

# CRASH

Clean-slate design of **R**esilient, **A**daptive,  
**S**ecure **H**osts

---



**PROPOSERS DAY**

Arlington, VA

4 June 2010





# CRASH

## Clean-slate design of **R**esilient, **A**daptive, **S**ecure **H**osts



Blow off the Legacy Computational Base  
Inspired by biological mechanisms for resilience.

- Provably removes whole classes of vulnerabilities.
- Learns how to respond to new threats.
- Defense in depth.
- Diversity, randomization, variability.
- Diagnosis, adaptation & self-regeneration.



# We Are At War, And We're Losing

"If these trends continue through the end of 2009, there would be a **60 percent increase in malicious cyber activity compared to 2008**. .... in just the preceding six months, the U.S. military alone had spent more than \$100 million ... to remediate attacks on its networks" 2009 REPORT TO CONGRESS of the U.S.-CHINA ECONOMIC AND SECURITY REVIEW COMMISSION ONE HUNDRED ELEVENTH CONGRESS, NOVEMBER 2009

Figure 2: Yearly Dollar Loss (in millions) of Referred Complaints

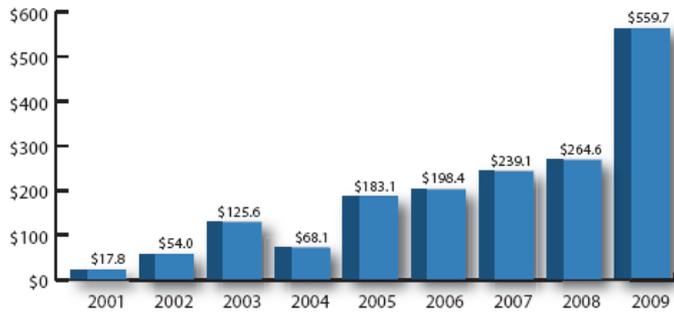
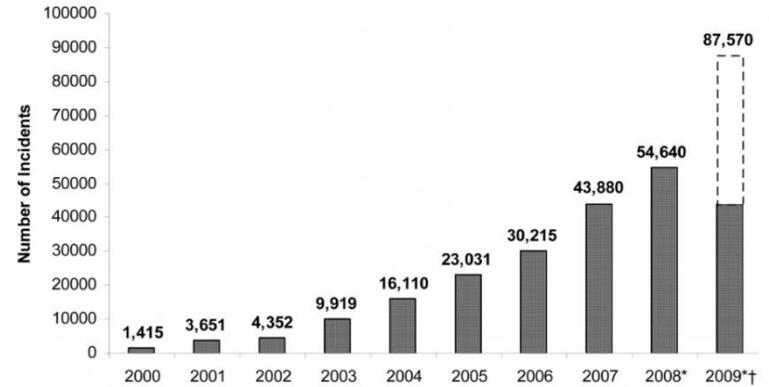


Figure 1: DoD Reported Incidents of Malicious Cyber Activity, 2000-2008, With Projection for 2009

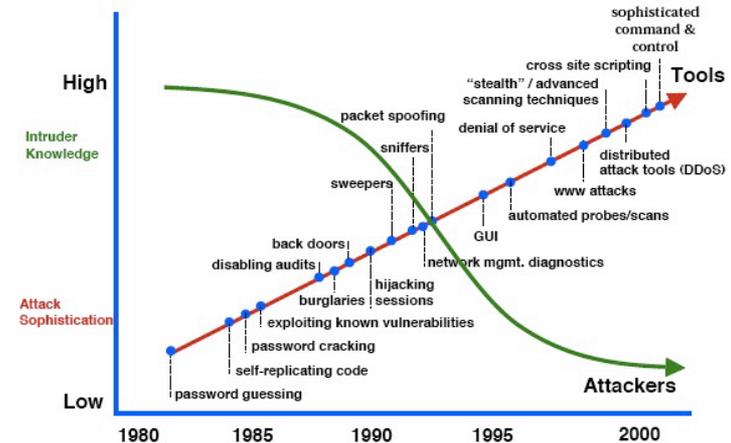


## We Would Lose Cyberwar says former DNI Mike McConnell

This was written by Michael Cheek on Wednesday, February 24, 2010, 11:33.

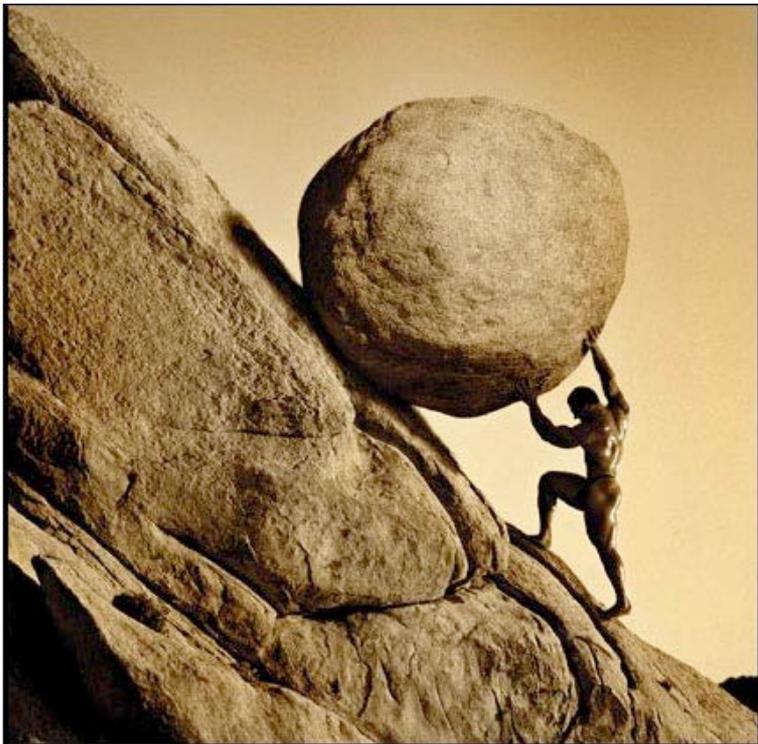


Cyberwar is increasingly entering into the mindset of policy month, DNI Dennis Blair outlined the cyber threat in his Annual 1 the US Intelligence Community, saying that "The United States combination of known and unknown vulnerabilities, strong an adversary capabilities, and a lack of comprehensive threat aware





## Today's Practice: P4 Perimeter Protection, Patch & Pray



- It's not about better firewalls
- It's not about better virus detection
- It's not about better intrusion detection
- It's not about better programmer practices
- **It's about being more adaptive and resilient**

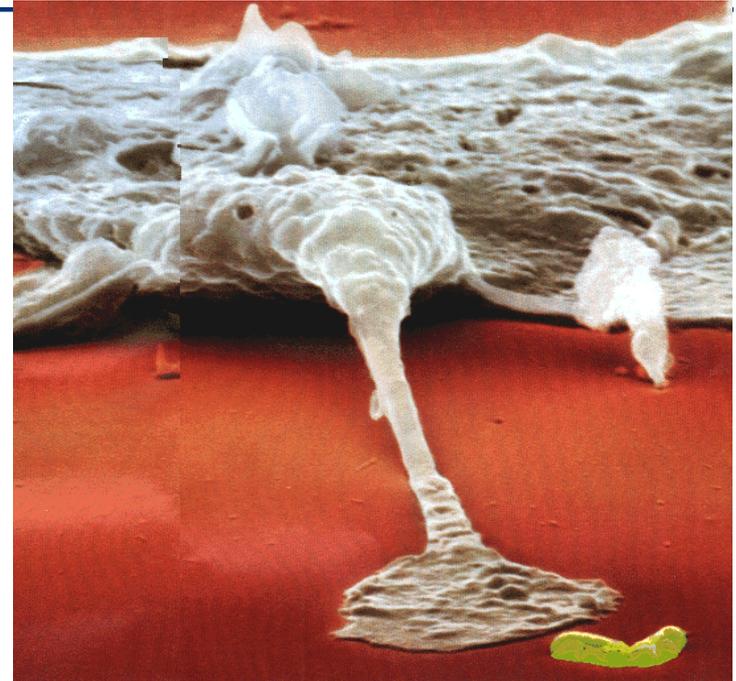


# Two Models of Survivability



## Fortress (traditional)

- Impenetrable (hopefully)
- Monolithic
- Single Layer
- Rigid
- Immobile

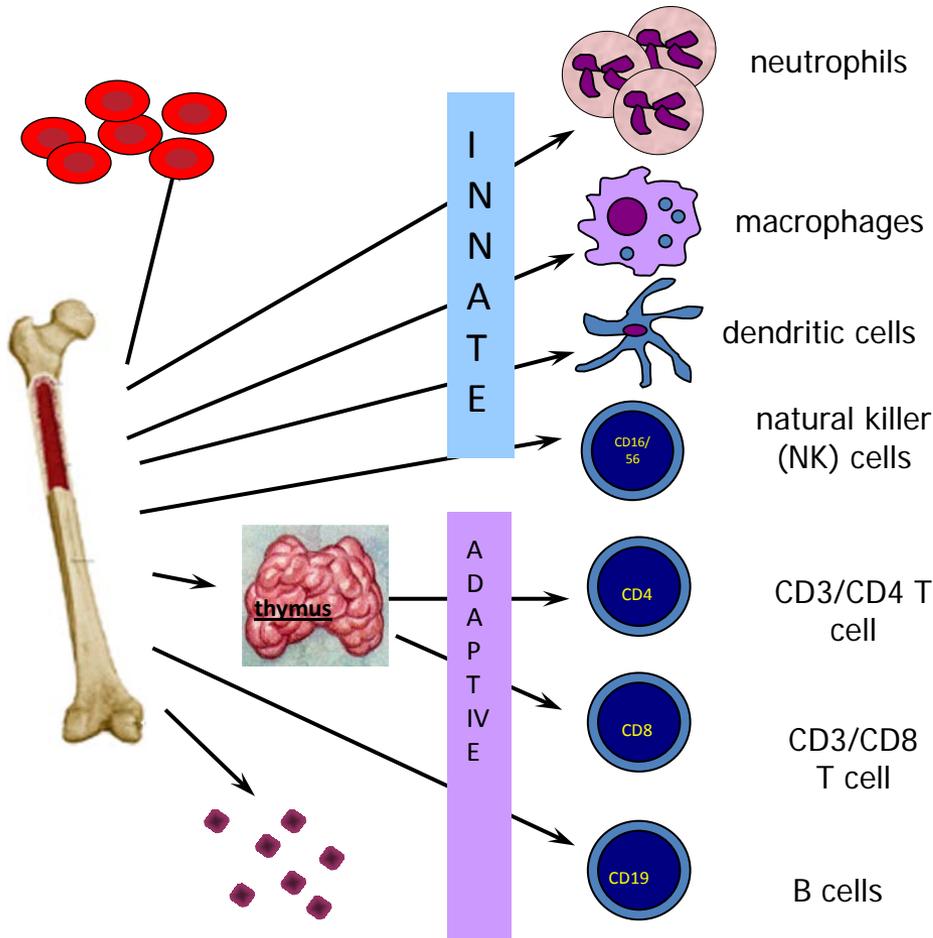


## Organism

- Many partial barriers
- Heterogeneous
- Defense in depth & Self Healing
- Adapts, Learns, Evolves
- Mobile



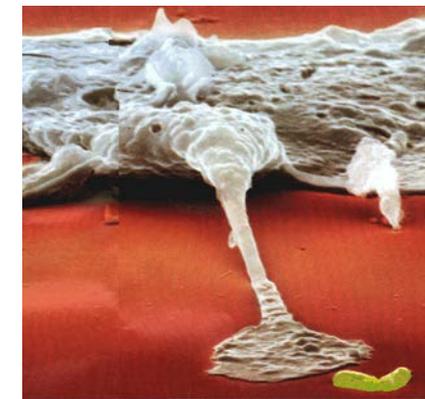
# Humans have Two Immune Systems: Innate and Adaptive



Fast, but inflexible, covers fixed sets of pathogen that are always present. Supports the adaptive immune system.

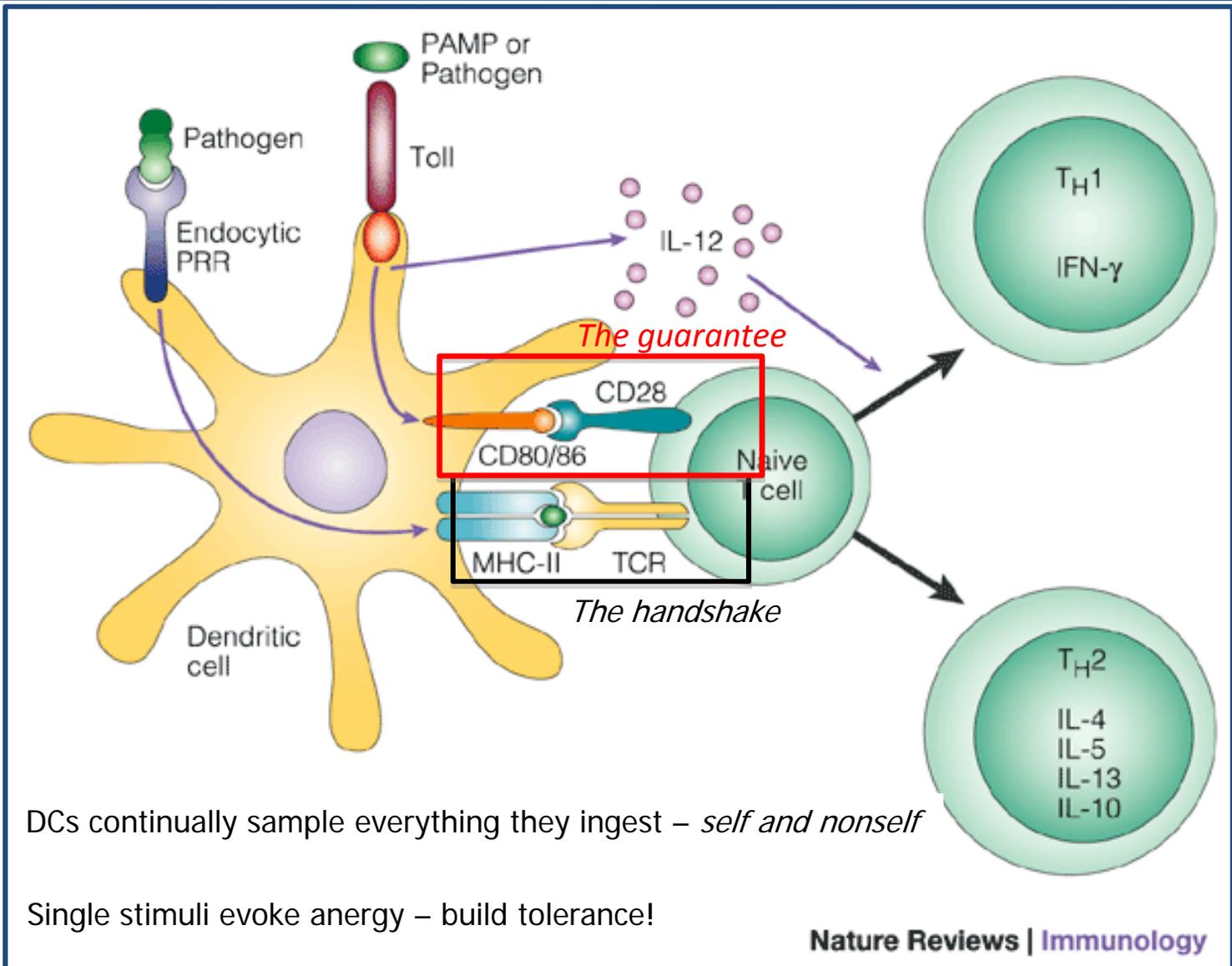


Slower, learns to recognize new sets of pathogens, distinguishes self from non-self, retains memory to guard against future attacks.



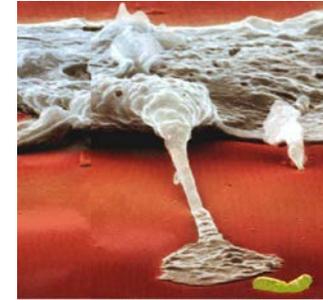
**At least 20 – 30% of the body's resources are involved in constant surveillance and containment.**

# Part of the innate immune system engages the adaptive immune system



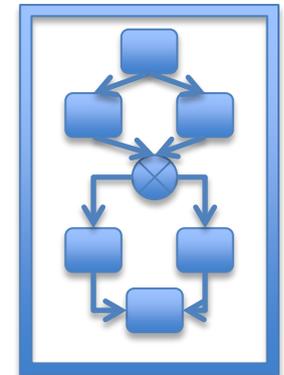
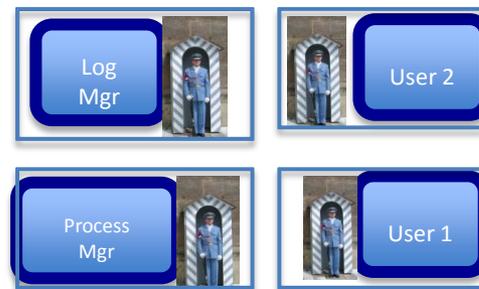
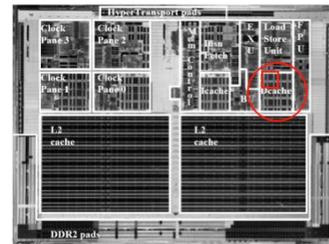
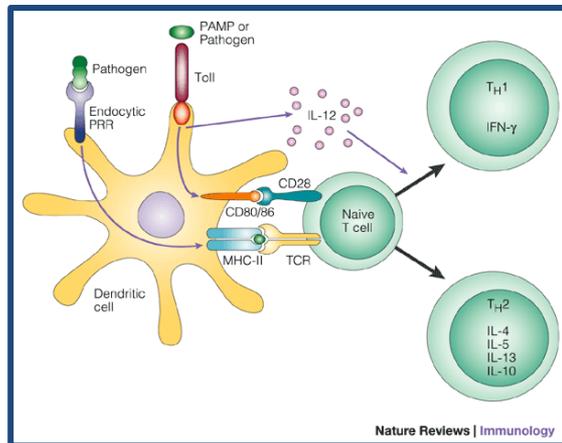


# Biology and Computation: Two Design Styles



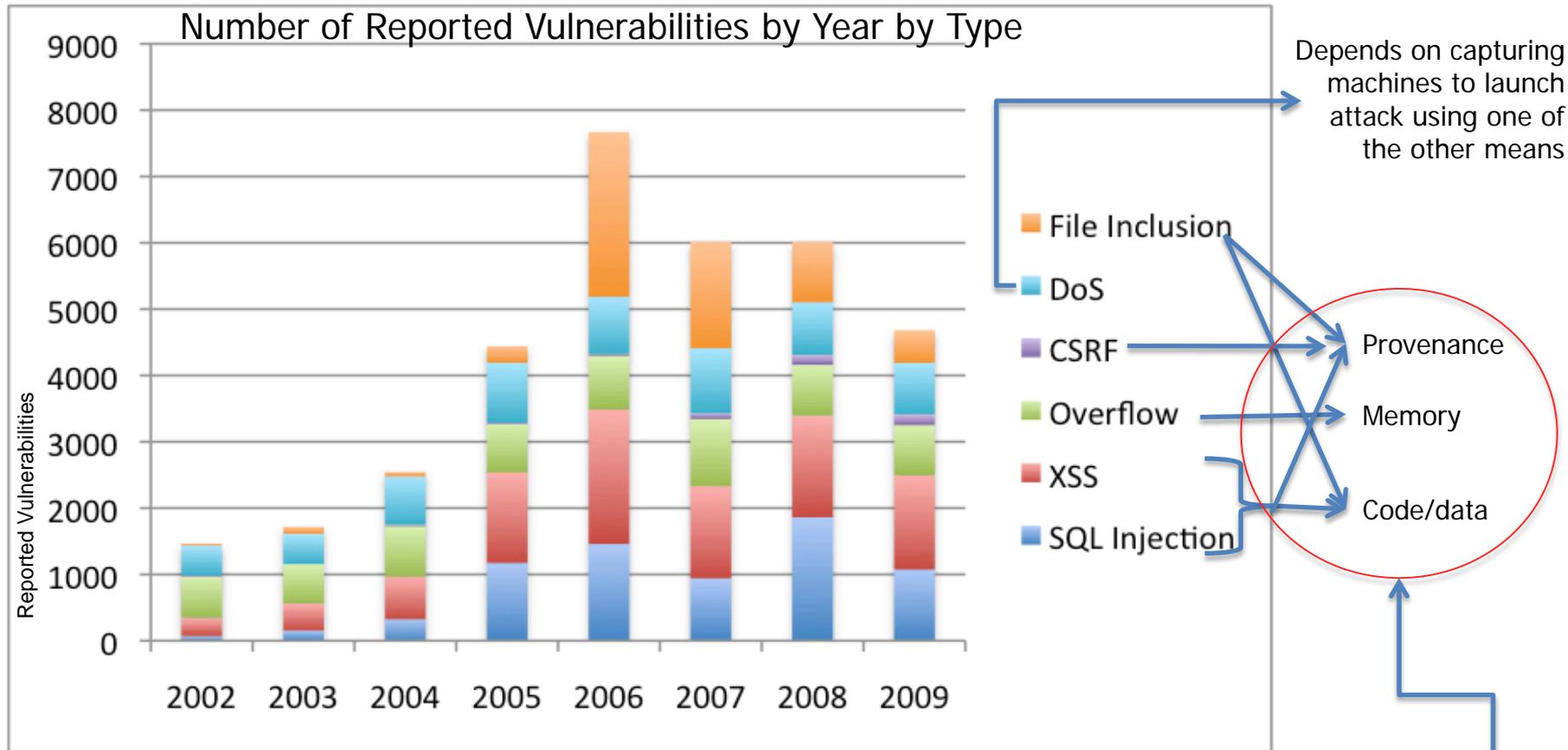
<b>Computation</b>	<b>Biological</b>
Near Perfect Components	Fallible components
Core design formed in era of scarcity	Abundance of resources
Core design formed in isolated environment	Evolution in ecosystem of predators and parasites
Evolutionary pressure from market: price, performance and features	Evolutionary pressure from ecosystem: survivability
Self-regulation and adaptation rarely considered. Runs open-loop.	Self-regulation and adaptation are core mechanisms. Closed loop control.
No enterprise-wide survivability mechanisms	Diversity for population survival Public-health systems in human society

- Computational realization of innate immunity
- Computational realization of adaptive immunity
- These achieve resilience with modest overhead (< 1% area overhead)





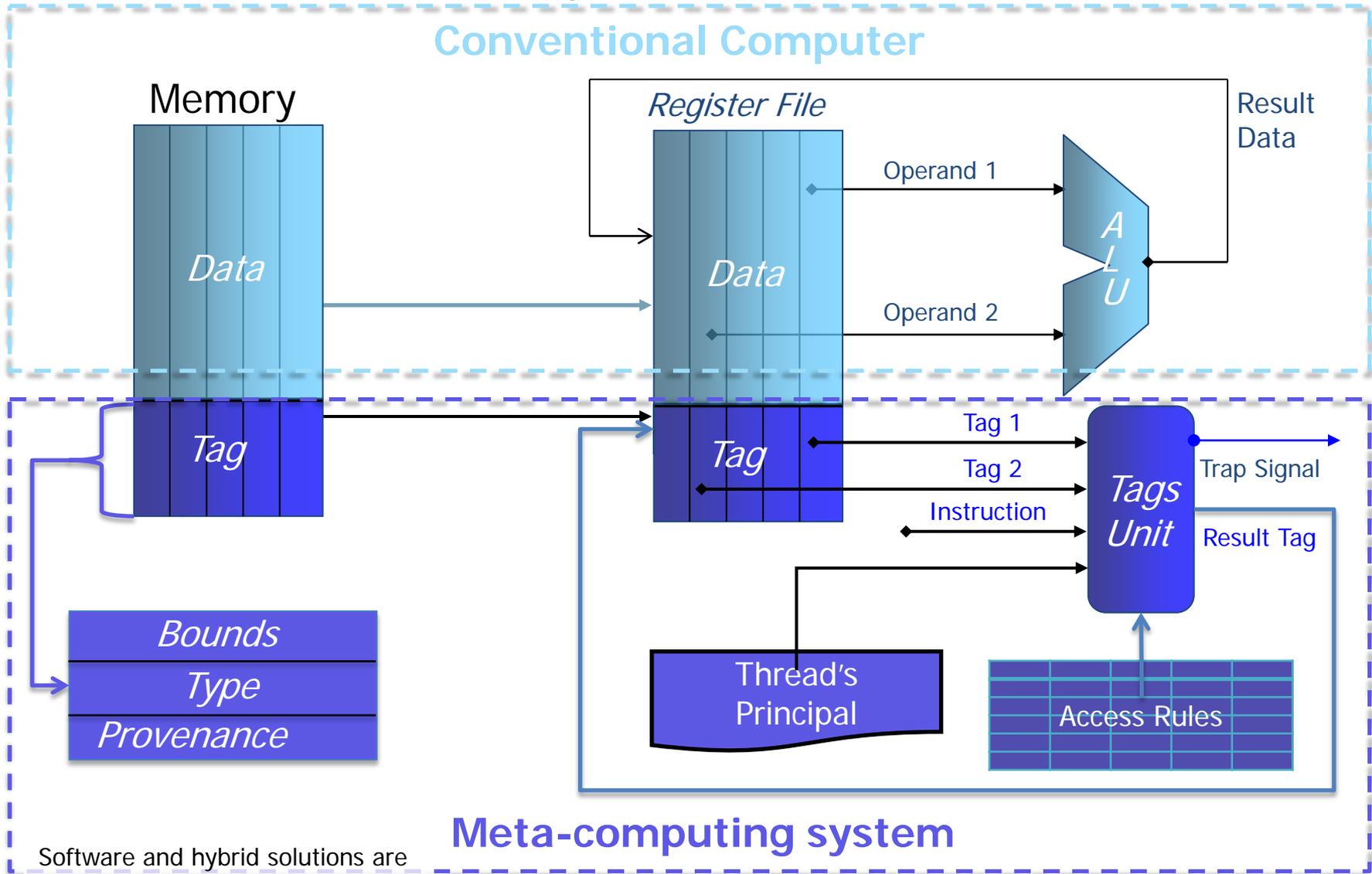
# Almost all current technical vulnerabilities are manifestations of a few root causes



The innate system only has to protect these very few key semantic properties



# Innate immunity: An example hardware solution

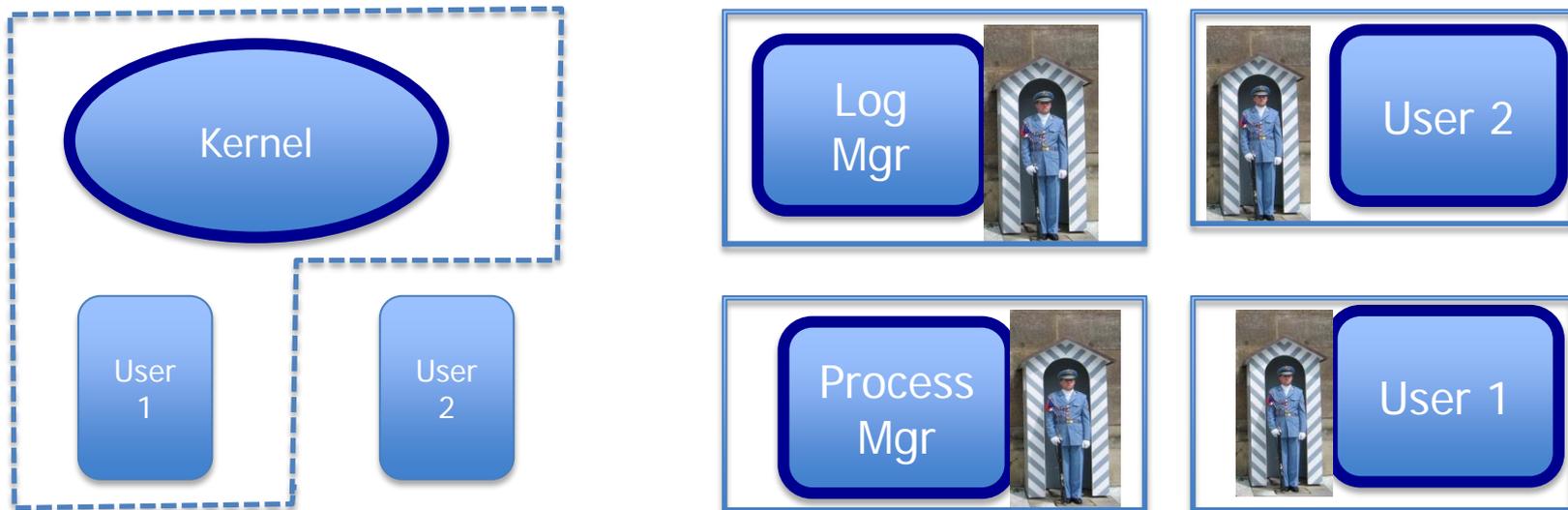


Software and hybrid solutions are also possible (e.g. PROCEED)



# CRASH Innate Immunity Features

- Decomposed kernel
- Many, loosely interacting components
- Components have limited privilege
- Components are isolated using provenance tags
- Components are mutually suspicious
- Controlled information flow
- Non-hierarchical privilege
- OS tools are available to application software.
- Uniform memory management
- Selective Undo as a basic primitive



Achieving these features for innate immunity is possible with a Clean Slate approach.

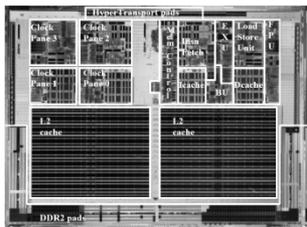
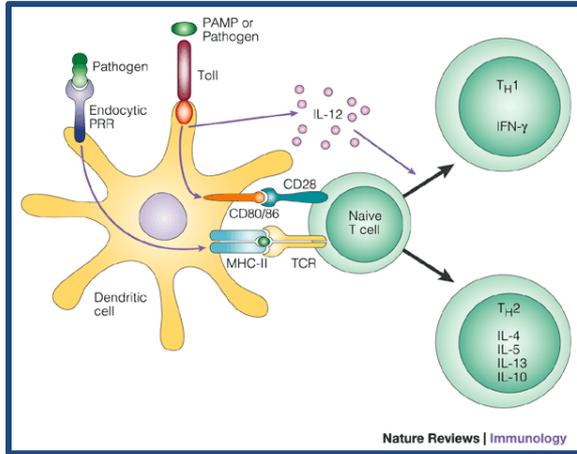


# Existence Proof: Innate Immunity

Task	Technology	
Type Safety	<ul style="list-style-type: none"><li>•Type-safe programming languages</li><li>•Static Analysis</li></ul>	<ul style="list-style-type: none"><li>•Type Tagged hardware</li><li>•Virtual machines</li></ul>
Code-data Separation	<ul style="list-style-type: none"><li>•Different code &amp; data memories</li><li>•Tainting injected interpreted code</li></ul>	<ul style="list-style-type: none"><li>•Type-tagging for instructions</li></ul>
Memory Safety	<ul style="list-style-type: none"><li>•Garbage collected memory systems</li><li>•Software bounds maps</li></ul>	<ul style="list-style-type: none"><li>•Bounds tagged hardware</li><li>•Memory safe pgm'ing language</li></ul>
Provenance	<ul style="list-style-type: none"><li>•"Tainting" tagged hardware</li><li>•Capabilities oriented language</li></ul>	<ul style="list-style-type: none"><li>•Coarse-grained software labels</li><li>•Information flow languages</li></ul>
Separation	<ul style="list-style-type: none"><li>•Separation kernel</li><li>•Clark-Wilson structures</li></ul>	<ul style="list-style-type: none"><li>•Hardware supported compartments</li></ul>
Privilege Management	<ul style="list-style-type: none"><li>•Multics style gates</li><li>•Capabilities</li></ul>	<ul style="list-style-type: none"><li>•Hardware supported procedure access control</li></ul>

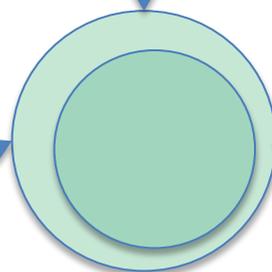
Representative Projects: Microsoft Singularity, MILS, EROS, HiStar, Asbestos, Fox Project, MIT Lisp machine, Multics, Burroughs 5700

# Adaptive Immunity

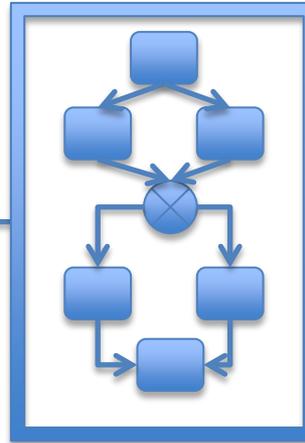


1. Hardware analog of innate immune system detects anomaly

2. Software system analog of adaptive immune system is signaled



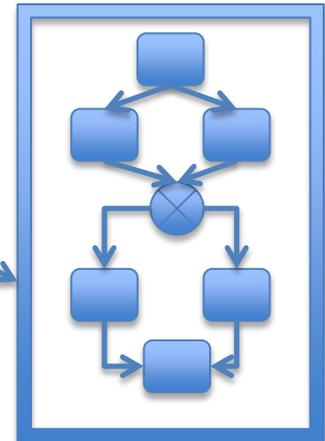
System Model



4. System model is adapted with new attack-specific detector

3. System model is used to perform diagnosis (e.g. localization and characterization)

5. Adaptive immune system synthesizes plan to get around problem and patch to remove specific vulnerability



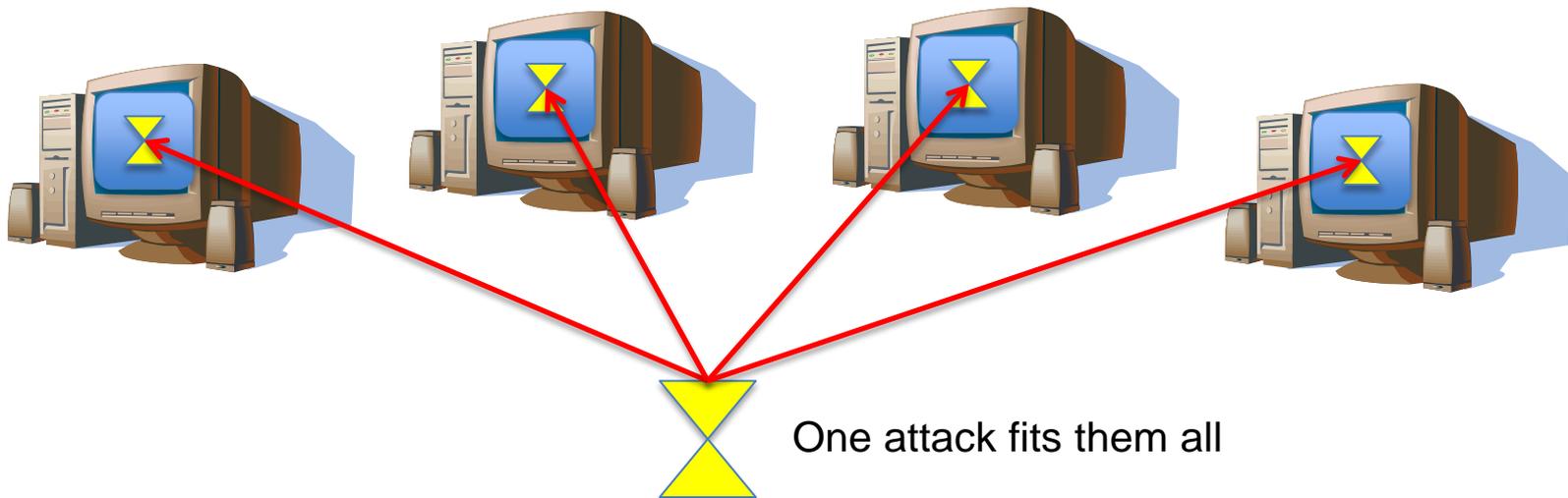


# Existence Proof Adaptive Immunity

Task	Technology	
Detection	Meta-computing System modeling	Architectural Differencing Inline reference monitor
Model Construction	Invariant learning Static analysis	System modeling language support Call sequence learning
Recovery	Selective Undo & Replay Failure Oblivious Computing	Functional Redundancy & Decision Theoretic Dispatch
Repair	Evolutionary Programming	Invariant insertion
	Model-based diagnosis	Model repair

Representative Projects: Clearview(MIT), AWD RAT (Teknowledge/MIT), pH(UNM), DAIKON (MIT), Vernier (SRI/Parc), CORTEX (Honeywell), Assure (Columbia)

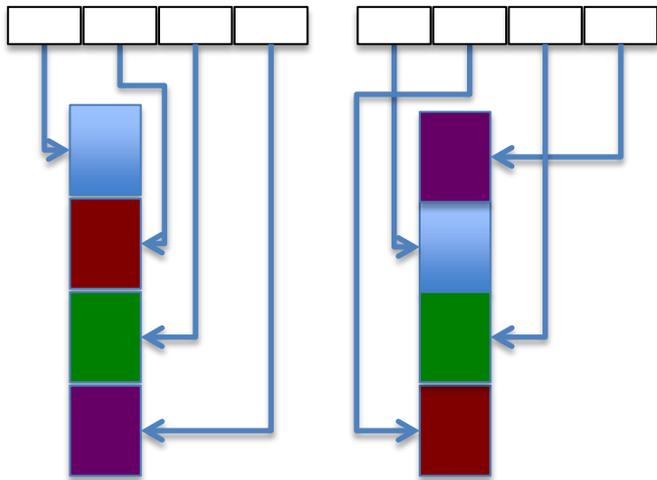
- The attacker's work factor is proportional to Entropy
  - When all systems are the same, a single attack disables them all
  - When a single system never changes, the same attack will work repeatedly
- We currently have a computational monoculture.





# Dynamic Diversity makes a single host different from moment to moment

## Address space randomization



Code and/or data blocks are periodically repositioned in memory so that attacker has to work harder to find a target. Garbage-Collected memory has the property inherently, new methods may optimize for increased entropy.

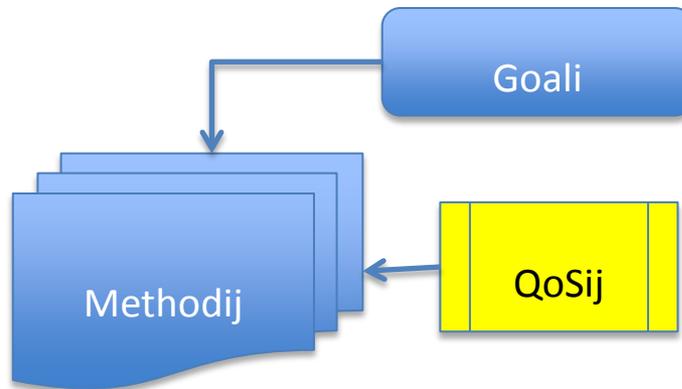
## Instruction set randomization

Disk	Memory	ICache
Instruction-1	Encrypted-1	instruction-1
Instruction-2	Encrypted-2	instruction-2
Instruction-3	Encrypted-3	instruction-3
Instruction-4	Encrypted-4	instruction-4
Instruction-5	Injected-1	Encrypted-1
Instruction-6	Injected-2	Encrypted-1
	Encrypted-5	instruction-5
	Encrypted-6	instruction-6

Code is encrypted as it enters memory and Decrypted as it enters the instruction cache (or translation buffer). Injected code in native instruction set is then encrypted and not executable. Encryption key can be varied by process and time.

## Functional Redundancy & Decision theoretic dispatch

There are multiple methods for achieving each goal ("n-version programming"). Each distinct method has different qualities of service. Method selection is driven both by preferences over QoS and by need for unpredictability.





# Make The Enemy Push the Rock



**Innate immunity** rules out all the standard attacks using hardware mechanisms that cannot be bypassed. There are at least two reasons why any attack won't work, both of which would need to be subverted for the attacker to make progress. Even if an attacker gains some access, his ability to exploit the penetration is limited by the hardware enforced access rules.



**Adaptive immunity** learns to recognize the footprints of novel techniques used by the attacker, catches him earlier in the exploit, prevents him from achieving his goals, and facilitates quicker recovery and regeneration. Innate immunity buys us time for adaptive immunity to take over and increase the attacker's work factor yet further. As time goes on we know more and more about the attacker and how to stop him.



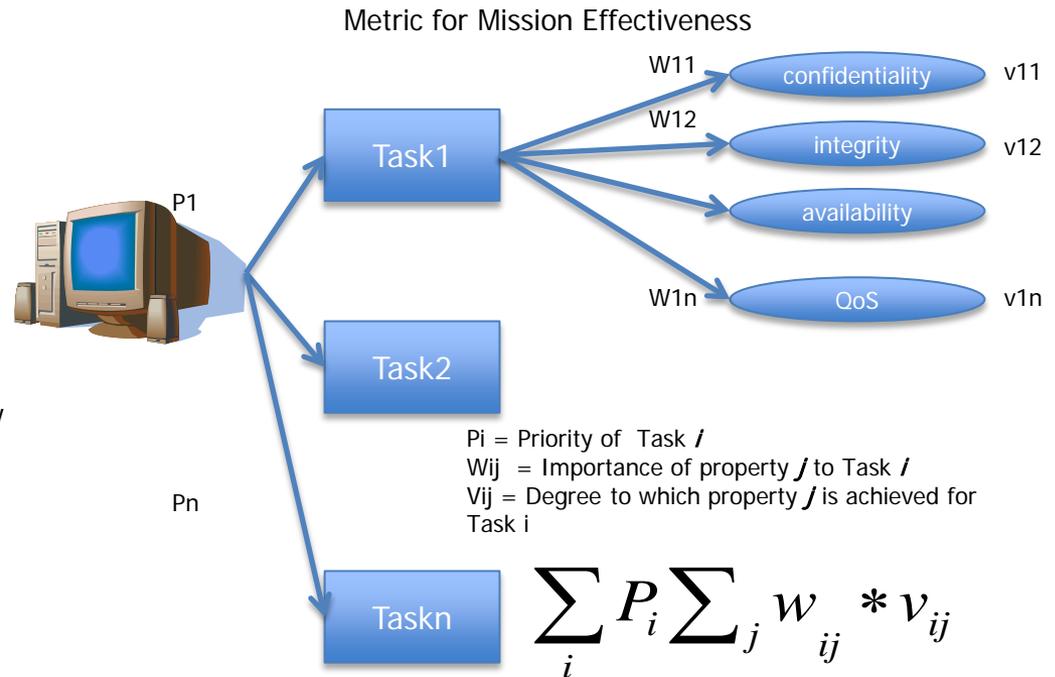
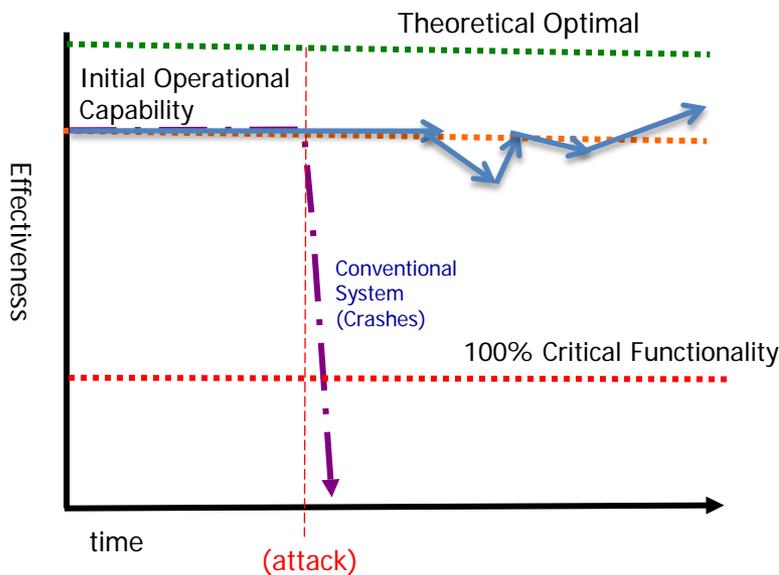
**Dynamic Diversity** guarantees that even if an attacker gets past both innate and adaptive immunity, he still has more work to do because what he thought he knew about us is no longer true. As time goes on we know more and more about the attacker while he knows less about us.



# Objectives

The objective is to sustain mission effectiveness. Different mission components have different security needs and will make different trade-offs at run-time between these, quality of service, and even correctness.

- Provide 100% critical functions at all times in spite of attacks.
- Design out the root causes of all current technical vulnerabilities.
- Adapt around corruptions and recover to initial capabilities after penetrations.
- Learn how to defend against new vulnerabilities to improve robustness over time.





# What Will We Get

## Demonstration computer systems

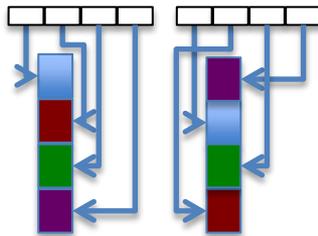


New hardware & operating system architectures that eliminate all common technical vulnerabilities



Adaptive Software that:

- Diagnoses root causes of vulnerabilities and builds situational assessment
- Quickly adapts & reconfigures
- Learns from previous attacks and gets better at self-protection



Diversity techniques that:

- Increase entropy in time and space
- Make every system unique
- Raise work factor of attacker for each system

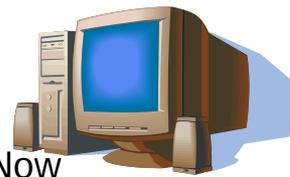


# Transition targets within DARPA & DoD

- Conventional computing will represent a diminishing part of computing base
- 3 new areas will come to dominate
- These have high security requirements and low legacy constraints
- CRASH technology is aimed at these 3 areas



Conventional desktop computing's share of the market will shrink over the next 5 – 10 years





# Program Structure

Program Area	Topics
New Processor Design	<b>Innate immunity</b> Type & memory safety, Meta-data processing
New OS and System Design	<b>Innate &amp; Adaptive Immunity, Diversification</b> Decomposition, least privilege, separation of privilege, information flow management
Languages & Environments	<b>Innate &amp; Adaptive Immunity,</b> Information flow, type safety, model capture, interaction with formal methods
Adaptation	<b>Adaptive Immunity</b> System modeling & machine learning, self monitoring and diagnosis, self-adaptive software frameworks, automatic patching,
Dynamic Diversity	<b>Diversification</b> memory and instruction set randomization
Formal methods & analysis techniques	Assessing resilience, metrics, co-design of hardware, languages, OS and formal methods, information flow proofs, verification of security properties
Application demonstrators	New demo min-apps built to exploit, demo, & test full framework
Red teams	Voice of the offense in the design process throughout program lifetime
Incentives and market analysis	Workshops and analyses of opportunities & incentives for transitioning technologies into DoD & mainstream



[www.darpa.mil](http://www.darpa.mil)