# Achieving High Survivability in Distributed Systems through Automated Intrusion Response

**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

Joint work with: Yu-Sung Wu, Bingrui Foo, Matt Glause, Yu-chun Mao, Gunjan Khanna (Students); Eugene H. Spafford (Faculty)

Work supported by:
NSF, Indiana 21st Century, IBM, Avaya

# A Brief History of Me

- 1996-2001: MS/PhD student in Computer Science, University of Illinois at Urbana-Champaign
  - Advisor: Ravi Iyer and Zbigniew Kalbarczyk
  - Thesis: Distributed Error Detection in Software Implemented Fault Tolerance Middleware (Chameleon)
- 2002-present: Assistant Professor in the School of Electrical and Computer Engineering
  - Courtesy Appointment in Computer Science
  - Group with 6 PhD students
- Attended and presented at FTCS/DSN in 1999, 2002-now
  - PDS PC member 2003-now

PURDUE
U N I V E R S I T Y

# Research Focus

- Payload system: Distributed system of interacting services

- Automated diagnosis
  - Accidental failure that can cascade
  - Diagnosis through monitoring inter-service messages

- Automated containment and response
  - Malicious failure
  - Multi-stage failure

- Concrete problem areas
  - Distributed e-learning application (Purdue)
  - Distributed e-commerce application (IBM)
  - Distributed VoIP application (Avaya)

# Intrusion Response in Distributed Systems: Basics

- ## Distributed System
  - Interconnected entities and services
    - Example: An eCommerce system (customers, bank, warehouse, database, web applications, and etc.)
  - A favorable target of cyber attacks and insider attacks
  - Denial-of-service, Vandalizing, Stealing information, Illegal transactions

- ## Challenges in protecting distributed systems
  - Interactions between services allow "infection" to spread
  - Heterogeneous services, some of them black box
  - Need to limit impact to normal transactions or normal users

# Existing IRS

- Manual: Typically requires the administrator to check the detection log files, identify the compromised region, and enforce the containment
  - Not automatic. Long reaction time

- Local response: Response taken at the site of detection
  - Example: Snort cutting connection from suspicious host
  - Possibly too late and infection has spread

- Static response: Pre-configured table from detector alarm to response
  - Example: RBAC systems
  - Limited applicability to simple systems
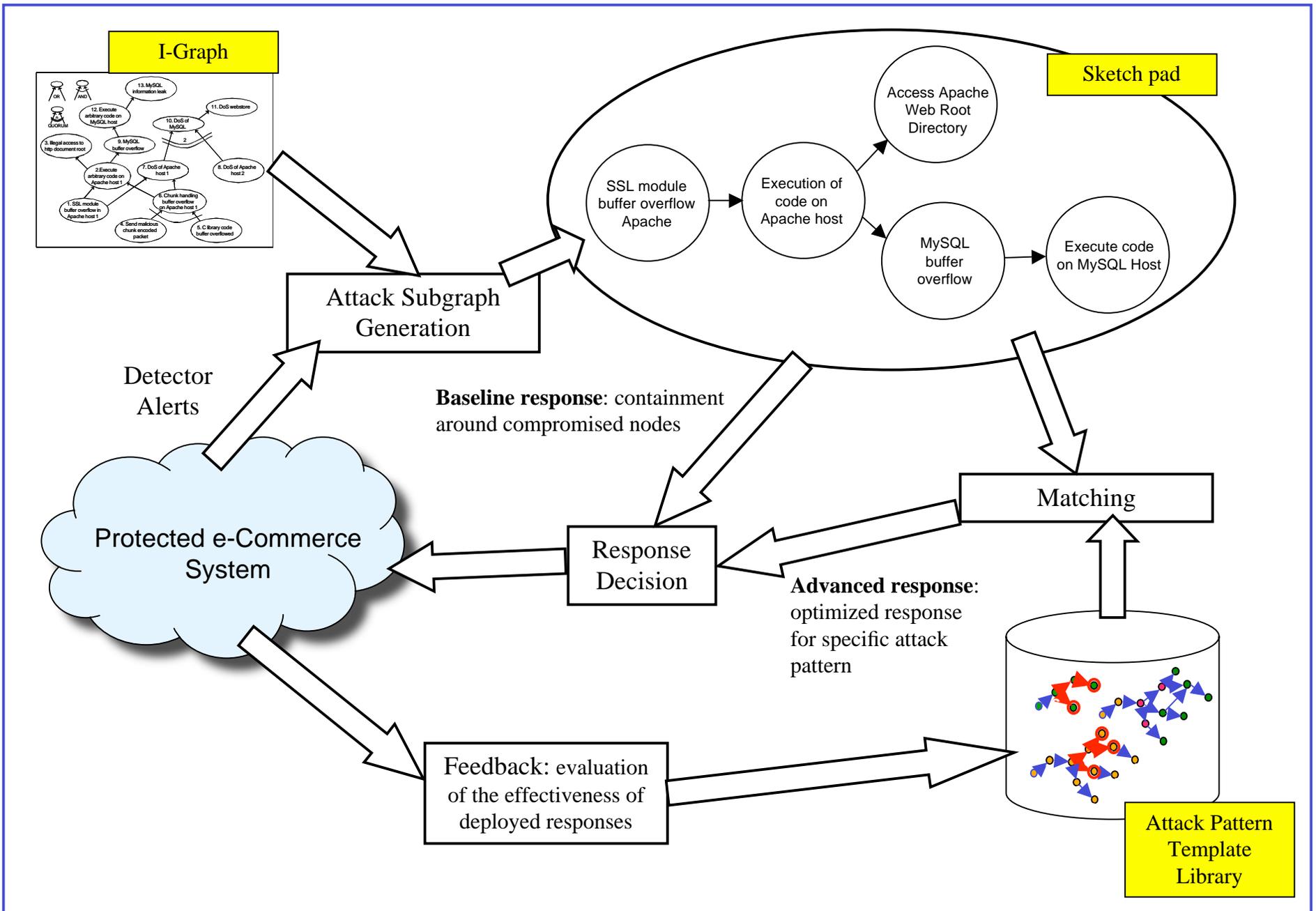
PURDUE
UNIVERSITY

# State-of-the-Art

- Dynamic response creation
- Responses created based on various factors
  - Virulence of the attack
  - Certainty that an attack is in progress
  - Examples: CSM, Emerald
- Attacks are verified using network topology
- Alert fusion: Multiple alerts are aggregated to determine the attack and response is taken for the attack

PURDUE
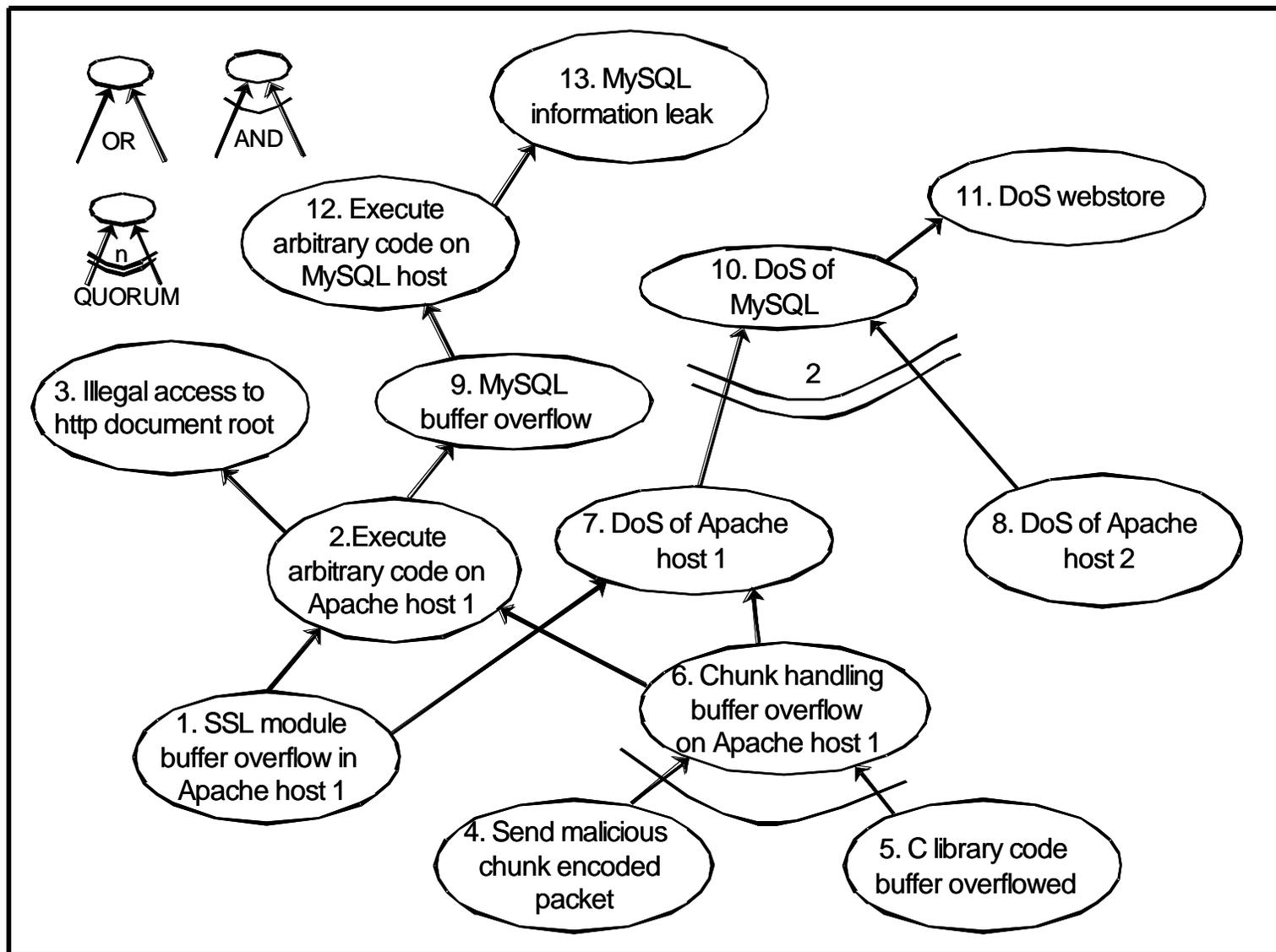UNIVERSITY

# Design Goals/Challenges

- Provide online response and containment while the attack is in progress

- Maximize combination of survivability of the system and resilience to future attacks

- Handle unanticipated attacks

- Work with incomplete knowledge of vulnerabilities and attack paths

- Work with imperfect detectors

# Design Approach

- We know the (legitimate) interactions of services in the system
- We know the manifestations of the attack on the service, but not the attack path
- Use a knowledge representation for the attack goals, rather than the attack path
- Evaluate suitability of response based on disruptivity of response, effectiveness of response to prior attacks of this type, likelihood that attack is in progress
- Build in capability to leverage expert or administrator knowledge and regulatory policies
- Result: ADEPTS – a system for adaptive intrusion response and containment

**I-Graph**

**Sketch pad**

Access Apache Web Root Directory

SSL module buffer overflow Apache → Execution of code on Apache host → Access Apache Web Root Directory

MySQL buffer overflow → Execute code on MySQL Host

Attack Subgraph Generation

Detector Alerts

**Baseline response**: containment around compromised nodes

Protected e-Commerce System

Response Decision

Matching

**Advanced response**: optimized response for specific attack pattern

Feedback: evaluation of the effectiveness of deployed responses

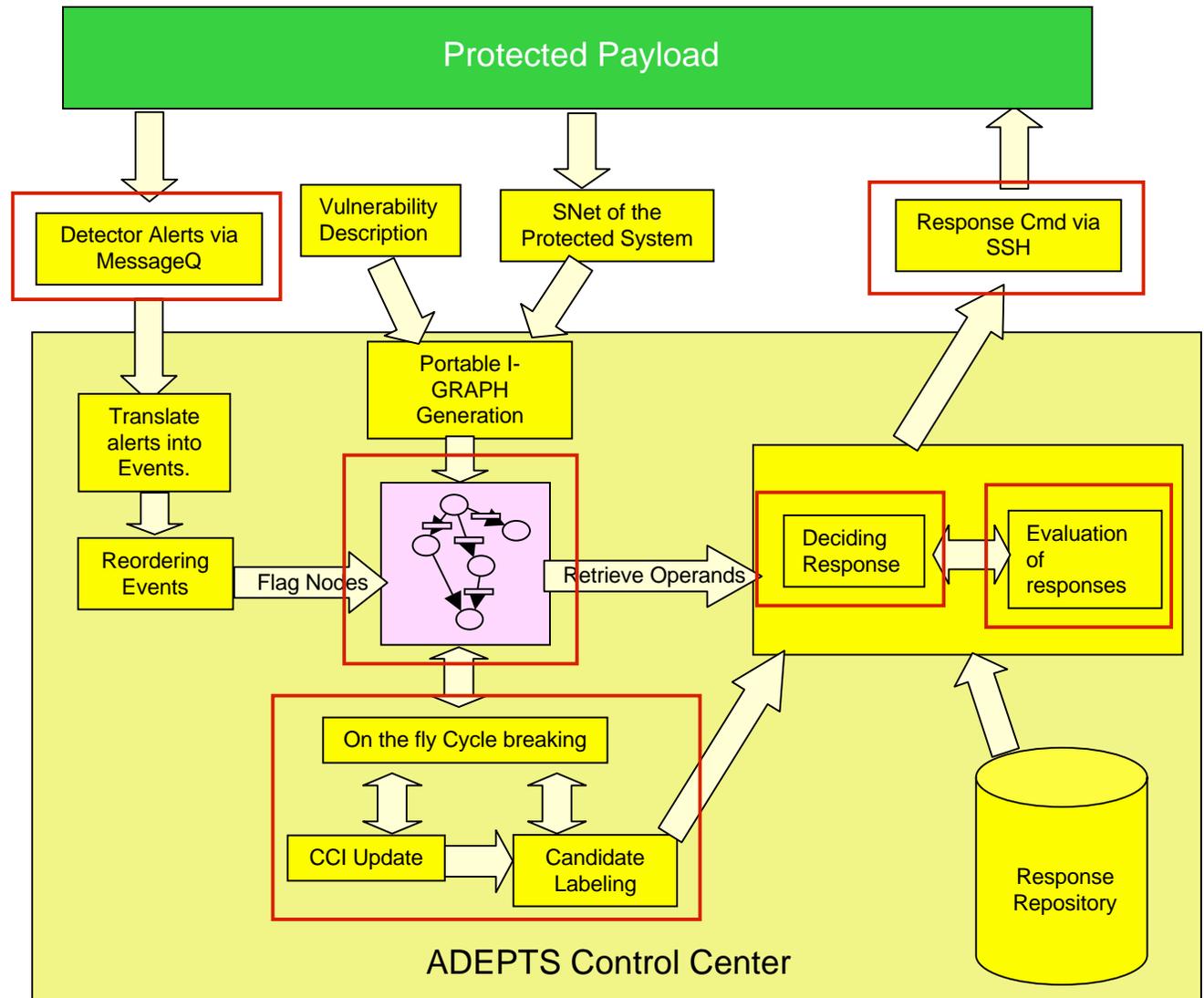**Attack Pattern Template Library**

# ADEPTS Knowledge Representation: I-GRAPH

PURDUE
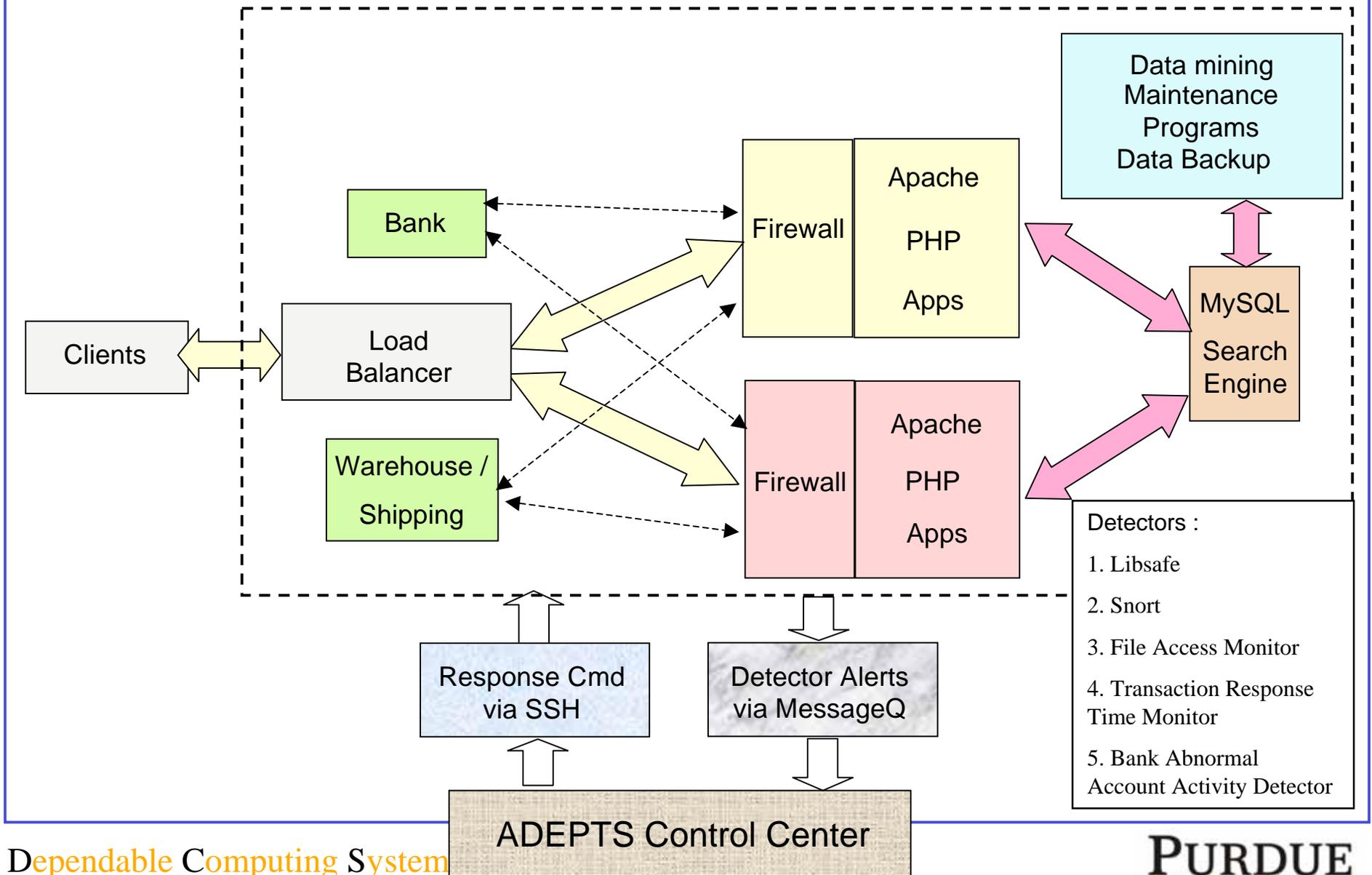UNIVERSITY

# Process Flow & Architecture View of ADEPTS

1. Detection framework flags alerts
2. I-GRAPH parameters updated
3. Determine locations to take responses
4. Available responses determined based on attack parameters and I-GRAPH
5. Responses chosen and deployed
6. Evaluation of deployed responses



Protected Payload

Detector Alerts via MessageQ

Vulnerability Description

SNet of the Protected System

Response Cmd via SSH

Portable I-GRAPH Generation

Translate alerts into Events.

Reordering Events

Flag Nodes

Retrieve Operands

Deciding Response

Evaluation of responses

On the fly Cycle breaking

CCI Update

Candidate Labeling

Response Repository

ADEPTS Control Center

# Handling Unanticipated Attacks

- Unanticipated attack has two manifestations
  1. No detector and therefore no alert, or
  2. Alert generated but no corresponding node in the I-GRAPH
- For (1)
  - Deduce the presence of missed alerts through placement in the I-GRAPH
  - Draw edges between disjoint parts of I-GRAPH
- For (2)
  - Grow the I-GRAPH with *general nodes* (nodes formed based on the alert)
  - Connect general nodes to the rest of I-GRAPH with *general edges*
  - Weight on the general edge indicates likelihood that the alert is part of attack scenario

PURDUE
UNIVERSITY

# Current System



**ADEPTS Control Center**

Clients

Load Balancer

Bank

Warehouse / Shipping

Firewall — Apache PHP Apps

Firewall — Apache PHP Apps

Data mining Maintenance Programs Data Backup

MySQL Search Engine

Response Cmd via SSH

Detector Alerts via MessageQ

Detectors :

1. Libsafe

2. Snort

3. File Access Monitor

4. Transaction Response Time Monitor

5. Bank Abnormal Account Activity Detector

Dependable Computing System

PURDUE
UNIVERSITY

# Survivability

- Survivability is the high level metric – based on two factors
  - Transactions that are supported (in the face of attacks)
  - System level goals that continue to be maintained

| Name | Services involved | Weight |
|------|-------------------|--------|
| Browse webstore | Apache, MySQL | 50 |
| Add to shopping car t | Apache, MySQL | 100 |
| Place order | Apache, MySQL | 100 |
| Charge credit card | Warehouse, Bank | 100 |
| Maintenance work | Variable | 50 |

Illegal read of file (20)

Illegal write to file (30)

Unauthorized credit card charges (80)

Cracked administrator password (90)

Illegal process being run (50)

Corruption of Apache docs/MySQL db (70)

Confidentiality leak of customer info (100)

Unauthorized orders created or shipped (80)

PURDUE
UNIVERSITY

# Response Repository

- Each response has two parts
  - Opcode: Depends on intrusion-centric channel between services
  - Operand: Instantiated from the alerts
- Evaluation of entire response = opcode + operand
  - Wildcards allowed for operands

| Intrusion -centric channel | Opcode | Operand |
|---|---|---|
| General Responses (channel independent) | KillProcess | ProcessID |
|  | Shutdown | Service/ Host |
|  | Restart | Service/Host |
| Shared File Channel | Disable | UserAccount |
|  | DenyFileAccess | FileName UserPrivilege |
|  | DisableRead | FileName UserPrivilege |
|  | DisableWrite | FileName UserPrivilege |

# Experiment #1 Survivability Improvement

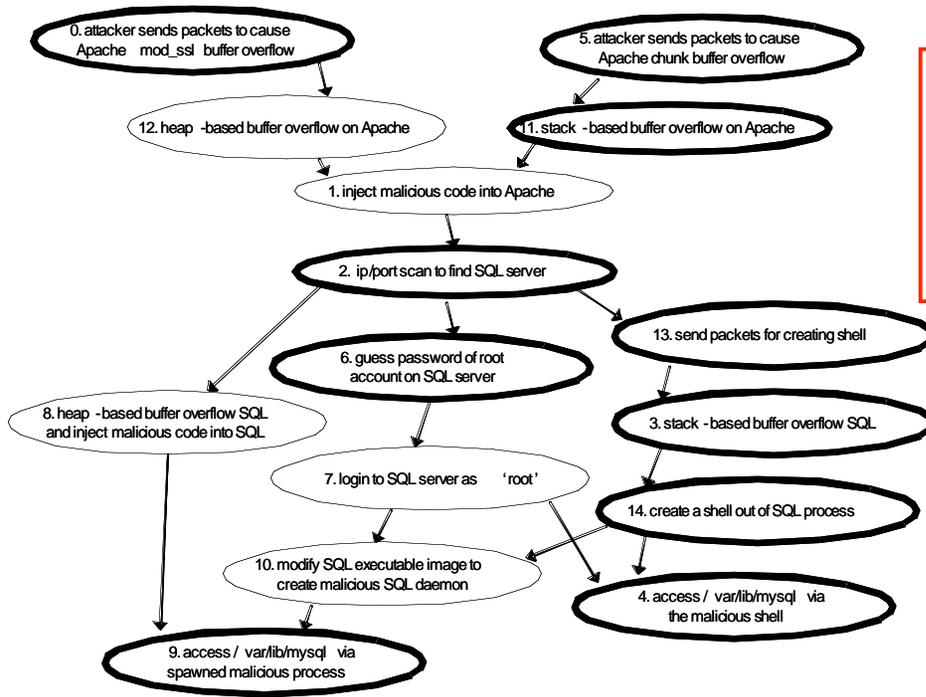Effect of illegal transactions on survivability
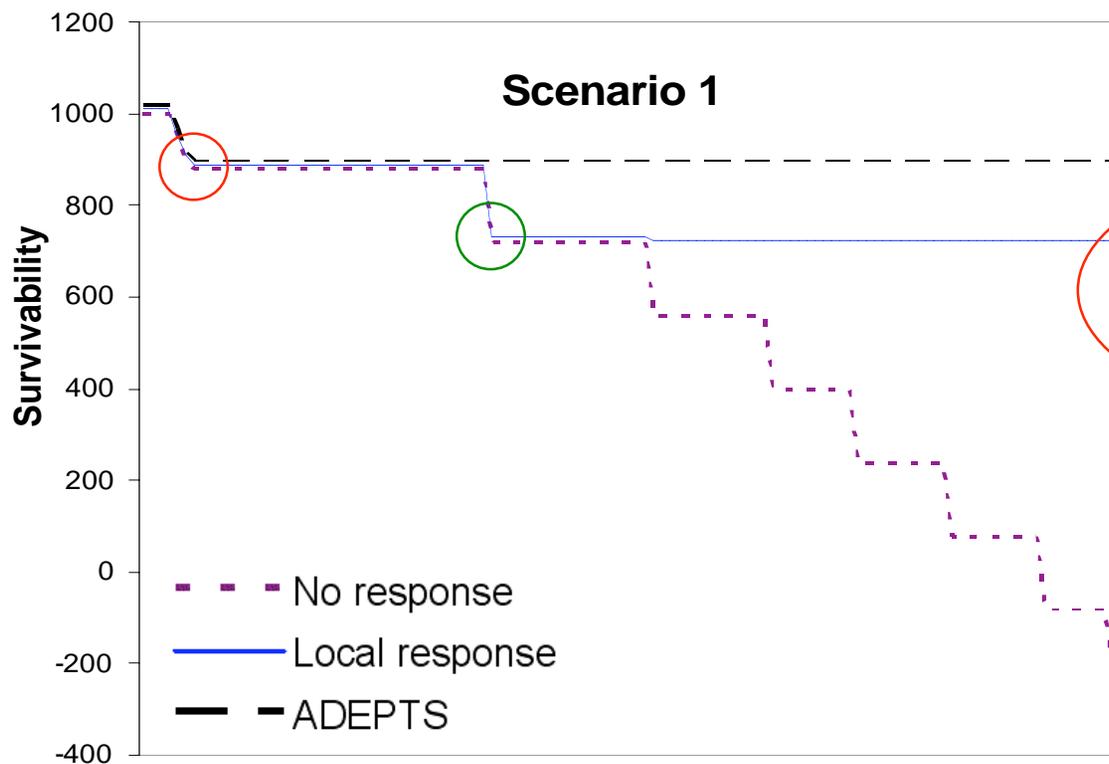


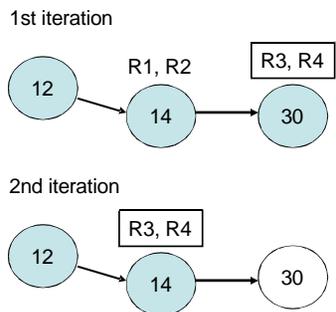| Scenario 1 |
| --- |
| Use php_mime_split (CVE-2002-0081) buffer overflow to insert malicious code into Apache. |
| 'ls' to list webstore document root and identify the script code informing the warehouse to do shipments. |
| Send shipping request to warehouse and craft the request form so that a warehouse side buffer overrrun bug fills the form with a victim's credit card number. |
| Unauthorized orders are made. |

Chart labels: Scenario 1; Survivability; 1200, 1000, 800, 600, 400, 200, 0, -200, -400

Legend: No response; Local response; ADEPTS

PURDUE UNIVERSITY

# Multiple instances of attacks v.s. Survivability

PURDUE
UNIVERSITY

# Handling Unanticipated Attacks



0. attacker sends packets to cause Apache mod_ssl buffer overflow

5. attacker sends packets to cause Apache chunk buffer overflow

12. heap -based buffer overflow on Apache

11. stack -based buffer overflow on Apache

1. inject malicious code into Apache

2. ip/port scan to find SQL server

6. guess password of root account on SQL server

13. send packets for creating shell

8. heap -based buffer overflow SQL and inject malicious code into SQL

3. stack -based buffer overflow SQL

7. login to SQL server as 'root'

14. create a shell out of SQL process

10. modify SQL executable image to create malicious SQL daemon

4. access /var/lib/mysql via the malicious shell

9. access /var/lib/mysql via spawned malicious process

Remove node 12 from the attack graph and run the experiments

**Complete attack graph**

1st iteration: 12 → 14 (R1, R2) → 30 (R3, R4)

2nd iteration: 12 → 14 (R3, R4) → 30
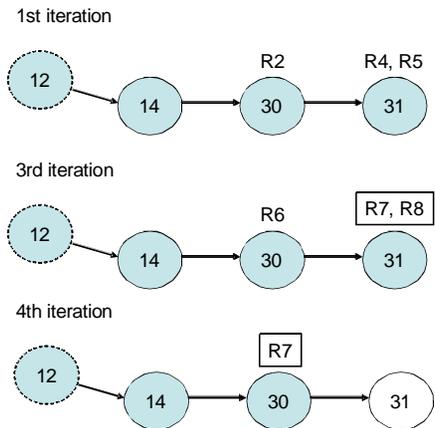
**Incomplete attack graph without capability for unanticipated attack handling**

1st iteration: 12 → 14 → 30 (R2) → 31 (R4, R5)

3rd iteration: 12 → 14 → 30 (R6) → 31 (R7, R8)

4th iteration: 12 → 14 → 30 (R7) → 31
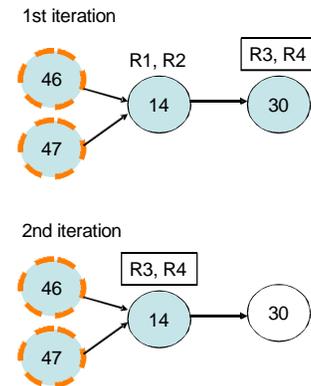
**Incomplete attack graph with capability for unanticipated attack handling**

1st iteration: 46, 47 → 14 (R1, R2) → 30 (R3, R4)

2nd iteration: 46, 47 → 14 (R3, R4) → 30

# Conclusion

- We have a system (ADEPTS) for online reasoning about multi-stage attacks for containment

- ADEPTS uses a knowledge representation of attack consequences and service connections that can be grown

- ADEPTS learns about effectiveness of responses for containing future attacks

- ADEPTS can respond to unanticipated attacks, albeit not optimally

PURDUE
UNIVERSITY

# What's in the works

- Attack template library – attack patterns with pre-configured responses
  - Optimized responses for specific attack manifestations or policy based response
  - ADEPTS can further deduce the potential connections between an unanticipated alert and the other nodes in the I-GRAPH
  - Challenges: How to match with the pattern? How to aggregate multiple patterns? How to move an existing attack to a pattern?
- Synthetic diversity for improving survivability
  - Leverage work on synthetically introducing diversity to create diverse replicas for services
  - Use knowledge of diversity introducing technique to build I-GRAPH

PURDUE
UNIVERSITY

# Publications

- Gunjan Khanna, Saurabh Bagchi, Kirk Beaty, Andrew Kochut, and Gautam Kar, "Providing Automated Detection of Problems in Virtualized Servers using Monitor framework," In the Workshop on Applied Software Reliability (WASR), held with the IEEE International Conference on Dependable Systems and Networks (DSN), 6 pages, June 25-28, 2006.
- Gunjan Khanna, Padma Varadharajan, and Saurabh Bagchi, "Automated Online Monitoring of Distributed Applications through External Monitors," IEEE Transactions on Dependable and Secure Computing, vol. 3, no. 2, pp. 115-129, Apr-Jun, 2006.
- Yu-Sung Wu, Bingrui Foo, Yu-Chun Mao, Saurabh Bagchi, Eugene H. Spafford, "Automated Adaptive Intrusion Containment in Systems of Interacting Services," Accepted to appear in Journal of Computer Networks, special issue on "Security through Self-Protecting and Self-Healing Systems", to appear Fall 2006.
- Bingrui Foo, Yu-Sung Wu, Yu-Chun Mao, Saurabh Bagchi, and Eugene Spafford, "ADEPTS: Adaptive Intrusion Response using Attack Graphs in an E-Commerce Environment," In the International Conference on Dependable Systems and Networks (DSN), pp. 508-517, Yokohama, Japan, June 28 - July 1, 2005.

PURDUE
UNIVERSITY