An Evolutionary Strategy for Resilient Cyber Defense

Errin W. Fulp, H. Donald Gage, David J. John, Matthew R. McNiece, William H. Turkett, and Xin Zhou

Wake Forest University
Department of Computer Science

Cisco Research & Open Innovation

GLOBECOM
December 9, 2015
• Many attacks can be attributed to poor software configurations
  – Misconfigurations are a common security issue for web-services

• Often these errors are high-impact and difficult to resolve
Difficulty of the Problem

- Consider a single configuration with \( n \) parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Possible Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeepAlive, allow requests over the same connection</td>
<td>0, 1</td>
</tr>
<tr>
<td>KeepAliveTimeout, wait time to wait for requests</td>
<td>0 - 1800</td>
</tr>
<tr>
<td>Indexes, automatic directory indexing</td>
<td>0, 1</td>
</tr>
<tr>
<td>LimitRequestBody, limit the message size</td>
<td>0 - 65535</td>
</tr>
<tr>
<td>LimitRequestFields, limit number of HTTP requests</td>
<td>1, 0</td>
</tr>
<tr>
<td>LimitRequestFieldSize, limit HTTP header field size</td>
<td>0 - 32767</td>
</tr>
</tbody>
</table>

A configuration is a set of parameter settings

\[
C = \{ 1, 800, 1, 32767, 0, 1 \}
\]

- Want to find parameter settings that are feasible and secure
  - Number of parameters \( (n) \) can be extremely large
  - Parameters may be interdependent, forming *parameter chains*

- Can model the configuration as a Boolean expression
  - Dependencies can be expressed using AND, OR, and NOT
  - Finding a secure configuration is similar to the satisfiability problem
Finding a Secure Configuration

- Security guidelines (e.g. DISA STIG and NIST checklists) are helpful
  - Settings are often too basic, may be out-of-date, or not applicable
- Can apply different search algorithms to find secure configurations

- However, security is **not** a static problem
  - Computing environment and threats continually change
• What is secure today, is not necessarily secure tomorrow
  – New exploits, new applications, and better settings

• Consider DISA STIG recommendation changes (fixes?) over time

<table>
<thead>
<tr>
<th>Application</th>
<th>Additions</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache2.2 from 2011 until 2015</td>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>RHEL5.0 from 2012 until 2015</td>
<td>4</td>
<td>135</td>
</tr>
</tbody>
</table>

• Therefore, configuration discovery must continually adapt and improve

\(^a\text{These numbers perhaps overstate this point, since all differences are considered.}\)
Evolutionary Algorithms

- Evolutionary Algorithms (EAs) mimic evolution to find solutions
  - Better solutions are created from good solutions
- EAs have several beneficial characteristics
  - Continually adapt to the current environment
  - Randomness (mutation) can be used to create diversity
Modeling Configurations as Chromosomes

- Configuration consists of multiple parameters
  - Individual configuration parameter settings are chromosome traits

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>file permission</td>
<td>permissions for the secret file</td>
<td>222</td>
</tr>
<tr>
<td>login banner</td>
<td>message displayed upon login</td>
<td>2</td>
</tr>
<tr>
<td>file ownership</td>
<td>ownership of the secret file</td>
<td>774</td>
</tr>
<tr>
<td>max open files</td>
<td>change max number of open files</td>
<td>220</td>
</tr>
<tr>
<td>max file</td>
<td>change max file descriptor</td>
<td>409</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Some parameters settings will affect security, however which are security related is possibly unknown

- Therefore a chromosome will be a configuration

- Chromosomes will be be ranked, need a measure of fitness

\[ C = \{22, 2, 774, 220, 409, \ldots\} \]
Chromosome (Configuration) Fitness

- Fitness (*security provided*) is based on the attacks encountered

- Common Vulnerability Scoring System (CVSS) based fitness
  - CVSS has a 6 part vector that yields a 0 - 10 score
    
    | CVE            | CVSS Vector | CVSS Score |
    |----------------|-------------|------------|
    | Default ssh password | AV:N/AC:L/Au:N / C:P/I:P/A:P | 7.2 (high) |

  - Detail is helpful, but may not be available at the time of attack

- Attack signatures, count the types of attacks (*currently used*)
Evolutionary Algorithm Processes

- Maintain a population (pool) of chromosomes and perform 3 processes
  
  ![Selection Diagram](image)
  
  ![Crossover Diagram](image)
  
  ![Mutation Diagram](image)
  
  - Selection identifies parents for new chromosomes (configurations)
    - Roulette wheel selection, chromosomes weighted based on fitness
  - Crossover combines selected chromosomes to create new chromosomes
    - Multi-point crossover used
  - Randomly change traits (configuration parameter settings)
    - Provides diversity and exploration of solution space
Evolution as a Resilient Defense

- System does not converge to a single value, it continually iterates
  - Demonstrated to discover secure settings for a given attack
- Interested in performance when attacks change over time
Experimental Setup

• Experiments using 200 RHEL5 and Apache 2.2 configurations
  – Managed configurations have 102 parameters

• Interested in the resiliency of the approach
  – Alternated attacks (red and blue) each target 5 unique parameters
  – Parameters differed on the number of possible settings

• Performance measured security provided and diversity
  – Scored parameters, zero if vulnerable or 100 if secure
  – Pool diversity measured using Hamming distance

• 100 experiments performed and the average results were recorded
Experimental Results for Two Phases

- Secure settings are found for the attacked parameters for both phases
  - Some differences in the number of generations required
  - *Do lose the non-attacked parameter settings*
- Diversity also improves as the configurations become more secure
  - Agree on secure settings, remaining parameters are diverse
Experimental Results for Four Phases

- Similar to the 1st and 2nd phases, EA is able to find secure settings at 3rd and 4th phases
Experimental Results: Parameter Fitness

- Parameters with more possible settings require more generations
Conclusions and Future Work

• EAs provide a method for finding secure configuration settings
  – Relies on selection, crossover, and mutation

• Interested in the resiliency of the approach
  – EA is able to discover secure settings as attacks changes
  – *Does lose previously secure settings however*...

• While promising, there are some interesting open questions
  – Maintain *history* of secure parameter settings
  – May be possible to identify attacked parameters
History-Based Mutation

- Mutation encourages the EA to discover new solutions
  - Mutation may also cause the loss of previously discovered solutions
- Record the security (or lack of) for each tested parameter-setting

Create a counter for each setting of each parameter
Initialize setting-counters to zero

Given an *assessed* configuration
foreach setting used in the configuration
  if the configuration is secure then
    Increment the associated setting-counter by 1
  else
    Decrement the associated setting-counter by 1

- Over time this builds a density estimate (*for this threat vector*)
  - Use estimate to bias the mutation towards secure settings
Parameter Setting Densities

- Consider a parameter with settings GET, POST, OPTIONS, PUT, and CONNECT

- Counts initially collected during a training phase
  - Weights are created based on the counts
  - Note, settings with negative counts can receive non-zero weights to allow future exploration

- Weights are then used for to bias the mutation for this parameter
Mutation Process

- A new pool of chromosomes (configurations) is created per generation

- Only a percentage of the chromosomes use history-based mutation
  - History-based configurations learn from previous settings (history)
  - Remaining configurations continue to search for alternative settings

- The most fit will most-likely survive to the next generation
Experimental Results: Max Fitness

- At 3\textsuperscript{rd} phase, history-based mutation is able to quickly reestablish secure settings, similar for 4\textsuperscript{th} phase