U.S. DEPARTMENT OF ENERGY | OFFICE OF CYBERSECURITY, ENERGY SECURITY, AND EMERGENCY RESPONSE

Firmware Indicator Translator (FIT)
Idaho National Laboratory (INL)

Rita Foster, Jed Haile, Christian Hunt, and New Context

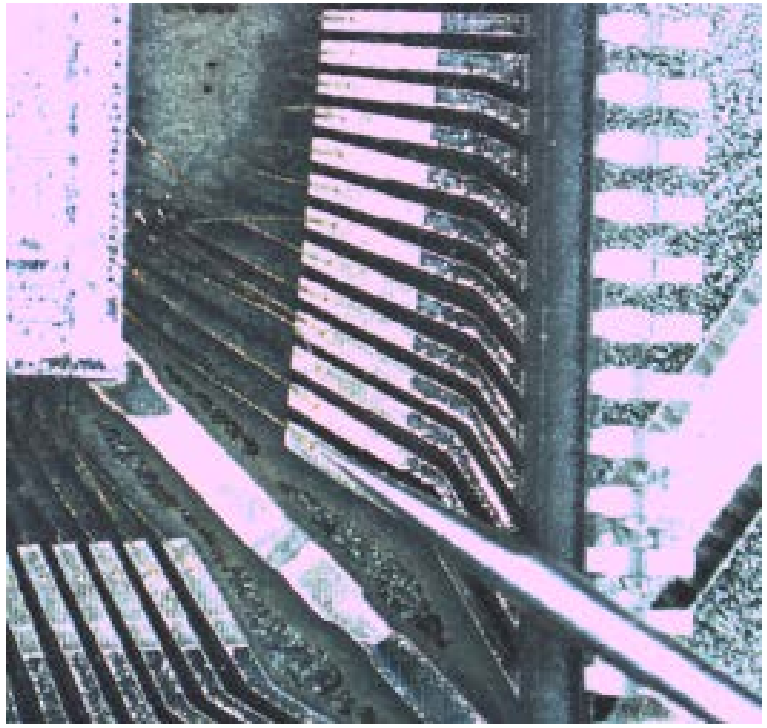
Cybersecurity for Energy Delivery Systems Peer Review

November 6-8, 2018

Summary: Firmware Indicator Translation

Objective

- Enable firmware indicator and response capabilities via binary and translated code analysis methods to visualize layers of firmware code complexity behavior.
- Solving the adversaries are “racing to the bottom” – Spectre, Meltdown, Ryzenfall, Chimera, Trisis, Supply chain backdoors challenges

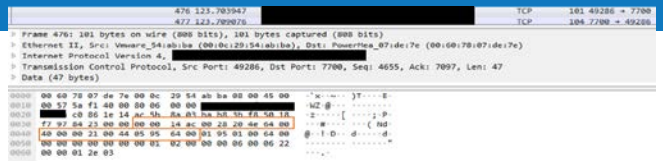


Schedule

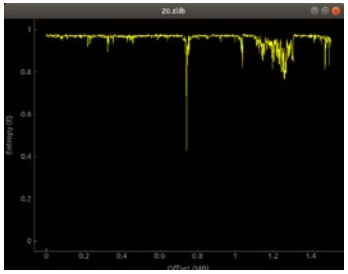
- FY18: Concepts prototyped
- FY19: Refine, Best-Fit; Scale & Test Use Cases
- FY20: Demo & Open Source Tools

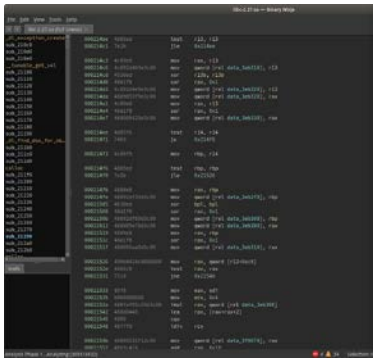
Total Value of Award:	**$ 2.3M**
Funds Expended to Date:	**% 30%**
Performer:	**Idaho National Laboratory**
Partners:	**DTE; SCE; PG&E; Siemens; New Context**

Advancing the State of the Art (SOA)

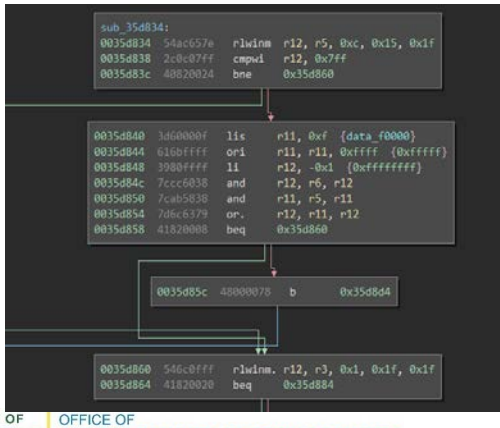


- **SOA Gaps:**
  - Firmware analysis tools are limited and static
  - Current Adversaries are focused on the sub components unseen, not monitored and undetected in firmware
  - Many Ontologies exist for code and architecture but none describe firmware complexities





- **FIT is:**
  - Untangling complexities in firmware
  - Agnostic to Vendor – Binary is Ground Truth
  - Sheds light on previously hidden ‘features’ in firmware
- **FIT end use will be broad:**
  - Visual representation of code behavior
  - Predictive code behavior
  - Highlight differences for firmware update
  - Enable the creation of indicators and remediation actions
  - Validate vendors and integrators products



Challenges to Success

Challenge 1 - Ontology

- Defined analysis ontology to identify components of firmware

Challenge 2 – Categories for Code

- Defined 17 feature matrix categories

Challenge 3 - Repeatability

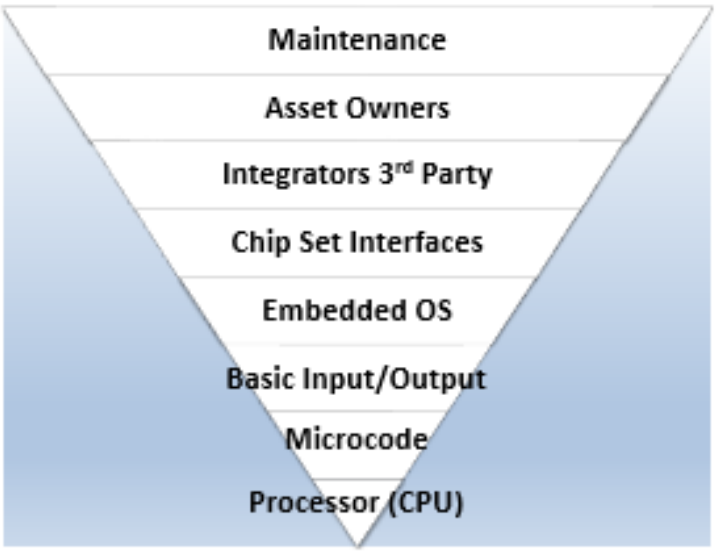
- 4 distinct platforms ready for analysis

Challenge 4 – Heterogeneity

- Volume of firmware and platforms will increase likelihood of all layers analysis

Challenge 5 - Scalability

- Related internal research is working with high-performance computing



Progress to Date

Major Accomplishments

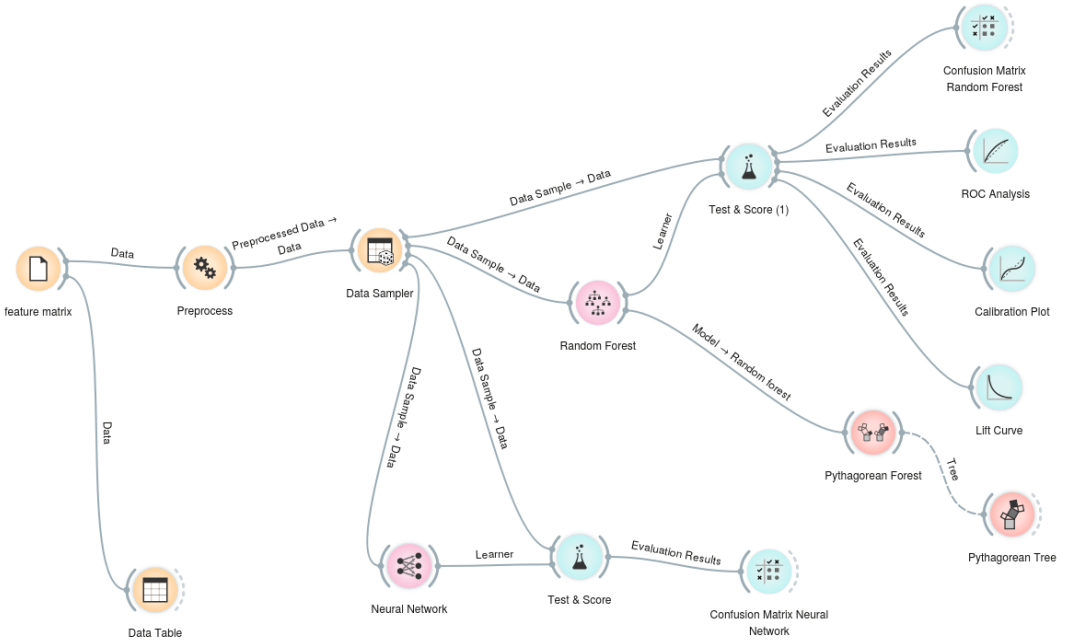
- Use of existing binary firmware analysis tools
- Creation of Firmware layers Ontology model
- Set up of 3 components/test environments provided by asset owners for analysis
- Indicator creation for Firmware Load/Extraction over a network & USB
- Demo proof of concept SIEM/STIX;

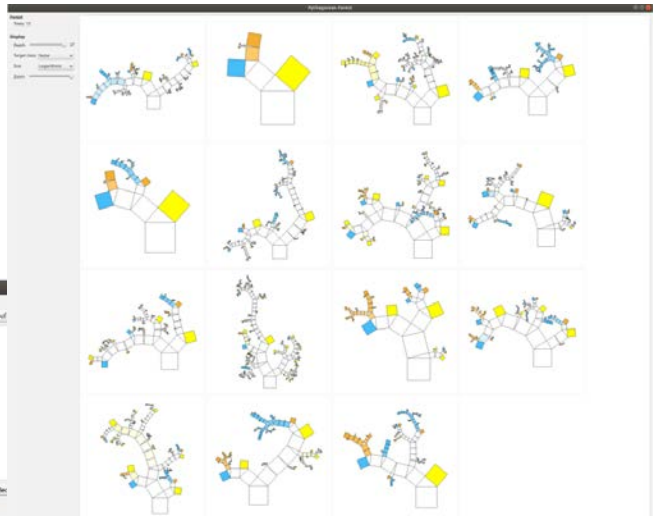
  Compromise vs Non
- Dis-assembled and translated sample libraries loaded into a graph database – many views of code
- Creation of dis-assembled translated Firmware Analysis Tool framework
- Identification and refinement of feature matrix via machine learning
- Multiple machine learning techniques used on sample libraries to visualize code behavior

Progress to Date

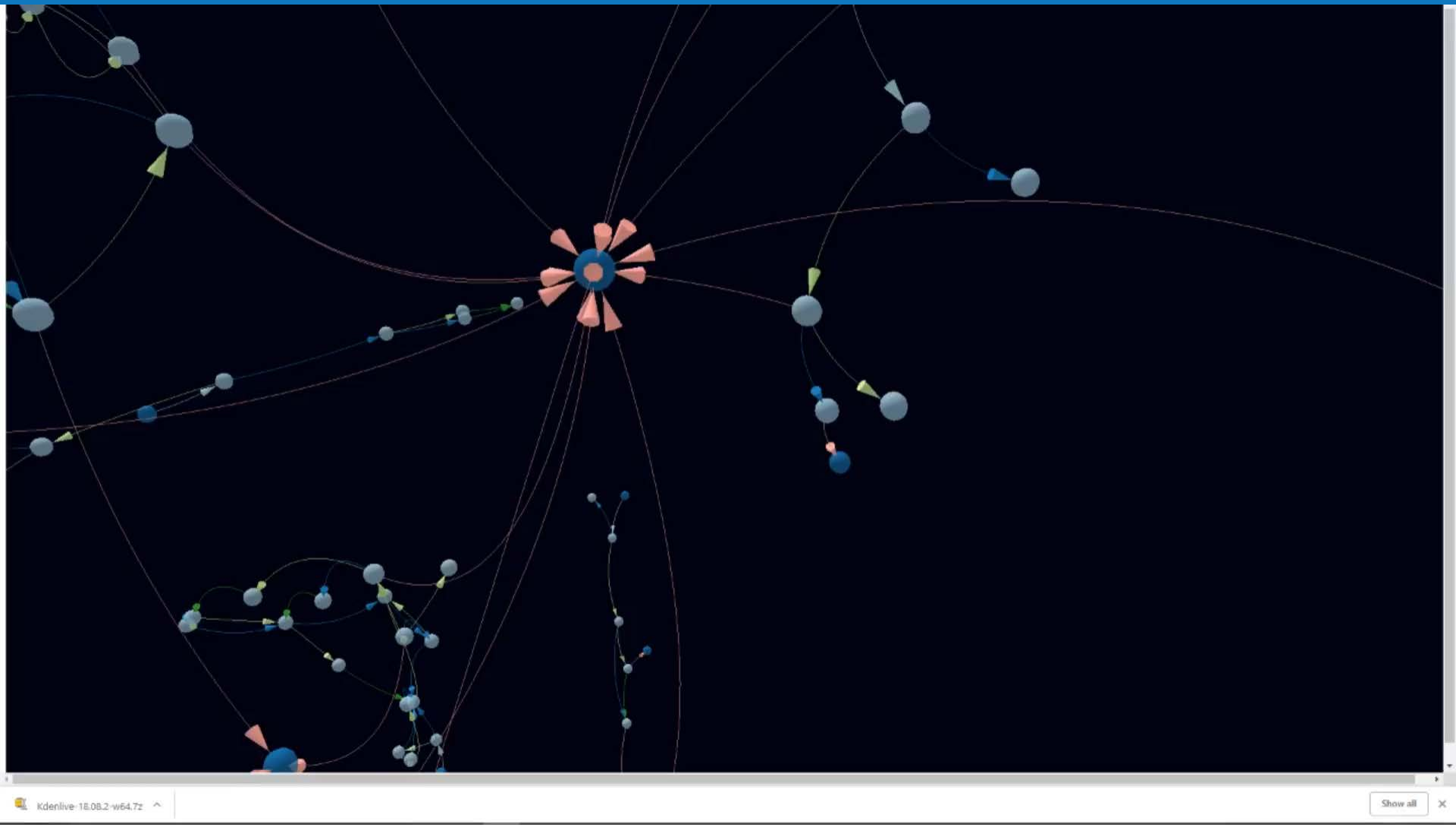
Major Accomplishments

- Binary Analysis
  - Code Tools with Exploit
  - Indicator of Firmware Egress
- Translation Proof
  - SIEM/STIX
- Dis-assembled and Translated Code Behavior Analysis Tool Set
  - Graph Database
  - Hieratical View
  - Machine Learning Techniques

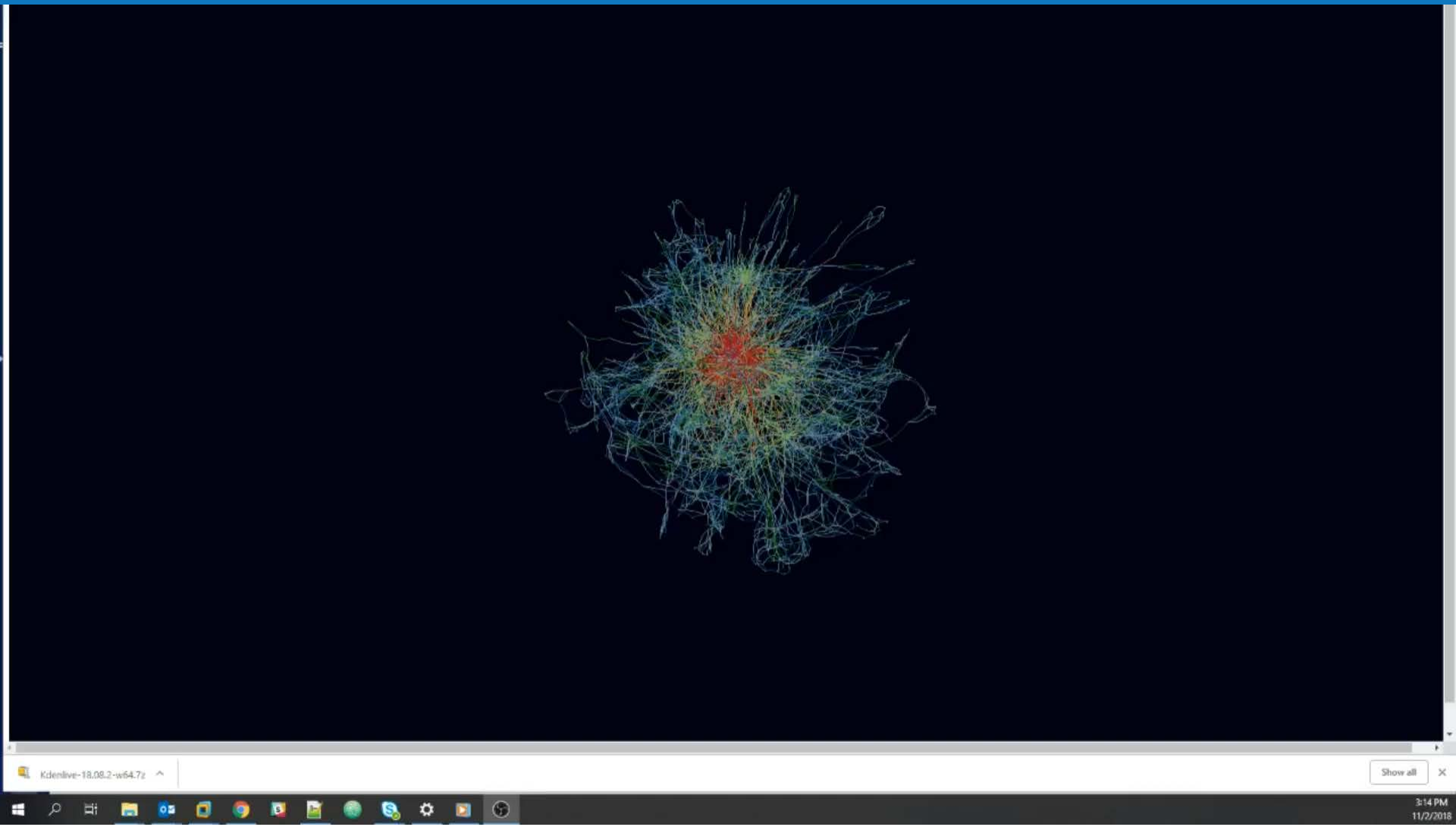




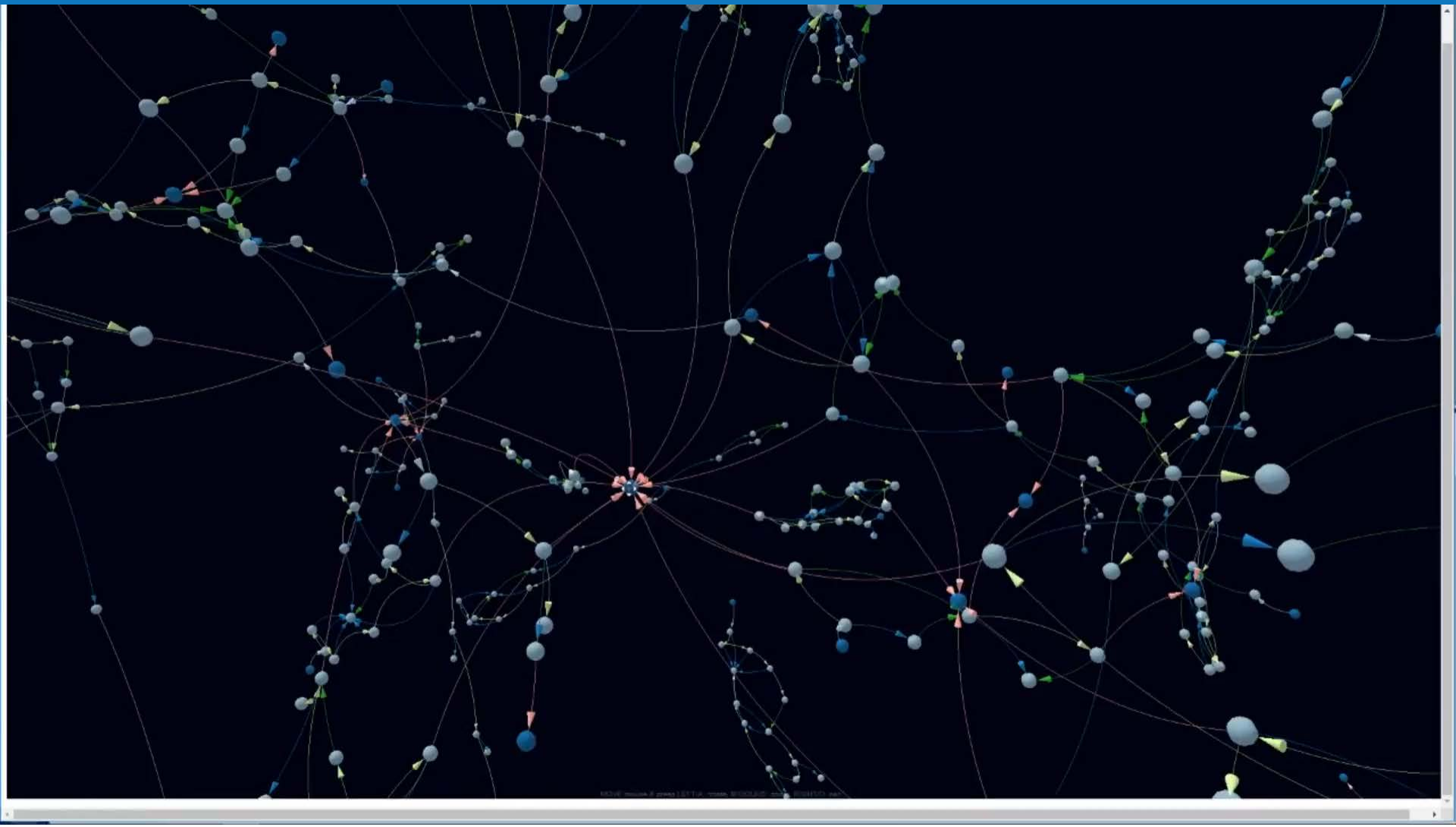
Graph Code Behavior



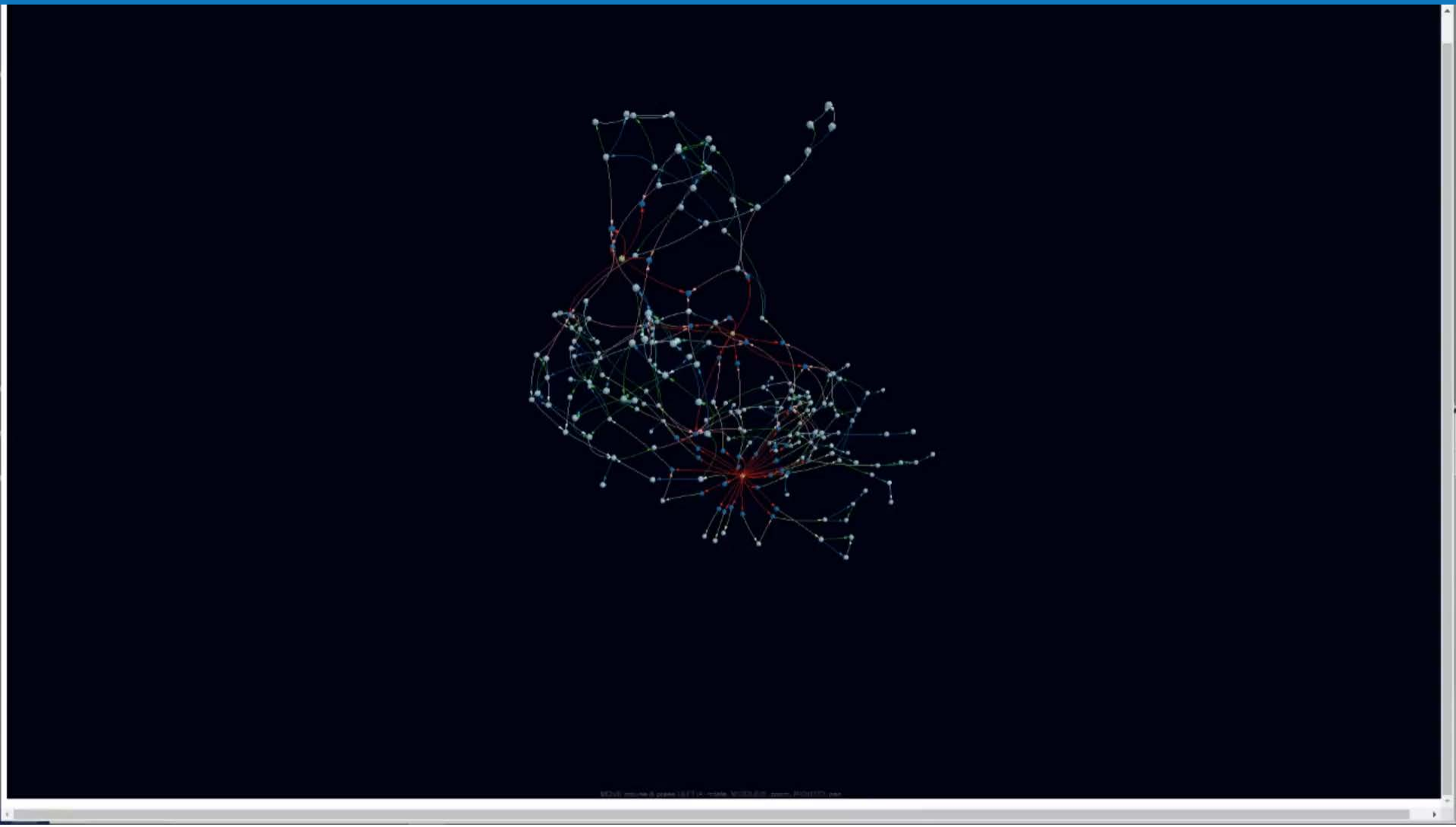
Larger Library



Stranded Code

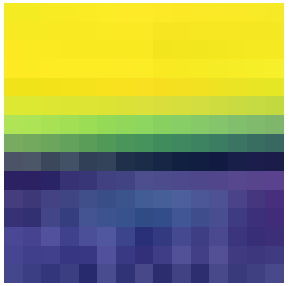


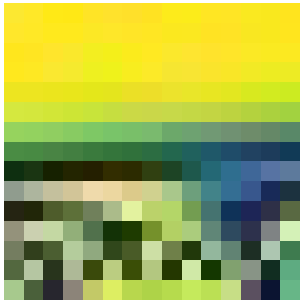
Smaller Code



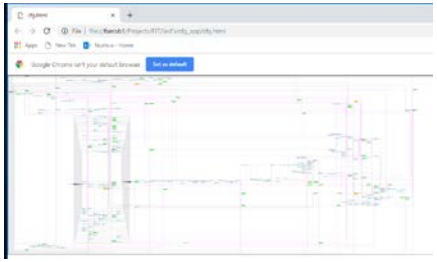
Collaboration/Technology Transfer

Plans to transfer technology – Open Source Knowledge to end user via use cases





- Conferences and published articles on lessons learned from binary analysis (DHS ICS-CERT 2018); results of using machine learning on translated code; lessons learned: data analytics from multiple threat feeds
- Asset Owner Use: abandon technology; out of band analysis to known good; indicator and remediation creation to manage cyber threat to firmware on most critical embedded systems
- Vendor use for analysis of firmware code and interfaces



- Original equipment manufacturer use validation of code sources
- Government use potential for identifying unknown embedded code in supply chain; validating critical embedded systems; understanding malware code behavior

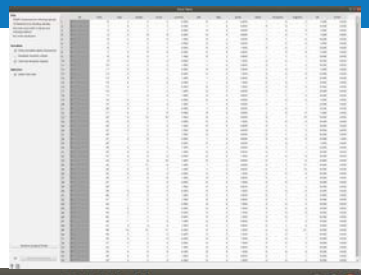
Next Steps for this Project

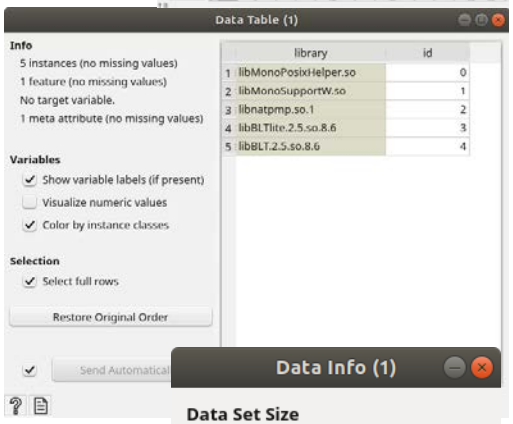
Approach for the next year

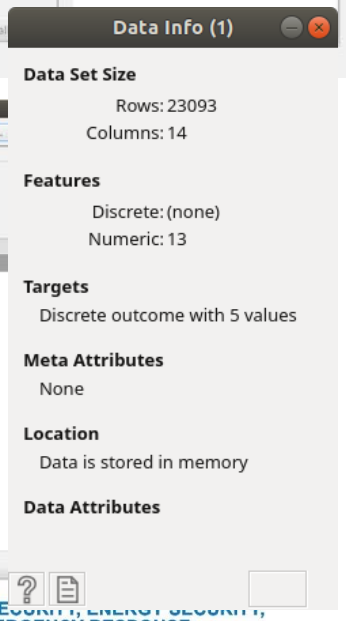
- Identify valuable tasks from binary analysis for potential use in translated code analysis
- Assess use of cyber injects/binary patch vs firmware versions
- Indicator analysis test set
  - Data analytics for heterogeneous threat source
  - Create indicators and remediation actions
- Scale up to one complete firmware base
  - Highlight known version/binary patch differences
  - Identify previously unknown and/or stranded

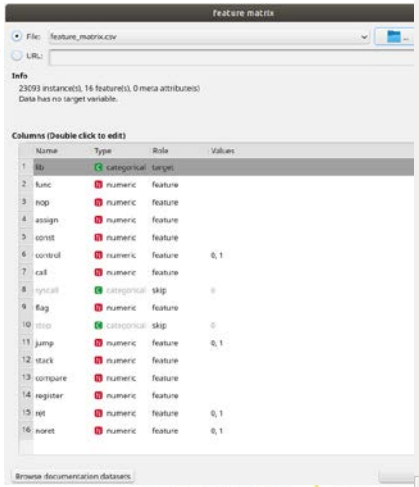
Approach for the final year

- Scale up to multiple firmware bases
- Identify demonstration and test
- Host on open source repository









Graphical Code