

Collaborative Intrusion Detection System

Laylon Mokry, Jose Quiroz, Patrick Bishop, Paul Slife

September 16, 2020

Meeting Notes

16 September 2020

- GUI Demonstration and Feedback:
 - The GUI for our Prototype is demonstrating layout; specifically, that our **basic** model will look similar to a **SIEMS** tool discussed in the previous meeting. Our scratch-up is below:

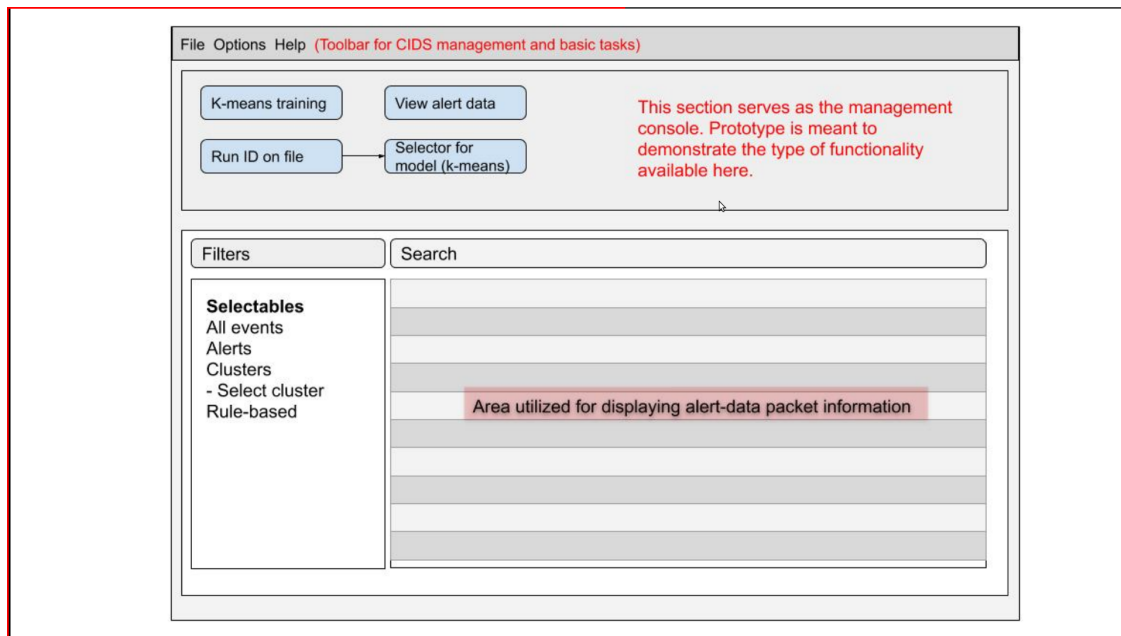


Figure 1: Concept

- Our initial “prototye” is shown below. The prototype is meant to demonstrate three user areas:
 1. Action panel (north). This is where the user selects actions to execute.
 2. Filter panel (bottom left). This is where the user will select filters, clusters, etc to display on the right hand side.
 3. Packet panel (bottom right). This is where the data packets will be displayed. This is not currently broken into fields, but will be.



Figure 2: Full console

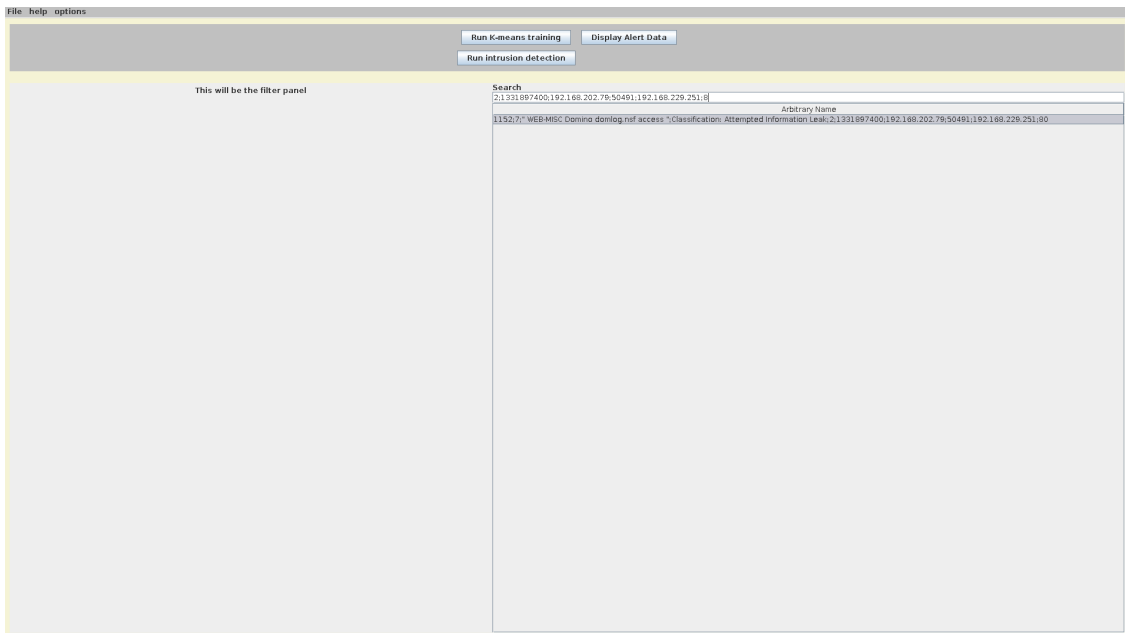


Figure 3: Conducting a search

- More specifically, the prototype aims to complete the functional requirements and acceptance testing shown in the table below.

| | |
|---|---|
| 4. User Interaction System provides a graphical user interface for the user to conduct intrusion detection requirements. | 4.1 The program opens a file with snort alert-level data. |
| | Expected Result-> The data from the file is parsed into an alert data class. (<i>normal</i>) |
| | 4.2 The system displays packet information for the user. |
| | Expected Result-> The data from the file is displayed in a content pane. (<i>normal</i>) |
| | 4.2 The user searches for a packet |
| | Expected Result-> The system displays only the packet which meets the search specification. (<i>normal</i>) |

- The current milestone requires customer feedback on the milestone:

“Customer feedback: Customer Involvement Alert => Prior to delivering this milestone, demo your operational Prototype and acceptance test case for your Customer. Present the feedback from your Customer regarding this demo to include whether your Customer agrees that you have demonstrated completion of the applicable acceptance test cases and any changes they want made.”
- Instructor feedback:
 - * Separate clusters by color in the data panel
 - * Data should be able to come from multiple sources
 - * Be able to connect to other users. TCP/IP
 - * One user should be able to initiate collaboration

Other discussion

- Differential privacy review
 - Achieved via addition of random noise
 - Benefit – provably secure
- The rest of the discussion was on clustering and on the differential privacy. The discussion was included in the following link: https://docs.google.com/document/d/19XImfqUX-phXZn_5oWGkPs97NRoG7dks_SjsjLCiTS0/edit

09 September 2020

- Improve implementation on categorical attributes (this is the bottleneck in latest implementation)
- Combine differential privacy with secure sharing to achieve better efficiency (differential privacy is more efficient but less accurate due to noise added)
 - Homomorphic Encryption
 - Summary of Homomorphic Differential Privacy
 - Differential Privacy Tutorial

- Privacy preserving solutions to other collaborative IDS tasks, e.g., rule-based, sequence-based. Currently we only do clustering of similar alerts but users can use more complex rule/sequential patterns. E.g., to identify a compromised machine used to launch attacks. (so there will be two alerts, first showing attack on the machine, second showing attack launched by that machine).
- Implement a GUI that allow users to detect IDS collaboratively and visualize results. It may look similar to SIEM tools. The following link give you some idea common SIEM tools SIEM Tools (free)
- Functional Requirements, in short:
 - Improve implementation
 - Implement differential privacy
 - * Combine with secret sharing
 - Pattern based to determine attacks
 - * Rule based
 - Snort alerts
 - * Different data sets
 - * Security data set repository
 - GUI
 - * Take the low level alerts as input
 - * Take the sensors and file as input
 - * Tasks and result of tasks
 - * Display super alerts
 - * Same interfaces for everyone
 - Communicating with peers