An Evolutionary-Inspired Approach for Moving Target Defenses

Wake Forest University
Department of Computer Science

Errin W. Fulp, Daniel Cañas, David J. John, *and* William H. Turkett
Department of Computer Science • Wake Forest University • Winston-Salem, NC

supported by
National Science Foundation
Secure and Trustworthy Cyberspace
Award 1252551

Motivation

Moving Target (MT) environments provide security through diversity by changing various system properties that are explicitly defined in the computer configuration. Temporal diversity can be achieved by making periodic configuration changes; however in an infrastructure of multiple similarly purposed computers diversity must also be spatial, ensuring multiple computers do not simultaneously share the same configuration and potential vulnerabilities. Given the number of possible changes and their potential interdependencies discovering computer configurations that are secure, functional, and diverse, is challenging.
A Genetic Algorithm (GA) can be employed to find temporally and spatially diverse secure computer configurations. In the proposed approach a computer configuration is modeled as a chromosome, where an individual configuration setting is a trait or allele. The GA operates by combining multiple chromosomes (configurations) which are tested for feasibility and ranked based on performance which will be measured as resistance to attack. Successive iterations of the GA yield configurations that are often more secure and diverse due to the crossover and mutation processes.

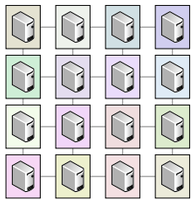
1 Computer Infrastructures

- Computers are often deployed in a deterministic fashion
  - Computing Clusters, Database farms, VM Servers
  - Configured in an automated and predictable fashion
- The predictable and uniform nature is beneficial for *attackers*
  - A single exploit is applicable to a large number of machines
- One defense is to create a **moving target** environment

2 Attack Process

- Attacker Process above demonstrates a simplified version of the steps that an attacker takes [2]
- Moving target environments provide security through diversity
  - Should make reconnaissance phase ineffective
- Moving Target (MT) environment creates uncertainty for attackers
  - Lead an attacker to making suboptimal decisions
  - Or to the attacker to not attack at all

3 Moving Target Environments



- Want to create a moving target defense for computers.
  - Change configurations over time, remain secure and functional
- Configurations should be diverse, temporally and spatially
  - **Temporally** - difference in a single computer over time.
  - **Spatially** - difference between computers at any point in time.

4 Computer Configurations

```
;
# Create "/keygen" if it doesn't exist.
class gen_class
    {
        file { "/etc/cipher/keygen.list":
            ensure => present,
            mode => 644,
            owner => root,
            group => root
        }
    }
```

```
;
NameVirtualHost 172.20.30.40
<VirtualHost 172.20.30.40>
# primary vhost
DocumentRoot /www/subdomain
RewriteEngine On
RewriteRule /.* /www/subdomain/index.html
ServerSignature Off
ServerTokens Prod
# end definition
</VirtualHost>
;
```

```
;
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
# increase Linux autotuning TCP buffer limits
# min, default, and max number of bytes to use
# (only change 3rd value, make it 16 MB or more)
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216
# recommended to increase this for 10G NICS
net.core.netdev_max_backlog = 30000
net.ipv4.tcp_congestion_control=cubic
# are you really reading this?
;
```

- Attacker observes computer properties during reconnaissance phase
  - Configurations define properties of the OS and applications
- Defining one good configuration is difficult
  - Windows system typically has over 200,000 registry entries

5 Difficulty of Finding Secure Configurations

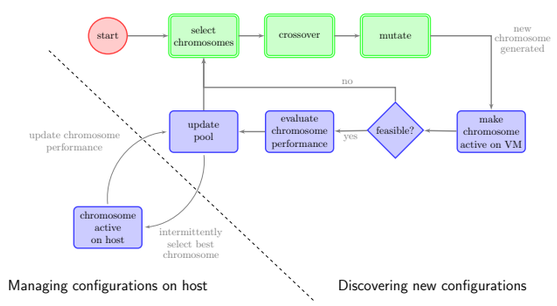
- Let C be a configuration of n individual settings, $C = \{p_1, p_2, ..., p_n\}$

Description	Possible Values
max handles /proc/sys/fs/file-max	0 - 10000
max TCP rev /proc/sys/net/core/rmem_max	0 - 256960
max TCP send /proc/sys/net/core/wmem_max	0 - 256960
timestamps /proc/sys/net/ipv4/tcp_timestamps	0, 1
buffer size /proc/sys/net/ipv4/tcp_rmem	0 - 256960
scaling /proc/sys/net/ipv4/tcp_windowscaling	0, 1

$C = \{10000, 4093, 4093, 0, 4093, 1\}$

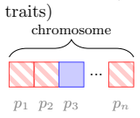
- Parameters may be interdependent
  - Certain subset of settings may be required for feasibility or security
  - Requiring `ssh` is not beneficial with weak passwords, or enable scaling if the max TCP window ≥ 65535
- Therefore finding feasible, secure configurations is difficult

6 Genetic Algorithms and Configurations



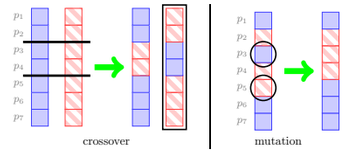
- Genetic Algorithms (GAs) mimic evolution to find solutions [3].
  - Better solutions are created from good solutions
  - Can perform well with difficult search problems
- Using a GA to create a MT environment
  - Model a computer configuration as a chromosome (configuration parameters are the chromosome traits)

$$C = \{p_1, p_2, p_3, ..., p_n\} \Rightarrow$$



  - Perform GA processes (selection, crossover, and mutation) to chromosomes (configurations)
  - Potentially update chromosome pool with resulting configuration based on *feasibility* and *fitness*
  - Intermittently replace existing configuration with a configuration from chromosome pool

7 Genetic Algorithm Processes



- **Selection** - Determine the best pair of chromosomes from pool
  - Requires fitness values to rank configurations
- **Crossover** - Combine chromosomes to produce new chromosomes
  - Combining parts of good solutions to create better solutions
- **Mutation** - Randomly change traits in the new chromosome
  - Help move away from local minima/maxima
  - Increases diversity of configurations, important for MT

8 Feasibility and Fitness

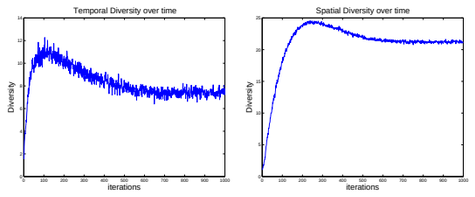
- **Feasibility** determines if the configuration is functional
  - Use a set of test predicates to determine basic functionality
- **Fitness** measures the security of the configuration
  - Measurement is not known in advance, determined over time
  - **Vulnerability score** based on the number and type of security attacks, a_i, over period of time, potentially weighted

$$v = \sum_{i=1}^{n} w_i a_i$$

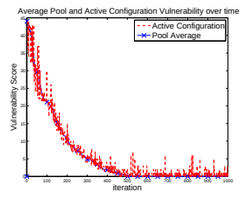
- Chromosome fitness value will not always be correct

9 Initial Simulation Results

- Simulated 256 computers, same initial configuration [1]
  - Configuration has 80 binary parameters, initially insecure
  - 2^{80} possible, but not necessarily secure or functional
  - Pool consisted of 10 chromosomes per computer
- Diversity is measured as the Hamming distance
  - Temporal is average distance between successive configurations
  - Spatial is average distance between any two computers



- Vulnerability score is a weighted value that measures the vulnerability of the configuration (lower score is better)



10 CVE, CCE, and CVSS Information

Example CVSS Score

`9.4 (CVSS#AV:L/AC:M/Au:N/C:N/I:P/A:C)`

Attack has an access vector that is low (`AC:L`), access complexity that is medium (`AC:M`), no authentication required (`Au:N`), no confidentiality impact (`C:N`), an integrity impact that is partial (`I:P`), and an availability impact that is complete (`A:C`).

- NIST Common Vulnerability Scoring System (CVSS)
  - Enumerates vulnerabilities and the severity
  - Information includes impact on information confidentiality, integrity, and access
  - Information can be used to provide a fitness basis
  - Nessus security scanner provides CVSS information
- Common Vulnerabilities and Exposures (CVE) and Common Configuration Enumeration (CCE)
  - Help identify security vulnerabilities and important configuration parameters

11 Future Directions

- **Collecting and Analyzing Configurations**
  - Consider NIST CVE and CCE for configuration parameters
  - Identify set of interdependent configuration parameters
  - Determine parameter roles in terms of security or diversity
- **Development of Feasibility Tests and Fitness Functions**
  - Develop predicates to determine feasibility and fitness [4]
  - Use NIST CVSS to rate configuration fitness
  - Investigate Nessus for configuration fitness assessment
- **Genetic Algorithm and MT Development**
  - Investigate crossover rate and mutation rates in the context of uncertain fitness values
- **Configuration Diversity**
  - Effectively measuring diversity (how far apart configurations are) is critical for assessing the performance of any MT defense.
  - Initial investigation will also be done on attacker discernible configuration settings.
- **Prototype implementation and Testing**
  - Understand performance under realistic conditions

12 References

[1] Michael B. Crouse and Errin W. Fulp. A moving target environment for computer configurations using genetic algorithms. In *Proceedings of the 4th Symposium on Configuration Analytics and Automation (SafeConfig 2011)*, 2011.

[2] Dorene Kewley, Russ Fink, John Lowry, and Mike Dean. Dynamic approaches to thwart adversary intelligence gathering. In *Proc. of the DARPA Information Survivability Conference & Exposition II (DISCEX '01)*, volume 1, pages 176–185, 2001.

[3] Melanie Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998.

[4] Ya-Yunn Su, Mona Attariyan, and Jason Flinn. Autobash: improving configuration management with operating system causality analysis. *SIGOPS Oper. Syst. Rev.*, 41:237–250, October 2007.