Research Topics

Chris Hankin
Research Institute in Trustworthy Industrial Control Systems

£2.4M programme, 5 coordinated projects.
Phase 1 (Directorship) awarded 01/01/14, Chris Hankin, Imperial College London.
Phase 2 awarded 01/10/14.

MUMBA: Multifaceted metrics for ICS business risk analysis
CAPRICA: Converged approach towards resilient industrial control systems and cyber assurance
CEDRICS: Communicating and evaluating cyber risk and dependencies in ICS
SCEPTICS: A systematic evaluation process for threats to ICS (incl. national grid and rail networks)
RITICS: Novel, effective and efficient interventions
Key Questions / Challenges

Do we understand the harm threats pose to our ICS systems and business?

Can we confidently articulate these threats as business risk?

What could be novel effective and efficient interventions?
Impact: Policy
Impact: Testbeds
The SCEPTICS project aims to raise the awareness of ICS owners to the vulnerabilities within their ICS, by providing a toolkit of analysis techniques they can use to perform their own risk assessments and identify threats.

**SCEPTICS is:**

- Developing processes for system scoping and analysis from an industrial perspective
- Establishing appropriate methods for identification of harm threats and vulnerabilities
- Documenting and packaging the tools and processes for use by industry stakeholders on their own networks


Last accessed 2nd December 2014.
TRAKS: Universal Key Management for ERTMS

- Developed to simplify distribution of message authentication keys to vehicles & RBCs under ERTMS
- NCSC and RSSB engagement
- Provides mechanisms for secure:
  - Key generation (primarily RBC / vehicle but other pairings equally valid e.g. Eurobalise / vehicle)
  - Key distribution
  - Lifecycle management
- Backwards-compatible with existing key management in ERTMS / EuroRadio
- Utilises post-quantum secure pseudo-random functions
- Also suitable for prevention of “man-in-the-middle” with PLCs in environments where communication protocol is unprotected (e.g. MODBUS/PROFIBUS)
CEDRICS
Communicating and evaluating risk and dependencies

Stochastic models of systems and adversaries

Decision analysis and communication based on Claims, Arguments, Evidence
Methodology for structured security-informed cases

Developed a claims, arguments and evidence (CAE) framework for creating rigorous, structured argumentation for complex engineering systems:

- Cybersecurity explicitly addressed
- Increases the depth and range of analysis
- Addresses the trustworthiness of the evidence
- Helps to communicate cyber risks to different stakeholders.
Mumba
Developing Effective Metrics for Cyber-Risk Decision Making
The Outcomes

Case Studies
Latent Design Conditions

Fieldwork
Grey Area and Limbo Design

Game play
Decision Patterns and Risk Thinking

Testbed
Technological Factors
Practical Challenges in Security Decision Making
“Converged Approach: Control system operates in coordination with network defence, situational monitoring and novel control strategies, to ensure resilient operation of ICS and rapid recovery in the face of an attack.”
Project Outputs and Key Achievements

**Secure SCADA Protocol Gateway (SSPG)**
- First published security implementation of IEC61850-90-5
- Demonstrated successfully with multiple commercial PMUs
- Demonstrated in real-time control loop

**Distributed SCADA-Specific Intrusion Detection Prevention System (SCADA-IDS/IPS)**
- First published distributed SCADA IDS/IPS with decentralised threat monitoring and analytics
- Active security monitoring for legacy devices and communication protocols in substations
- Targeting defence in depth complementing boundary protection of SSPG

- Worldwide only open source PMU
- Better performance, latest protocols, more secure than commercial PMUs
Key Facts about RITICS2

• Research Institute in Trustworthy Inter-connected Cyber-physical Systems
• 16 university partners
• 22 organisations involved in RITICS Council
• Links with NCSC Community of Interest in ICS
• Links to PETRAS IoT Hub
• Inter- and multi-disciplinary focus
Programme

• NIS Directive – baseline, barriers, impact
• Safety and Security
• Autonomous Systems
• Incident Response and Forensics
• Cyber Controls
• Interconnected Systems
• Supply Chain
Topics

• Network Intrusion Detection and Machine Learning

• Diversity as Defence

• Security Metrics
Network Intrusion Detection and Machine Learning
NIDS – Anomaly Detection (Cheng, Li and Chana)

Package Level Detection by *Bloom Filter*
- Construct a *signature database* by observing regular communication patterns.
- Incorporate the signature database into the *bloom filter detector*.
- *Detect* anomalous data packages *at package-content level*.

Time-series Level Detection by *Long Short Term Memory (LSTM)*
- Address *temporal dependence* between consecutive packages
- *Learn the most likely* package signatures from seen packages by *LSTM*.
- *Further classification* of packages *at time-series level*.

Evaluation by Public ICS Database and Comparison
- Apply to a public *ICS dataset* created from a SCADA system for a gas pipeline.
- Significantly outperform other existing approach and produce *state-of-the-art* results.
Mississippi ICS Attack Dataset

https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets

- A lab-scale testbed of a gas pipeline SCADA
- Deep inspection of Modbus data log
- **214,580** normal packets + **60,048** attack packets
- **20** unique features in ARFF Format.
- **7** common types of attacks.

Data Collection Model *(Turnipseed 2015)*

A gas pipeline SCADA *(Turnipseed 2015)*
**Mississippi ICS Attack Dataset**

**ARFF Data Format**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Network</td>
</tr>
<tr>
<td>function</td>
<td>Command Payload</td>
</tr>
<tr>
<td>length</td>
<td>Network</td>
</tr>
<tr>
<td>setpoint</td>
<td>Command Payload</td>
</tr>
<tr>
<td>gain</td>
<td>Command Payload</td>
</tr>
<tr>
<td>reset rate</td>
<td>Command Payload</td>
</tr>
<tr>
<td>deadband</td>
<td>Command Payload</td>
</tr>
<tr>
<td>cycle time</td>
<td>Command Payload</td>
</tr>
<tr>
<td>rate</td>
<td>Command Payload</td>
</tr>
<tr>
<td>system mode</td>
<td>Command Payload</td>
</tr>
<tr>
<td>control scheme</td>
<td>Command Payload</td>
</tr>
<tr>
<td>pump</td>
<td>Command Payload</td>
</tr>
<tr>
<td>solenoid</td>
<td>Command Payload</td>
</tr>
<tr>
<td>pressure measurement</td>
<td>Response Payload</td>
</tr>
<tr>
<td>crc rate</td>
<td>Network</td>
</tr>
<tr>
<td>command response</td>
<td>Network</td>
</tr>
<tr>
<td>time</td>
<td>Network</td>
</tr>
<tr>
<td>binary attack</td>
<td>Label</td>
</tr>
<tr>
<td>categorized attack</td>
<td>Label</td>
</tr>
<tr>
<td>specific attack</td>
<td>Label</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of Attacks</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Normal(0)</td>
</tr>
<tr>
<td>Naïve Malicious Response Injection</td>
<td>NMRI(1)</td>
</tr>
<tr>
<td>Complex Malicious Response Injection</td>
<td>CMRI(2)</td>
</tr>
<tr>
<td>Malicious State Command Injection</td>
<td>MSCI(3)</td>
</tr>
<tr>
<td>Malicious Parameter Command Injection</td>
<td>MPCI(4)</td>
</tr>
<tr>
<td>Malicious Function Code Injection</td>
<td>MFCI(5)</td>
</tr>
<tr>
<td>Denial of Service</td>
<td>DOS(6)</td>
</tr>
<tr>
<td>Reconnaissance</td>
<td>Recon(7)</td>
</tr>
</tbody>
</table>
Package Level Detection by Bloom Filters

Signature database

- observe a large normal time-series dataset.
- A sequence of data packets:
  \[ X = \{ x^{(1)}, x^{(2)}, \ldots, x^{(n)} \} \]
- A single packet with \( m \) features:
  \[ x^{(t)} = \{ x_1^{(t)}, x_2^{(t)}, \ldots, x_m^{(t)} \} \]
- A signature of a packet:
  \[ s(x^{(t)}) = g(c_1^{(t)}, c_2^{(t)}, \ldots, c_o^{(t)}) \]
- Assign a unique value to each different combination of original features.

Feature Discretization

- Find optimal granularity of discretization (too coarse \( \rightarrow \) high FN; too fine \( \rightarrow \) high FP)

\[ \arg \max \sum_{i=1}^{l} w_i n_i, \quad \text{err}_v < \theta \]

\( n_1, \ldots, n_l \) – the number of discretized values; \( \text{err}_v \) – validation error/FP rate
\( w_1, \ldots, w_l \) – relative importance of a feature; \( \theta \) – acceptable FP rate
- Find the finest-grained discretization whose FP rate is below the acceptable FP.
Package Level Detection by Bloom Filters

Bloom-filter (BF) Anomaly Detector

- BF is a light-weight data structure; test if an element is a member of a set.
- Insert all the normal packet signatures into the BF detector
- Hash functions map each element to the corresponding positions of a bit vector.
- Lookup an element by hashing it by the same functions, and check:

\[ F_p(x^{(t)}) = \begin{cases} 1 & \text{if } s(x^{(t)}) \notin \mathcal{B} \\ 0 & \text{otherwise} \end{cases} \]

\[ x^{(t)} \text{ is classified as anomaly if } F_p(x^{(t)}) = 1, \]

- otherwise the package passed our package level anomaly detector.
Time-series Level Detection by LSTM

Long Short-Term Memory (LSTM)

- Advanced anomalies can only be detected by observing preceding packets.
- Recurrent neutral network (RNN) specialise in sequence learning to predict time series.
- Memorize/back-propagate through time for training.
- Identify anomalies with long time lags in between.

A Memory Cell in LSTM

Time Series Learning of LSTM
Time-series Level Detection by LSTM

Stacked LSTM Anomaly Detector

- Store the normal signature database into the LSTM detector
- Input previous network packets (in discretized representation)
- Output the predicted probability of each signature of next packet
  \[
  \Pr(s_i \mid c^{(t-1)}, c^{(t-2)}, \ldots) = \frac{e^{z_i}}{\sum_{k=1}^{|S|} e^{z_k}} \quad \forall i \in \{1, 2, \ldots, |S|\}
  \]
  \[
  \sum_{i=1}^{|S|} \Pr(s_i \mid c^{(t-1)}, c^{(t-2)}, \ldots) = 1.
  \]
- The top $k$th most probable signatures are used to classify:
  \[
  F_t(x^{(t)} \mid c^{(t-1)}, c^{(t-2)}, \ldots) = \begin{cases} 
  1 & \text{if } s(x^{(t)}) \notin S^{(k)} \\
  0 & \text{otherwise.}
  \end{cases}
  \]
- Optimal choice of $k$; similar as granularity of discretization.
- Add probabilistic noise in training to weaken overfitting.
Training & Validation

- Split dataset: 60% training + 20% validation + 20% testing
- Discretization of continuous features by k-means and the validation error.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Discretization method</th>
<th>Value No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>time interval</td>
<td>Kmeans clustering</td>
<td>2+1</td>
</tr>
<tr>
<td>crc rate</td>
<td>Kmeans clustering</td>
<td>2+1</td>
</tr>
<tr>
<td>pressure measurement</td>
<td>Even interval partition</td>
<td>20+1</td>
</tr>
<tr>
<td>setpoint</td>
<td>Even interval partition</td>
<td>10+1</td>
</tr>
<tr>
<td>PID parameters</td>
<td>Kmeans clustering</td>
<td>32+1</td>
</tr>
</tbody>
</table>

- A stacked two layer LSTM; each layer has 256 memory units; output 613 signatures.
- Train the LSTM with/without noise for 50 training epochs.
- Error ratio converges quickly to 0.
- Similar top-k (after k>3) error for models with/without noise.
• Precision is the fraction of positive elements that are true positives.

• Recall: is the fraction of relevant (false negative) elements that are classified as true positives.

• Accuracy is the fraction of the whole sample that is correctly classified as either a true positive or a true negative.
... F1

• F-measure combines precision and recall. The most common measure is the F1-measure which is computed as:
  \[
  \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}
  \]

• which can be interpreted as a weighted average.
Experiments – Comparison

Comparison with Other Anomaly Detection Methods

- Evaluation metrics – *Precision, Recall, Accuracy and F-score*.
- Compare with other anomaly detection methods.
- Detected ratio (recall) for seven types of attacks.

<table>
<thead>
<tr>
<th>Attack Type</th>
<th>Model</th>
<th>Detected Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMRI</td>
<td>Our model</td>
<td>0.88</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.45</td>
</tr>
<tr>
<td>CMRI</td>
<td>Our model</td>
<td>0.67</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>0.53</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>0.53</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.33</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.19</td>
</tr>
<tr>
<td>MSCI</td>
<td>Our model</td>
<td>0.66</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>0.18</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.66</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.62</td>
</tr>
<tr>
<td>MPCI</td>
<td>Our model</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>0.26</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.64</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.66</td>
</tr>
<tr>
<td>MFCI</td>
<td>Our model</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.32</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.54</td>
</tr>
<tr>
<td>DOS</td>
<td>Our model</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>0.40</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>0.12</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.15</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.58</td>
</tr>
<tr>
<td>Recon.</td>
<td>Our model</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>BF</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>SVDD</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>0.12</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td>PCA-SVD</td>
<td>0.54</td>
</tr>
</tbody>
</table>
Evasion Attacks

Originally discovered by researchers when trying to better interpret neural networks.

Stealthy Attacks (Cheng, Li and Chana)

Objectives

- A framework for conducting stealthy attacks with minimal knowledge of the target ICS

- Better understanding of the limitations of current detection mechanisms, and the real threat posed by stealthy attacks to ICS.

Main Contributions

- Demonstrated attacks can be automatically achieved by intercepting the sensor/control signals for a period of time using a particularly designed real-time learning method.

- Used adversarial training technique – Wasserstein GAN to generate false data that can successfully bypass the IDS and still deliver specific attack goals.

- Two real-world datasets are used to validate the effectiveness of our framework.
  - A gas pipeline SCADA system.
  - Secure Water treatment testbed from iTrust@SUTD.
Stealthy Attacks against ICS

- Intercept the expected behaviours of the system via compromised channels.
- *Injected malicious sensor reading* at each time step to achieve certain attack goals.
- Attackers attempt to hide their manipulation; *remain undetected by ADS.*
Anomaly Detection Mechanism for Securing Industrial Processes

- Protect from physical faults and cyber attacks by monitoring the sensor reading & control commands.
- Rely on a Predictive Model which predicts the next sensor measurements based on previous signals.

\[
\text{Observed: } \mathbf{x}(t) = \{y(t), u(t)\} = \{y_1(t), y_2(t), \ldots, y_m(t), u_1(t), u_2(t), \ldots, u_n(t)\}
\]

\[
\text{Predicted: } \hat{\mathbf{y}}(t) = \{\hat{y}_1(t), \hat{y}_2(t), \ldots, \hat{y}_m(t)\}
\]

Existing Predictive Models

- Auto-Regressive (AR) model
  - Fit a linear regression model for each reading based on its \(p\) previous readings.

- Linear Dynamic State-space (LDS) model
  - A vector \(w\) for physical states
  - Matrices for system dynamics.
  - Noise vectors.
  - Known as the State Estimator

- Long Short-Term Memory (LSTM) [HASE’17]
  - State-of-the-art prediction accuracy
  - hidden vector computed iteratively
  - Weight matrix and bias vector
Anomaly Detection for Industrial Processes

Detection Methods
- An anomalous signal is detected when the residual error between the predicted and observed:

\[ |\hat{y}_i^{(t)} - y_i^{(t)}| > \tau_i \quad \forall i \in \{1, 2, \ldots, m\} \]

\[ ||\hat{y}^{(t)} - y^{(t)}|| > \tau \]

- Use the history of residual errors to detect collective anomalies, by Cumulative Sum (CUSUM) based on an accumulated statistic:

\[ H(t) = \max(0, H^{(t-1)} + r(t) - \mu - \omega) > \tau \]

Formally Define Stealthy Attacks
- A general anomaly detector as a function:

\[ F(y^{(t)} | X^{t-1}) = \begin{cases} 1 & \text{if } y^{(t)} \text{ triggers an alarm} \\ 0 & \text{otherwise} \end{cases} \]

- Consider the ADS is a black-box for attackers.
- A certain number of PLC-sensor and PLC-actuator channels \( y_c^{(t)} \) are compromised.
- Define a set of stealthy attack goals -- inject malicious values that are deviant with the true values

\[ \tilde{y}_g^{(t)} \in \mathcal{G} v_g \quad \text{where } \mathcal{G} = \{>, <, \leq, \geq, =\} \land \tilde{y}_g^{(t)} \in \tilde{y}_c^{(t)} \]
Deep Learning Framework for Stealthy Attacks

**WGAN-based Training Model for Stealthy Attacks**

- Reconnaissance phase + Attacking phase

- A Wasserstein Generative Adversarial Net (W-GAN) is constructed for training

- W-GAN: an iterative game between two players (Generator and Discriminator)
  - **Generator**: generates data with the same distribution as the training data
  - **Discriminator**: distinguish generated data from training data
  - At each step, either G or D is trained to optimize its objective function.
  - Until Discriminator fails...

---

![Diagram of deep learning framework for stealthy attacks]

- **True** data up to t-1
- **False** data up to t-1

- **G** (Generator)
- **D** (Discriminator)

- **True / False** data up to t-1

- **Anomalous?**
Deep Learning Framework for Stealthy Attacks

**Malicious Measurement Generator**
- Generate malicious data achieving attack goals.
- As a sequence learning problem, solved by LSTM-FNN.
- Two sliding windows:
  - \( S^t_e = \{ x^{(t-l)}_e, x^{(t-l+1)}_e, \ldots x^{(t-1)}_e \} \)
  - \( \tilde{S}^t_e = \{ \tilde{x}^{(t-l)}_e, \tilde{x}^{(t-l+1)}_e, \ldots \tilde{x}^{(t-1)}_e \} \)
- Generator as an overall function:
  \[
  \tilde{y}^{(t)}_c = G(S^t_e, \tilde{S}^t_e; \Theta_G)
  \]
- Minimize the chance being detected and deliver the goal:
  \[
  \arg\min_{\Theta_G} \frac{1}{|T|} \sum_{t \in T} \mathcal{F}(G(S^t_e, \tilde{S}^t_e; \Theta_G) \mid X^{t-1})
  \]
  subject to \( \tilde{y}^{(t)}_g \ominus v_g \) \( \forall g \in G, t \in T \)

**Substitute Anomaly Detector**
- Approximate the black-box anomaly detector.
- Input the window of previous data and the current data; Output the classification.
  \( \eta = D(\tilde{y}^{(t)}_c, \tilde{S}^t_e; \Theta_D) \)
- Detector as an overall function:
  \[
  \arg\min_{\Theta_D} \frac{1}{|T_1|} \sum_{t \in T_1} D(y^{(t)}_c, S^t_e; \Theta_D) - \frac{1}{|T_2|} \sum_{t \in T_2} D(\tilde{y}^{(t)}_c, \tilde{S}^t_e; \Theta_D)
  \]
- Larger values for malicious; smaller values for true data.

**GAN: Generator + Detector**
- Generating malicious data which makes the detector output smallest possible values whilst achieving the goals.
GAS Pipeline Case Study

**Mississippi Dataset of a gas pipeline SCADA**
- Controls the air pressure in a pipeline; contains a PLC, a sensor and several actuators.
- Pressure measurements at every 2s, 68,803 time series signals are collected.

**Experiment Setup**
- Baseline Anomaly Detector uses LSTM model.
- Four Attack Scenarios: being 4 or 8 units smaller than real values; different compromised channels

---

<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Setpoint</strong></td>
<td>The pressure set point</td>
</tr>
<tr>
<td><strong>Gain</strong></td>
<td>PID gain</td>
</tr>
<tr>
<td><strong>Reset rate</strong></td>
<td>PID reset rate</td>
</tr>
<tr>
<td><strong>Deadband</strong></td>
<td>PID dead band</td>
</tr>
<tr>
<td><strong>Cycle time</strong></td>
<td>PID cycle time</td>
</tr>
<tr>
<td><strong>Rate</strong></td>
<td>PID rate</td>
</tr>
<tr>
<td><strong>System mode</strong></td>
<td>Automatic (2), manual (1) or off (0)</td>
</tr>
<tr>
<td><strong>Control scheme</strong></td>
<td>Pump (0) or valve (1)</td>
</tr>
<tr>
<td><strong>Pump</strong></td>
<td>Open(1) or off (0) – for manual mode</td>
</tr>
<tr>
<td><strong>Valve</strong></td>
<td>Open(1) or off (0) – for manual mode</td>
</tr>
<tr>
<td><strong>Pressure measurement</strong></td>
<td>Pressure measurement</td>
</tr>
</tbody>
</table>
Results and Evaluation

- Generated malicious measurements successfully capture the trend of the real trace.

- Generated malicious measurements mostly can bypass the anomaly detector
  - Most malicious values have similar or less residual error than the true values.
  - Outliers are caused by HMI human input at manual mode.

- Ratio of attack goal achieved the detection ratio of malicious measurements
  - Ignored the outliers (residual error > 0.05)
  - Less detection ratio for attack scenario 3 and 4.
  - Only compromising PLC-sensor channel still generates high-quality attacks.

<table>
<thead>
<tr>
<th>Attack Scenario</th>
<th>Ratio of goal achieved</th>
<th>Detected ratio by residual error</th>
<th>Detected ratio by CUSUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>88.1%</td>
<td>2.6%</td>
<td>0.2%</td>
</tr>
<tr>
<td>2</td>
<td>86.0%</td>
<td>2.4%</td>
<td>0.1%</td>
</tr>
<tr>
<td>3</td>
<td>85.9%</td>
<td>1.1%</td>
<td>0.01%</td>
</tr>
<tr>
<td>4</td>
<td>90.5%</td>
<td>1.2%</td>
<td>0.01%</td>
</tr>
</tbody>
</table>
UK/Singapore cyber security research

Security by Design for Interconnected Critical Infrastructures
Water Treatment System Case Study

Experiment Setup
- A water treatment plant (SWaT from iTrust@SUTD) maintains the water quality within acceptable limits.
- 51 sensors extracted every second, in total 496,800 signals for normal operation are collected.

<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIT201</td>
<td>Measures NaCl level</td>
</tr>
<tr>
<td>AIT202</td>
<td>Measures HCl level</td>
</tr>
<tr>
<td>AIT203</td>
<td>Measures NaOCl level</td>
</tr>
<tr>
<td>FIT201</td>
<td>Flow transmitter for dosing pumps</td>
</tr>
<tr>
<td>P101</td>
<td>Raw water tank pump state</td>
</tr>
<tr>
<td>MV201</td>
<td>Motorized vale state</td>
</tr>
<tr>
<td>P201</td>
<td>NaCl dosing pump state</td>
</tr>
<tr>
<td>P203</td>
<td>HCl dosing pump state</td>
</tr>
<tr>
<td>P205</td>
<td>NaOCl dosing pump state</td>
</tr>
</tbody>
</table>

- Focus on generating malicious HCl and NaOCl measurements, still within normal range.

\[
\tilde{y}_{g_1}^{(t)} \geq \min(y_{g_1}^{(t)} + 0.1, 1) \quad \tilde{y}_{g_2}^{(t)} \leq \max(\tilde{y}_{g_2}^{(t)} - 0.1, 0)
\]

Simulation and Evaluation
- A successful attack -- either the HCl (>0.99) or the NaOCl (<0.01) dosing pump is turned on unexpectedly by the injected malicious measurements + bypassed the detector.

<table>
<thead>
<tr>
<th>Compromised Channels</th>
<th>Successful Ratio by residual error</th>
<th>Successful Ratio by CUSUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only PLC-AIT202, PLC-AIT203</td>
<td>90.1%</td>
<td>93.8%</td>
</tr>
<tr>
<td>all channels</td>
<td>92.4%</td>
<td>94.6%</td>
</tr>
</tbody>
</table>
Future work

- Proposed a novel GAN based stealthy attack framework, required a much lower \textit{a-priori} knowledge of the targeted ICS.

- Developed a real-time adversarial learning method allowing attackers to inject malicious data to automatically conduct stealthy attacks without being detected.

- Indicated that with recent development in deep learning, the widely recognized effectiveness of existing anomaly detection techniques might be overestimated.

- More advanced anomaly detection frameworks are needed.
Diversity as Defence
Motivation

Software Diversity -- an Effective Defense Strategy

- Software mono-culture promotes and accelerates the spread of malware.
- Diversification can mitigate the infection of malware between similar products and reduce the likelihood of repeating application of single exploits.
- More essential when combating zero-day exploits.
Motivation

Software Diversity -- an Effective Defense Strategy

- Software mono-culture promotes and accelerates the spread of malware.
- Diversification can mitigate the infection of malware between similar products and reduce the likelihood of repeating application of single exploits.
- More essential when combating zero-day exploits.

Existing work on Diversity-inspired defence

- From the software development: \emph{n-version programming, code randomization, etc.}.
- From the perspective of security management:
  - Optimal product assignment by distributed colouring algorithm [CCS’04]
  - Diversifying routing nodes in a network [TDSC’15]
  - Security metrics to evaluate network diversify and resilience. [ESORICS’10, TIFS’16]
Motivation

Software Diversity -- an Effective Defense Strategy

- Software mono-culture promotes and accelerates the spread of malware.
- Diversification can mitigate the infection of malware between similar products and reduce the likelihood of repeating application of single exploits.
- More essential when combating zero-day exploits.

Existing work on Diversity-inspired defence

- From the software development: n-version programming, code randomization, etc..
- From the perspective of security management:
  - Optimal product assignment by distributed colouring algorithm [CCS’04]
  - Diversifying routing nodes in a network [TDSC’15]
  - Security metrics to evaluate network diversify and resilience. [ESORICS’10, TIFS’16]

Assumption -- Products share no vulnerabilities between each other.
Assumption -- only one vulnerable product/service running at each host.
**Motivation**

**Software Diversity -- an Effective Defense Strategy**

- Software mono-culture promotes and accelerates the spread of malware.
- Diversification can mitigate the infection of malware between similar products and reduce the likelihood of repeating application of single exploits.
- More essential when combating zero-day exploits.

**Existing work on Diversity-inspired defence**

- From the software development: *n-version programming, code randomization, etc.*.
- From the perspective of security management:
  - Optimal product assignment by distributed colouring algorithm [CCS’04]
  - Diversifying routing nodes in a network [TDSC’15]
  - Security metrics to evaluate network diversify and resilience. [ESORICS’10, TIFS’16]

- **Assumption** -- Products share no vulnerabilities between each other.
- **Assumption** -- only one vulnerable product/service running at each host.
Motivation

Software Diversity -- an Effective Defense Strategy

- Software mono-culture promotes and accelerates the spread of malware.
- Diversification can mitigate the infection of malware between similar products and reduce the likelihood of repeating application of single exploits.
- More essential when combating zero-day exploits.

Existing work on Diversity-inspired defence

- From the software development: n-version programming, code randomization, etc..
- From the perspective of security management:
  - Optimal product assignment by distributed colouring algorithm [CCS’04]
  - Diversifying routing nodes in a network [TDSC’15]
  - Security metrics to evaluate network diversify and resilience. [ESORICS’10, TIFS’16]
- Assumption -- Products share no vulnerabilities between each other.
- Assumption -- only one vulnerable product/service running at each host.
Main Contributions

Objectives

- An accurate model to determine the infection of potential exploits across a network.
- An efficient way to find product assignments to minimise the prevalence of 0day exploits.
Main Contributions

Objectives

- An accurate model to determine the infection of potential exploits across a network.
- An efficient way to find product assignments to minimise the prevalence of 0day exploits.

Main Contributions

- Modelled various services/products at each host and exposed attack vectors.

- Proposed the metric *vulnerability similarity of products*; Statistical study of CVE/NVD.

- Formally model the multi-labelling network by a discrete *Markov Random Field (MRF)*; Optimised by the *sequential tree-reweighted message passing (TRW-S)* algorithm.

- A case study inspired by *Stuxnet Propagation* to
  - Find the optimal assignment against the collaboration of multiple 0day exploits
  - Evaluate the optimal result in a *NetLogo simulation* in terms of MTTC.

- Scalability analysis of our optimisation method against
  - Large-scale networks with *up to 10,000 hosts*.
  - High-density networks with *up to 50 degrees (# edges) per host*.
  - High-complexity networks with *up to 30 products/services per host*.
  - Most heavy cases converged from a couple of seconds to ~3 minutes.
Similarity Metrics

Similarity of Products Vulnerability based on CVE/NVD

- Firstly define the similarity between a pair of products;
  - capture statistically how similar the vulnerabilities found on two products are.
  - likelihood of being compromised by the same exploit.

**Definition 1** Let $x_i$, $x_j$ be a pair of products, $V_{x_i}$ and $V_{x_j}$ are vulnerabilities of $x_i$ and $x_j$ respectively. The vulnerability similarity between $x_i$ and $x_j$ can be obtained by the Jaccard similarity coefficient:

$$
sim(x_i, x_j) = \frac{|V_{x_i} \cap V_{x_j}|}{|V_{x_i} \cup V_{x_j}|}
$$
Similarity Metrics

Similarity of Products Vulnerability based on CVE/NVD

- Firstly define the similarity between a pair of products;
  - capture statistically how similar the vulnerabilities found on two products are.
  - likelihood of being compromised by the same exploit.

**Definition 1** Let $x_i, x_j$ be a pair of products, $V_{x_i}$ and $V_{x_j}$ are vulnerabilities of $x_i$ and $x_j$ respectively. The vulnerability similarity between $x_i$ and $x_j$ can be obtained by the Jaccard similarity coefficient:

$$sim(x_i, x_j) = \frac{|V_{x_i} \cap V_{x_j}|}{|V_{x_i} \cup V_{x_j}|}$$

- Common Vulnerability Enumerations (CVE) and National Vulnerability Database (NVD)

<table>
<thead>
<tr>
<th>CVE-ID</th>
<th>CVE-2016-7153</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>The HTTP/2 protocol does not consider the role of the TCP congestion window in providing information about content length, which makes it easier for remote attackers to obtain cleartext data by leveraging a web-browser configuration in which third-party cookies are sent, aka a &quot;HEIST&quot; attack.</td>
</tr>
<tr>
<td>Release Date</td>
<td>September 6th, 2016</td>
</tr>
<tr>
<td>CVSS Severity</td>
<td>Base Score: 5.0 MEDIUM; Vector: AV:N/AC:L/Au:N/C:P/I:N/A:N ; Impact Subscore: 2.9; Exploitability Subscore: 10.0</td>
</tr>
<tr>
<td>CVSS V2 Metrics</td>
<td>Access Vector: Network exploitable; Access Complexity: Low; Authentication: Not required to exploit;</td>
</tr>
</tbody>
</table>
## Similarity Metrics

### Similarity of Products Vulnerability based on CVE/NVD

- 84,229 vulnerabilities in NVD; CPE serves to sort vulnerabilities according to affected products.
- Compare most vulnerable OS products and Web Browsers [CVE Details] from 1999 to 2016.

<table>
<thead>
<tr>
<th></th>
<th>WinXP2</th>
<th>Win7</th>
<th>Win 8.1</th>
<th>Win10</th>
<th>Ubt14.04</th>
<th>Deb8.0</th>
<th>Mac10.5</th>
<th>Suse13.2</th>
<th>Fedora</th>
</tr>
</thead>
<tbody>
<tr>
<td>WinXP2</td>
<td>1.00 (479)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Win7</td>
<td>0.278 (328)</td>
<td>1.00 (1028)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Win8.1</td>
<td>0.009 (10)</td>
<td>0.228 (298)</td>
<td>1.00 (572)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Win10</td>
<td>0 (0)</td>
<td>0.124 (164)</td>
<td>0.697 (421)</td>
<td>1.00 (453)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ubt14.04</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>1.00 (612)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deb8.0</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.208 (195)</td>
<td>1.00 (519)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mac10.5</td>
<td>0 (0)</td>
<td>0.081 (109)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>1.00 (424)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Suse13.2</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.170 (161)</td>
<td>0.112 (102)</td>
<td>0 (0)</td>
<td>1.00 (492)</td>
<td></td>
</tr>
<tr>
<td>Fedora</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.083 (75)</td>
<td>0.049 (41)</td>
<td>0.001 (1)</td>
<td>0.116 (89)</td>
<td>1.00 (367)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>IE8</th>
<th>IE10</th>
<th>Edge</th>
<th>Chrome</th>
<th>Firefox</th>
<th>Safari</th>
<th>SM</th>
<th>Opera</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE8</td>
<td>1.0 (349)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IE10</td>
<td>0.386 (240)</td>
<td>1.0 (513)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edge</td>
<td>0.014 (7)</td>
<td>0.121 (73)</td>
<td>1.0 (194)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chrome</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.001 (2)</td>
<td>1.0 (1661)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.001 (2)</td>
<td>0.005 (15)</td>
<td>1.0 (1502)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Safari</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.002 (2)</td>
<td>0.009 (21)</td>
<td>0.003 (6)</td>
<td>1.0 (766)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SeaMonkey</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.001 (3)</td>
<td>0.450 (683)</td>
<td>0.001 (1)</td>
<td>1.0 (492)</td>
<td></td>
</tr>
<tr>
<td>Opera</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0.003 (1)</td>
<td>0.003 (6)</td>
<td>0.004 (7)</td>
<td>0.004 (4)</td>
<td>1.00 (492)</td>
<td>1.00 (225)</td>
</tr>
</tbody>
</table>
Similarity Metrics

Similarity of Hosts and Infection Models

- Each host runs a set of services: $S_{h_i} = \{s_1, \ldots, s_k\}$, where $S_{h_i} \in 2^S$
- Each service is provided by a set of products: $p(s_j) = \{p_{s_j}^i, \ldots, p_{s_j}^k\}$, where $p_{s_j}^i \in P$
- An assignment of products for a host: $\alpha(h_i, S_{h_i}) = (\alpha'(h_i, s_1), \ldots, \alpha'(h_i, s_k)) = (p_{s_1}^m, \ldots, p_{s_k}^n)$
- Estimate the infection rate by comparing the assigned products (i.e. similarity of hosts).

$$sim(\alpha(h_i, S_{h_i}), \alpha(h_j, S_{h_j})) =$$

$$\begin{cases} \max \{sim(\alpha'(h_i, s_k), \alpha'(h_j, s_k))\}, & \text{for sophisticated attackers} \\ random \{sim(\alpha'(h_i, s_k), \alpha'(h_j, s_k))\}, & \text{for naive attackers} \end{cases}$$

$$r(h_i, h_j) = sim(\alpha(h_i, S_{h_i}), \alpha(h_j, S_{h_j}))$$
Similarity Metrics

Similarity of Hosts and Infection Models

- Each host runs a set of services: $S_{h_i} = \{s_1, \ldots, s_k\}$, where $S_{h_i} \subseteq 2^S$
- Each service is provided by a set of products: $p(s_j) = \{p_{s_j}^1, \ldots, p_{s_j}^k\}$, where $p_{s_j}^i \in P$
- An assignment of products for a host: $\alpha(h_i, S_{h_i}) = (\alpha'(h_i, s_1), \ldots, \alpha'(h_i, s_k)) = (p_{s_1}^m, \ldots, p_{s_k}^n)$
- Estimate the infection rate by comparing the assigned products (i.e. similarity of hosts).

$$sim(\alpha(h_i, S_{h_i}), \alpha(h_j, S_{h_j})) =$$

$$\begin{cases} 
\max_{s_k \in S_{h_i} \cap S_{h_j}} \{sim(\alpha'(h_i, s_k), \alpha'(h_j, s_k))\}, & \text{for sophisticated attackers} \\
\text{random} \left\{ sim(\alpha'(h_i, s_k), \alpha'(h_j, s_k)) \right\}, & \text{for naive attackers}
\end{cases}$$

$$r(h_i, h_j) = sim(\alpha(h_i, S_{h_i}), \alpha(h_j, S_{h_j}))$$

(a) available products

(b) a network

(c) the infection model with assigned products
Optimal Assignment of Diverse Products

Formal Model by Markov Random Field (MRF)

- Each service has various selections of products $\rightarrow$ up to $|P|$ labels.
- Each host provides multiple services $\rightarrow$ up to $|P| \times |S|$ labels.
- Sufficient flexibility and generality.
- Existence of feasible/efficient optimisation solution.
- The optimal diversification problem $\rightarrow$ find an optimal label for each service at each host

Optimisation

- Given an infection model $G := (H, L, \alpha)$, the energy function is given as:

  $$E(G) = \sum_{h_i \in H, s_k \in S_i} \phi(h_i, s_k) + \sum_{(h_i, h_j) \in L} \psi(\alpha(h_i, S_{h_i}), \alpha(h_j, S_{h_j}))$$

  $$\hat{\alpha} = \arg\min_{\alpha} E(G)$$

  $$= \arg\min_{\alpha} \sum_{h_i \in H} \sum_{s_j \in S_{h_i}} P_{r\text{const}} + \sum_{(h_i, h_j) \in L} \sum_{s_k \in S_{h_i} \cap S_{h_j}} \text{sim}(\alpha(h_i, S_{h_i}), \alpha(h_j, S_{h_j}))$$

- Solved by tree-rewighted message passing algorithm (TRW-S) [TPAMI’15].
- Generally guarantees an optimal solution; outperforms others algorithms in heavy tasks.
Case Study– Stuxnet Propagation

(a) A four-zone networked ICS architecture

[Tofino Security’11]

(b) The corresponding network topology

(c) Available products for each host [WinCC Manual]
The solution uses product similarities obtained from CVE/NVD statistically.
Minimizes the infection rate at each edge by choosing most diverse product pairs.
Locally optimal assignment might be discarded for global optimum (e.g. c4 and c2).
Case Study– Stuxnet Propagation

NetLogo Simulation for Evaluation

- NetLogo is an agent-based modelling tool.
- Programmable modelling environment
- Simulate behaviours of systems and natural phenomena over time.

- Two attacker modes.
- # Ticks = MTTC
- Six hosts are seamlessly protected.
- Real-time plot of MTTC
Case Study– Stuxnet Propagation

NetLogo Simulation for Evaluation

- Compare the OPTIMAL assignment with a RANDOM assignment in terms of MTTC.

- An extra experiment with a larger network (100 nodes, ~300 edges and 5 services per host).
Scalability Analysis on Random Networks

Scalability with Randomly generated networks

- Computational time consumed by optimising a set of randomly generated networks.
- Three key parameters: \# hosts, \# degrees (edges per host), \# services per host.

(A) fixed \# degree = 3, \# services = 3
(B) fixed \# hosts = 100, \# services = 3
(C) fixed \# hosts = 100, \# degrees = 30

Mid-range computer: Intel i5 2.8GHz CPU, a 8GB RAM and a nVidia GTX 750

The \# hosts has a major impact. Still converged within 7.342 seconds for 10,000 hosts.
Scalability Analysis on Random Networks

Scalability with Randomly generated networks

- Scalability analysis against high-density, high-complexity and large-scale networks.
- Performs well and converges within about 3 minutes.

Table 1: Computational time (in seconds) for networks of various densities over different # hosts

<table>
<thead>
<tr>
<th># deg.</th>
<th># serv.</th>
<th># hosts</th>
<th>100</th>
<th>200</th>
<th>400</th>
<th>600</th>
<th>800</th>
<th>1000</th>
<th>2000</th>
<th>4000</th>
<th>6000</th>
</tr>
</thead>
<tbody>
<tr>
<td>mid-density</td>
<td>20</td>
<td>15</td>
<td>0.239</td>
<td>0.438</td>
<td>1.099</td>
<td>1.478</td>
<td>1.944</td>
<td>2.784</td>
<td>6.706</td>
<td>16.517</td>
<td>33.392</td>
</tr>
<tr>
<td>high-density</td>
<td>40</td>
<td>25</td>
<td>0.640</td>
<td>1.766</td>
<td>3.553</td>
<td>5.881</td>
<td>8.135</td>
<td>10.999</td>
<td>27.484</td>
<td>82.500</td>
<td>151.110</td>
</tr>
</tbody>
</table>

Table 2: Computational time (in seconds) for various sizes of networks over different # degrees

<table>
<thead>
<tr>
<th># hosts</th>
<th># serv.</th>
<th>#deg.</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
<th>45</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>mid-scale</td>
<td>1000</td>
<td>15</td>
<td>0.759</td>
<td>1.577</td>
<td>1.954</td>
<td>2.693</td>
<td>3.294</td>
<td>4.040</td>
<td>4.652</td>
<td>5.174</td>
<td>5.758</td>
<td>6.309</td>
</tr>
<tr>
<td>large-scale</td>
<td>6000</td>
<td>25</td>
<td>21.239</td>
<td>40.940</td>
<td>59.216</td>
<td>77.583</td>
<td>95.750</td>
<td>117.810</td>
<td>144.470</td>
<td>152.040</td>
<td>167.190</td>
<td>189.710</td>
</tr>
</tbody>
</table>

Table 3: Computational time (in seconds) for various sizes of networks over different # services

<table>
<thead>
<tr>
<th># hosts</th>
<th># deg.</th>
<th># edges</th>
<th>#serv.</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>mid-scale</td>
<td>1000</td>
<td>20</td>
<td>~ 20,000</td>
<td>0.603</td>
<td>1.608</td>
<td>2.709</td>
<td>4.008</td>
<td>5.253</td>
<td>6.974</td>
</tr>
<tr>
<td>large-scale</td>
<td>6000</td>
<td>40</td>
<td>~ 240,000</td>
<td>10.306</td>
<td>27.214</td>
<td>51.587</td>
<td>90.407</td>
<td>134.340</td>
<td>188.050</td>
</tr>
</tbody>
</table>
Conclusion and Future work

- Proposed an efficient way to mitigate zero-day infection over a network by optimally diversifying products deployed on the hosts.

- Introduced the similarity metric to capture how similar the vulnerabilities of two products are. Applied in statistical study on CVE/NVD database.

- Estimated the infection rate of malware between products by the similarity metrics.

- A multi-label model is adopted to capture the spread of multiple zero-day exploits.

- Developed an efficient and scalable optimisation method.

- Other sources/ways to measure the similarity metric.

- Consider interdependency between services and preference over products.
Security Metrics
Assessing Cyber-Physical Security in Industrial Control Systems

Martín Barrère, Chris Hankin, Demetrios Eliades, Nicolas Nicolaou and Thomas Parisini

Institute for Security Science and Technology
Imperial College London, UK

September 2019 / Athens, Greece
Agenda

1. Introduction

2. Base security metric (weighted AND/OR graphs)

3. Extended security metric (AND/OR hypergraphs)

4. Analytical evaluation

5. Case study on water transport networks

6. Conclusion and future work
INTRODUCTION
Introduction

- **Goal:** security metric for ICS networks
- **AND/OR graphs** to model complex interdependencies between cyber-physical components
- Identify **critical ICS nodes**, with **minimal** compromise **cost**, that could disrupt the operation of the system
  - NP-complete problem
  - Multiple overlapping security measures
- Measure security levels, **compare different ICS settings**
BASE SECURITY METRIC

Weighted AND/OR graphs
ICS network model (simple example)

- AND/OR graph with sensors, software agents and actuators
- Adversarial model: an attacker can compromise any network node $n \in V_{AT}$ at a certain cost $\varphi(n)$ with $\varphi : V_{AT} \rightarrow \mathbb{R}_{\geq 0}$
- Compromised node: component unable to operate properly
Least-effort attack strategy (critical nodes)

- **Objective**: set of nodes, with minimal cost (effort) for an attacker, such that if compromised, the system would enter into a non-operational state

- **Problem**: identifying critical nodes in AND/OR graphs => NP-complete
  
  (Desmedt et al. (2004); Jakimoski et al. (2004); Souza et al. (2013))

- **Solution**:
  - Critical nodes: a, c
  - Total cost: 4

---

![Diagram of critical nodes and system components]
MAX-SAT resolution approach

MAX-SAT problem: find a truth assignment that maximises the weight of the satisfied clauses (or minimise the weight of falsified clauses)

1. AND/OR logical transformation:

   \[ f_G(c1) = c1 \land (d \land ((a \land b) \lor (b \land c))) \]

2. Attacker’s objective:

   \[ \neg f_G(c1) = \neg(c1 \land (d \land ((a \land b) \lor (b \land c)))) \]

3. MAX-SAT problem specification:
   - Falsification penalty scores
     
     \[
     \begin{array}{|c|c|c|c|c|}
     \hline
     a & b & c & d & c1 \\
     \hline
     \varphi(a) = 2 & \varphi(b) = 5 & \varphi(c) = 2 & \varphi(d) = 10 & \varphi(c1) = \text{inf} \\
     \hline
     \end{array}
     \]
   - MAX-SAT solution: minimises the penalty induced by falsified weighted variables
Visualisation system (META4ICS)

- **META4ICS**: Metric Analyser for Industrial Control Systems
  Available at: [https://github.com/mbarrere/meta4ics](https://github.com/mbarrere/meta4ics)
SECURITY METRIC (EXTENSION)

Problem generalisation with AND/OR hypergraphs
Multiple overlapping security measures

- Set of security measure instances: $S = \{s_1, s_2, \ldots\}$
- Cost function (attacker’s effort): $\psi : S \rightarrow \mathbb{R}_{\geq 0}$

<table>
<thead>
<tr>
<th>Measure instance</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_3$</th>
<th>$s_4$</th>
<th>$s_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measure type</td>
<td>$M_2$</td>
<td>$M_3$</td>
<td>$M_1$</td>
<td>$M_4$</td>
<td>$M_5$</td>
</tr>
<tr>
<td>Attacker’s cost</td>
<td>$3$</td>
<td>$7$</td>
<td>$2$</td>
<td>$12$</td>
<td>$\text{inf}$</td>
</tr>
<tr>
<td>Protection range</td>
<td>${a, c}$</td>
<td>${b}$</td>
<td>${a}$</td>
<td>${d}$</td>
<td>${c_1}$</td>
</tr>
</tbody>
</table>
Extended security metric (formulation)

\[ \mu(G, t) = \arg\min_{X \subseteq V_{AT}} \left( \sum_{x_i \in X} \varphi(x_i) + \sum_{s_j \in S(X)} \psi(s_j) \right) \]

\[ \text{s.t. } wcc(\sigma(G, X)) \geq 2 \ \forall \ X = \{t\} \]

- Inputs: AND/OR graph, target node
- Solution node set: \( X \subseteq V_{AT} \)
- Functions:
  - \( S(X) \) => set of measure instances \( \{s_i, \ldots, s_j\} \) protecting \( X \)
  - \( \sigma(G, X) \) => removes nodes in \( X \) from \( G \)
  - \( wcc(G) \) => weakly connected components
AND/OR hypergraph-based approach

- **Hypergraphs**: generalisation of standard graphs where graph edges (hyperedges) can connect any number of vertices

\[
f_H(e_5) = e_5 \land e_4 \land ((e_1 \land e_3) \lor (e_3 \land e_2))
\]

<table>
<thead>
<tr>
<th>(e_1)</th>
<th>(e_2)</th>
<th>(e_3)</th>
<th>(e_4)</th>
<th>(e_5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>{a, s1, s3}</td>
<td>{c, s1}</td>
<td>{b, s2}</td>
<td>{d, s4}</td>
<td>{c1, s5}</td>
</tr>
</tbody>
</table>
AND/OR hypergraph resolution

\[ h_G(c1) = (c1 \lor s5) \land (d \lor s4) \land ((a \lor s1 \lor s3) \land (b \lor s2)) \lor ((b \lor s2) \land (c \lor s1))) \]

- Attacker’s compromise costs (used as MAX-SAT penalty values):
  - Atomic nodes: \( \varphi(n) = 1, \forall n \in V_{AT} \)
  - Measure instances:

<table>
<thead>
<tr>
<th>Measure instance</th>
<th>s1</th>
<th>s2</th>
<th>s3</th>
<th>s4</th>
<th>s5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attacker’s cost  ( \psi(s_i) )</td>
<td>3</td>
<td>7</td>
<td>2</td>
<td>12</td>
<td>inf</td>
</tr>
</tbody>
</table>

- First attempt: falsify \( b \lor s2 \)
  - Cost: \( 1 + 7 = 8 \)
- Second attempt: falsify \( a \lor s1 \lor s3 \) and \( c \lor s1 \)
  - Cost for set \( \{a, s1, s3, c\} \): \( 1 + 3 + 2 + 1 = 7 \) (MIN)
Analytical experiments
Disjoint security measures

- Scalability evaluation while increasing graph size
- Fast resolution for base problem (one measure per node)
Overlapping security measures (1)

- Variation analysis of overlapping measures
- Graph with 1000 nodes
Overlapping security measures (2)

- Overlapping analysis on graphs of different sizes
- Same pattern (more overlapping => faster resolution)
Case study
Water Transport Networks
Case study

- Focus: water transport networks (base sub-system)

- Pressure sensors s1 (before) and s2 (after) the pump P1
- Flow sensor s3 (pump outflow)
- Water level sensor s5 at tank T1 with flow sensors (s4 and s6)
- Two PLCs: agent a1 (tank T1) and agent a2 (pump P1)
Case study

- Base sub-system repeated in larger infrastructures
Data collection and preparation

- Measures acquired from utility operators and public sources
- Attacker’s cost: three-point rating scale

<table>
<thead>
<tr>
<th>Factor / Rate</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skills ($f_1$)</td>
<td>no special skills/knowledge</td>
<td>advanced skills/knowledge</td>
<td>expert skills/knowledge</td>
</tr>
<tr>
<td>Tools ($f_2$)</td>
<td>off-the-shelf tools</td>
<td>non-conventional tools required</td>
<td>specialized tools</td>
</tr>
<tr>
<td>Time ($f_3$)</td>
<td>$\leq 10$ min</td>
<td>10-30 min</td>
<td>$\geq 30$ min</td>
</tr>
</tbody>
</table>

- Cost function

$$\psi(m) = f_1 \times f_2 \times f_3$$
Base subsystem (no redundancy)

<table>
<thead>
<tr>
<th>Components</th>
<th>Security measures</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>s3</td>
<td>{ F2-1, P1-2, A2-2 }</td>
<td>22</td>
</tr>
<tr>
<td>s5</td>
<td>{ F1-2, B1-1, A3-1, P2-2 }</td>
<td>14</td>
</tr>
<tr>
<td>a1</td>
<td>{ F1-2, B1-1, A3-1 }</td>
<td>6</td>
</tr>
<tr>
<td>a2</td>
<td>{ F1-1, B2-1, P1-1, A2-1 }</td>
<td>29</td>
</tr>
<tr>
<td>c1</td>
<td>{ F1-1, B2-1 }</td>
<td>9 + inf (special case)</td>
</tr>
</tbody>
</table>

Solution:
- Critical nodes: agent a1
- Security measures:
  - F1-2, B1-1, A3-1
- Total cost: 6
Extended subsystem (with redundancy)

<table>
<thead>
<tr>
<th>Components</th>
<th>Security measures</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2, a7, a8, a10</td>
<td>{ F1-1, B2-1, P1-1, A2-1}</td>
<td>29</td>
</tr>
<tr>
<td>a1, a3, a9</td>
<td>{ F1-2, B1-1, A3-1}</td>
<td>6</td>
</tr>
<tr>
<td>s1, s2</td>
<td>{ F1-1, B2-1}</td>
<td>9</td>
</tr>
<tr>
<td>c1</td>
<td>{ F1-1, B2-1}</td>
<td>9 + inf (special case)</td>
</tr>
<tr>
<td>s3</td>
<td>{ F2-1, P1-2, A2-2}</td>
<td>22</td>
</tr>
<tr>
<td>s4</td>
<td>{ F1-2, B1-1, A3-1, P2-1}</td>
<td>14</td>
</tr>
<tr>
<td>s5</td>
<td>{ F1-2, B1-1, A3-1, P2-2}</td>
<td>14</td>
</tr>
<tr>
<td>s6</td>
<td>{ F2-2, P1-3, A2-3, A3-1}</td>
<td>25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measure instance</th>
<th>Measure type</th>
<th>Attacker cost</th>
<th>Protection range</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1-1</td>
<td>F1</td>
<td>1</td>
<td>{ a2, a7, a8, a10, c1, s1, s2}</td>
</tr>
<tr>
<td>F1-2</td>
<td>F1</td>
<td>1</td>
<td>{ a1, a3, a9, s4, s5}</td>
</tr>
<tr>
<td>F2-1</td>
<td>F2</td>
<td>2</td>
<td>{ s3}</td>
</tr>
<tr>
<td>F2-2</td>
<td>F2</td>
<td>2</td>
<td>{ s6}</td>
</tr>
<tr>
<td>B1-1</td>
<td>B1</td>
<td>2</td>
<td>{ a1, a3, a9, s4, s5}</td>
</tr>
<tr>
<td>B2-1</td>
<td>B2</td>
<td>8</td>
<td>{ a2, a7, a8, a10, c1, s1, s2}</td>
</tr>
<tr>
<td>A2-1</td>
<td>A2</td>
<td>18</td>
<td>{ a2, a7, a8, a10}</td>
</tr>
<tr>
<td>A2-2</td>
<td>A2</td>
<td>18</td>
<td>{ s3}</td>
</tr>
<tr>
<td>A2-3</td>
<td>A2</td>
<td>18</td>
<td>{ s6}</td>
</tr>
<tr>
<td>A3-1</td>
<td>A3</td>
<td>3</td>
<td>{ a1, a3, a9, s4, s5, s6}</td>
</tr>
<tr>
<td>P1-1</td>
<td>P1</td>
<td>2</td>
<td>{ a2, a7, a8, a10}</td>
</tr>
<tr>
<td>P1-2</td>
<td>P1</td>
<td>2</td>
<td>{ s3}</td>
</tr>
<tr>
<td>P1-3</td>
<td>P1</td>
<td>2</td>
<td>{ s6}</td>
</tr>
<tr>
<td>P2-1</td>
<td>P2</td>
<td>8</td>
<td>{ s4}</td>
</tr>
<tr>
<td>P2-2</td>
<td>P2</td>
<td>8</td>
<td>{ s5}</td>
</tr>
</tbody>
</table>
Extended scenario (META4ICS display)

- Solution: nodes a1 and s2, instances F1-2, B1-1, A3-1, F1-1, B2-1
- Total cost: 15
Conclusion and future work
Conclusion

- Identification of security-critical nodes in ICS environments
- Security metric as least-effort attack strategy
- AND/OR graph-based models
  - Base problem (weighted AND/OR graphs)
  - Multiple overlapping security measures (AND/OR hypergraphs)
- Combination of AND/OR graphs with MAX-SAT optimisation techniques
- Experimental results indicate very good scalability
- Practical analysis of a realistic water transport network
Future work

- Evaluation on other ICS environments
  - Smart grid, power plants

- Integrate attack graphs at the cyber level

- Consider budget constraints

- Automated generation of AND/OR graph models for ICS