ATTACK-AWARENESS FOR SPIRE (INTRUSION-TOLERANT SCADA)

Tiger Gao, Dan Qian, Elaine Wong, & Jason Wong
1. BACKGROUND

What is Spire? What is SCADA?
What is SCADA?

- Supervisory Control and Data Acquisition
- Allows for centralized control over systems that are spread out over large distances
- Monitors and controls devices that collect information from and interact with the physical world such as power breakers, valves, HVAC controllers, factory machine computers
- Used widely in critical infrastructure:
  - Electrical grids, water treatment facilities, power plants buildings, factories, facilities etc.
SCADA System Layout

- **PLCs/RTUs**
  - Sensors and/or control units
  - Closest level of interaction with process

- **SCADA Master**
  - Coordinates network of PLCs/RTUs

- **HMI**
  - Interface through which human operators can monitor and give commands to the system
Current SCADA Vulnerabilities

- Use in critical infrastructure makes SCADA systems valuable targets to attack, especially by state actors
  - Stuxnet
- Compromised SCADA systems can disable or potentially permanently destroy critical infrastructure
  - 2015 Ukrainian Power Grid Attack
    - Power cut for 230,000 people
- Transition from closed networks to IP exposes SCADA systems to the internet, easier to attack
  - Experiment with honeypot of PLCs were attacked 39 times from 14 countries in a month (Aron)
What is Spire?

- Spire is an open-source, intrusion tolerant SCADA solution over IP
- Many components work together to prevent attacks
  - Spines: Networking
  - Prime: Timely Byzantine Fault Tolerance
  - Multicompiler: Entropy
  - Scheduled Resets
- RTU/PLC Proxy
How It Actually Works

Spire Architecture: Single Control Center
Effectiveness - Does It Work?

- **Short Answer:** ✓
- Resisted an extensive attack by a Sandia National Laboratories
- Also retains timeliness consistently, with some variations because of special circumstances
2. OUR WORK

Introducing attack-awareness to an intrusion-tolerant system
Attack-Awareness

- Spire handles many attacks silently, without notifying a human operator
  - Bad leader in byzantine agreement protocol
  - Dumb DDoS attacks (from compromised device)
- These problems could be easily resolved with human awareness (i.e. unplugging a compromised master)
- Our goal: Displaying HMI alerts to notify operators of ongoing attacks of different types
Attack Types of Note

DDoS Attacks

▪ Dumb attacks from adversaries who may have compromised part of the system and are spamming it with random messages
▪ Can happen at HMI, proxy, and firewall levels

Bad Leader Attacks

▪ A compromised SCADA master who leads the agreement protocol sends inconsistent messages to other masters, delaying instruction execution
▪ This occurs inside prime
What We Have Accomplished (Part I)

- First few weeks: Just getting to know the code
  - Exploring the codebase, learning about Spire in general
  - Setting up VMs to work with HMIs on PVBrowser
  - Getting to know PVBrowser
- Then: Learning how to set up the full system (HMIs, PLCs, SCADA masters, oh my!)
What We Have Accomplished (Part II)

- First: A plain alert message on HMI for internal DDoS attacks
- This later became a table displaying which SCADA master is spamming the HMI, and is therefore the compromised machine
- Similarly, we display an alert when the proxy is being spammed
- Adding an alert for bad leaders in the Prime agreement protocol
- Adding logging for possible firewall spam
DEMOS!!!!!!
Challenges

- Getting through the codebase
- PVBrowser/Ubuntu/Centos issues with freezing
- Mostly, didn’t/haven’t had enough experience with the system to develop an intuition of where different kinds of bugs could be coming from
  - Weird problems like clock synchronization for Spines communication or temp files that we needed sudo access to delete
  - Sometimes just had to restart the whole thing
Future Work

▪ Generalizing the alert system
  ▫ Ex. There are a lot of places where Prime can detect suspicious activity, but doesn’t alert
  ▫ Detecting replay attacks (a less dumb DDoS attack) which forces system to decrypt before discarding message
▪ Integrating logging for firewall spam into the HMI
▪ Cleaning up; this was mostly proof of concept
THANK YOU

to Amy Babay, for setting us up, coming to our meetings, guiding us through code and bugs

to Sam Beckley, for helping us get started with the HMI

to Yair Amir, for giving us the opportunity to work on this incredibly impactful and important project