ICS\SCADA Cyber security
Seminar Agenda

1. CYBERBIT Background
2. ICS\SCADA networks overview
3. ICS\SCADA Challenges, threats and vulnerabilities
4. CYBERBIT SCADA Shield
5. SCADA Shield Demo
Company Profile

A wholly-owned subsidiary of Elbit Systems

More than 15 years of vast experience in cyber & intelligence

providing solutions and products to the commercial defense markets

Annual sales three digit number (in M USD) and a high growth rate
protecting the core assets of high risk - high value enterprises and critical infrastructures from advanced cyber threats
Four Must-Haves for Advanced Threat Response

Sensors
Powerful and well spread

Cyber Data Warehouse
Comprehensive and Behavior Driven

Incident Response Platform
Efficient and Centralized

Training and Simulation
That's Exactly What We Do

CYBER SHIELD TnS
(TRAINING AND SIMULATION)

CYBER SHIELD Sensors
Powerful and Well-Spread

CYBER SHIELD AnD
Behavior Driven Data Warehouse

CYBER SHIELD MnR
SOC Incident Response Platform

CS-ICS (SCADA)
CS-IT
CS-Mobiles
Overview

• SCADA is a system for remote monitoring and control that operates over communication channels.

• It is a type of industrial control system (ICS). Industrial control systems are computer-based systems that monitor and control industrial processes that exist in the physical world.

• SCADA systems historically distinguish themselves from other ICS systems by being large-scale processes that can include multiple sites, and large distances.

• These processes include industrial, infrastructure, and facility-based processes such as:
  ➢ Power generation, fabrication, and refining.
  ➢ Water treatment and distribution, wastewater collection and treatment.
  ➢ Oil and gas pipelines, electrical power transmission.
  ➢ Buildings, airports, ships, and space stations.
Common SCADA systems components

- **PLC** - Programmable Logic Device
- **RTU** - Remote Terminal Units
- **HMI** - Human Machine Interface
- **GW\FEP** - Gateway of Front end processor
- **SCADA server** - Responsible for communication between CC applications and field devices such as RTUs and PLCs
- **Historian\Reporting server** - Stores data, events and alarms
Typical ICS/SCADA Systems - Transportation
(R)Evolution of Control Systems

1st generation: “Monolithic”

- Direct connection
- Or serial bus
- Fieldbus (ModBus, PROfibus)

2nd generation: “Distributed”

- Ethernet TCP/IP (Controls network)
- Fieldbus (ModBus, PROfibus)

3rd generation: “Networked”

- Ethernet TCP/IP (Controls network)
- SMS
- Wireless access point

4th generation: “IoT”

- Ethernet TCP/IP (Controls network)
- Office PCs
- Data Warehouse
- Web server
- Firewall
- Engineering station
- Operator Terminal (HMI)
- Database, data historian file
- INTERNET
- Office PCs
- Engineering station
- Operator Terminal (HMI)
- Database, data historian file
- INTERNET
ICS\SCADA Challenges, threats and vulnerabilities
So many concerns...

Assuring operational continuity is harder than ever
With So Many Built-in Challenges

Old unsecured technology
Increasing network connectivity
Geographically dispersed sites
Multiple vendors and protocols
Exacerbating Regulation
Vulnerabilities by ICS level

Based on Firerye report - 2016
ICS-Specific vulnerability disclosures

Based on Firerey report - 2016
ICS\SCADA Cyber attacks
Conficker (Warm)
Target: French Navy
Impact: Failure to download flight plans.

Night Dragon (Trojan)
Target: Exxon, BP, Shell and others
Impact: Collect data from SCADA system

Flame (Malware)
Target: Iranian Oil Ministry, Iranian National Oil co.
Impact: Steal and delete information from ICS\SCADA systems

Black Energy (Malware)
Target: Ukrainian Power Grid
Impact: Massive data deletion and power shutdown to more than 225,000 people

Stuxnet (Warm)
Target: Iran’s nuclear facility
Impact: Destroyed multiple centrifuges

DUQO (Malware)
Target: Western countries and others
Impact: Conduct reconnaissance on ICS\SCADA

Havex (Malware)
Target: General Electric and others
Impact: Scan for ICS\SCADA devices\servers and send data to C&C servers

IRONGATE (Malware)
Target: Siemens S7-315 PLCSIM
Impact: Process manipulation, sending false data to HMI and malicious traffic to PLC

Based on Checkpoint article - 2016
Examples of ICS/SCADA protocols vulnerabilities
SCADA vendors & protocols examples

- DF1
- Modbus
- TCP/RTU/ASCII
- DNP3 / DNPi
- Profibus
- DNP
- SIEMENS Teleperm XP/ S7COMM
- MDLC / MDLC over IP
- IEC60870-5-101/104
- IEC60870-6/IEC61850 - MMS
- IEC
## General ICS\SCADA protocols threats

<table>
<thead>
<tr>
<th>Threat</th>
<th>Description</th>
<th>Recommended action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unauthorized devices in the network</td>
<td>Unauthorized devices in the network can be from different types, but all needs to attended:</td>
<td>Create automatically allowed communication baseline (Whitelist rules) between all network devices based on:</td>
</tr>
<tr>
<td></td>
<td>1. Unfamiliar laptop\PC.</td>
<td>1. 4-tuple (In case of TCP\UDP transmissions):</td>
</tr>
<tr>
<td></td>
<td>2. SCADA device (PLC, RTU, etc.).</td>
<td>2. ICS\SCADA protocol unique device identification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. ICS\SCADA protocol unique message type\operation ID.</td>
</tr>
<tr>
<td>Buffer overflows</td>
<td>Integer type field size that is not complied with the standard.</td>
<td>Alert on every integer field size that is not complied with the standard.</td>
</tr>
<tr>
<td></td>
<td>For example, a field size of type UINT (Unsigned int) can be only between 0 to 65535.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If a message with a bigger value at the receiving side (PLC) and it’s logic does not validate the size it can easily cause a buffer overflow which can result in PLC malfunction.</td>
<td></td>
</tr>
<tr>
<td>Text overflows</td>
<td>Text type field length that is not complied with the standard.</td>
<td>Alert on every text field length that is not complied with the standard.</td>
</tr>
<tr>
<td></td>
<td>For e.g. A string field length can be only between 20 to 25 chars, and a message with a string that its length is bigger than the allowed value at the receiving side (PLC) it can easily cause a text overflow which can result in PLC malfunction.</td>
<td></td>
</tr>
<tr>
<td>Threat</td>
<td>Description</td>
<td>Recommended action</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Illegal APDU length</td>
<td>APDU's with illegal APDU length. Modbus Frame length shall not exceed 260 bytes</td>
<td>Alert on a packet that it’s APDU length exceeded the maximum size.</td>
</tr>
<tr>
<td>Unauthorized Ladder Logic Upload/Download</td>
<td>Retrieve existing logic and to upload new logic commands.</td>
<td>Alert on retrieve existing logic and to upload new logic commands from unknown device.</td>
</tr>
<tr>
<td>Field to field attacks</td>
<td>unauthorized communication from a field device breaching the allowed process policy</td>
<td>Alert on unauthorized communication from a field device breaching the allowed process policy</td>
</tr>
</tbody>
</table>
# CIP (Common Industrial) protocol threats

<table>
<thead>
<tr>
<th>Threat</th>
<th>Description</th>
<th>Recommended action</th>
</tr>
</thead>
</table>
| Crash PLC         | For some specific PLC’s models that communicate using CIP protocol there are vulnerabilities that allows the attacker to send malicious commands that can cause the PLC to crash. For e.g. Rockwell Automation products:  
  - MicroLogix 1100/1400  
  - ControlLogix  
  - 1756-ENBT communication module  
  - And more                                                                 | Alert on Messages sent to CIP Class code: 0xc0 with Service code: 0x97 service      |
| Stop CPU command  | A legitimate command that can be sent to a PLC that will stop it’s functionality.                                                | Alert on:  
  CIP service code = 0x07 from not allowed IP address.                               |
| Crash CPU command | For some specific PLC’s models that communicate using CIP protocol there are vulnerabilities that allows the attacker to send a specific bytes string that can cause the CPU to crash. | Alert on DATA bytes:  
  ENIP->command_specific_data is Equal to = 52022006240103f00c000a0220022401f4f00909880401000100 |
Known attack vectors

**Security**
- Unauthorized communications between two devices
- Unauthorized actions
- Unauthorized maintenance activity

**Operational**
- Malform packets
- Reset commands and crash messages
- Changes in network volumes/speeds/rates
Let's Look at SCADA Shield
SCADA Shield

Assuring Operational Continuity
Cyberbit SCADA Shield – Trustworthy SCADA

Network detection and response – providing **visibility, discovery and security** of ICS networks

- **non-intrusive plug&play**
  - network DPI sensor

- **protocol and hardware agnostic**

- **alerts, forensics & mapping**

**Applications:**
- Power Plants
- Refineries
- Water Supply
- Airports
- Distribution systems
Industrial Control Systems

- SCADA server
- Historian

SCADA Shield

- Blackbox
- Netmap
- Insight
- Alerter

- Real network map
- Overview of all network communications
- Security and malfunction alarms
- “Keep alive” monitoring
- Alarm investigation and analysis
- Network forensics

- unreliable network schematic representation (manual update)
- Alarm handling
- Meter readings
- Remote configuration

The New Operational Toolbox
First TIER European Power Utility Secures its OT Network with CyberShield for SCADA

Selected Solution: Cyber Shield AnD SCADA

A major European power utility (power generation and transmission)

Deployed in country-wide OT transmission network

visibility of the OT network, full network communications in-depth analysis, and enhanced security

IDS and IPS mode
Key Features

- Serial and IP communications monitoring
- White and black listing
- Plug&play deployment – no shutdowns
- Forensics investigation
- IDS and IPS mode
- Netmap – full visualization of the OT network
Main Benefits

- Minimize downtime
- Detect and respond to cyber threats
  - Identify system malfunctions and human errors before damage occurs
- Obtain reliable and genuine network map
- Prevent cyber attacks and misconfigurations (in IPS mode)
- Minimize time to response
- Conduct forensics and investigations for root cause analysis
- Supports NERC CIPv5 compliance
Forensics Hands-on practice
Practice environment

**Attacker**

**AnD Components**

**Existing System**

**Communication Links**

**SCADA Network**

**Switch**

Fan 1

IEC104 PLC 192.168.0.102

Schneider Modicon PLC’s

Modbus PLC 192.168.0.100

Unit ID: 0

Fan 2

**Mirror\Tapping port**

Ethernet\Serial

**AnD Blackbox**

192.168.0.10

ECC

**AnD\SOC Operator**

HMI 192.168.0.111

**AnD management server (FMS)**

192.168.0.20

192.168.0.50

**NMS Server**

Syslog \ SNMP

**SIEM**

**Attendees network**

Switch

SCADASHield_AP

DHCP Server

Management UI IP: 192.168.0.50

Username: admin

Password: Aa123456 (Also for WiFi)

© 2015 by CYBERBIT │ CYBERBIT Proprietary
The “Range”

Simulated Network

Blue Team

Traffic Generator

Threat Generator

Red Team (optional)

Instructors
OT Module

- Physical OT hardware.
- Integration with the simulated IT and SCADA environment.
- Variety of SCADA scenarios and threat vectors
Modbus TCP overview
Modbus and Modbus TCP

- Client/Server communications protocol designed by Modicon in 1979 for use with its PLCs.
- Originally a serial protocol very similar to RS-485, it now has a TCP variant.
- All MODBUS/TCP Packets are sent via TCP to registered port 502.
- The MODBUS messaging services (Client / Server Model) are used for real time information exchange:
  - between two device applications.
  - between device application and other device.
  - between HMI/SCADA applications and devices.
  - between a PC and a device program providing on line services.
Modbus and Modbus TCP

- The MODBUS messaging service provides a Client/Server communication between devices connected on an Ethernet TCP/IP network.
- This client / server model is based on four type of messages:
  - A MODBUS Request is the message sent on the network by the Client to initiate a transaction.
  - A MODBUS Indication is the Request message received on the Server side.
  - A MODBUS Response is the Response message sent by the Server.
  - A MODBUS Confirmation is the Response Message received on the Client side.
The MBAP Header contains the following fields:

- **Transaction Identifier** - It is used for transaction pairing, the MODBUS server copies in the response the transaction identifier of the request.
- **Protocol Identifier** – It is used for intra-system multiplexing. The MODBUS protocol is identified by the value 0.
- **Length** - The length field is a byte count of the following fields, including the Unit Identifier and data fields.
- **Unit Identifier** – This field is used for intra-system routing purpose. It is typically used to communicate to a MODBUS+ or a MODBUS serial line slave through a gateway between an Ethernet TCP-IP network and a MODBUS serial line. This field is set by the MODBUS Client in the request and must be returned with the same value in the response by the server.
# MODBUS Function Codes

<table>
<thead>
<tr>
<th>Function code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Read Coils</td>
</tr>
<tr>
<td>2</td>
<td>Read Discrete Inputs</td>
</tr>
<tr>
<td>3</td>
<td>Read Multiple Registers</td>
</tr>
<tr>
<td>4</td>
<td>Read Input Registers</td>
</tr>
<tr>
<td>5</td>
<td>Write Single Coil</td>
</tr>
<tr>
<td>6</td>
<td>Write Single Register</td>
</tr>
<tr>
<td>7</td>
<td>Read Exception Status</td>
</tr>
<tr>
<td>15</td>
<td>Write Multiple Coils</td>
</tr>
<tr>
<td>16</td>
<td>Write Multiple Registers</td>
</tr>
<tr>
<td>20</td>
<td>Read File Record</td>
</tr>
<tr>
<td>21</td>
<td>Write File Record</td>
</tr>
<tr>
<td>22</td>
<td>Mask Write Register</td>
</tr>
<tr>
<td>23</td>
<td>Read/Write Multiple Registers</td>
</tr>
<tr>
<td>24</td>
<td>Read FIFO</td>
</tr>
</tbody>
</table>
Thank You

Liran.Eldar@cyberbitsolutions.com
Dave.Stanley@cyberbitsolutions.com
Modbus TCP cheat sheet
Modbus TCP help

- Modbus TCP messages have a constant format:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Size</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transaction Identifier</td>
<td>16 BITS</td>
<td>Counts up for message synchronization</td>
</tr>
<tr>
<td>Protocol Identifier</td>
<td>16 BITS</td>
<td>Always 0 (In 'Normal' ModbusTCP)</td>
</tr>
<tr>
<td>Length Field</td>
<td>16 BITS</td>
<td>Length of frame (From this point on, without the first 2 fields)</td>
</tr>
<tr>
<td>Unit Identifier</td>
<td>8 BITS</td>
<td>Station Identifier</td>
</tr>
<tr>
<td>Function code</td>
<td>8 BITS</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td>N BYTES</td>
<td>Optional data for the function code</td>
</tr>
</tbody>
</table>

- The Schneider Modicon with Unity series of PLCs that use Modbus function code 90 (0x5a) to perform administrative commands without authentication

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Size</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transaction Identifier</td>
<td>16 Bits</td>
<td>Counts up for message synchronization</td>
</tr>
<tr>
<td>Protocol Identifier</td>
<td>16 Bits</td>
<td>Always 0 (In 'Normal' ModbusTCP)</td>
</tr>
<tr>
<td>Length Field</td>
<td>16 Bits</td>
<td>Length of frame (From this point on, without the first 2 fields)</td>
</tr>
<tr>
<td>Unit Identifier</td>
<td>8 Bits</td>
<td>Station Identifier</td>
</tr>
<tr>
<td>Function code</td>
<td>8 Bits</td>
<td>= 0x5A</td>
</tr>
<tr>
<td>Session ID</td>
<td>8 Bits</td>
<td>Current session identifier (The session ID is determined on communication initialization with the PLC)</td>
</tr>
<tr>
<td>Operation Code</td>
<td>8 Bits</td>
<td>Command opcode</td>
</tr>
<tr>
<td>Data</td>
<td>N Bits</td>
<td>Optional data for the command</td>
</tr>
</tbody>
</table>
Schneider's Modbus TCP protocol dictates an ACK telemetry for each command. A successful ACK is a Modbus TCP packet with Operation Code = 0xFE, and a NACK is a Modbus TCP packet with Operation Code = 0xFD.

<table>
<thead>
<tr>
<th>Command Opcode</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x10</td>
<td>New session. The session identifier should be set by the sender. The PLC should respond with an ACK to signal a session successfully opened.</td>
</tr>
<tr>
<td>0x11</td>
<td>Close Session. The session identifier that is set in this command packet will be the session that is closed.</td>
</tr>
<tr>
<td>0x40</td>
<td>Start PLC</td>
</tr>
<tr>
<td>0x41</td>
<td>Stop PLC</td>
</tr>
</tbody>
</table>
Modbus TCP help

- **PLC Stop\Start**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 00 00 00 00 06 ff 5a 1b 41 ff 00</td>
<td>00 00 00 00 00 04 ff 5a 1b fe</td>
</tr>
</tbody>
</table>

  Command: 00 00 00 00 00 06 ff 5a 1b 40 ff 00

  Response: 00 00 00 00 00 04 ff 5a 1b fe

- **Deep inspection**

  | 00 00 00 00 06 ff 5a 1b 40 FF 00 |
  | Count 0 | 00 | 00 | 00 | 06 | ff | 5a | 1b | 40 | FF | 00 |
  | Prot. ID 0 | Length 6 | Unit ID | Function Code | Ses. ID | Start PLC |