WCIS: A Prototype for Detecting Zero-Day Attacks in Web Server Requests
Presentation Outline

- Web Classifying Immune System (WCIS)
  - Traditional Artificial Immune System (AIS) features
  - Differences from traditional AIS
  - Classification Scheme
  - Web Server Request Model
  - Population Lifecycle

- Experimental Results
  - Accuracy at detecting attacks in specific classifications
  - Detection of unknown attacks

- Future Research

Dr. Melissa Danforth
Dept. of Computer & Electrical Engineering & Computer Science
California State University, Bakersfield
Web Classifying Immune System (WCIS)
Artificial Immune System (AIS)

- Inspired by biological immune systems
  - Ability to adapt to variants and new pathogens
  - Pattern matching for “antibody” and “antigen” binding
- AIS tries to distinguish “self” from “non-self”
  - “Self” is “normal” traffic, “non-self” is “abnormal” traffic
- Uses several key biological features
  - Negative selection
  - Affinity maturation
  - Immunization
  - Peripheral tolerance
Web Classifying Immune System (WCIS) 
Differences from Traditional AIS

- Add classifications to ‘non-self’ patterns
  - Enables specialization of sensors for specific areas
  - Enables “inoculation” for specific attack class(es)
  - Provides more information about zero-day attack than just “an attack has been detected”

- Separate evolutionary process from detection
  - Do costly processes “offline” on back-end system
  - Live traffic detection collects statistics to enable further refinement by back-end system
## WCIS – Request Classifications

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Info</td>
<td>Gather information about server</td>
</tr>
<tr>
<td>Traversal</td>
<td>Read-only directory traversal</td>
</tr>
<tr>
<td>SQL</td>
<td>SQL injection attack</td>
</tr>
<tr>
<td>Buffer</td>
<td>Buffer overflow attack</td>
</tr>
<tr>
<td>Script</td>
<td>Execute a script on the webserver</td>
</tr>
<tr>
<td>XSS</td>
<td>Cross-site scripting</td>
</tr>
</tbody>
</table>
## Characteristics of Request

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Version</td>
<td>+</td>
</tr>
<tr>
<td>HTTP Command</td>
<td>..</td>
</tr>
<tr>
<td>Number of Variables</td>
<td>\</td>
</tr>
<tr>
<td>Length of URI</td>
<td>( or )</td>
</tr>
<tr>
<td>%</td>
<td>&lt; or &gt;</td>
</tr>
<tr>
<td>`</td>
<td>//</td>
</tr>
</tbody>
</table>
WCIS – Request Parsing

- **Pattern/chromosome structure**
  - Contains full set of request fingerprint features
  - Flags indicate active/inactive features for sensor
  - Each sensor has at least two active features
    - Example: Length of 50-75 characters and 5-10 + characters

- **Pattern matching**
  - Sensor compares active features to request
  - Detects request as attack when sensor matches
    - Must fall within range for ranged features
    - Must match set bit for bitmap features
    - Example: Length 65 with 7 + characters
WCIS – Sensor Population Lifecycle

- Random generation of sensors
  - Select features randomly & initialize with random values

- Iterative affinity maturation
  - Perform negative selection
  - Test against attacks in population’s classification
  - Breed sensors with best affinity using genetic algorithm
    - Single point crossover and rank selection with elitism
    - Children feature selection based on union of parents’ active features and random active features from each parent

- Mutate subset of new sensors
  - Select random feature and alter it
WCIS – Sensor Population Lifecycle

- Deploy sensors on live environment
  - Currently just test sensors against unlabeled data
  - Record accuracy at detection and false positives
  - Compare classification decisions by sensor populations
- Refine sensors in response to live detection
  - Export statistical information to back-end system
  - Enter a modified affinity maturation loop
  - Code supports concept, but untested due to red tape
- Received clearance to test live deployment and refinement during this academic term
Experimental Results
Results – Experimental Setup

“Normal” dataset – 52977 requests
- Web server requests from DARPA Lincoln Labs logs
- Verified normal requests from live web server logs

“Attack” dataset – 179 attacks
- Buqtraq proof of concepts
- Verified attacks from live web server logs
- Logs of tests run on isolated machine

“Unknown” dataset – 11659 requests
- Random entries from Apache access.log repository for the department web server
## Results – Experimental Setup

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pop</td>
<td>Population size for each classification</td>
</tr>
<tr>
<td>Gen</td>
<td>Max iterations for affinity maturation</td>
</tr>
<tr>
<td>Xover</td>
<td>Percent selected as parents by GA</td>
</tr>
<tr>
<td>Mut</td>
<td>Mutation rate for population</td>
</tr>
<tr>
<td>Thresh</td>
<td>Threshold affinity for negative select.</td>
</tr>
<tr>
<td>Agree</td>
<td>Attack alert agreement threshold</td>
</tr>
</tbody>
</table>
Results – Classification Accuracy

Pop=25 Gen=40 Mut=1%

![Graph showing classification accuracy over generations for different categories: Info Traversal, SQL Buffer, Script XSS.](image)
Results – Classification Accuracy

Pop=50 Gen=10 Mut=2.5%
Results – Classification Accuracy

Pop=75 Gen=20 Mut=5%
## Results – Unknown Attacks Detected

<table>
<thead>
<tr>
<th>Class</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traversal</td>
<td><code>.php?index=../../../../proc/self/environ%00</code></td>
</tr>
<tr>
<td>Script</td>
<td><code>/* .php?option=com_dump&amp;controller=../../../..// ../../../..///proc/self/environ%0000</code></td>
</tr>
<tr>
<td>Traversal</td>
<td>Same as previous line</td>
</tr>
<tr>
<td>Script</td>
<td><code>/faculty/interests/../../../index.html</code></td>
</tr>
<tr>
<td>Script</td>
<td><code>/cs150/index.php?p=../../../</code></td>
</tr>
<tr>
<td>Script</td>
<td><code>../../../../../ports_labeled.jpg</code></td>
</tr>
</tbody>
</table>
Future Research
Future Research

- Detection against modeled data (real-time)
  - Isolated network is now functional
- Detection against live data – clearance received
- Expand fingerprint to include other parts of request
  - Attack data can be in other fields in request
- Explore other genetic algorithms
  - Single objective algorithm may not be best
  - Try multi-objective algorithms
  - Try variations on genetic algorithms
- Investigate other networking problem domains
Questions?