
APOD

Applications that Participate in their Own Defense

Franklin Webber (BBN):

APOD Overview

Doug Reeves (NCSU):

SE-RSVP Overview

Bill Nelson (BBN):

Experimental Results

John Clem (Sandia):

Attacks and Red Team Observations

Fault Tolerant Networks PI Meeting, Newport RI

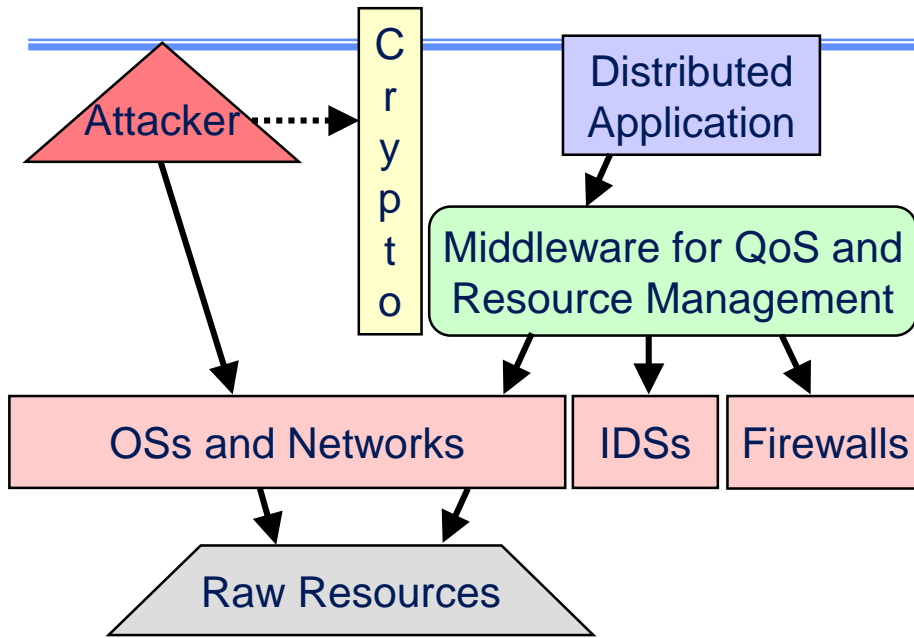
25 July 2002

APOD Overview

Franklin Webber, BBN Technologies
franklin@eutaxy.net

Fault Tolerant Networks PI Meeting, Newport RI
25 July 2002

Applications That Participate in Their Own Defense



New Ideas

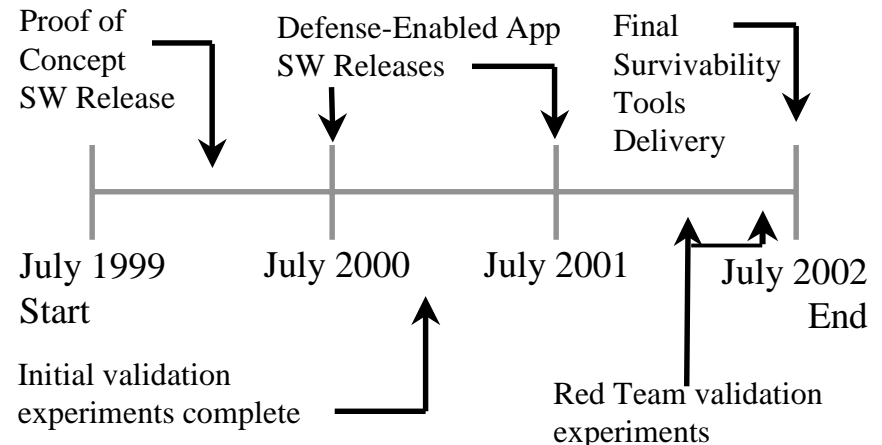
- Involve application software in its own defense
- Use quality-of-service (QoS) management to aid intrusion diagnosis and to resist attacks
- Use middleware to coordinate basic intrusion response mechanisms according to overall defense strategy for application
- Provide tools to help developers configure application defenses
- Use Red Team experiments to measure survival value of adaptive defense

Impact

- Systems with more survivability, built with less effort
- Defense of critical applications without need for perfect security in infrastructure; cost-efficient: can use COTS
- A set of example defense-enabled applications
- A collection of validated defense strategies that application developers and/or security specialists can apply

Principal Investigator: Franklin Webber

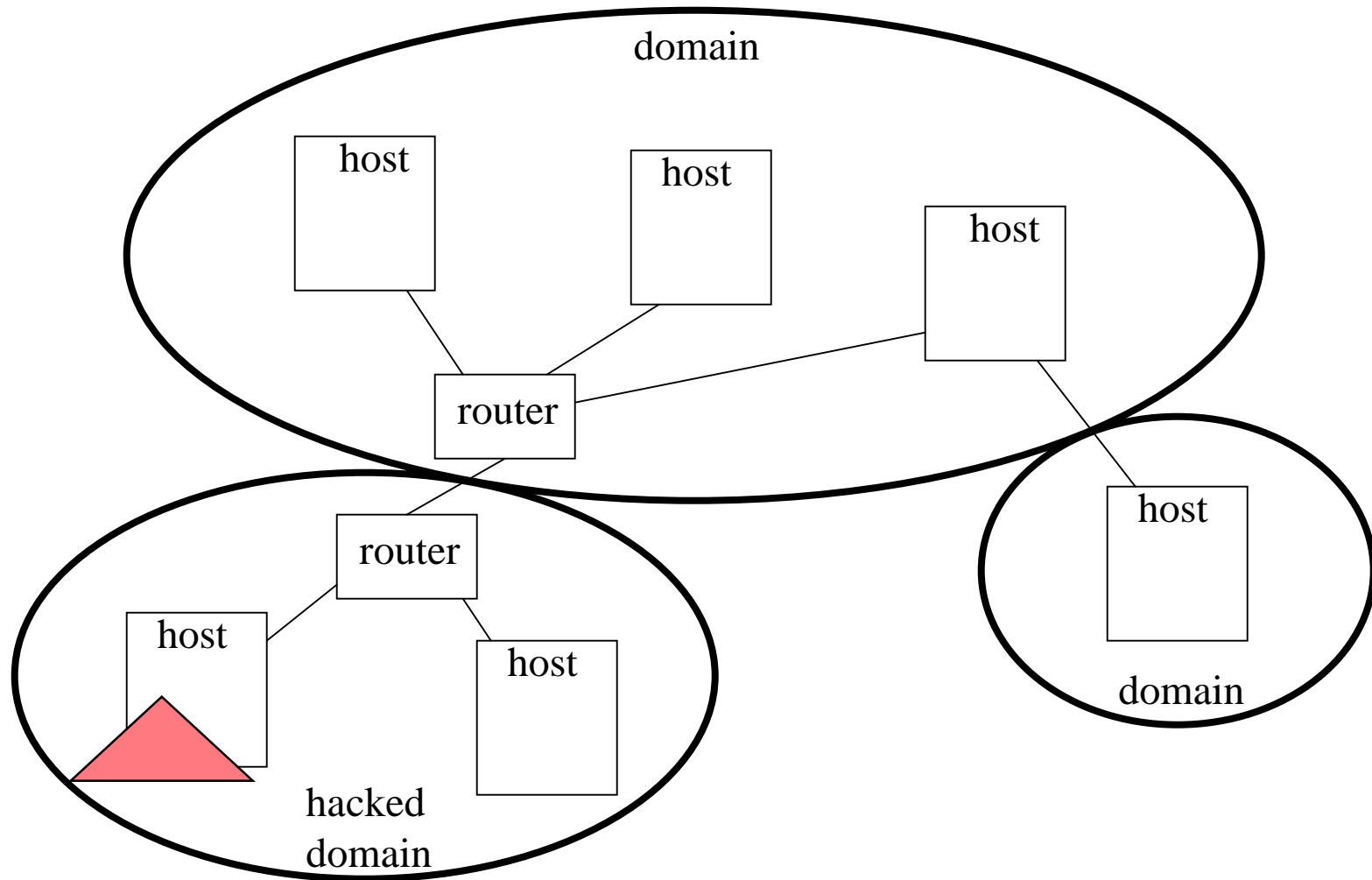
Schedule



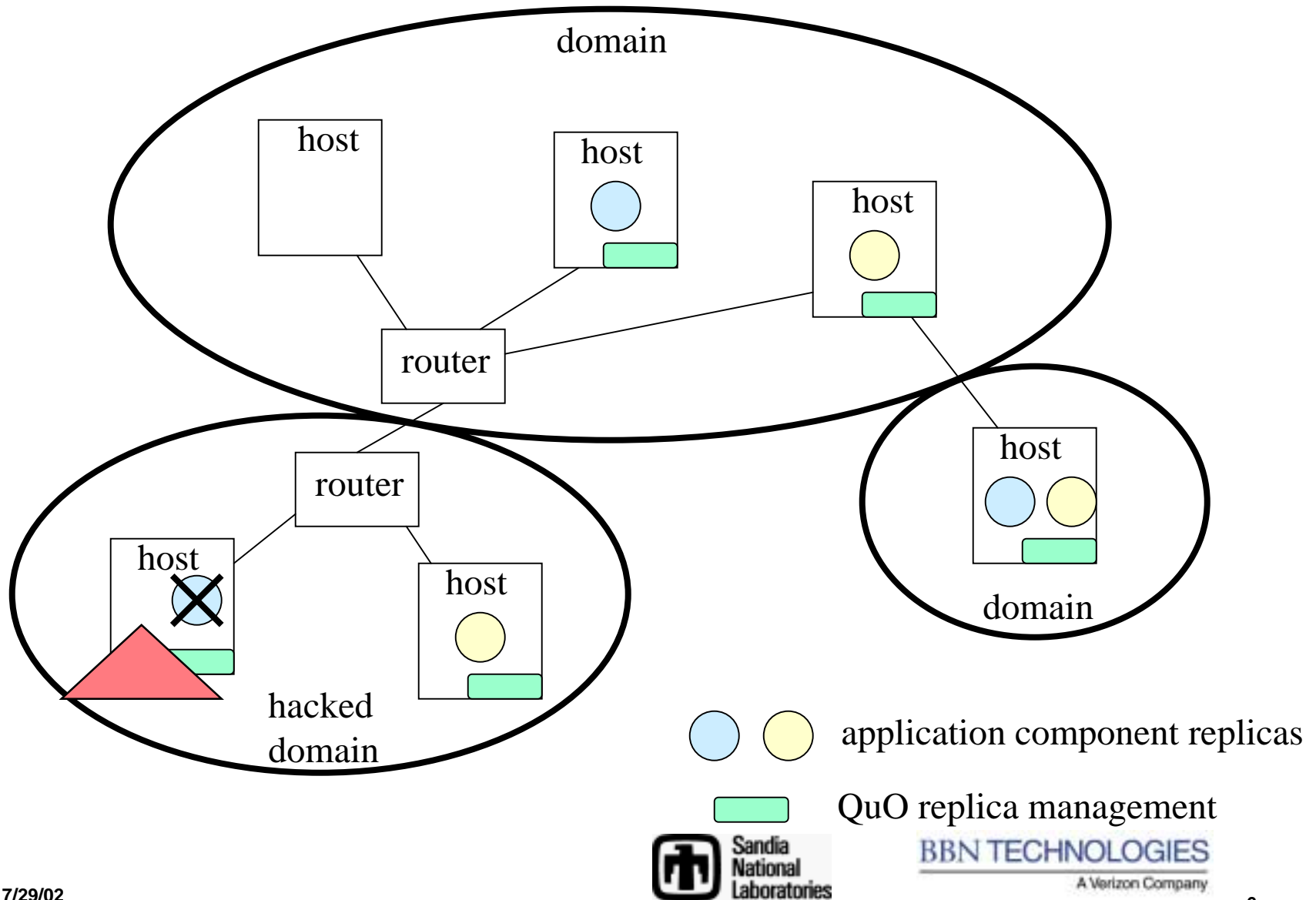
Implementing Defenses in Middleware

- APOD defenses are implemented using the QuO (Quality-of-Service for Objects) adaptive middleware developed at BBN:
- for simplicity:
 - ◆ QoS concerns separated from functionality of application
- for practicality:
 - ◆ Middleware defenses will augment, not replace, defense mechanisms available in lower system layers
- for uniformity:
 - ◆ Advanced middleware such as QuO provides a systematic way to integrate defense mechanisms
- for reusability:
 - ◆ Middleware can support a wide variety of applications

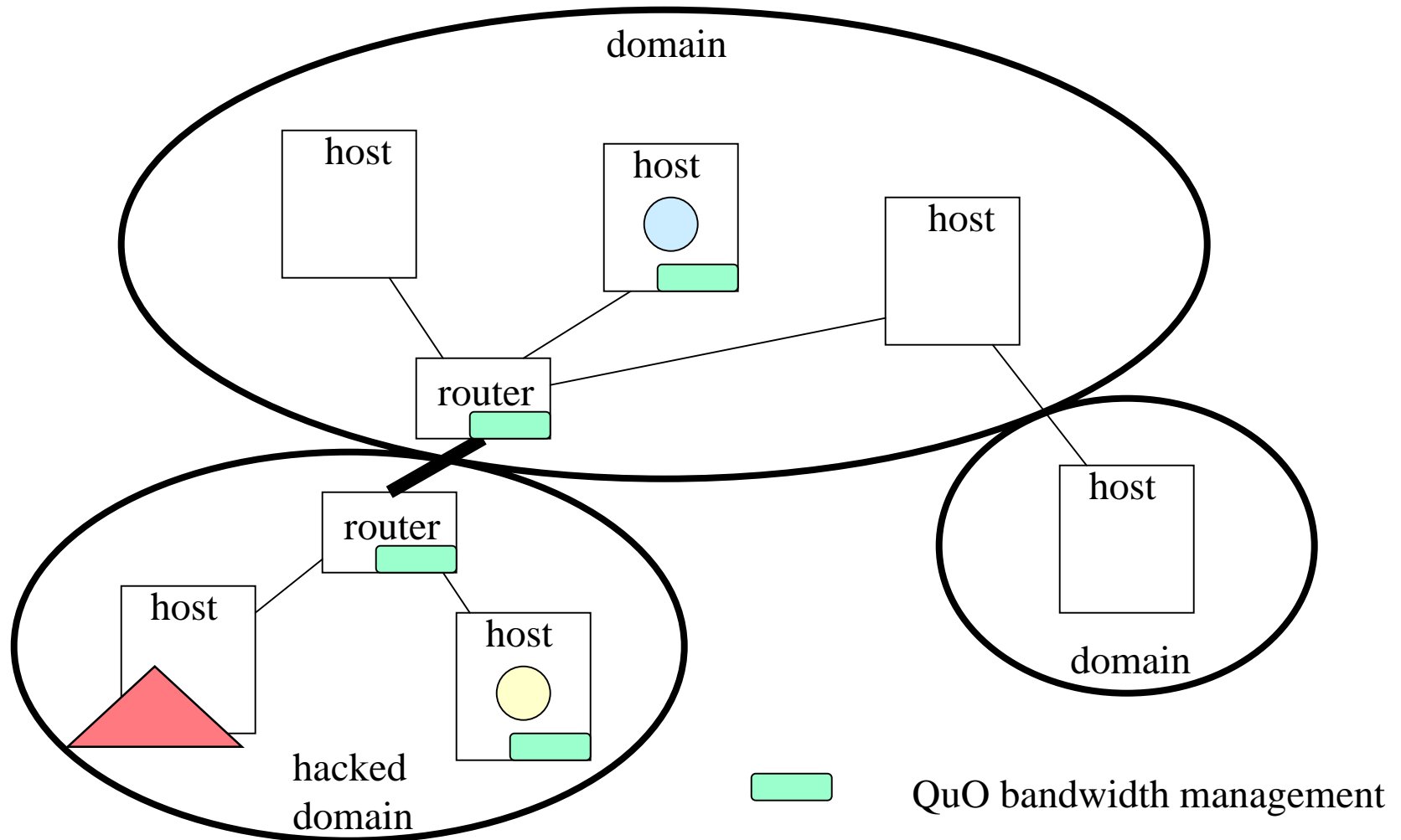
Security Domains Limit the Damage From A Single Intrusion



Replication Management Can Replace Killed Processes



Bandwidth Management Can Counter Flooding Between Routers



Other APOD Defensive Adaptations

- Dynamically configure firewalls to block traffic
- Dynamically configure routers to limit traffic
- Dynamically change communication ports
 - ◆ not used in Red Team experiments
- Dynamically change communication protocols
 - ◆ not used in Red Team experiments

Defense Strategy

- Use QuO middleware to coordinate all available defense mechanisms in a coherent strategy.
- Our best current strategy has two parts:
 - ◆ “outrun”: move application component replicas off bad hosts and on to good ones
 - pure replacement only (a policy decision)
 - ◆ “contain”: quarantine bad hosts and bad LANs by limiting or blocking network traffic from them and, within limits, shutting them down
 - triggered by intrusion detectors, flooding, other anomalies

Red Team Experiments

- Two experiments:
- First experiment prohibited Red Team from using network flooding.
- Second experiment allowed flooding; defense used a security-enhanced RSVP plus an improved 'containment' strategy.
- Both experiments were preceded by a 'whiteboard' session with the Red Team to look for conceptual weakness in defenses.
- Both experiments gave the Red Team "root" privilege on a subset of application hosts.

Conclusions I

- A defense of critical distributed applications can be organized in middleware and should include several complementary defense mechanisms, each of which adapts to changes in a different QoS aspect.
- The Red Team experiments were the most valuable part of our validation effort.

Conclusions II

- Is it worthwhile to defense-enable a critical application?
- The attacker may be able to 'outrun' the defense if every security domain is vulnerable. We do not know how likely this may be in the real world.
- To minimize this possibility, a defense-enabled application must be fielded on a heterogeneous set of platforms.
- An attack that infiltrates only a subset of security domains will face substantial resistance from a defense-enabled application.

APOD-1

Experimental Results

William H. Nelson, BBN Technologies
wnelson@bbn.com

Fault Tolerant Networks PI Meeting, Newport RI
25 July 2002

Overview

- **Experiment design** (3 slides)
 - ◆ Experiment methodology
 - ◆ Hypothesis, flags, & metrics
 - ◆ Topology & operational components
- **Experimental results** (5 slides)
 - ◆ Scope - summary of runs (table)
 - ◆ Baseline behavior plot
 - ◆ Sample attack #1 plot
 - ◆ Sample attack #2 plot
 - ◆ Time to denial (bar graph)
- **Summary** (2 slides)
 - ◆ Conclusions
 - ◆ Lessons learned

Schedule

- Mid Oct. 2001
 - ◆ Evaluate APOD for experiment
- Nov. 2001
 - ◆ Define experiment, draft plan
- Dec. 2001
 - ◆ Hold White Board (12,13th)
- Jan. 2002
 - ◆ Final plan published
 - ◆ Integration & testing
- Feb. 2002
 - ◆ Begin execution
- Mar. 2002
 - ◆ Finish execution
 - ◆ Data reduction & analysis
 - ◆ Hot Wash (26th)
- Final report published (6-4-02)

People

- Blue Team (BBNT)
 - ◆ Franklin Webber
 - ◆ Partha Pal
 - ◆ Michael Atighetchi
 - ◆ Chris Jones
 - ◆ Paul Rubel
- White team (BBNT)
 - ◆ Ken Theriault
 - ◆ Bill Nelson
 - ◆ Wilson Farrell
 - ◆ Lyle Sudin
 - ◆ Marla Shepard
- Red Team (Sandia)
 - ◆ Steve Kaufman
 - ◆ Dino Dai Zovi

Hypothesis, flags, & metrics

- Hypothesis

- ◆ Top-level: APOD improves immunity to cyber attacks relative to systems that do not use APOD

- APOD improves immunity to availability attacks relative to systems that do not use APOD
- APOD improves immunity to availability attacks at an acceptable cost to the defender

- Opponent flags

- ◆ Degrade image service (e.g. increase latency)
- ◆ Deny availability of images to users

- Principal metrics

- ◆ Image serving latency across a suite of image clients (delay flag)
- ◆ Fraction of image requests that do not succeed (deny flag)

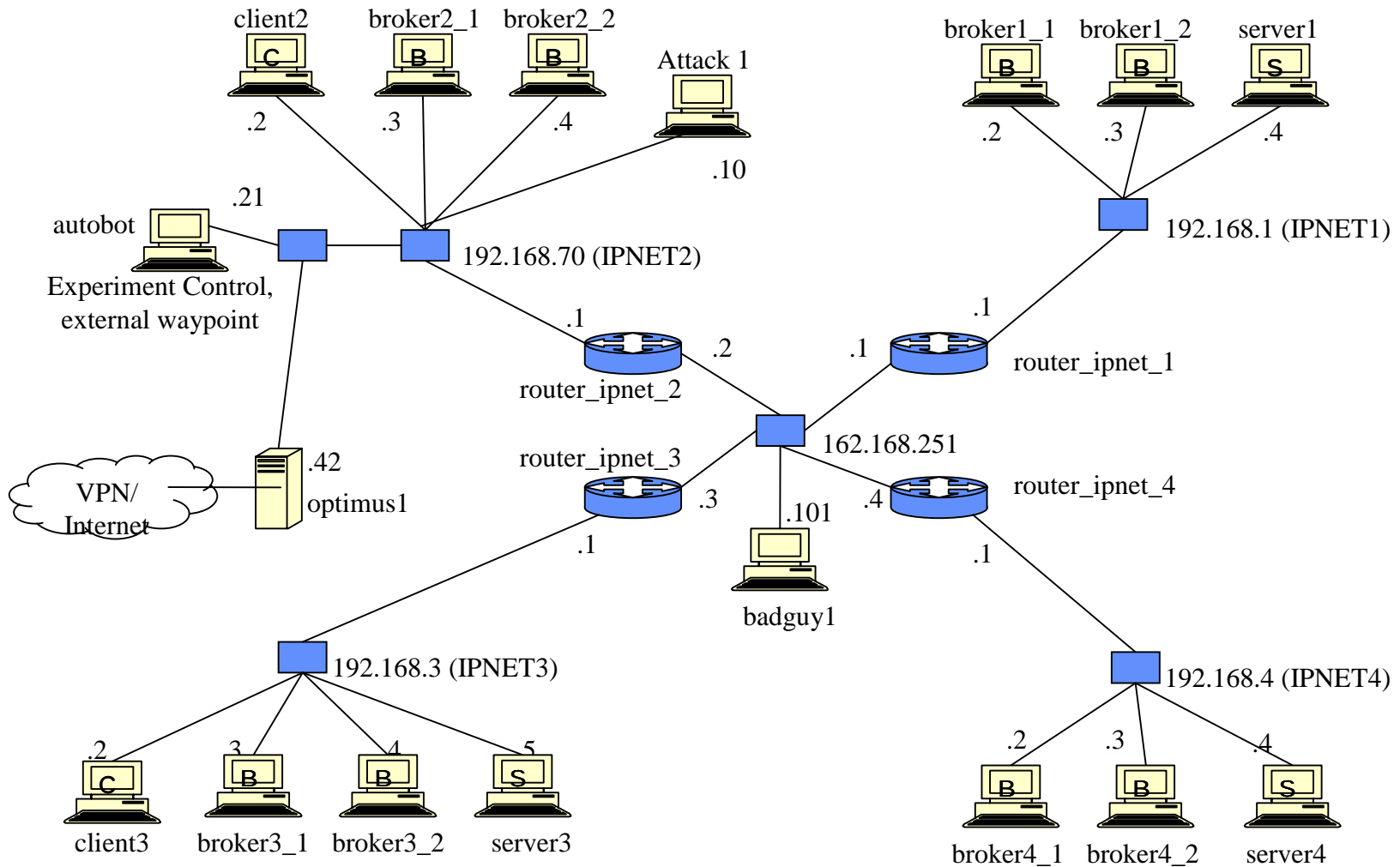
Experiment Methodology

- Utilize a laboratory test network
 - ◆ Networked image serving system (brokers, image servers, and clients)
 - ◆ Mission traffic (images)
 - ◆ Defense (APOD replication/replacement of compromised brokers)
- Controlled introduction of attacks on brokers
- Scripted clients repeatedly request images from a data base and accumulate latency and delivery statistics

APOD Present	Attacks	Purpose
No	No	Baseline metric measurement
No	Yes*	Baseline attack performance
Yes	No	Performance cost of APOD
Yes	Yes	Defense performance of APOD

* Attacks do not target APOD; difficult to compare with results of APOD-specific attacks

Topology



Experiment scope

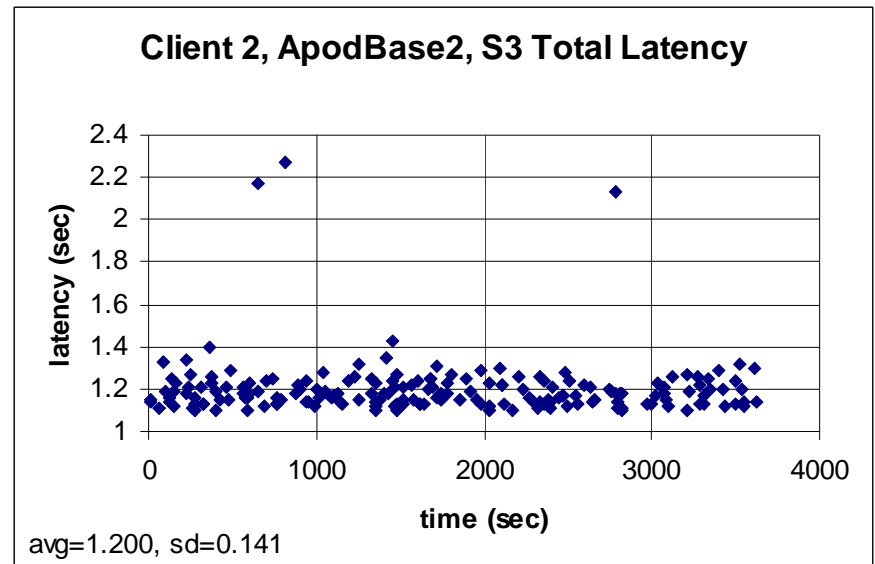
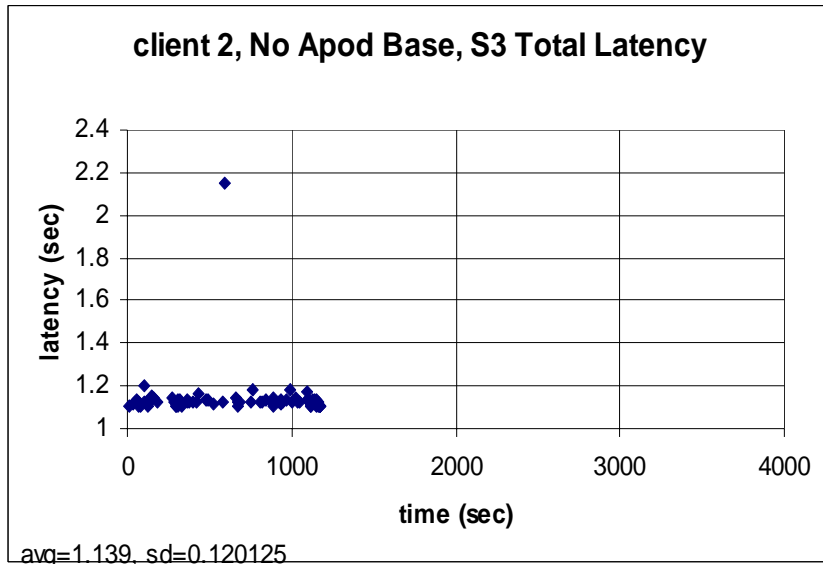
APOD	Attack Used	System Configuration					
		Number of runs	Client. Timeout (sec)	Normal Period (sec)	Number of Clients	Flood Protection	Denial
No	None	1	20	16	2	N/A	No
	TcpKill	1	20	16	2	N/A	Yes
Yes	None	3	20	16	2	N/A	No
		1	20	16	8	N/A	No
	Isolate_Scan	1	20	16	2	N/A	No
	ScanAll	3	20	16	2	N/A	No
	Scan_Isolate_Flood	3	20	16	2	N/A	1 of 3
	Scan_Flood_Isolate	2	20	16	2	N/A	Yes
		1	10	16	4	N/A	Yes
		1	10	16	8	N/A	Yes
		1	2.5	16	2	N/A	No
		1	5	16	2	N/A	No
		3	10	16	2	N/A	1 of 3
		1	40	16	2	N/A	Yes
		1	80	16	2	N/A	Yes
		1	20	4	2	N/A	No
		1	20	8	2	N/A	Yes
		1	20	32	2	N/A	No
		3	20	64	2	N/A	1 of 3
		1	20	16	2	Yes	No
	TcpKill *	1	20	16	2	Yes	Spikes
					8		
Scan_TcpKill_Isolate *	1	20	16	2	Yes	Yes	
					No		

Baseline system behavior (no attacks)

APOD-1 introduces some overhead. The average latency increases ~5%

Client 2, No APOD Base (seconds)			
Server	Average	Std. Dev.	# spikes/tot
3	1.139	0.120	0/175
1	1.126	0.018	0/82
4	1.137	0.127	0/61

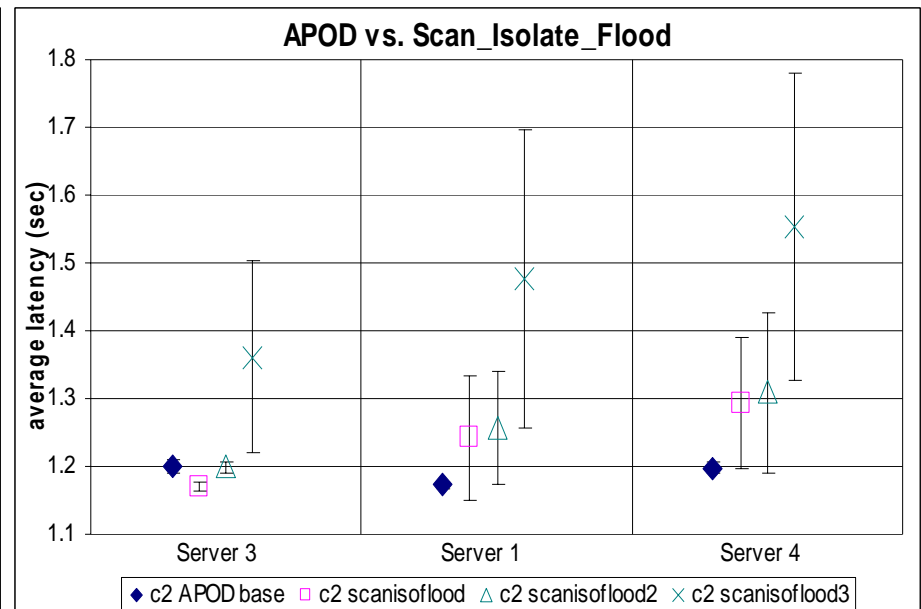
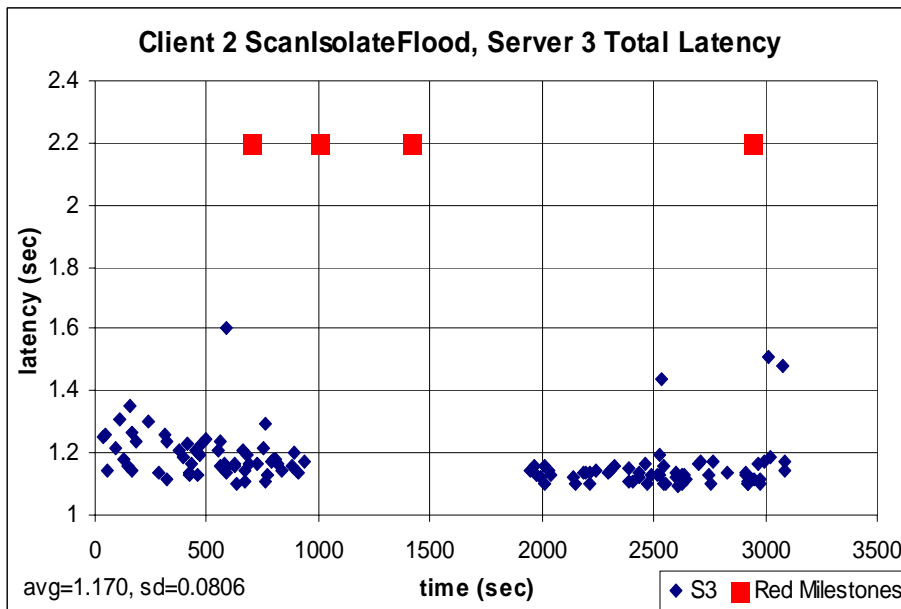
Client 2, APOD Base (seconds)			
Server	Average	Std. Dev.	# spikes/tot
3	1.200	0.141	0/189
1	1.172	0.064	0/196
4	1.198	0.132	0/211



APOD-1 vs. Scan_Isolate_Flood attack

- A denial in image service occurred while clients could not access a broker.

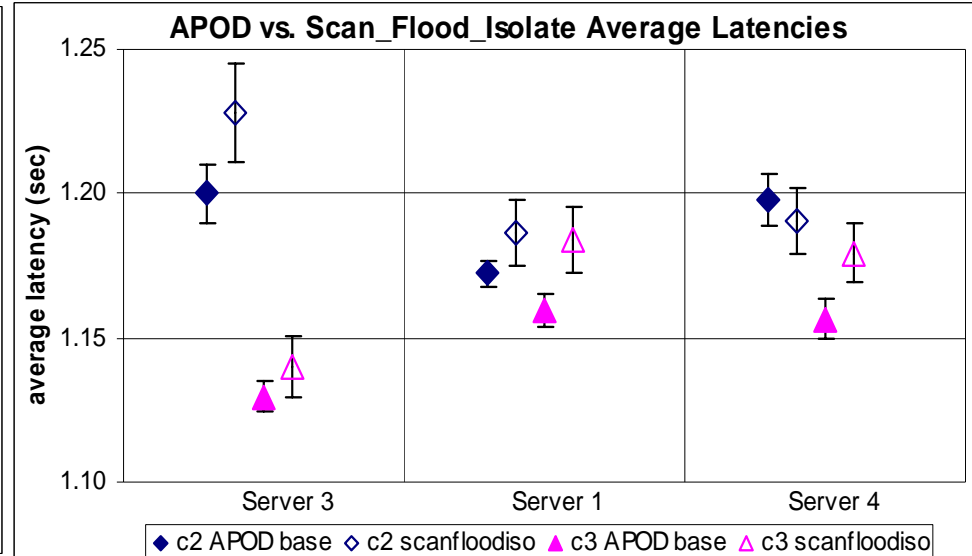
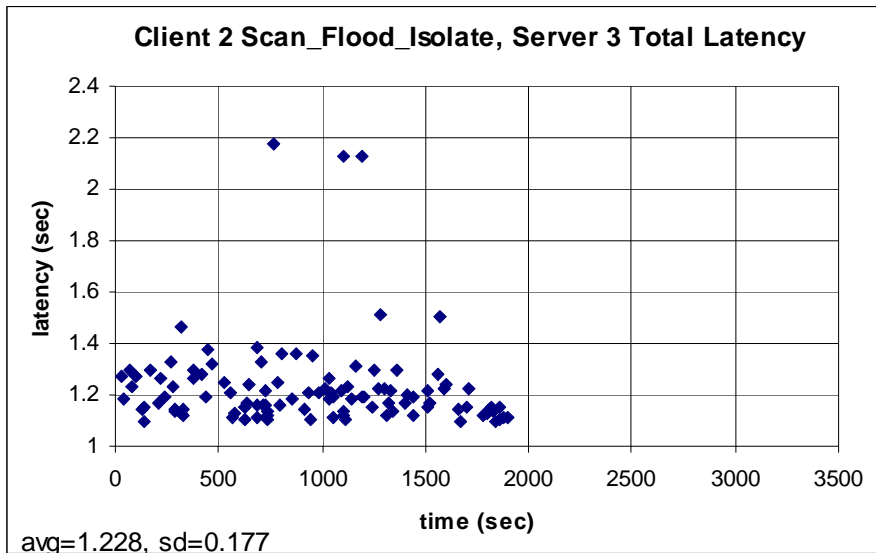
Client 2, Scan_Isolate_Flood Attack			
Server	Average (no spikes)	Std. Dev.	# spikes/tot
3	1.170 sec	0.081 sec	0/121
1	1.151	0.046	2/112
4	1.294	1.059	0/123



APOD-1 vs. Scan_Flood_Isolate attack

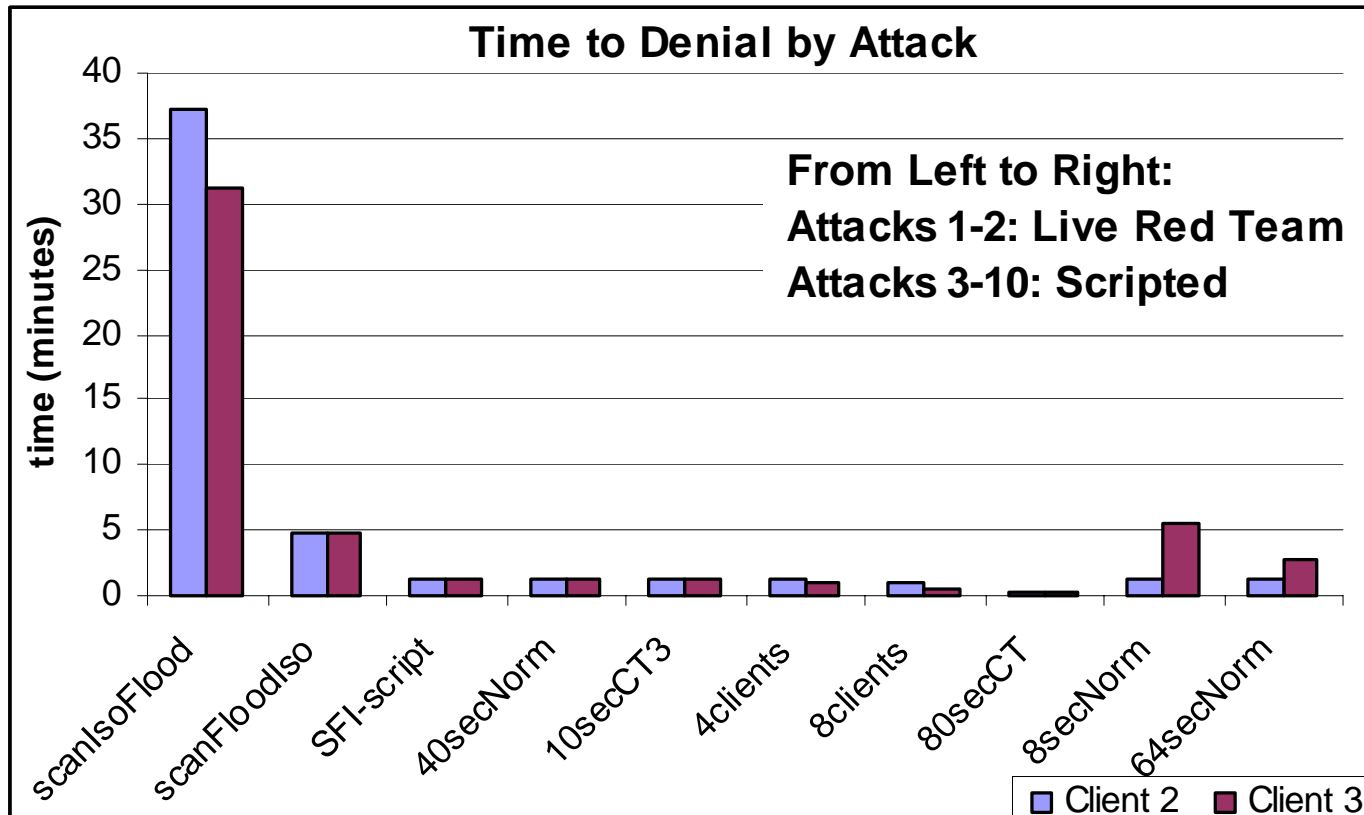
- Both clients stopped receiving images. This run went for 3454 seconds.

Client 2, Scan_Flood_Isolate			
Server	Average (no spikes)	Std. Dev	# spikes/tot
3	1.228 sec	0.177 sec	0/109
1	1.186	0.117	0/103
4	1.190	0.118	0/105



Time to Denial (TTD)

- Scripted Attacks decrease the Time to Denial
 - ◆ Live Red Team, <TTD> = 19.5 minutes
 - ◆ Scripted attacks, <TTD> = 2.56 minutes
- Some changes in APOD parameters increase Time to Denial (i.e. attacks 9-10)



Conclusions

- **Broker replication (with a dynamic firewall) provides effective defense against attacks on image brokers**
 - ◆ APOD-1 kept the image-serving system up ~ 66% of attacks
 - But, sustained broader attacks still succeeded (deny ~ 33% of attacks)
 - ◆ When the deny flag was achieved, APOD-1 prolonged image serving system's life
 - ~ 20 minutes (live Red Team)
 - ~ 2.5 minutes (scripted attacks)
- **Performance (attack dependant)**
 - ◆ Isolated point latencies – due to temporary broker outages
 - Low attack rate - no measurable effect
 - High attack rate - noticeable increase in average latency
 - ◆ Image service denial
 - Mildly aggressive attack - outage until APOD recovers
 - Aggressive / persistent attack - outage until manual intervention
- **Cost**
 - ◆ Added complexity and implementation effort (extra machines)
 - ◆ APOD-1 mechanism had little operational overhead
 - With APOD-1, no attacks: ~ 5% increase in latency
 - With APOD-1, with attacks: ~ 0.5% additional latency increase

APOD-1

Attacks and Red Team Observations

John F. Clem, Sandia National Labs
jfclem@sandia.gov

Fault Tolerant Networks PI Meeting, Newport RI
25 July 2002

Defined Flags

- Degrade
 - ◆ Introduction of latency to the system
- Deny
 - ◆ Halting image service for 15 minutes

APOD-1 Attack Strategies

Goal	Shutdown brokers	Isolate brokers	Flood brokers	Reset connections
Methods	Spoofed scans to shutdown brokers	ARP-spoof (ARP Cache Poison)	TCP connection flood	TCPKill
Effects	Broker containment rules react to scans	Effectively isolates APOD hosts	Consumes available ports Adds latency	Resets connections
Notes	Attacker can be anywhere in this architecture	Need root on subnet host	Encryption irrelevant	Encryption prevents this

Red Team Attacks

Name	ScanAll	Scan_Isolate_Flood	Scan_Flood_Isolate
Attack Strategy Goal	Shutdown all brokers in one round	Shutdown brokers Isolate client and brokers on IPnet2 Add additional obstacle to remaining brokers	Shutdown brokers Interrupt communications between APOD hosts Isolate remaining
How it works (Method)	NMAP scans using Xmas tree packets and null packets that trigger the IDS.	Multi-stage, sequential attack	Multi-stage, sequential attack
Flag Captured?	No	Yes(Deny)* *1 time in 3 attempts	Yes(Deny)* *11 times in 18 attempts

Red Team Observations

- APOD-1 makes the attacker's job harder
- Good set of basic mechanisms
- Defensive strategies/mechanisms need development
- APOD-1 speed of response very good

Experiment lessons learned

- Experiment was of great value to the APOD technology developers, showing how APOD-1 behaved under actual attack
- Capturing attacks in scripts is essential for repeatability, and testing improvements to the defense in a controlled manner
- APOD-1 made the attacker's job harder
 - ◆ Three techniques had to be used to pin down APOD-1
 - ◆ Order and timing of the attacks was important
 - ◆ Had to be tailored to topology, image serving system, and APOD-1
- It is difficult to instrument a live Red Team with a computer
 - ◆ Automated attacks can be instrumented, run more quickly, and easily repeated
 - ◆ Close coordination with the Red Team is required to capture research, development, and reconnaissance times

More Information

- **APOD Experiment Web Site (on Docushare)**

<https://archive.ia.isotic.org/>

- **To download copies of:**
 - ◆ **APOD Experimental Plans**
 - ◆ **APOD Hot Wash's**
 - ◆ **APOD FTN PI Meeting Presentations**
 - ◆ **APOD Final Reports**
- **Go to:**
 - ◆ **DARPA IA&S**
 - ◆ **FTN/DC Experiments**
 - ◆ **APOD Experiments**