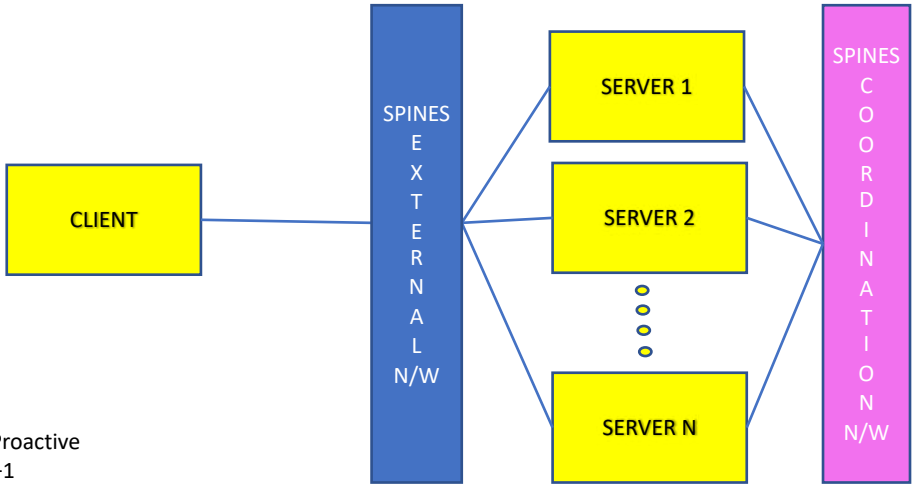
Recap : Fault Tolerant System



Fail-Stop : 2f+1
Byzantine: 3f+1

Byzantine with Proactive Recovery:3f+2k+1

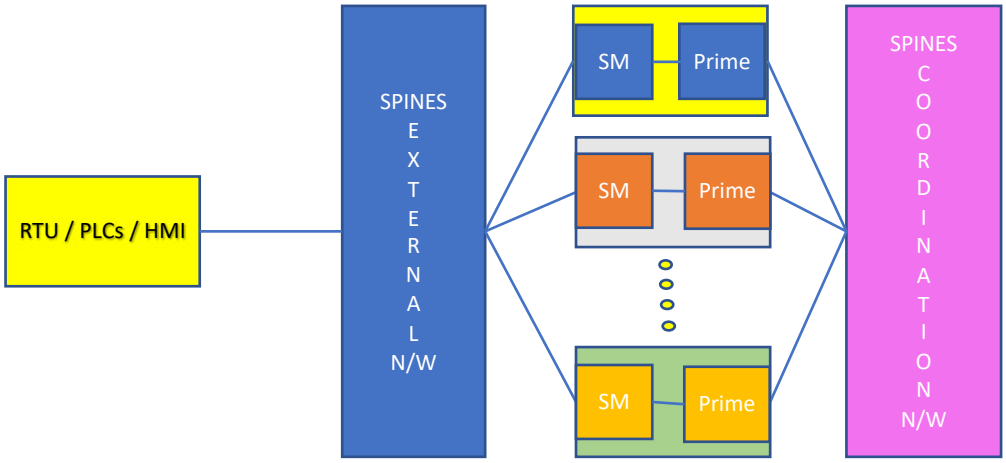
Recap : Fault Tolerant System + Network Intrusion Tolerant



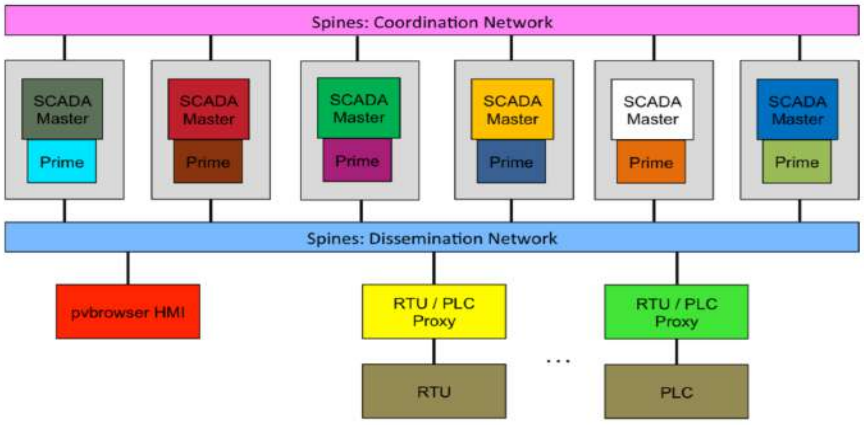
Fail-Stop : 2f+1
Byzantine: 3f+1

Byzantine with Proactive Recovery:3f+2k+1

Network-Attack-Resilient Intrusion-Tolerant SCADA for the Power Grid



SPIRE Architecture



Example Spire system deployment with six replicas.

Performance Measurements

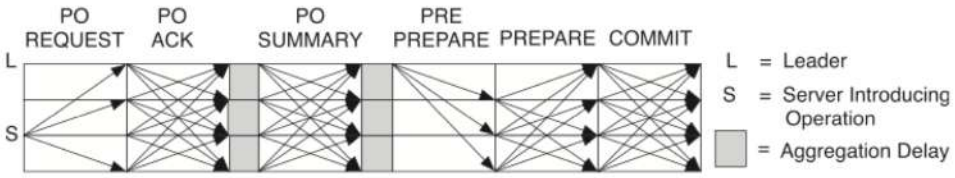
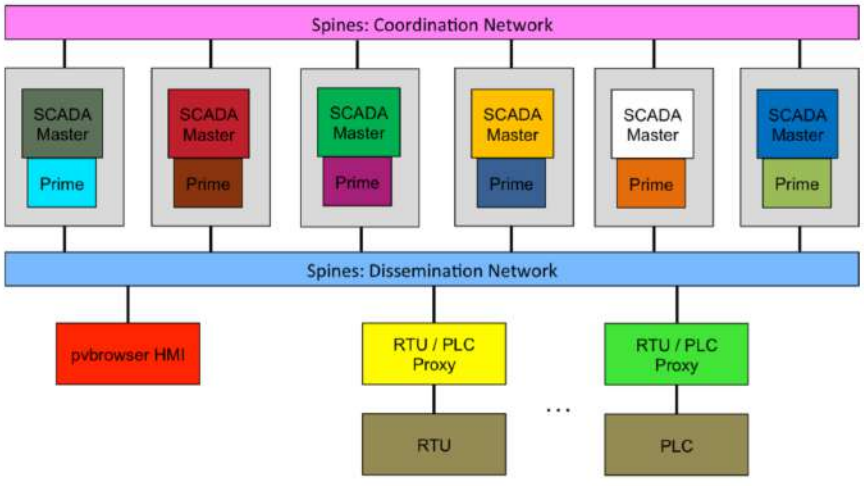


Fig. 2. Fault-free operation of Prime ($f = 1$).

Single Site 6 Configuration Features



Example Spire system deployment with six replicas.

Architecture Choices

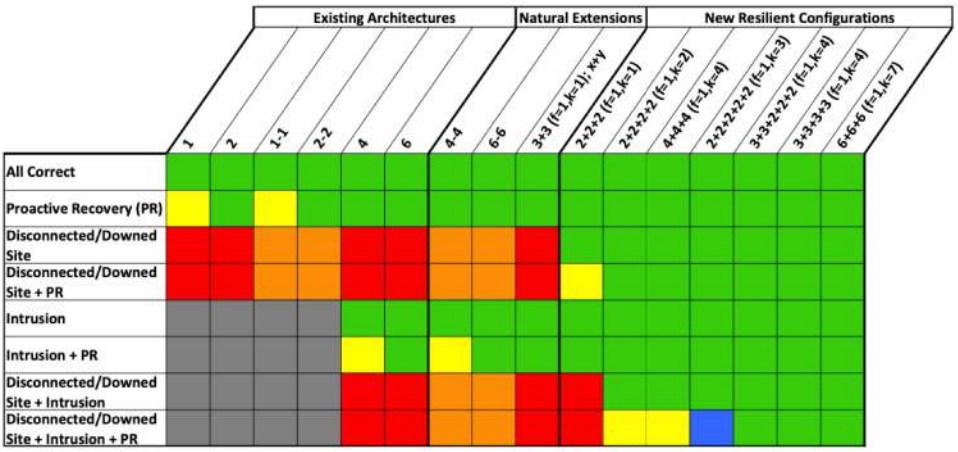


Fig. 2. Illustration of specific SCADA system configurations' ability to support the threat model we consider, including all combinations of a replica being unavailable due to proactive recovery, a site disconnection due to network attack or failure, and an intrusion (SCADA master compromise).

3+3+3+3

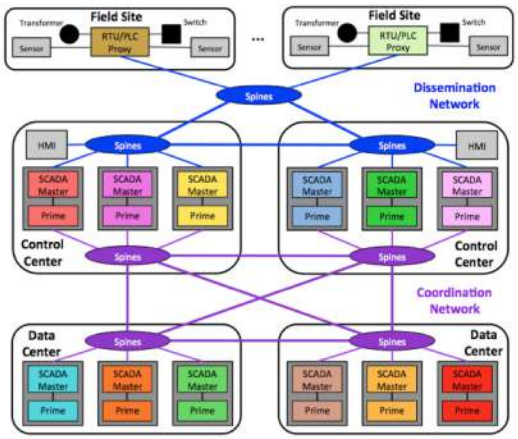


Fig. 6. Spire software architecture for configuration "3+3+3+3"

SPIRE Performance

3+3+3+3 Configuration Performance

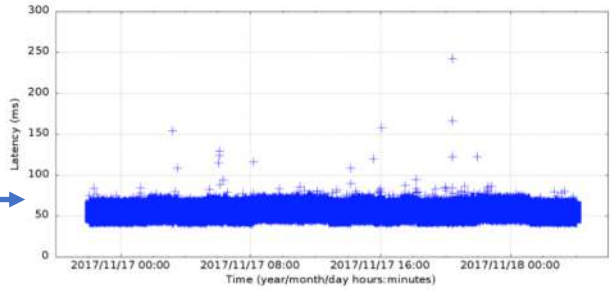


Fig. 8. Update latencies over 30-hour wide-area deployment

	Avg Latency	% < 100ms	% < 200ms	0.1 percentile	1 percentile	50 percentile	99 percentile	99.9 percentile
6+6+6	51.4 ms	100.00	100.00	39.5 ms	40.6 ms	51.3 ms	63.8 ms	68.8 ms
3+3+3+3	54.7 ms	100.00	100.00	43.1 ms	44.2 ms	54.7 ms	65.4 ms	67.1 ms
3+3+2+2+2	56.4 ms	100.00	100.00	44.5 ms	45.8 ms	56.3 ms	67.3 ms	69.5 ms
5+5+5+4	57.4 ms	100.00	100.00	45.4 ms	46.6 ms	57.4 ms	68.8 ms	71.8 ms
6+6+6+6	64.8 ms	99.9111	99.9667	50.4 ms	52.2 ms	64.5 ms	82.7 ms	97.7 ms

TABLE II
SCADA CONFIGURATION PERFORMANCE ON LAN WITH EMULATED LATENCIES BETWEEN SITES FOR 36000 UPDATES OVER 1 HOUR

Current Lab Research Directions

- Real-time Byzantine Resiliency
- Resilient Systems under Cascade Failure

Real-time Byzantine Resiliency

We work at the level of Power Grid Substations

We want to build resilient systems that have least latency

Specific Use Case: High Voltage Protection Relays

- Relays are devices that can protect the grid
- When there is fault in grid, relay trips the breaker to protect grid
- The current state-of-the-art systems employ multiple protective relays with unilateral power to each protective relays

Issue in Current Systems

- A protective relay that does not trip when they should can cause irreparable damage to the grid and its connected customers
- A protective relay that does unnecessarily trip causes a major disruption to a large number of customers.

As a consequence, protective relays become an attractive target for malicious actors, especially at high voltage levels

Scope the Problem

Several rigid factors as design constraints

- Very exact real time constraint :
  - The relay has to react within a quarter of a power cycle
  - In a 60Hz system (e.g. in North America), a quarter cycle amounts to 4.166ms
- Economic Factors :
  - The protective relay is a reasonably expensive device (tens of thousands of dollars), and there are many of them in the system to support every substation (~1000s).
- Require resiliency with long system life for continuous availability.
- Require seamless substation integration into existing environments

Current Lab Research Directions

- Real-time Byzantine Resiliency
- Resilient Systems under Compound Threats

Resilient Systems under Compound Threat

Specifically:

Let us say we do build intrusion tolerant system
This system is impacted by natural disaster (e.g. Hurricane)
In real world, this weakened system is lucrative target for cyber attacks.

How to build resilient intrusion tolerant systems under such compound threats?

References

- http://www.dsn.jhu.edu/papers/scada_DSN_2018.pdf