The Search for Optimality in Automated Intrusion Response

Yu-Sung Wu and Saurabh Bagchi
Dependable Computing Systems Lab (DCSL) &
The Center for Education and Research in Information Assurance and Security (CERIAS)
School of Electrical and Computer Engineering
Purdue University

Work supported by:
Purdue Research Foundation, Avaya, CERIAS

Survivable Systems and Intrusion Response

- Modern life heavily depends on computer systems
- Intrusions/security breaches to these systems occur
- Ways to make a system survivable
  - At design/implementation phase
    - Eliminate vulnerabilities
    - Policy/Access Control/Cryptography/Formal Verification
  - In production phase
    - Use IDS (system logs checking/network packet sniffing/virus, worms scanning, detecting files modifications…) to identify misuses/anomalies
    - Perform incident/intrusion response (IRS) to detected misuses/anomalies
      - Containment and Recovery

Focus of this work

Intrusion Response System

- The need for IRS
  - A survivable system needs to provide functionality through intrusions
  - Human intervention after IDS alert can be costly and slow
  - IRS takes reports from IDS (usually bundled together), processes it, and carries out actions to counter the intrusion
- Existing examples of IRS
  - Anti-virus software which disables access to worm executables or files infected with virus
  - Routers/firewalls which actively block worm traffic
  - Laptops equipped with motion sensor and TPM module that can lock up the computer when unauthorized movement (usually occurs when the laptop is being stolen) is detected

Intrusion Response System

- Summary on existing IRS
  - Most of them are stand-alone and are tied with one single and specific detector (IDS)
  - Target mostly at one machine box only
- IRS for Distributed Systems
  - An environment of multiple interconnected boxes with heterogeneous and cooperating services
  - Few general-purpose IRS solutions exist for distributed systems
  - The most common way is to use the stand-alone solutions separately and independently on the boxes
    - E.g., Have McAfee anti-virus software installed on the workstation boxes, and CISCO IPS on the network joints.

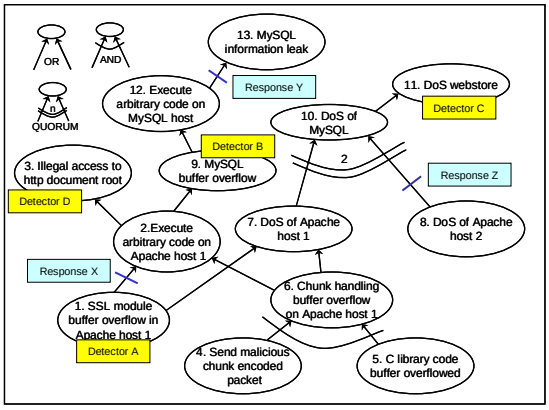
IRS for Distributed Systems

- Drawback of the "common way"
  - Each IRS/IDS pair does not leverage the detection reports from the other IDSs
    - Existing research on correlation IDS have shown clear advantages of doing so
  - The IRS/IDS pair does not consider the effects from the response actions carried out by the other IRS/IDS pairs
    - This can lead to redundant response actions and denial-of-service of the system at worst.
  - At best, locally optimal decisions from each IRS/IDS pair
  - There is no guarantee of system-wide global optimality

ADEPTS IRS: Basics

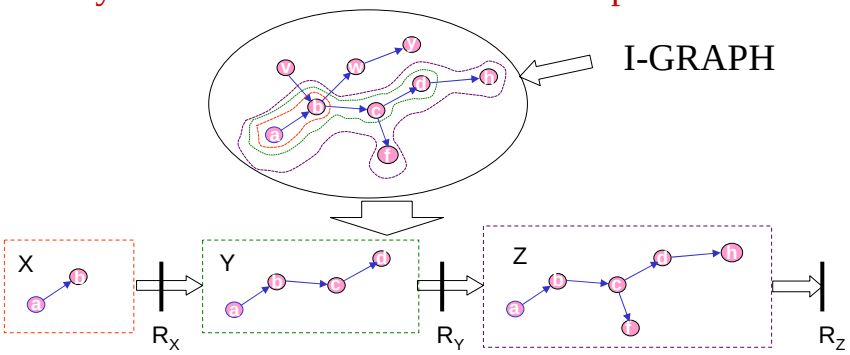
- Specifically designed for distributed systems
- Use I-GRAPH (a variant of attack graph) as the core binding between detectors (stand-alone IDSs) and response actions (stand-alone IRSs)

- A detector is associated with an I-GRAPH node to tell the confidence index of that node being compromised
- The compromise confidence index of nodes without associated detectors can be inferred through the graph structure
- Response actions are associated with edges
- A response is meant to stop the progression of the intrusion from the source node to the destination node



Attack Phases

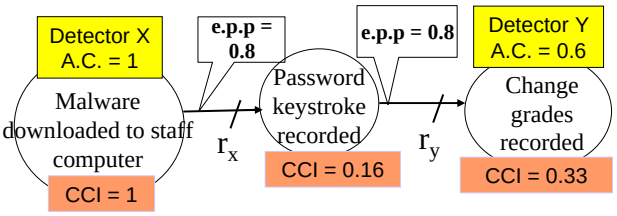
- The dynamics between intrusion and response



- Assuming an attack includes three "snapshots" X, Y, and Z – called *attack phases*
- Each snapshot includes I-GRAPH nodes which have been achieved as part of the attack thus far
- Following each snapshot *k*, ADEPTS determines a set of response actions R_k for deployment

Diagnosis of Achieved I-GRAPH Nodes

- Responses in I-GRAPH



Assume: Prob(r_x fails) = 0.2 Prob(r_y fails) = 0.4

For an edge e connecting node a to b in I-GRAPH with response r :

$$cci(b) = \begin{cases} cci(a) \times Prob(r\ fails) \times epp(e) & \text{, if b has no detector} \\ [cci(a) \times Prob(r\ fails) \times epp(e) + AC(x)]/2 & \text{, if b has detector x} \end{cases}$$

epp(e) : The edge propagation probability of edge e. This models an adversary's likelihood of taking this edge

ADEPTS deploys responses on the attack phase based on the CCI value of a node

Impact Vector

- A system has transaction goals and security goals that it needs to meet through the time of operation
  - Example: providing e-mail service and ensuring the confidentiality of sensitive data
- Attacks are meant to impact some of these goals
- Deployed responses also impact some of these goals
- Assume the impact from an attack to the system can be quantified through a vector IV with each element in the IV corresponding to the impact on each transaction/security goal $\in [0, 1]$
  - Impact vector for adversary reaching I-GRAPH node n_k is $IV(n_k)$
  - Impact vector for response r_k is $IV(r_k)$

Optimality of Response Actions

- We formally define the cost for a response combination (a set of response actions) RC_i as:

$$Cost(RC_i) = | Iv(RC_i) | = \left| \sum_{k=1}^{m} Iv(n_k) \Pr ob(n_k) + \sum_{k=1}^{n} Iv(r_k) \right|$$

In our attack graph model, $Prob(n_k)$ is estimated as $CCI(n_k)$

Accurately speaking, $Prob(n_k)$ is conditioned on many factors, and determining its value is by itself a challenging research problem

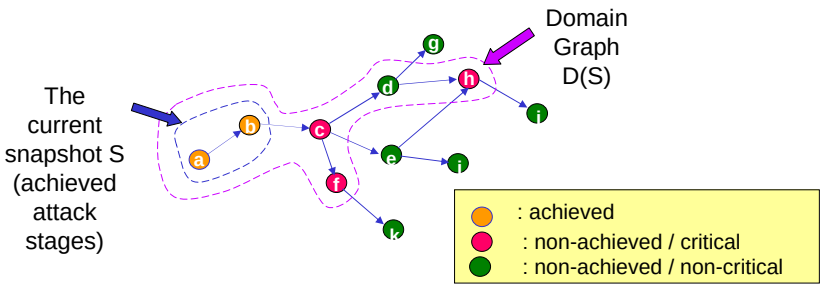
- The response combination RC_i is said to be optimal for the given attack if it achieves minimum $Cost(RC_i)$

Types of Response Actions

- Given a snapshot *s* and I-GRAPH *G*
  - In terms of containment, ADEPTS should consider all response actions applicable to the graph (*G*-*s*)
  - In terms of recovery, ADEPTS should consider all response actions applicable to the graph *s*
- *s* is typically not huge, as its size is linear in the number of detectable steps in a multi-stage attack
- But (*G*-*s*) can be huge
  - A function of applicable attack steps (vulnerabilities) in all services in the application system
  - However, for nodes in (*G*-*s*) which are far away from *s*, the likelihood of them being reached is lower than for closer ones

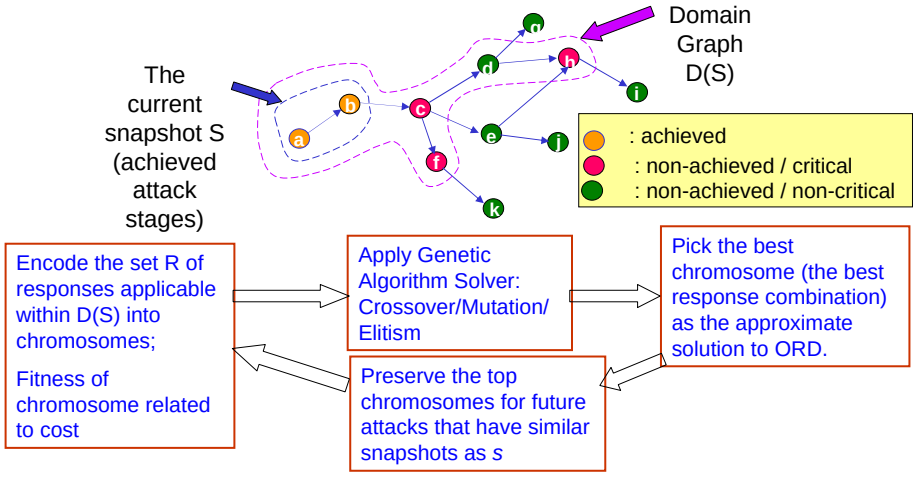
Domain Graph

- Our solution is to limit the response search space for a snapshot *s* to a subset of (*G*-*s*), namely the **domain graph** *D(s)*
- *D(s)* includes critical nodes from I-GRAPH
  - A node *n* is critical if $CCI_n * IV_n$ is greater than a given threshold



Approximate O.R.D. with Genetic Algorithm

- Optimal Response Determination is proved to be NP-hard by mapping the Set Covering Problem to it

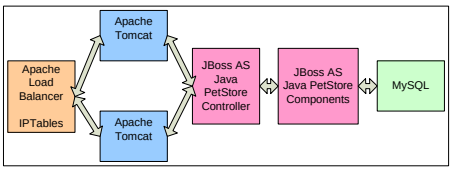


Genetic Algorithm based Solver for ORD

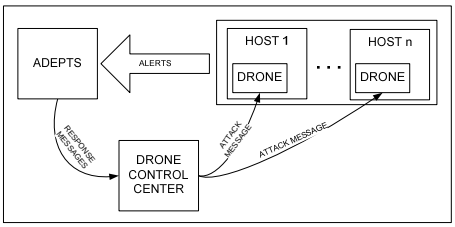
- Why choose G.A.?
  - Derivative based optimization techniques do not work as our objective function is discrete
  - Since exact problem is NP-hard, one has to look for approximate search
- Trade-off between the optimality of the solutions and the computational expense is adjustable by controlling the size of the domain graph and the number of evolutions in the GA based solver
- Good solutions (response combinations) from previous instances are preserved into the chromosome population for future instances of similar attacks
  - This speeds up the search of good response combinations for future attacks which are similar to the current one

Experiment and preliminary results

- The testbed
  - A three-tier eCommerce system as the reference basis for constructing attack scenarios.

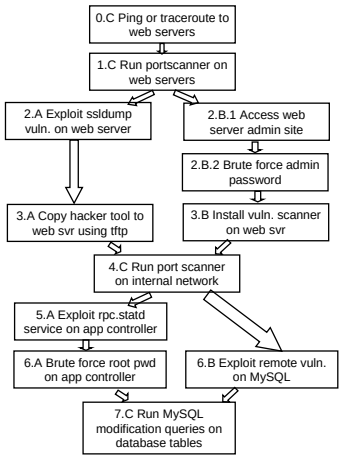


  - A drone based testing framework



Experiment and Preliminary results

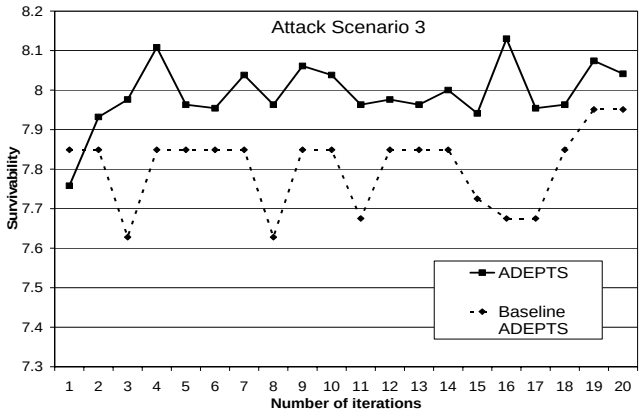
- Attack scenarios



Attack scenarios 3 and 4, used for experimental evaluation. Boxes with A and B denote the stages for scenario 3 and 4 respectively, while C denotes stages common to both.

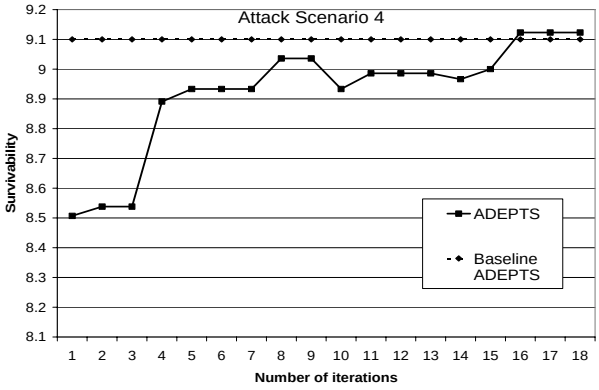
Experiment – Comparison against Baseline

- ADEPTS vs. the baseline (locally optimized responses)



Experiment – Incorrect Initial Conditions

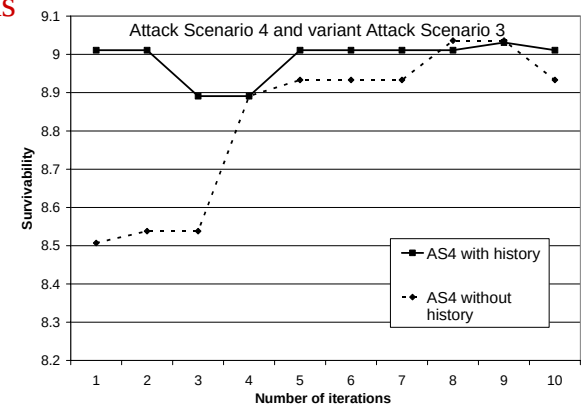
- ADEPTS where initial settings (effectiveness of responses, IV values, etc.) are incorrect, say due to inexperienced sysadmin



After 16 iterations of the attack, the effect of incorrect initial parameters disappears

Experiment – Learning from Similarity

- Utilizing the information (chromosomes) from similar attacks



For one case, AS4 is run after running its variant AS3 and generating history. For the other, AS4 is run without such history. It takes 8 iterations for the latter to catch up.

Wrap-up

- Defined a framework to reason about optimality of intrusion responses in distributed systems
- The framework is implemented using genetic algorithm based solver since exact solution is NP-hard
- Experiments with real multi-stage attacks indicate that globally optimizing response choices is beneficial
- What's coming next:
  - How can the number of evolutions of the GA be determined?
  - What happens if some detectors are misconfigured and attack phases are incomplete?
  - How to handle incomplete I-GRAPHs?