DARPA's Path to Self-Regenerative Systems

June 28, 2002

Operate Through Attacks!!

Dr. Jaynarayan Lala

Information Processing Technology Office

Defense Advanced Research Projects Agency

Cognitive Systems

Systems that know what they're doing

- Able to reason, using substantial amounts of appropriately represented knowledge.
- Learn from their experiences and improve their performance over time.
- Capable of explaining themselves and taking naturally expressed direction from humans.
- Aware of themselves and able to reflect on their own behavior.
- Able to respond robustly to surprises, in a very general way.

BAA 02-21 http://www.darpa.mil/ipto

SELF-REGENERATIVE INFORMATION SYSTEMS

Self-regenerative Systems: Program Goals

- Conceive, design, develop, implement, demonstrate and validate architectures, tools, and techniques that would allow fielding of systems that can learn.
- Develop the basic precepts of representation, reasoning and learning that will form the scientific foundation for all such future systems.

Self-Regenerative Systems: Envisioned Capabilities

- **Learn from its experience so it performs better tomorrow than it did today.**
- **Restore system capabilities to full functionality following an attack event or a component failure.**
- **Analyze a specific failure and diagnose the root cause of the failure.**
  - Determine if an attack focused on exploiting a specific vulnerability or a misconfiguration, or if the failure was caused by an operational error or a fundamental flaw in the architecture.

Self-Regenerative Systems: Envisioned Capabilities

- **Generalize a specific attack event to form a defense against a class of attacks.**
- **Adapt to changes in network traffic due to congestion or denial of service attacks or router and link failures.**
- **Continually create new deceptions as new threats emerge and old techniques become less effective.**
- **Monitor insider activity and develop profiles for appropriate and legitimate behavior.**
  - Take preventive and defensive measures as legitimate bounds are exceeded.

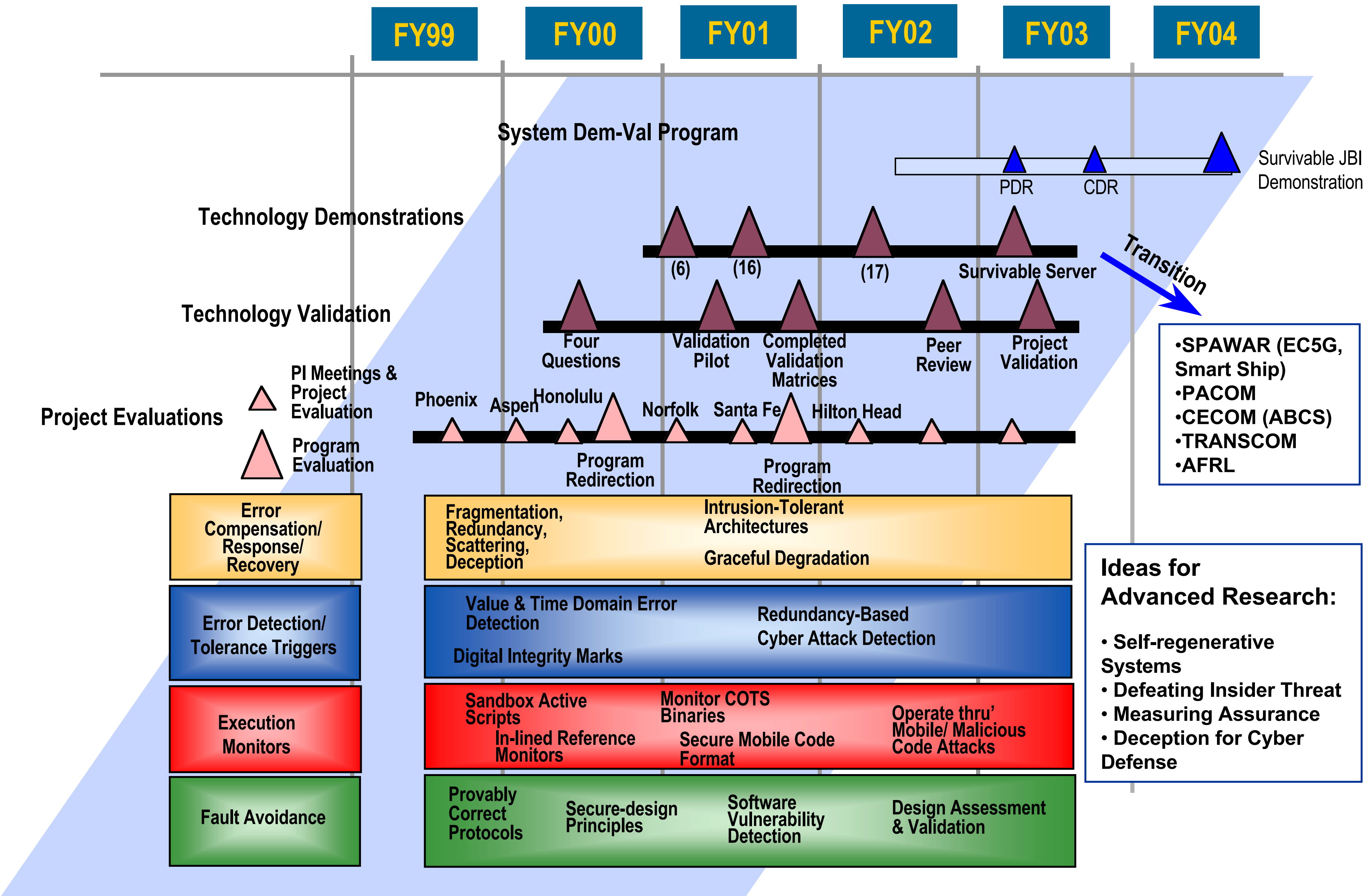
DARPA Investments

Self-Regenerative Information Systems

- **Seedling Programs**
  - Self-Healing Networked Information Systems:
    - Schneider Panel: 11/01 – 02/02
    - Automated Diversity, Scalable Redundancy, Deception Technologies, Defeating Insider Threats: 03/02 – 06/03
  - Measuring Assurance in Cyberspace: 07/02 – 06/03
  - Survivable Server: 07/02 – 06/03
- **OASIS Demonstration and Validation: Aug 2002 – July 2004**
- **Organically Assured and Survivable Information Systems (OASIS): July 1999- Dec 2003**

http://www.darpa.mil/ipto/research/oasis

Roadmap



FY99
FY00
FY01
FY02
FY03
FY04
System Dem-Val Program
Survivable JBI Demonstration
PDR
CDR
Technology Demonstrations
(6)
(16)
(17)
Survivable Server
Transition
Technology Validation
Four Questions
Validation Pilot
Completed Validation Matrices
Peer Review
Project Validation
Project Evaluations
PI Meetings & Project Evaluation
Program Evaluation
Phoenix
Aspen
Honolulu
Norfolk
Santa Fe
Hilton Head
Program Redirection
Program Redirection
•SPAWAR (EC5G, Smart Ship)
•PACOM
•CECOM (ABCS)
•TRANSCOM
•AFRL
Error Compensation/ Response/ Recovery
Fragmentation, Redundancy, Scattering, Deception
Intrusion-Tolerant Architectures
Graceful Degradation
Error Detection/ Tolerance Triggers
Value & Time Domain Error Detection
Digital Integrity Marks
Redundancy-Based Cyber Attack Detection
Execution Monitors
Sandbox Active Scripts
In-lined Reference Monitors
Monitor COTS Binaries
Secure Mobile Code Format
Operate thru' Mobile/ Malicious Code Attacks
Fault Avoidance
Provably Correct Protocols
Secure-design Principles
Software Vulnerability Detection
Design Assessment & Validation
Ideas for Advanced Research:
• Self-regenerative Systems
• Defeating Insider Threat
• Measuring Assurance
• Deception for Cyber Defense

HOLY GRAIL GOALS (1)

- Create self-healing systems that can operate through cyber attacks and provide continued, correct, and timely services to users.
- Adapt security posture to changing threat conditions and adjust performance and functionality.
- Always know how much reserve capability and attack margin are available.

HOLY GRAIL GOALS (2)

- Restore system capabilities to full functionality following an event
- Autonomously reassess success and failure of all actions before, during and after an event
- Autonomously incorporate lessons learned into all system aspects including architecture, operational procedures, and user interfaces

Study Committee

Fred B. Schneider, Cornell University - Chair

Jim Anderson, University of North Carolina

Stephanie Forrest, University of New Mexico

Carl Landwehr, National Science Foundation

Teresa Lunt, Palo Alto Research Center

Mike Reiter, Carnegie-Mellon University

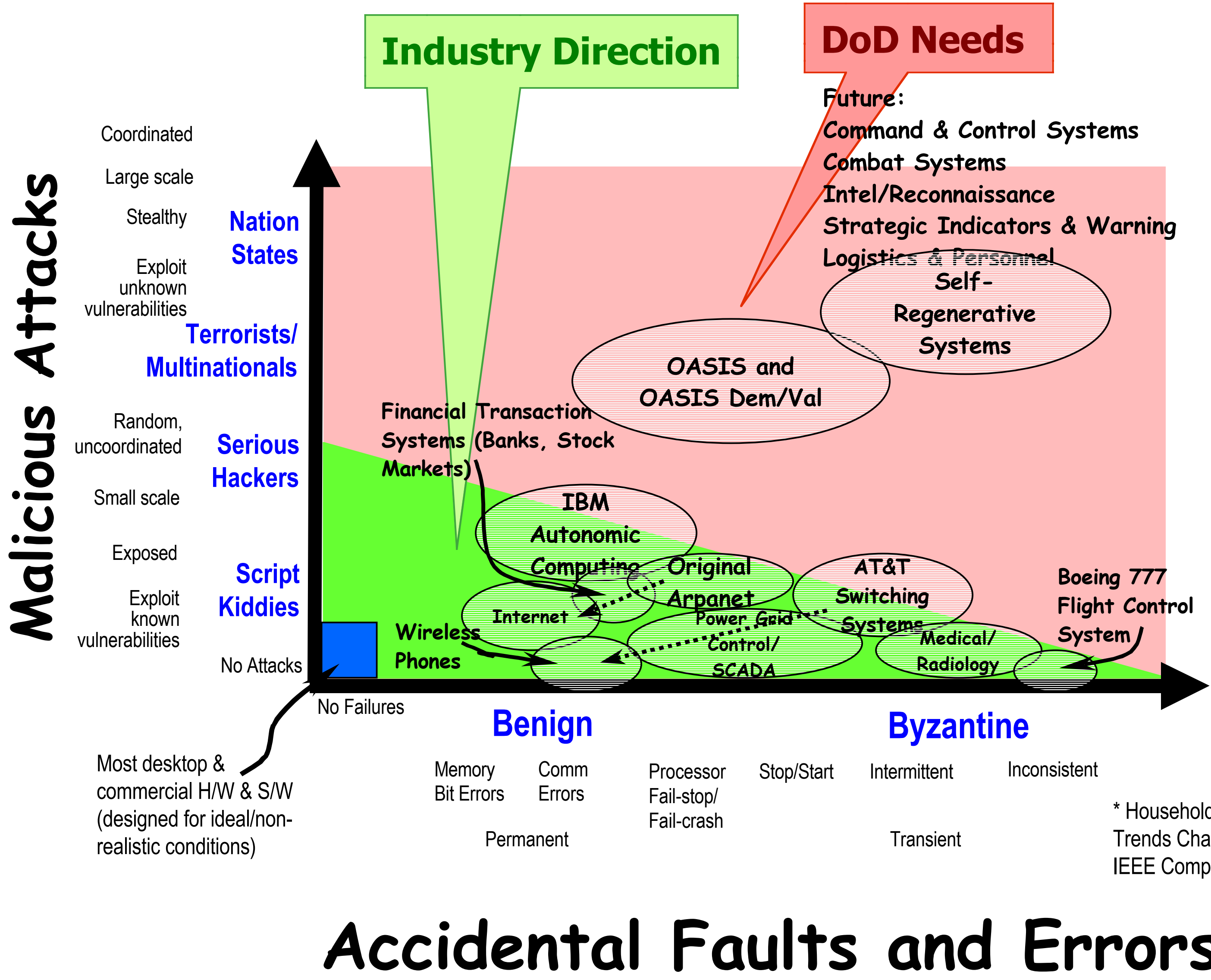
Kishor Trivedi, Duke University

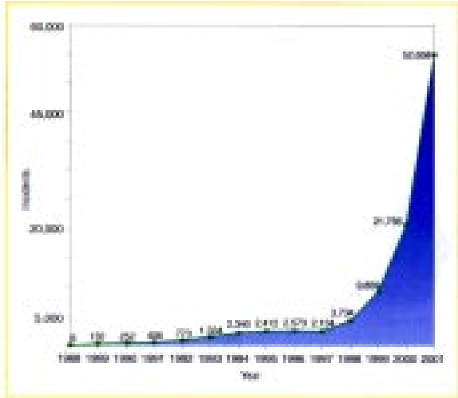
Study Process

- Two meetings in Washington, DC
- Briefings from subject-matter experts

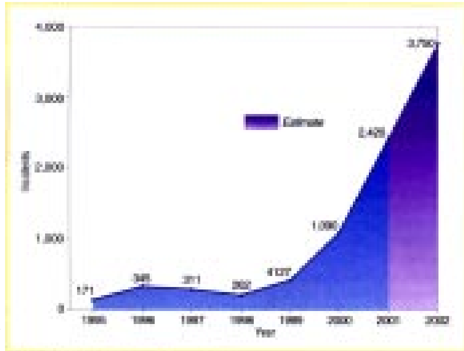
- Tarek Abdelzaher, Univ Virginia
- Massoud Amin, EPRI
- Anish Arora, Ohio State Univ
- Steve Bellovin, ATT
- Ken Birman, Cornell Univ
- Alan Demers, Cornell Univ
- Steve Goddard, Univ Nebraska
- Mohamed Gouda, Univ Texas
- Ted Herman, Univ Iowa
- Erica Jen, Santa Fe Institute
- Chandra Kintala, Avaya
- Simon Levin, Princeton Univ
- Alfred Spector, IBM Rsch
- Wietse Veneme, IBM Rsch

Industry versus DoD Needs





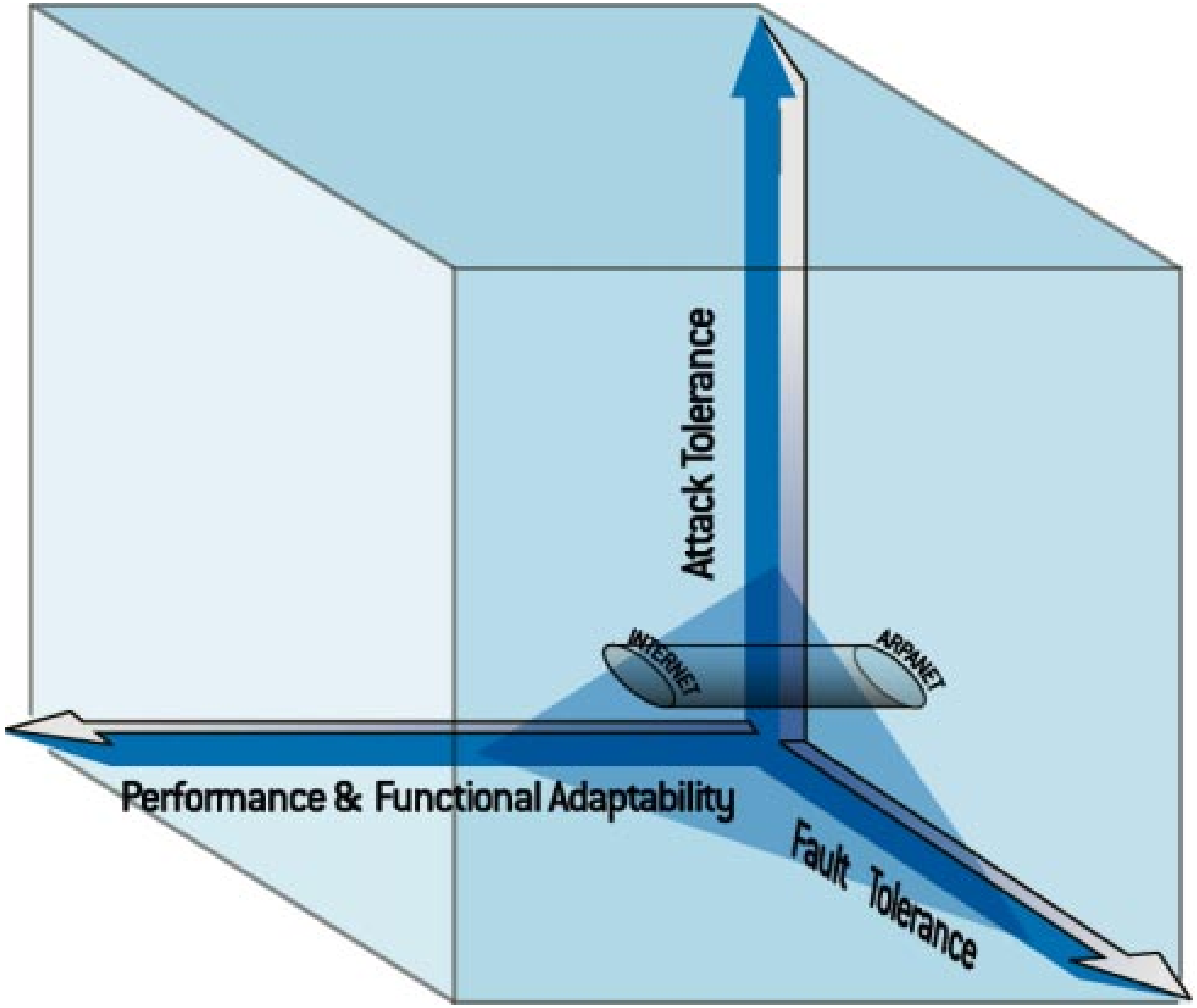
Incidents from 1988 to 2001*



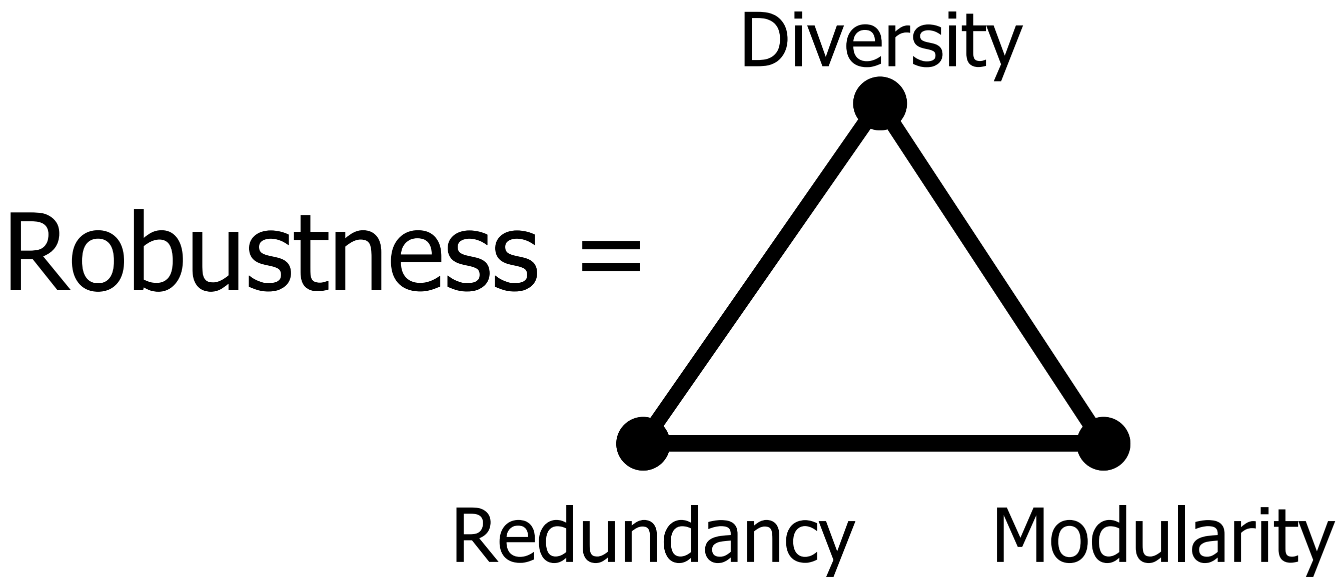
Vulnerabilities from 1995 to 2001*

* Householder, Houle, and Dougherty, "Computer Attack Trends Challenge Internet Security," Security & Privacy, IEEE Computer Society, Jan 2002

The Third Dimension



Addressing DoD Needs: Dimensions of Robustness [S. Levin]



The time is right to exploit new opportunities!

Addressing DoD Needs: New Research Opportunities

- **Primary Research Areas**
  - Temporal and spatial run-time diversity.
  - Scalable redundancy.
  - Self-stabilization.
  - Natural robustness via biological metaphors and systemic effects.
- **Complementary Research Areas**
  - Support for on-the-fly system change:
    - Software rejuvenation (refresh data or environment)
    - Control structure/data rep change
    - Adaptive fault-tolerance (ftol asmpt change)
    - Self-healing real-time schedulers
  - Enhanced detection:
    - Growing memory size, enables rollback to a previous state
    - Application-specific monitoring
  - Machine Learning
    - Reinforcement Learning (to adjust parameters in accordance with new information or feedback
    - Genetic programming (to evolve small software components)

Self-Regenerative Systems: Seedlings and SBIRs

Principal Investigator(s)	Project
Mike Reiter (CMU)/Stephanie Forrest (UNM	Automated Diversity in Computer Systems
Ken Birman (Cornell)	Scalable Network Redundancy for Network-Centric Military Applications
Mike Reiter (CMU)	Scalable Redundancy for Infrastructure Services
Fred Schneider	Beyond COCA: Quorums and Thresholds for Distributed Services
Scott Gerwehr (RAND Corporation)	Deception Technologies for Computer Network Defense
Steve Harp (Honeywell)	Skeptical Systems
S. Raj Rajagopalan (Telcordia)	Using Enhanced Credentials for Mitigating the Insider Threat in Enterprise Networks
Bob Balzer (Teknowledge)	CyberSafe: Autonomic Wrappers to Emasculate Malicious Code
Jayant Shukla (TRILKOM)	Applications for Multi-Terabit Networking
Matt Stillerman (ORA)	Efficient Code Certification for Open Firmware

Measuring Assurance: Program Goal

CONTEXT: Create robust software and hardware that are fault-tolerant, attack resilient, and easily adaptable to changes in functionality and performance over time.

PROGRAM GOAL: Create an underlying scientific foundation that will

- enable clear and concise specifications,
- measure the effectiveness of novel solutions, and
- test and evaluate systems in an objective manner.

Measuring Assurance: Challenges

- Unable to quantitatively state how assured systems and networks are.
- Unable to quantify ability of protective measures to keep out intruders.
- Difficult to characterize capabilities of intrusion detection systems to detect novel attacks.
- Benefits of novel response mechanisms cannot be measured comparatively or absolutely.

Measuring Assurance: Technical Approach

- Research the theoretic aspects of information assurance
- Develop measures of merit and metrics to characterize quantitatively various dimensions of security
- Show the relevance of the theory by applying theory to a realistic exemplar system

Measuring Assurance: Major Focus Areas

- Concepts and terminologies to succinctly express IA domain issues
- Threat, attack and vulnerability taxonomies
- Security models and models of attacker intent, objectives, and strategies
- Work factor metrics, survivability metrics, operational security metrics, cryptographic protocol metrics
- Methods for testing and validating protection mechanisms
- Security and survivability requirements specifications

Measuring Assurance: Seedling Performers

Principal Investigator	Project
Peng Liu (Penn State)	Measuring Quality of Information Assurance
Tom Van Vleck (NAI Labs)	Measuring Assurance
Dennis Hollingworth (NAI Labs)	Threat, Attack, and Vulnerability Taxonomies
Roy Maxion (CMU)	Developing a Defense-centric Taxonomy
Crispin Cowan (WireX)	Relative Vulnerability Approach to Predicting System Assurance
Brad Wood (SRI, International)	The Critical Security Rating
Bob Riemenschneider (SRI, International)	Global Measures of Assurance
Pradeep Khosla/Tom Longstaff (CMU/CERT)	Invited Workshop Series
Vladimir Gudkov (Univ of South Carolina)	The Quantitative Analysis of Cyberspace Utilizing Complex Systems Theory, Multi-dimensional Time-series Analysis, Wavelet Analysis and Generalized Entropy Measures
Mike St. Johns (NAI Labs)	Key Management within a Metric Analysis Framework
Bill Sanders (U of Illinois)/Partha Pal (BBN)	Probabilistic Quantification of Security Metrics in Cyberspace

Survivable Server Seedling

- **Objectives**
  - Create a survivable server using OASIS technologies that are suited to a selected military mission-critical applications
  - Demonstrate server survivability on a prototype platform in March 2003
  - Phase the project into the OPX program
- **Performers**
  - Teknowledge (HACQIT and integration)
  - Architecture Technology Corporation (VPNShield)
  - BBN (ITUA)
  - Secure Computing Corporation (ITSI)
  - Draper Laboratory (DB Transaction Mediator)
  - WireX (TRANSCOM WebMail Server with SCC)

OASIS Program Objectives

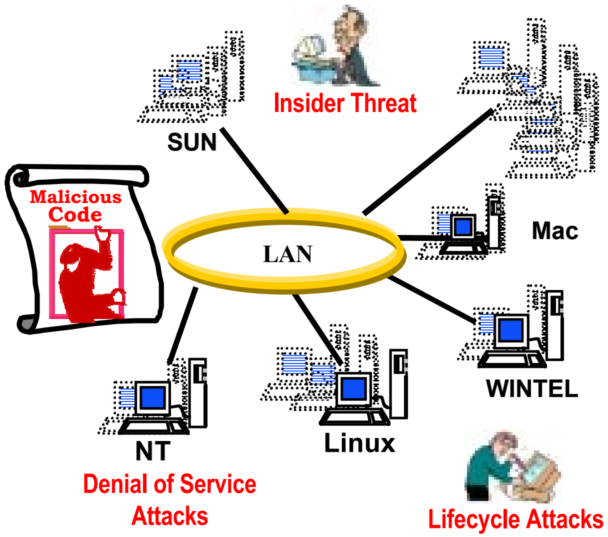
Prevent Intrusions
(Access Controls, Cryptography, Trusted Computing Base)

But intrusions will occur

Detect Intrusions, Limit Damage
(Firewalls, Intrusion Detection Systems, Virtual Private Networks, PKI)

But some attacks will succeed

Operate Through Attacks



OASIS Program Objectives

- To conceive, design, develop, implement, demonstrate, and validate architectures, tools and techniques that would allow fielding of organically survivable systems.
- To perform assessment and validation of organically survivable information systems.

Information Assurance Attributes*

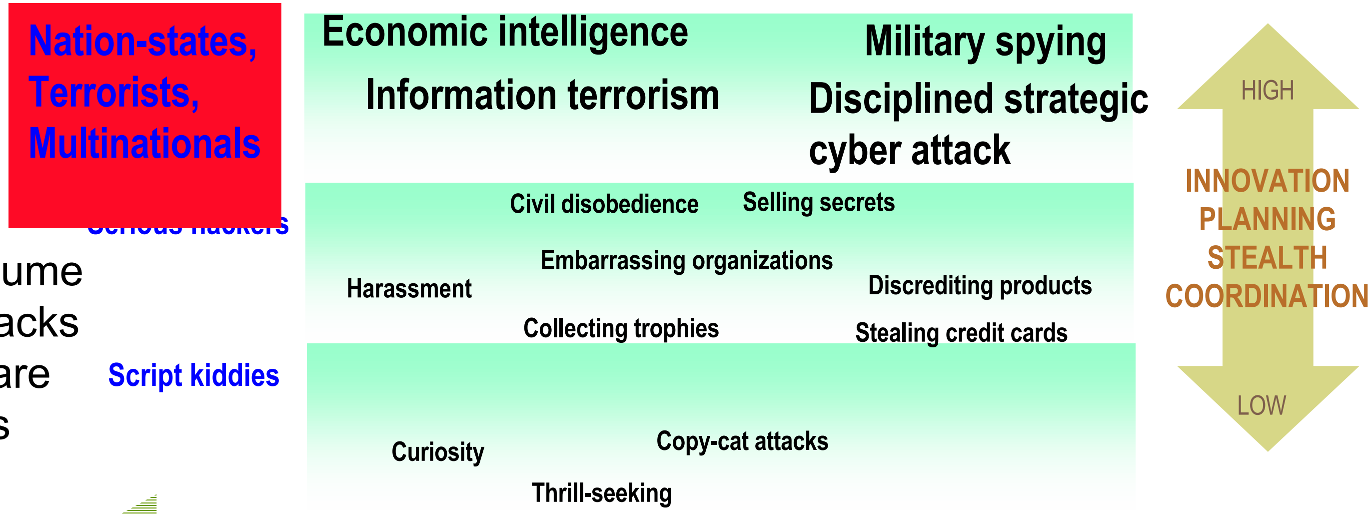
- **Integrity**
  - Maintain data and program integrity in the face of intrusions and malicious faults.
- **Availability**
  - Counter Denial-of-Service attacks and maintain high system availability.
- **Confidentiality**
  - Prevent unauthorized disclosure of information.
- **Authentication**
  - Prevent unauthorized access.
- **Non-repudiation**
  - Method by which the sender of data is provided with proof of delivery and the recipient is assured of the sender's identity, so that neither can later deny having processed the data.

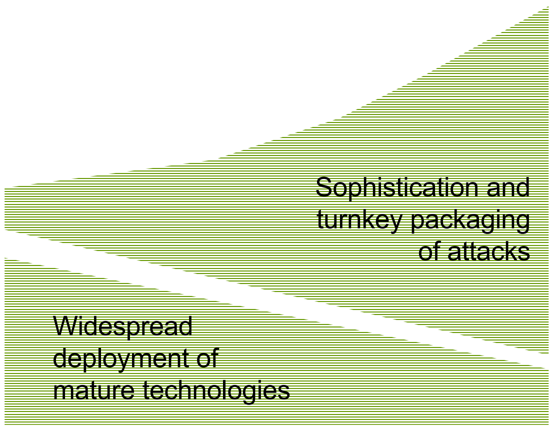
* Joint Pub 3-13 "Joint Doctrine for Information Operations"

Defending Against the Most Serious Attacks

The Daily Peacetime Problem

- Overwhelming volume of harassment attacks
- Can't tell if some are serious IW attacks





Increased population of attackers and access to damaging attacks

Reduced opportunities to attack DOD systems

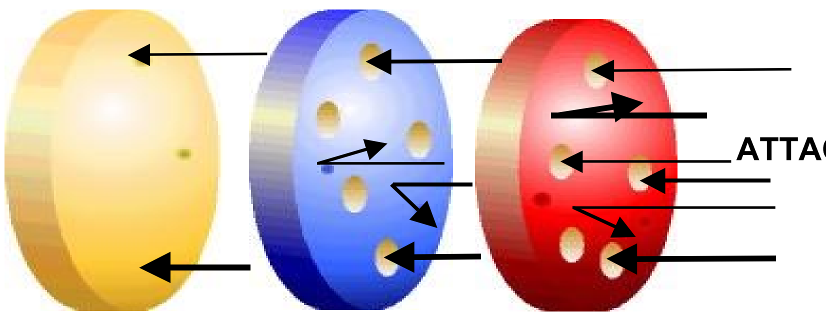
The Critical IW Attack Problem

- Still face high volume of harassment attacks
- Nation-state-level threats may use harassment attacks as cover, diversion, or disguise
- Determination and attribution of IW attacks is critical

OASIS Approaches, Challenges and Accomplishments

Approach

- Confine malicious code--compare actual behavior with predicted
- Detect errors: watermark, time/value domain anomalies, rear guards
- Error compensation and recovery: distributed computation, design diversity & deception



Top Technical Challenges

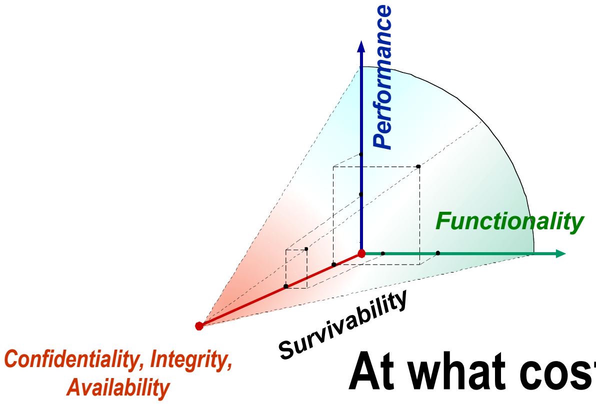
–Real-time trade of security, performance & functionality
–Cost-effective solutions
–Validation and verification

OASIS Program: Measures of Success

	Integrity	Availability	Confidentiality	Authentication	Non-repudiation
ILoveYou				N/A	N/A
Anna Kournikova				N/A	N/A
Nimda				N/A	N/A
Code Red I & II			N/A	N/A	N/A
Stachaldracht			N/A	N/A	N/A

Proof-Carrying Code Project	Integrity	Availability	Confidentiality	Authentication	Non-repudiation
Policy inconsistency					
Decision procedure					
Bug in protect. mech.					
Bug in decision proc.					
Illegal fetch/store					
Illegal jump					
Name resolution					
Check A, Execute B					
Forge certificate					
Compromised keys					
Unauthorized delete					
Invalid permissions					

Is intrusion tolerance feasible? - *Yes*



At what cost?

- Performance Overheads Quantified

Which security attributes are assured? Against which attacks/vulnerabilities?

OASIS Program	Integrity	Availability	Confidentiality	Authentication	Non-repudiation
Malicious Code					
DOS					
Insider Attack					

Coverage?

Proof-carrying Code Rationale

<table>
<tr><th></th><th></th><th>Availability</th><th>Integrity</th><th>Confidentiality</th><th>Authentication</th><th>Non-repudiation</th><th>Flexibility</th></tr>
<tr><td>Policy Inconsistency.</td><td>AV-1.1</td><td></td><td colspan="3">A2,M5</td><td></td><td rowspan="2">M1,M3, M6</td></tr>
<tr><td>Decision procedure</td><td>AV-1.2</td><td></td><td colspan="3">M4</td><td></td></tr>
<tr><td>Bug in protect. mech.</td><td>AV-2.1</td><td></td><td colspan="3">TCB</td><td></td><td rowspan="2">M1,M3, M6</td></tr>
<tr><td>Bug in decision proc.</td><td>AV-2.2</td><td></td><td colspan="2">M4</td><td>M4</td><td></td></tr>
<tr><td>Illegal fetch/store</td><td>AV-3.1</td><td></td><td colspan="2" rowspan="2">M2,M3,M4</td><td></td><td></td><td></td></tr>
<tr><td>Illegal jump</td><td>AV-3.2</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Name resolution</td><td>AV-3.3</td><td></td><td colspan="2">Note*</td><td></td><td></td><td></td></tr>
<tr><td>Check A, execute B</td><td>AV-3.4</td><td></td><td colspan="2">M2,M3,M4</td><td></td><td></td><td></td></tr>
<tr><td>Forge certificate</td><td>AV-3.5</td><td></td><td></td><td></td><td>M7</td><td></td><td></td></tr>
<tr><td>Compromised keys</td><td>AV-3.6</td><td></td><td></td><td></td><td>M8</td><td></td><td></td></tr>
</table>

M=Mechanisms
A=Assumptions

* May be addressed using Necula's strategy of safety-checking program after linking and loading. At this early stage of implementation we have not yet decided the issue.

Mechanisms: Formal Proofs

M1: Prover: constructs safety proof for untrusted application binary (Nec 97)

M2: Machine specification: axiomatizes instruction-set architecture (MA00)

M3: Safety policy: defines "theorem" to be proved (App01)

M4: Proof checker: determines whether proof matches theorem (PS99)

M5: Policy modeler: validation technique for safety policies (AF01)

M6: Semantics of types: used in constructing safety proofs (AF00)

M7: Digital signatures: can be generated only by holder of private key

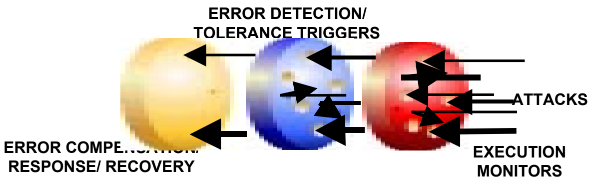
M8: Expiration: "freshness dating" certificates limits harm from key loss

Assumptions:

A1: Hardware (instruction-set architecture) executes correctly.

A2: Capability management: host's access control policy, written in expressive policy language, is appropriate to host's needs.

OASIS Technologies



ERROR COMPENSATION/ RESPONSE/ RECOVERY

Spatial, Temporal, Design, and Analytical Redundancies, Dynamic Reconfiguration, Quality of Service Trade-Offs, Fragmentation & Dispersal, Deception (Randomness, Uncertainty, Agility, Stealth), Graceful Degradation, Intrusion Tolerant Architectures

ERROR DETECTION/ TOLERANCE TRIGGERS

Watermarks, Mediated Interfaces, Rear Guard, Value & Time Domain Error Detectors, Comparison & Voting, Acceptance Checks, Redundancy-Based Cyber Attack Detection

EXECUTION MONITORS

In-Line Reference Monitors, Sandbox Active Scripts, Code Interposition, Wrappers, Proof Carrying Code, Graph Based Program Encoding, Monitor COTS Binaries, Secure Mobile Code Format, Operate through Mobile/ Malicious Code Attack

FAULT AVOIDANCE

Provably Correct Protocols, Secure-design Principles, Software Vulnerability Detection, Design Assessment and Validation

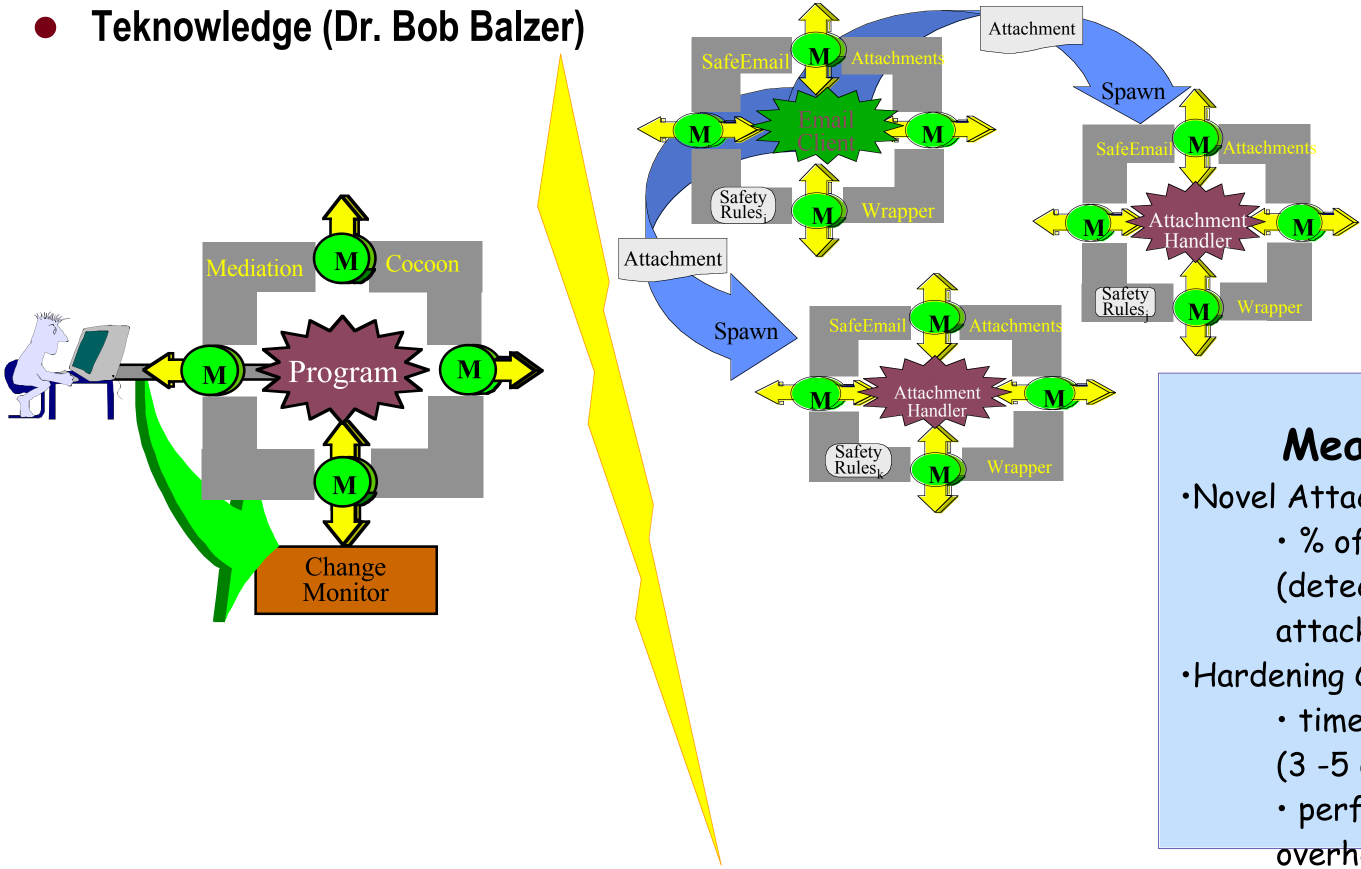
Active OASIS Projects

Category	Performer	Organization	Project
Error Detection/Tolerance Triggers; Error Compensation/Response/Recovery	Prof. Andrew Chien	UCSD	Agile Objects: Component-based Inherent Survivability
	Prof. Pradeep Khosla	CMU	Perpetually Available and Secure Information Systems
	Dr. Jim Just	Teknowledge	Hierarchical Adaptive Control for QoS Intrusion Tolerance (HACQIT)
	Dr. Peng Liu	UMBC	Engineering a Distributed Intrusion Tolerant Database System Using COTS Components
	Dr. Alexander Wolf	Univ. of Colorado	Tolerating Intrusions Through Secure System Reconfiguration
	Dr. Feiyi Wang	MCNC/Duke Univ.	Scalable Intrusion Tolerant Architecte (SITAR)
	Dr. Amjad Umar	Telcordia	Comprehensive Approach for IT Based on Intelligent Compensating Middleware
	Dr. Steve Chapin	Syracuse University	Computational Resiliency
	Mr. Alfonso Valdes	SRI, Intl.	Dependable Intrusion Tolerance
	Dr. Dick O'Brien	Secure Computing	Intrusion Tolerant Server Infrastructure
	Dr. Partha Pal	BBN	Intrusion Tolerance by Unpredictable Adaptation
	Ms. Janet Lepanto	Draper	Intrusion Tolerance Using Masking, Redundancy and Dispersion
	Mr. Lee Badger	NAI Labs	Self-Protecting Mobile Agents
	Mr. Gregg Tally	NAI Labs	Intrusion Tolerant Distributed Object Systems
Execution Monitors	Dr. Anup Ghosh	Cigital	An Investigation of Extensible Sys Sec for Highly Resource-Constrained Wireless Devices
	Dr. Robert Balzer	Teknowledge	Integrity Through Mediated Interfaces
	Prof. Anant Agarwal	InCert	A Binary Agent Technology for COTS Software Integrity
	Dr. Robert Balzer	Teknowledge	Enterprise Wrappers for Information Assurance(NT)
	Mr. Mark Feldman	NAI Labs	Enterprise Wrappers for Information Assurance (Unix)
	Prof. Andrew Appel	Princeton University	Scaling Proof-Carrying Code to Production Compilers and Security Policies
	Prof. Fred Schneider	Cornell University	Containment and Integrity for Mobile Code
Fault Avoidance	Dr. Tim Hollebeek	Cigital	An Aspect Oriented Security Assurance Solution
	Prof. Crispin Cowan	WireX	Autonomix: Component, System and Network Autonomy
	Dr. Victoria Stavridou	SRI, Intl.	Intrusion Tolerant Software Architecture
	Prof. Michael Franz	UC, Irvine	Reconciling Execution Efficiency With Provable Security
	Dr. Howard Shrobe	MIT	Active Trust Management for Autonomous Adaptive Survivable Systems
	Dr. Ranga Ramanujan	Architecture Technology	Randomized Failover Intrusion Tolerant Systems (RFITS)
	Prof. Tim Teitelbaum	Grammatech	Dependance Graphs for Information Assurance of Systems
	Dr. Tom Longstaff	CMU, SEI	Information Assurance Science and Engineering Project
	Dr. Victoria Stavridou	SRI, Intl.	Information Assurance Management Requirements

Number of Projects Started Under OASIS: 39 **Number of OASIS Projects Active Today: 25**

Safe E-mail Wrappers

- **Transitioning to PACOM for scalability tests and experience in military operational environment**
  - Demonstrated protection against mobile malicious code (malicious email attachments, scripts in email bodies, web applets, active-x controls, downloaded programs), corrupted executables and documents, and latent flaws in applications by several different techniques
  - Not signature based; techniques work on novel viruses without any customization
- **Teknowledge (Dr. Bob Balzer)**

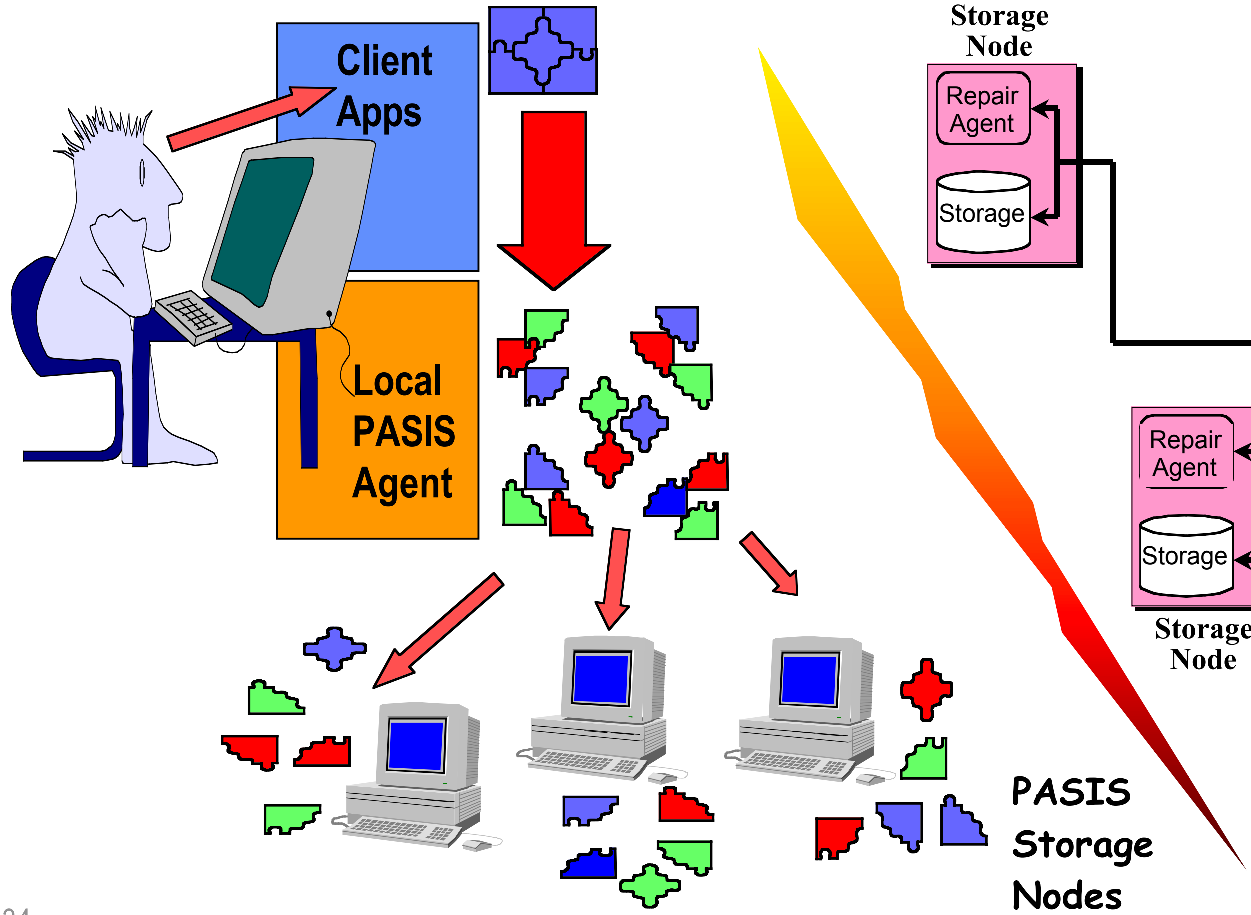


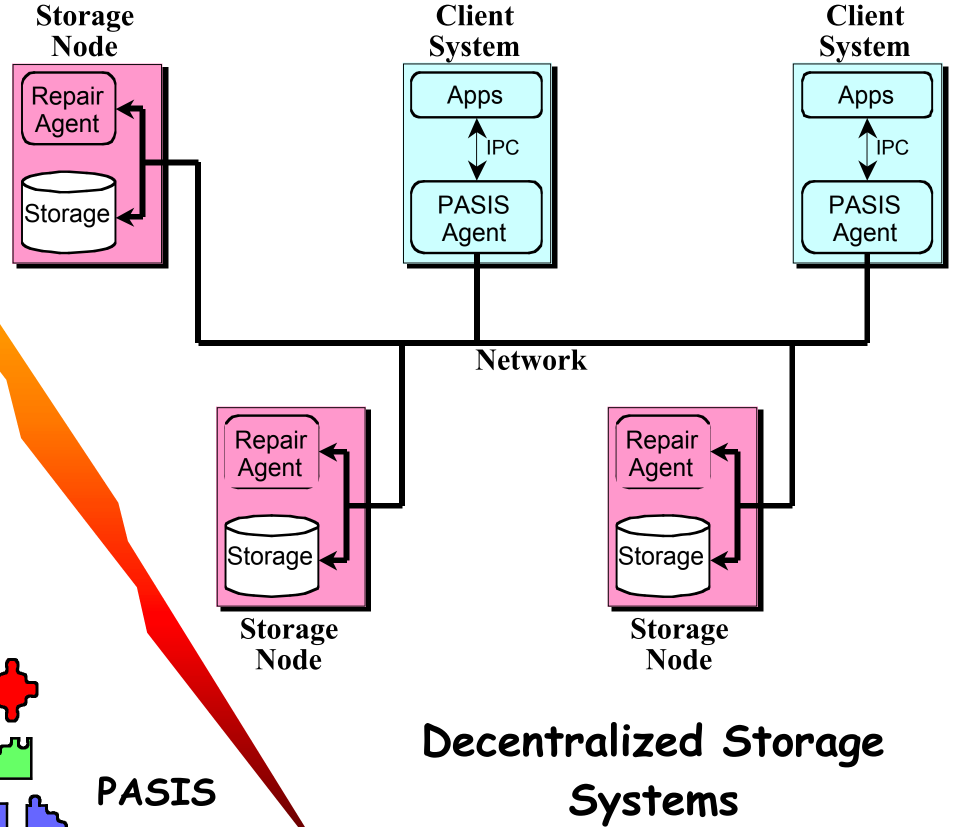
Measures of Merit

- Novel Attack Resistance:
  - % of novel attacks prevented (detected 13 of 13 malicious attacks)
- Hardening Costs:
  - time to tune security policies (3 -5 days)
  - performance degradation (7% overhead)

Intrusion Tolerant Data Storage

- **Perpetually Available and Secure Information Systems (PASIS)**
- **Transitioning to USAF Joint Battlespace Infosphere (JBI) - *Funded by AFRL***
  - To assure availability, integrity, and confidentiality of JBI "data repository"
  - Demonstrated intrusion tolerant data storage
- **Carnegie Mellon University (Prof. Pradeep Khosla)**

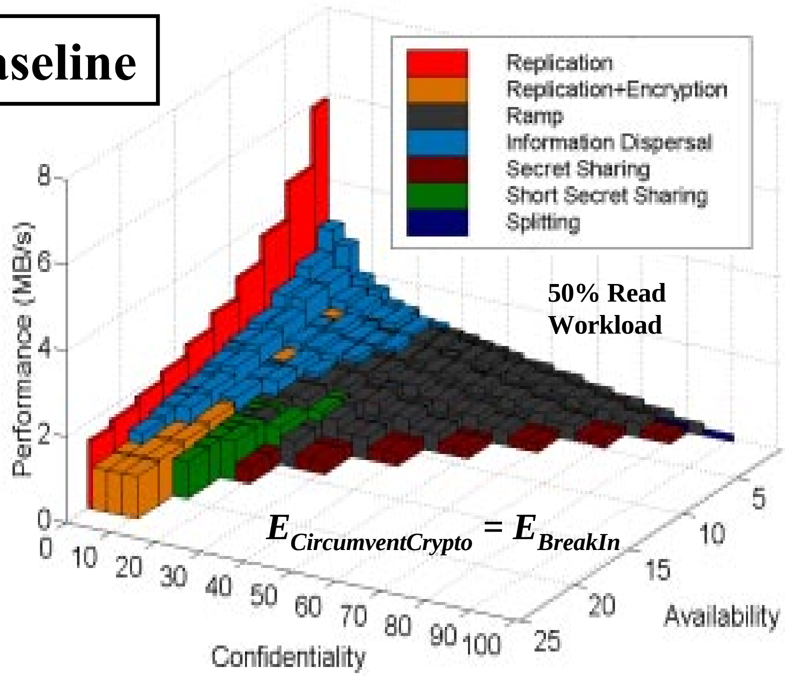




Intrusion Tolerant Data Storage

•PASIS (Performance Trade-offs)

Baseline

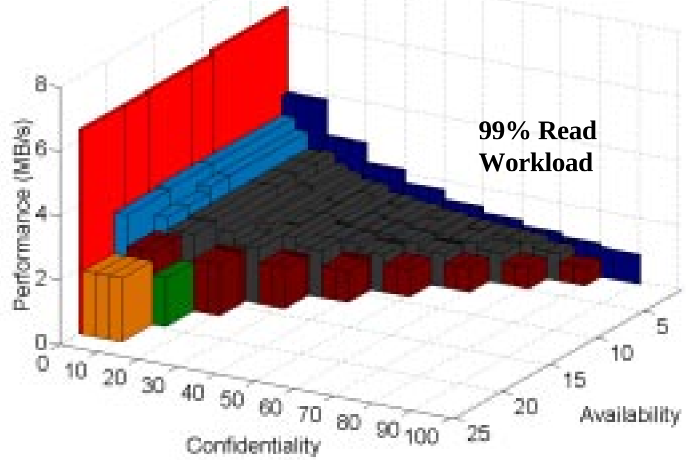


Performance (MB/s)
- •based on simple performance model
- •computed with standard performance eval. techniques

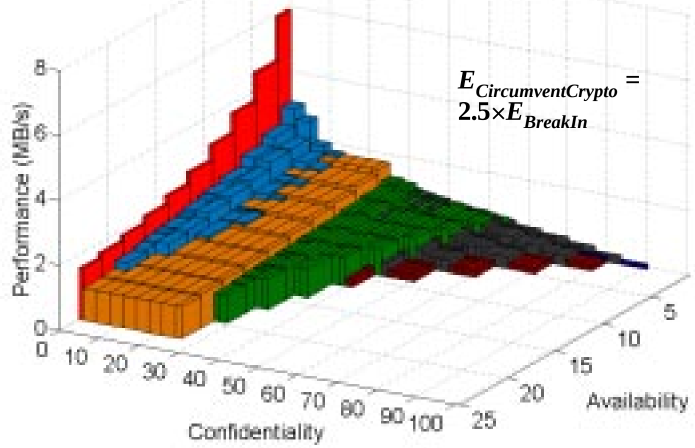
Availability ("nines")
- •standard fault tolerance math with independent failures
- •relative values are useful even if not independent

Confidentiality (Effort to compromise)
- •estimate effort involved with possible attack paths
- •overall effort is minimum of possible efforts

Extreme Read Workload

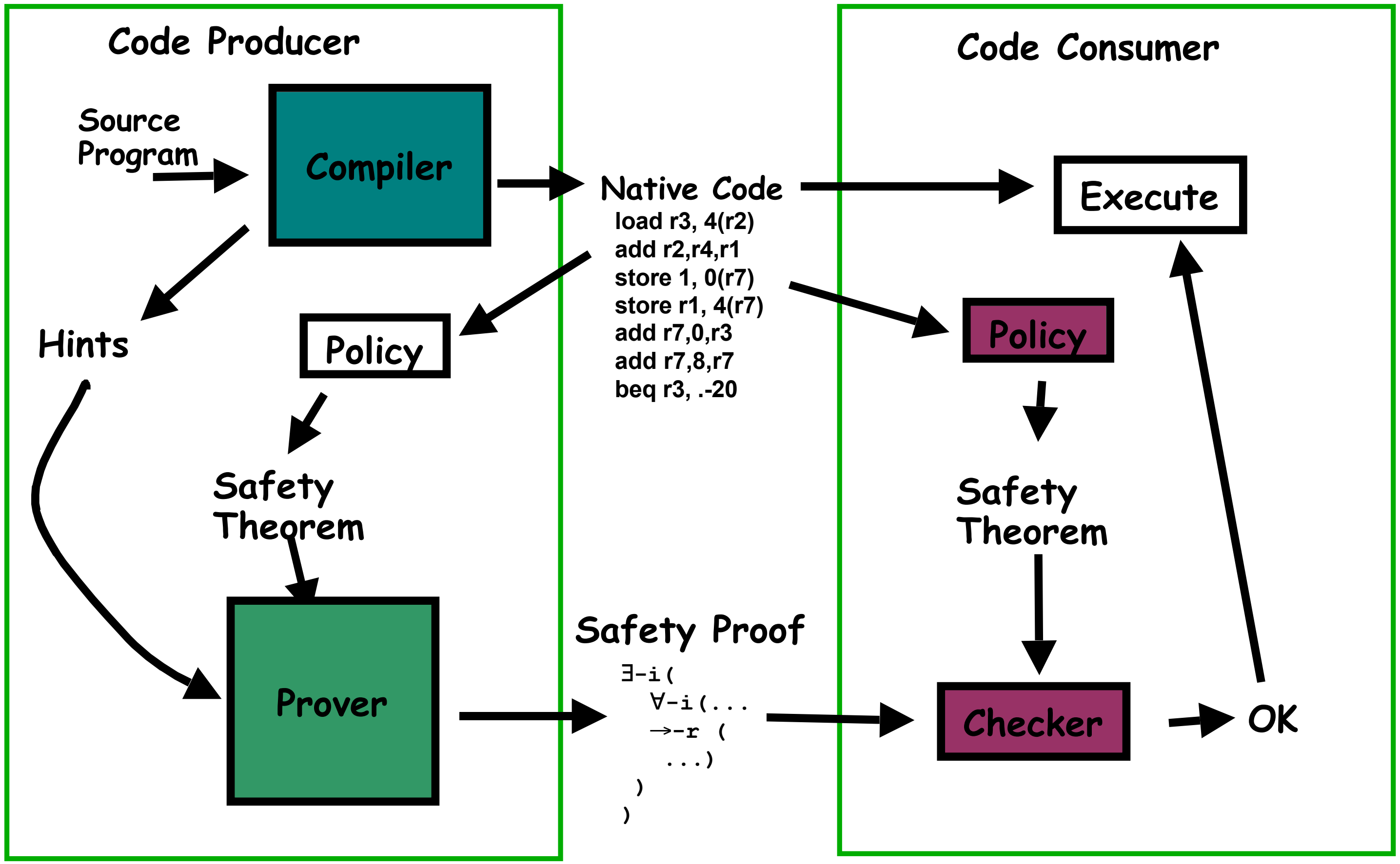


Security Model Sensitivity

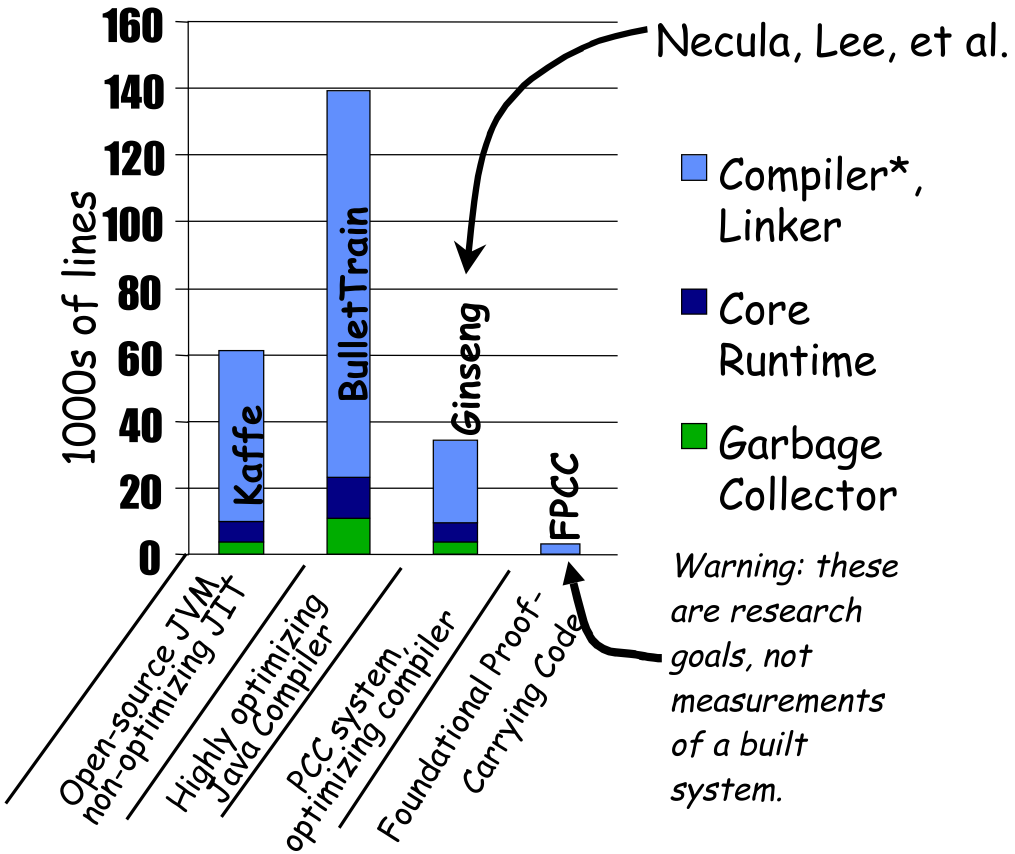


Proof-carrying Code

- **Princeton/Intel collaboration**
  - PCC Technology being applied to Intel's "Just in Time" compiler for Microsoft's Common Language Runtime (CLR).
  - Demonstrated scalable certifying compiler that produces proof of program behavior along with the code.
- **Princeton University (Prof. Andrew Appel)**
- **Yale University (Prof. Zhong Shao)**



Proof-carrying Code



Size of Trusted Computing Base

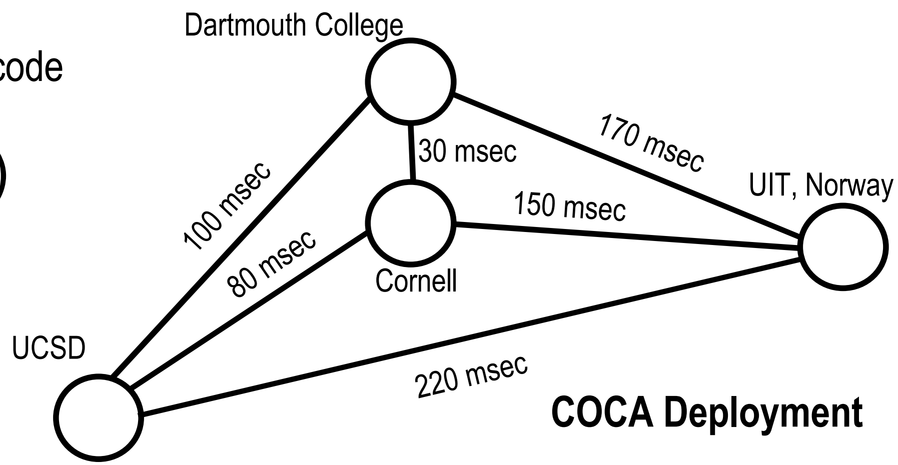
Measures of Merit

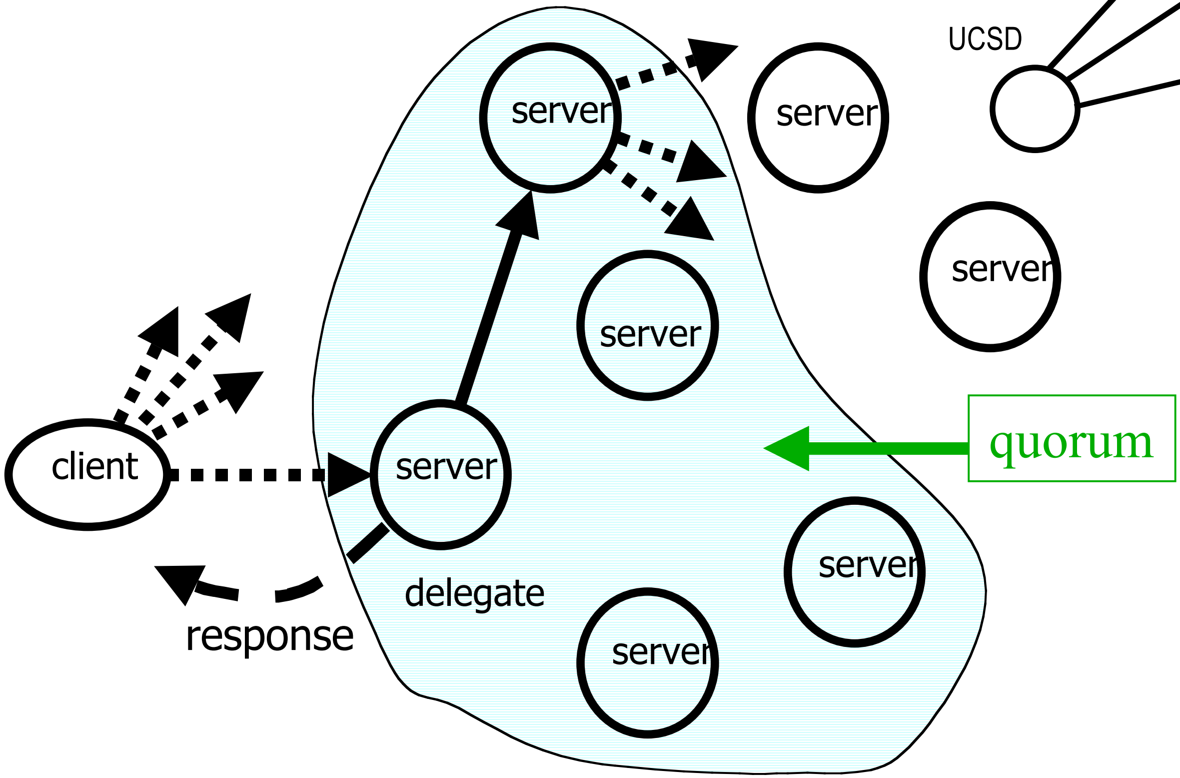
Goal:

- Reduce size of Trusted Computing Base to 4K Source Lines of Code
  - Approximately 10% of comparable functionality PCC compiler
- Actual TCB size achieved
  - 3K SLOC
  - 25% better than a very aggressive goal

Intrusion Tolerant Certificate Authority

- **Prototype implementation:**
  - Approximately 35K lines of new C source code
  - Certificates in accordance with X.509
- **Cornell University (Prof. Fred Schneider)**





server failure
↓ **disseminated Byzantine quorum**

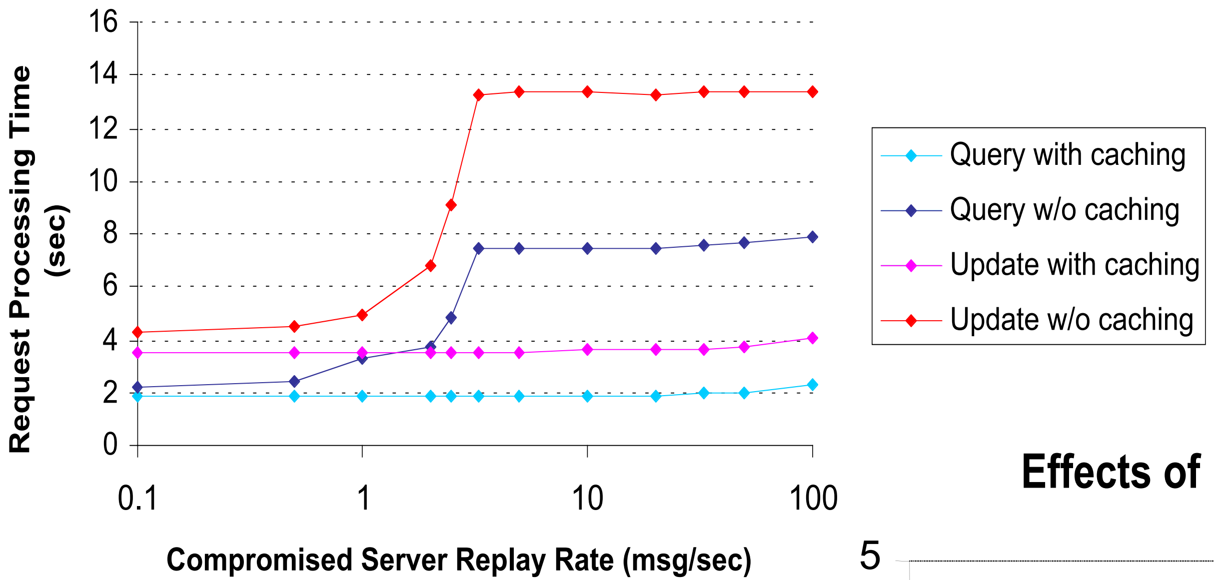
server compromise
↓ **threshold signature protocol**

mobile attack
↓ **proactive secret sharing (PSS)**

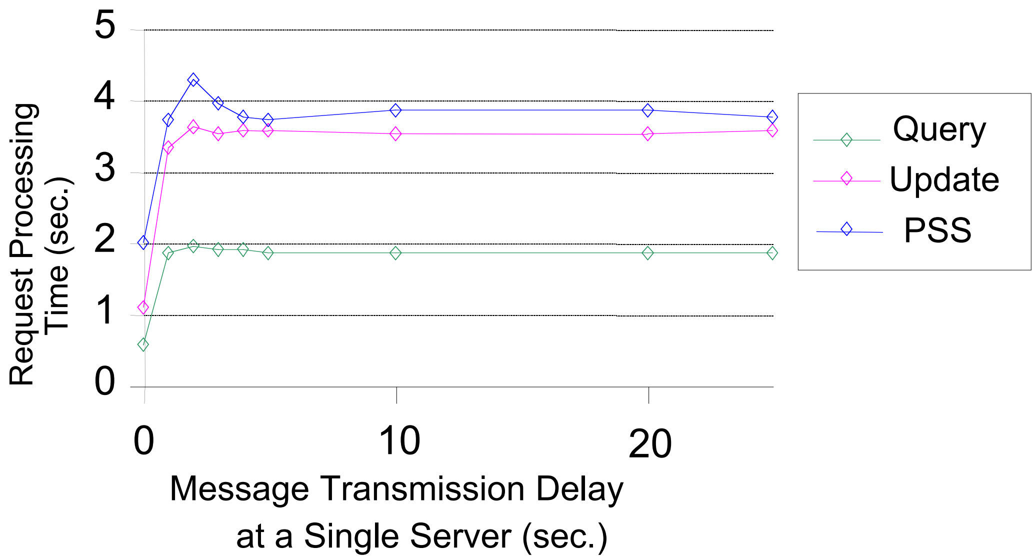
asynchrony
↓ **asynchronous PSS**

Denial of Service Defense

Effects of Caching



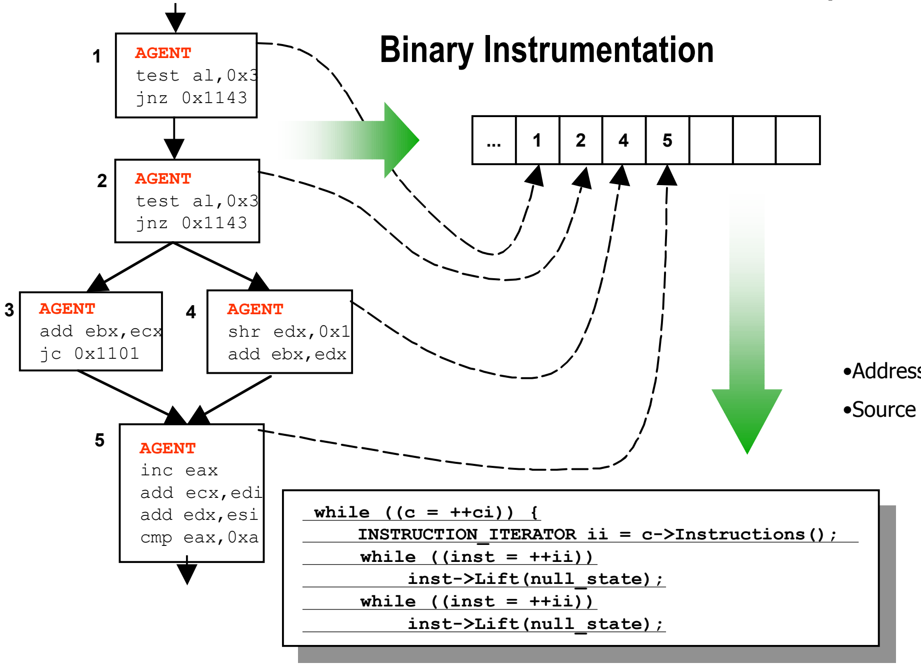
Effects of Message Delay



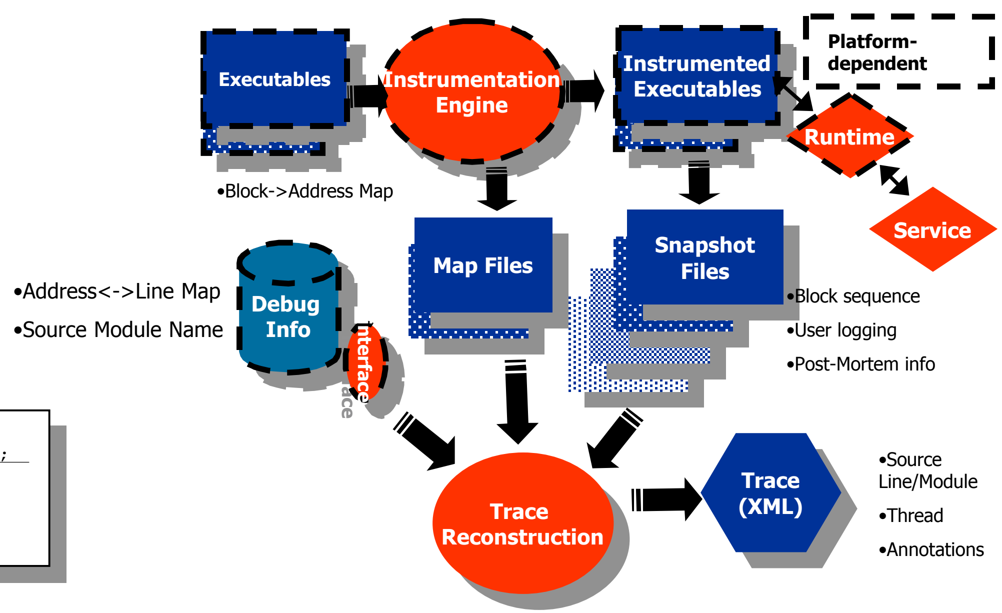
Monitoring Malicious Actions by Legacy Software

- **Transitioning to Sun Microsystems**
  - Transitioned to Phase Forward
  - Demonstrated insertion of code in C programs for Intel/NT platforms to monitor malicious actions by legacy software
- **InCert Software Corporation (Dr. Anant Agarwal)**

Binary Instrumentation



Major Components

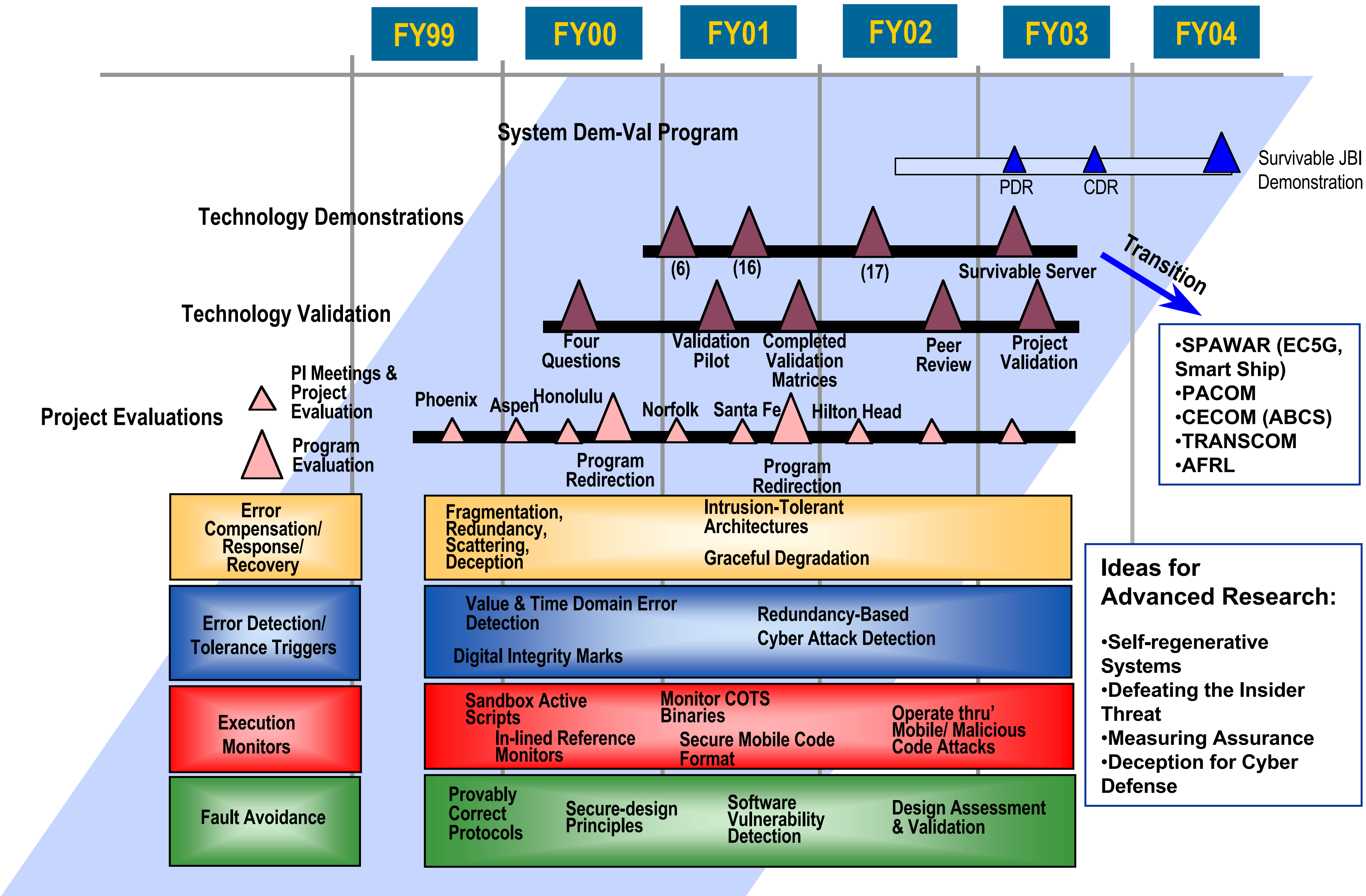


Competition Sensitive

Inserting Binary Agents: Measures of Success

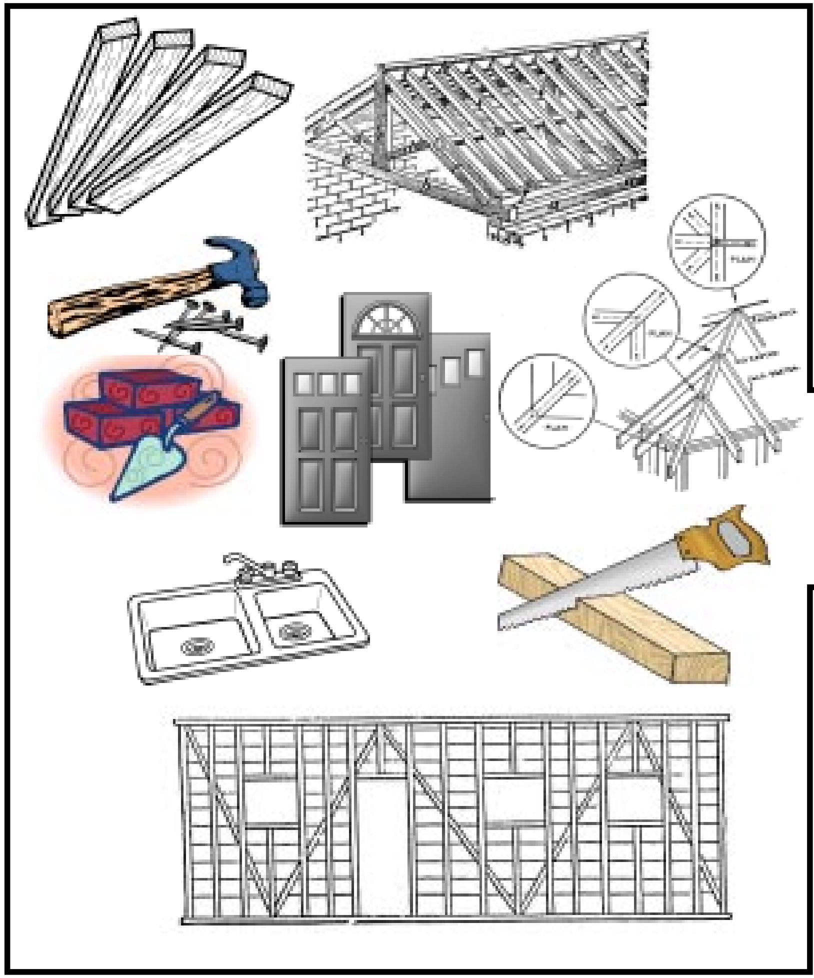
- **Percentage of executables successfully instrumented**
  - Goal: 100%
  - Accomplished to date: Virtually 100% (approx. 50 real world executables instrumented)
- **Performance degradation**
  - Goal: less than 5% overhead
  - Accomplished to date: 5-10% overhead when measured in real world scenarios.
- **Anomaly detection**
  - Goal: 100%
  - Accomplished to date: Detected 12 of 16 (75%) known problems in field tests.

OASIS Roadmap



OASIS Integration, Demonstration, and Validation Program (OASIS Dem/Val)

Dem-Val: Creating an Architecture



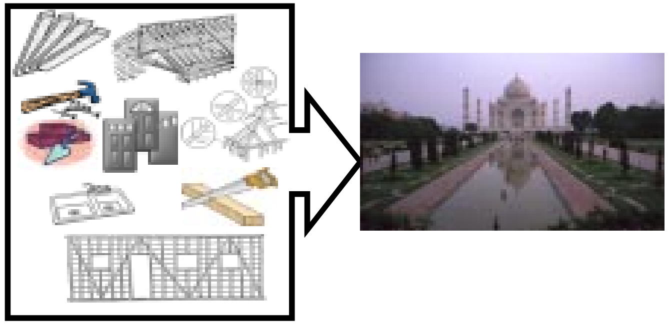
The OASIS, FTN, and other DARPA programs developed tools, components, architectures, mechanisms.

OASIS Dem-Val applies the DARPA program results and other technologies to produce an organically robust and dependable system architecture

OASIS Dem/Val

Program Objective

- Demonstrate and validate a working military mission critical system prototype that is highly dependable in the presence of cyber threats and imperfect hardware and software.



Key Milestones

6/02	1/03	6/03	1/04	6/04
Award	PDR	CDR/ Downselect		Demonstration

- Create a secure and survivable JBI architecture employing defense in depth layers of real-time execution monitors, adaptive re-configurable strategies
- Validate architectural approach using analytical models and formal proofs.
- Build a survivable JBI instantiation and demonstrate an Air Tasking Order creation, modification and execution under a sustained red team attack

Technical Challenges

1. Provide 100% of JBI critical functionality when under sustained attack by a "Class-A" red team with 3 months of planning.

   Currently many systems can be brought down in seconds to minutes with little planning.

2. Detect 95% of large scale attacks within 10 mins. of attack initiation and 99% of attacks within 4 hours with less than 1% false alarm rate.
3. Prevent 95% of attacks from achieving attacker objectives for 12 hours.

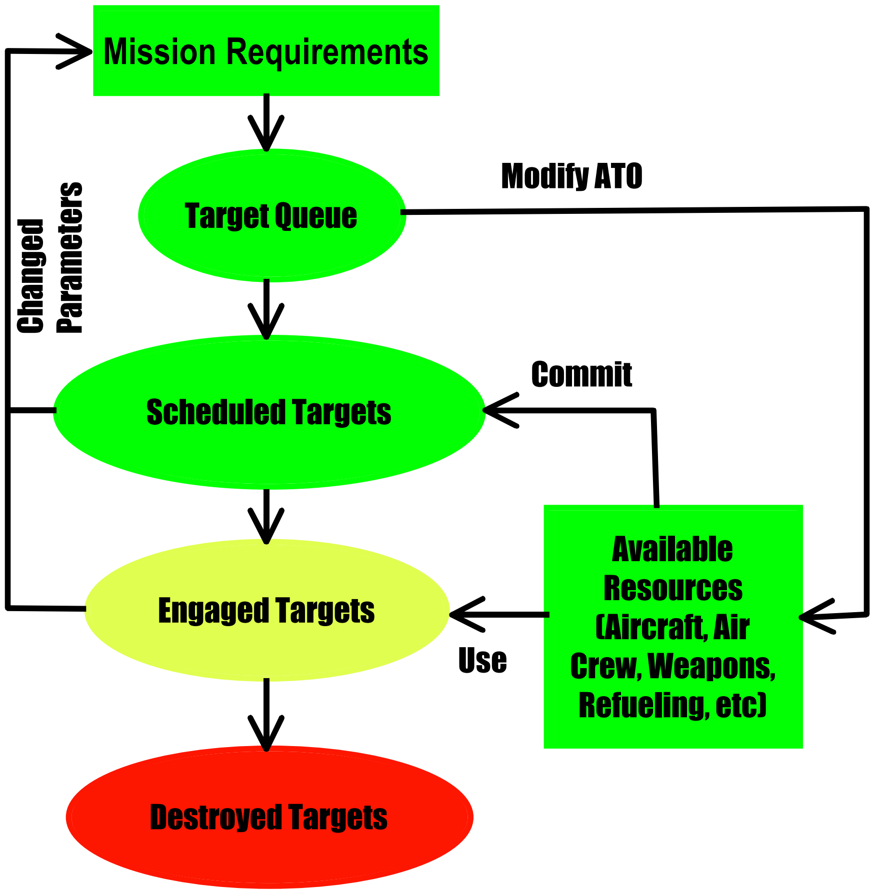
   In Integrated Feasibility Experiment (IFE) 3.1 fourteen out of fifteen flags were captured by the red team.

4. Reduce low-level alerts by a factor of 1000 and display meaningful attack state alarms .
5. Show survivability versus cost/performance trade-offs.

Technical Approach

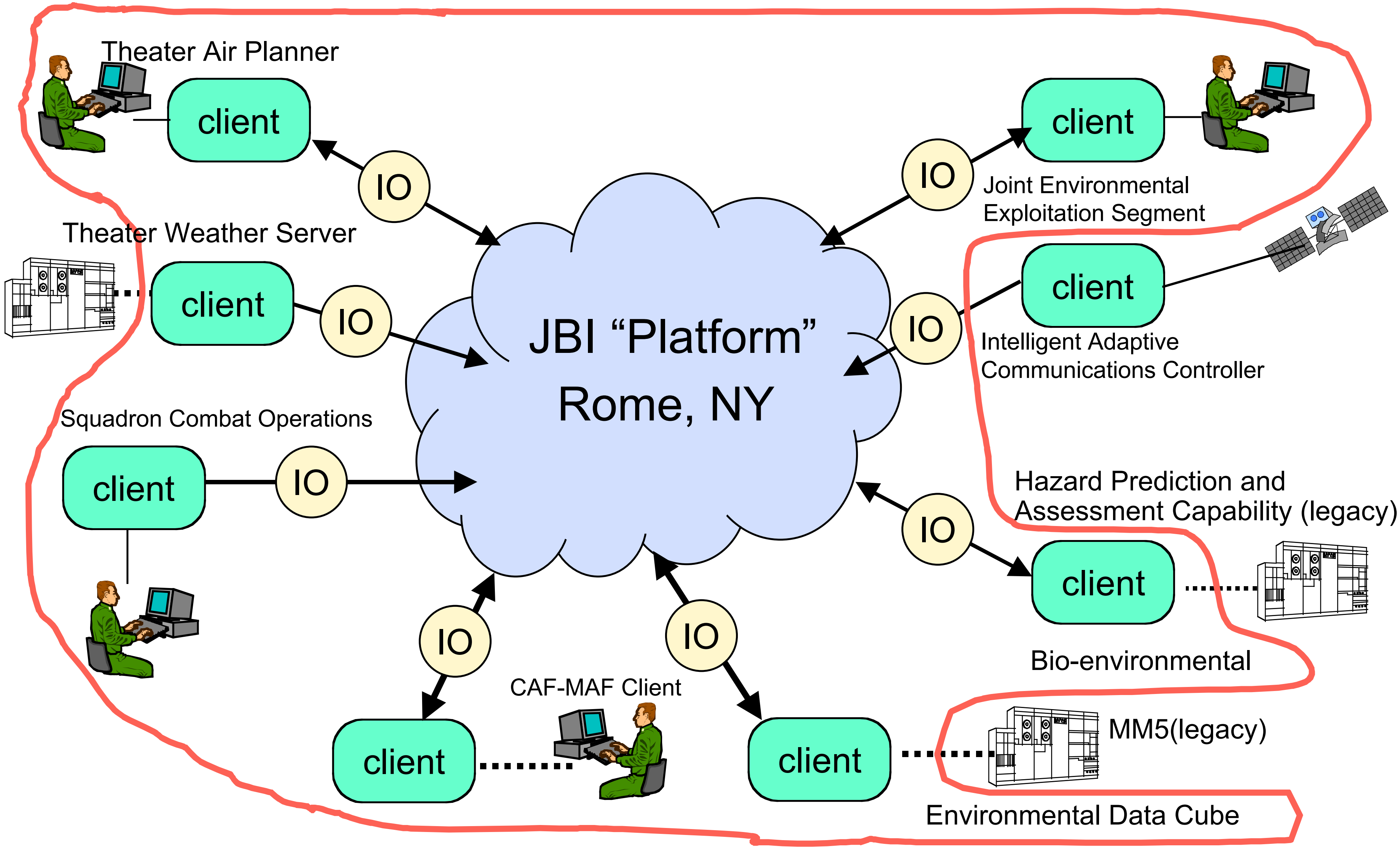
- Avoid single points of failure
- Design for graceful degradation
- Exploit diversity to increase the attacker's work factor
- Disperse and obscure sensitive data
- Make the system dynamic and unpredictable
- Deceive the attacker

Prototype Scenario



- **Mission Planning**
  - Establish mission objectives
  - Air Tasking Order (ATO) creation
  - ATO to operating units
  - minutes to hours
  - Air Mobility Command and Air Combat Command Coordination (CAF-MAF)
- **Mission Execution**
  - Monitor mission parameters
  - Mission parameters change
    - Weather change affects Chem-Bio plume dispersion forecast
  - Modify mission in progress
  - Re-direct mission elements
  - Real-time execution
  - Air Mobility Command and Air Combat Command Coordination (CAF-MAF)

JBI System Overview & Dem-Val Scope



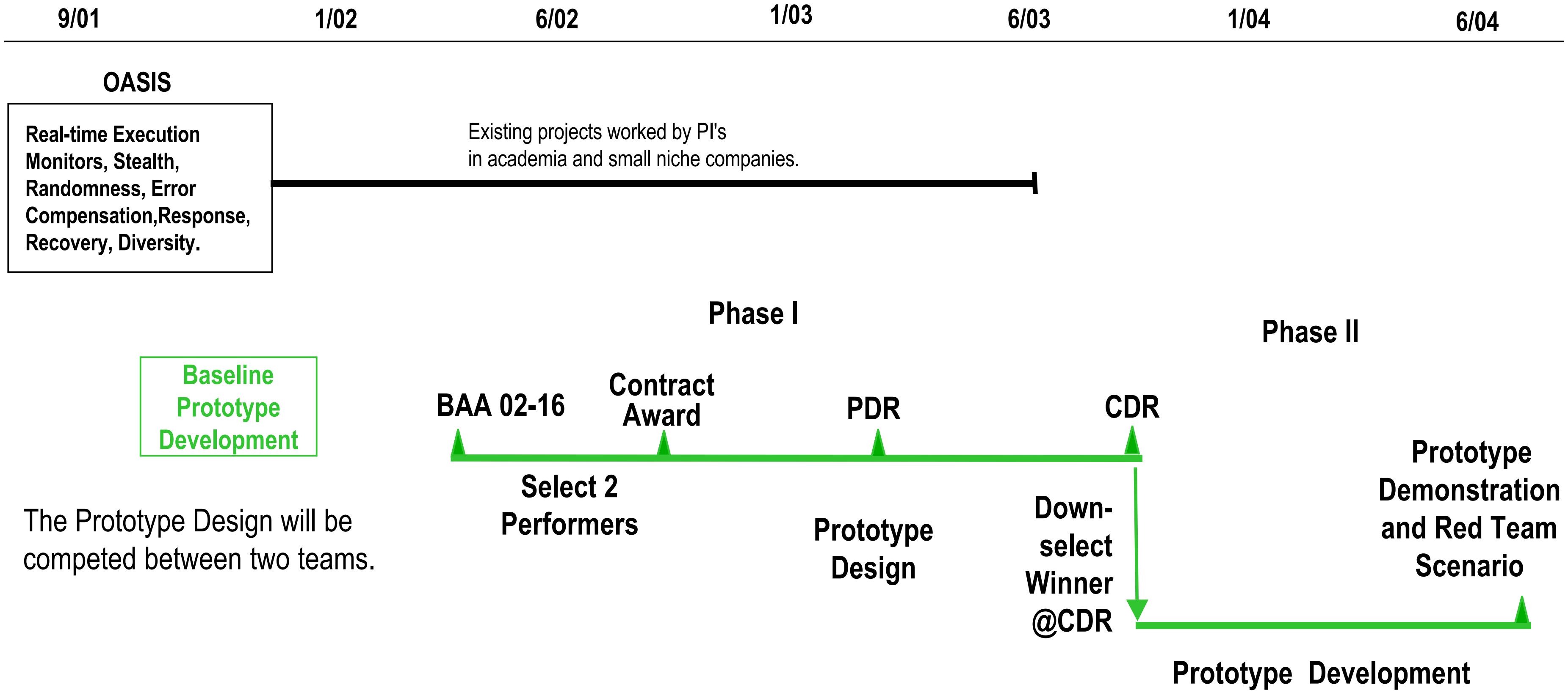
Goals, Requirements, and Measures of Success

- Provide 100% of JBI critical functionality when under sustained attack by a "Class-A" red team with 3 months of planning.
  - Currently many systems can be brought down in seconds to minutes with little planning.
- Detect 95% of large scale attacks within 10 mins. of attack initiation and 99% of attacks within 4 hours with less than 1% false alarm rate.
- Prevent 95% of attacks from achieving attacker objectives for 12 hours.
  - In Integrated Feasibility Experiment (IFE) 3.1 fourteen out of fifteen flags were captured by the red team.
- Reduce low-level alerts by a factor of 1000 and display meaningful attack state alarms .
- Show survivability versus cost/performance trade-offs.

Prototype Demonstration: Red Team Scenario

- **Red Team**
  - Competed
- **Attack Phases**
  - Determine Rules of Engagement
  - Planning Phase
    - Three to six months to provide for planning, innovation and stealth
  - Execution Phase
    - Two weeks to a month
- **Potential Attacks**
  - Wide coverage of known vulnerabilities and system components. (Denial of service, flooding, viruses, Trojans, worms)
- **Expected System Behavior under Attack**
  - System will dynamically reconfigure under changing threats
  - System will continue to provide essential services while under attack
  - System status will be displayed
- **Comparison to non-protected system under attack**
  - Similar resources expended against baseline JBI

Acquisition Strategy



The Prototype Design will be competed between two teams.