Towards High-Interaction Virtual ICS Honeypots-in-a-Box

DANIELE ANTONIOLI  ANAND AGRAWAL  N. O. TIPPENHAUER
Overview

In this work we:

- Present the design of a realistic ICS honeypot
  - Satisfying traditional, and ICS requirements
  - That is high-interaction, virtualized and low-cost
- Show an implementation of such a design
  - Targeting ICS based on Ethernet/IP
  - High-interaction without full virtualization
  - Compatible with Software-Defined Networking
- Discuss its evaluation
  - S3’s Capture-The-Flag (CTF) for ICS
Industrial Control Systems (ICS)

• Industrial Control Systems (ICS)
  ▶ Connected devices, managing an industrial process
  ▶ Control and monitor: PLC, SCADA, HMI
  ▶ Physical: sensors, actuators
  ▶ Cyber: switches, routers, gateways

• ICS security is a major challenge
  ▶ Internet-facing control networks
  ▶ Cyber and physical attacker surface
  ▶ Legacy-code, uncertified devices
Real Water Treatment ICS

daniele_antonioli@sutd.edu.sg Towards High-Interaction Virtual ICS Honeypots-in-a-Box
Real Water Treatment ICS

SCADA

Historian

VPN/Gateway

Attacker

Internet

Process 1

Process 2

Process n

PLC1a

PLC1b

PLC2a

PLC2b

PLCna

PLCnb

L0 Network

L1 Network

L0 Network

RIO

Actuators

Sensors

RIO

Actuators

Sensors

RIO

Actuators

Sensors

RIO

Actuators

Sensors

RIO

Actuators

Sensors

RIO

Actuators

Sensors

...
Our Idea: ICS Honeypots

daniele_antonioli@sutd.edu.sg Towards High-Interaction Virtual ICS Honeypots-in-a-Box  

Motivation
ICS Honeypots: Introduction

• Systems *intended* to be probed, attacked, and compromised
  ▶ Lures the attacker impersonating an ICS
  ▶ Stop, or slow-down the attack
  ▶ Study attacker’s behaviours

• Classifications
  ▶ Infrastructure: real vs. virtual (vs. hybrid)
  ▶ Realism: low-interaction vs. high-interaction
  ▶ Role: client vs. server
  ▶ Usage: research vs. production
Our Honeypot: Attacker Model

• Assumptions
  ▶ Honeypot reached over the Internet
  ▶ Vulnerable interface determines the attacker surface

• Capabilities
  ▶ Fingerprinting: addresses, ports, protocol
  ▶ Protocols: knowledge of all protocols used in system
  ▶ Physical system: limited knowledge of process and devices

• Interactions
  ▶ Denial-of-Service: flood the network
  ▶ Man-in-the-Middle: passive and active
  ▶ Device impersonation: valid and malformed packets
  ▶ Sabotage: trigger actions through malicious commands
Our Honeypot: Requirements

• High-interaction ICS honeypot
  ▶ Simulate the physical process
  ▶ Simulate the ICS devices: control logic, services
  ▶ Emulate the network infrastructure

• Low-cost
  ▶ Reconfigurable
  ▶ Scales

• ICS requirements
  ▶ *Time*: completion of tasks, and delivery of packets
  ▶ *Determinism*: schedule of tasks, and order of packets
Simple Design Approach

• How about an OpenPLC\(^1\) indexed on shodan.io?
  ▶ Classification: real, low-interaction, server
  ▶ Pros: low-cost, configuration
  ▶ Cons: realism, scale

\(^1\)http://www.openplcproject.com/
Our Honeypot: Design Choices

- **Virtual and high-interaction:**
  - Simulation of physical process and ICS devices
  - Lightweight network emulation
  - Runs in-a-Box (with SDN support)

- ICS requirements
  - Time: real-time emulation, and simulation
  - Determinism: scriptable environment
Our Honeypot: Architecture

Proposed Honeypot (top) vs. Real ICS (bottom).

High-Interaction virtual honeypot

Real ICS/SCADA system

Emulated network

Simulated PLC

Simulated HMI

Physical Process Simulation

VPN

Device

SSH

Telnet

Gateway

Internet

Attacker

Physical Process Simulation

Physical Process

HMI

PLC

VPN

Device

SSH

Telnet

Gateway

Proposed Honeypot (top) vs. Real ICS (bottom).
MiniCPS Framework [CPS-SPC 15]

"MiniCPS: A toolkit for security research on CPS Networks."

https://github.com/scy-phy/minicps

(C)yber → Network Emulator
(P)hysical → Physical Layer Simulation and API
(S)ytem → Devices Simulation
MiniCPS Framework [CPS-SPC 15]

"MiniCPS: A toolkit for security research on CPS Networks."

https://github.com/scy-phy/minicps

(C)yber → Network Emulator
(P)hysical → Physical Layer Simulation and API
(S)ytem → Devices Simulation
Honeypot Implementation

High-Interaction virtual honeypot

EtherNet/IP

SDN Controller

Switch

VPN
Device 192.168.1.76

SSH
Telnet
Gateway 192.168.1.77

HMI
192.168.1.100

PLC1
192.168.1.10

PLC2
192.168.1.20

PLC3
192.168.1.30

PLC4
192.168.1.40

Physical Layer API

Physical Process Simulation

Internet

Attacker

VPN Device 192.168.1.76

SSH Telnet Gateway 192.168.1.77

HMI 192.168.1.100

PLC1 192.168.1.10

PLC2 192.168.1.20

PLC3 192.168.1.30

PLC4 192.168.1.40

SDN Controller

Network

Component Logic

Component Logic

Physical Layer API

Physical Layer Simulation
Realistic Attack Propagation

Attack propagates over the simulated components
PLC Implementation

- Allen-Bradley ControlLogix
  - Same IP, MAC, and netmask
  - Simulated control logic (modifiable in real-time)
  - Ethernet/IP server on port 44818, and client
  - Same monitoring Webserver
Network Gateway Device Implementation

• Moxa OnCell IP gateway
  ▶ Eg: provide IP over 3G connection
  ▶ SSH server with default credentials
  ▶ Telnet server with default credentials (plaintext authentication)

• Virtual implementation
  ▶ Same IP, MAC, and netmask
  ▶ `sshd` on port 22 with default credentials
  ▶ `telnetd` on port 23 with default credentials
  ▶ Attacker gets a (chrooted) shell
• Capture-The-Flag (CTF)
  ▶ Cybersecurity competition (online and offline)
  ▶ Two types: attack-defense, and jeopardy-style

• S3 CTF was *online* and *jeopardy-style*
  ▶ Tasks divided into categories (cyber, physical)
  ▶ A task has a description, some clues, and reward points
  ▶ A task is solved finding and submitting the correct flag
  ▶ Team that captures most flags (scores most points) wins
Evaluation: S3 CTF Honeypot Setup

- Honeypots running on AWS EC2 instances\(^2\)
  - Linux, m3-medium: 1 vCPU, 3.75 GB RAM, 1 GB SSD
  - Set up a single instance (tricky)
  - Replicate it (easy, press a button)

- Vulnerable gateway interface
  - SSH’s credentials given (CTF)
  - Attacker has a (chrooted) shell

- Replicated part of a water treatment ICS
  - Two tanks, sensors, and actuators
  - Four PLCs and a HMI
  - Ethernet/IP protocol, star topology

\(^2\)https://aws.amazon.com/ec2/
Evaluation: S3 CTF Challenges

1. Network warm up
   - Task: eavesdrop what PLC2 sends to PLC3
   - Required: testbed’s topology, MitM attack skills
   - Solution: passive MitM attack between PLC2 and PLC3

2. Ethernet/IP warm up
   - Task: can you use cpppo\(^3\) to access README: 2 tag?
   - Required: Ethernet/IP industrial protocol
   - Solution: Ethernet/IP request (read)

3. Overflow the Raw water tank
   - Task: overflow the Raw water tank controlled by PLC1
   - Required: physical process setup
   - Solution: Ethernet/IP packets to overflow the tank

\(^3\)https://github.com/pjkundert/cpppo
4 Denial of Service HMI
   ▶ Task: change the keep alive value sent from the HMI to PLC3?
   ▶ Required: active MitM brute-force attacks
   ▶ Solution: active MitM with packet dropping

5 Overflow the Ultra-filtration tank
   ▶ Task: control PLC4 to overflow the Ultra-filtration tank
   ▶ Required: all the previous challenges
   ▶ Solution: active MitM with selective filter
# Evaluation: S3 CTF Results

## Table 1: CTF Results Summary.

<table>
<thead>
<tr>
<th>Teams</th>
<th># Captured Flags</th>
<th># Distinct Cmds</th>
<th># Executed LOC</th>
<th># Recon Tools</th>
<th># Attack Tools</th>
<th>Most Used Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team 1</td>
<td>2</td>
<td>20</td>
<td>1074</td>
<td>3</td>
<td>1</td>
<td>{1, 2, 6, 8}</td>
</tr>
<tr>
<td>Team 2</td>
<td>5</td>
<td>30</td>
<td>2488</td>
<td>6</td>
<td>2</td>
<td>{1, 2, 3, 4, 5, 6, 7, 8}</td>
</tr>
<tr>
<td>Team 3</td>
<td>3</td>
<td>23</td>
<td>2045</td>
<td>5</td>
<td>2</td>
<td>{1, 2, 3, 4, 6, 7, 8}</td>
</tr>
<tr>
<td>Team 4</td>
<td>4</td>
<td>27</td>
<td>963</td>
<td>5</td>
<td>2</td>
<td>{1, 2, 3, 4, 6, 7, 8}</td>
</tr>
<tr>
<td>Team 5</td>
<td>1</td>
<td>3</td>
<td>52</td>
<td>1</td>
<td>0</td>
<td>{1}</td>
</tr>
</tbody>
</table>

# : Number Of, LOC : Lines Of Code

*\{1: ettercap, 2: nmap, 3: netstat, 4: tcpdump, 5: tshark, 6: ifconfig, 7: cppo, 8: ping\}
Conclusions

In this work, we:

• Address the problem of designing a realistic honeypot for ICS
• Present the design of an *high-interaction, virtual*, low-cost ICS honeypot that runs *in-a-Box*
• Show an implementation of such a design based on the MiniCPS framework [CPS-SPC15]
• Discuss its evaluation in the context of an ICS CTF [paper draft]

**Acknowledgments:** Anand, Nils, and S3 participants’.

Thank you for your time!