The Evolution of System-call Monitoring

Stephanie Forrest
Steven Hofmeyr
Anil Somayaji

THE UNIVERSITY of NEW MEXICO

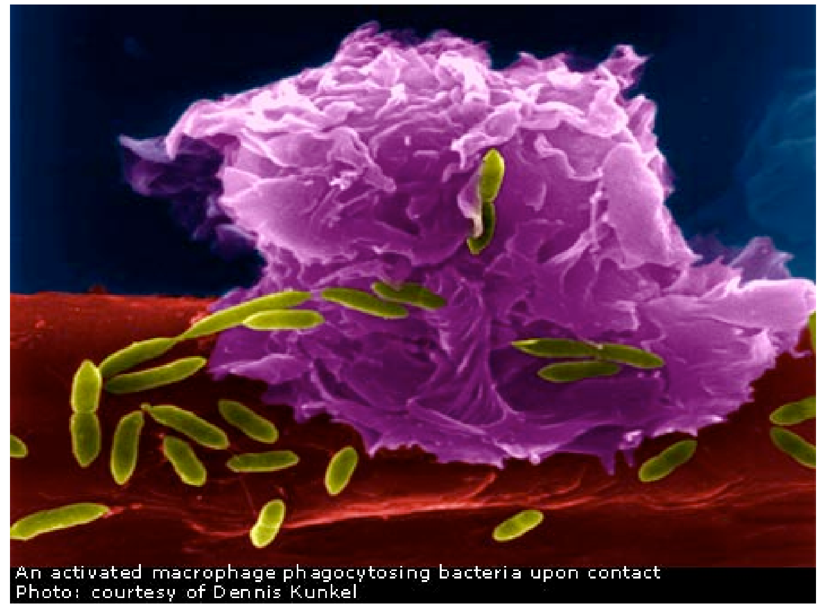
December, 2008

Outline of Talk

- A sense of self for Unix processes (Review)
  - Emphasize method rather than results
- Evolutionary innovations
- General principles and lessons learned

Background
The immunological perspective

- The problem the immune system solves for the body is (almost) the same as the problem we want computer security to solve for our computers:
  - Detecting unauthorized use of computers, computer viruses, etc.
  - Choosing and mounting an effective response.
- Sophisticated IDS and response
  - Detect and stop attacks automatically in real time
  - Focus on system call monitoring



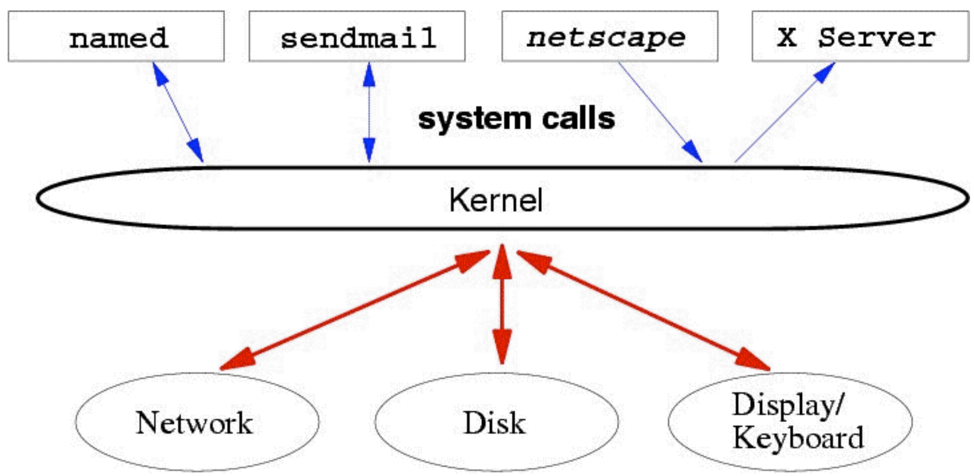
THE UNIVERSITY of NEW MEXICO

The biological perspective led to a set of general design principles

- Autonomy
  - On-line, real-time automated response
- Simple and generic
  - Anomaly detection, focus on executing code
- Adaptable to changing programs and environments
- Diversity
  - Of the defense mechanism and the host itself

A Sense of Self for Unix Processes *(IEEE S&P, 1996)*

- Collect system-call data for normally operating programs (time series)
- Build a profile of normal behavior based on these data
- Observe more (possibly anomalous) behavior
- Treat discrepancies as anomalies
- Sana Security *Primary Response*



THE UNIVERSITY *of* NEW MEXICO

Building the profile

- n-gram representation
- One profile per executable
- Store in fixed size array
- Profiles
  - 1 training array
  - 1 testing array
- Heuristics

open, read, mmap, mmap, open, getrlimit, mmap close

Call	Position 1	Position 2	Position 3
open	read,getrlimit	mmap	mmap, close
read	mmap	mmap	open
mmap	mmap, open, close	open,getrlimit	getrlimit,mmap
getrlimit	mmap	close	
close			

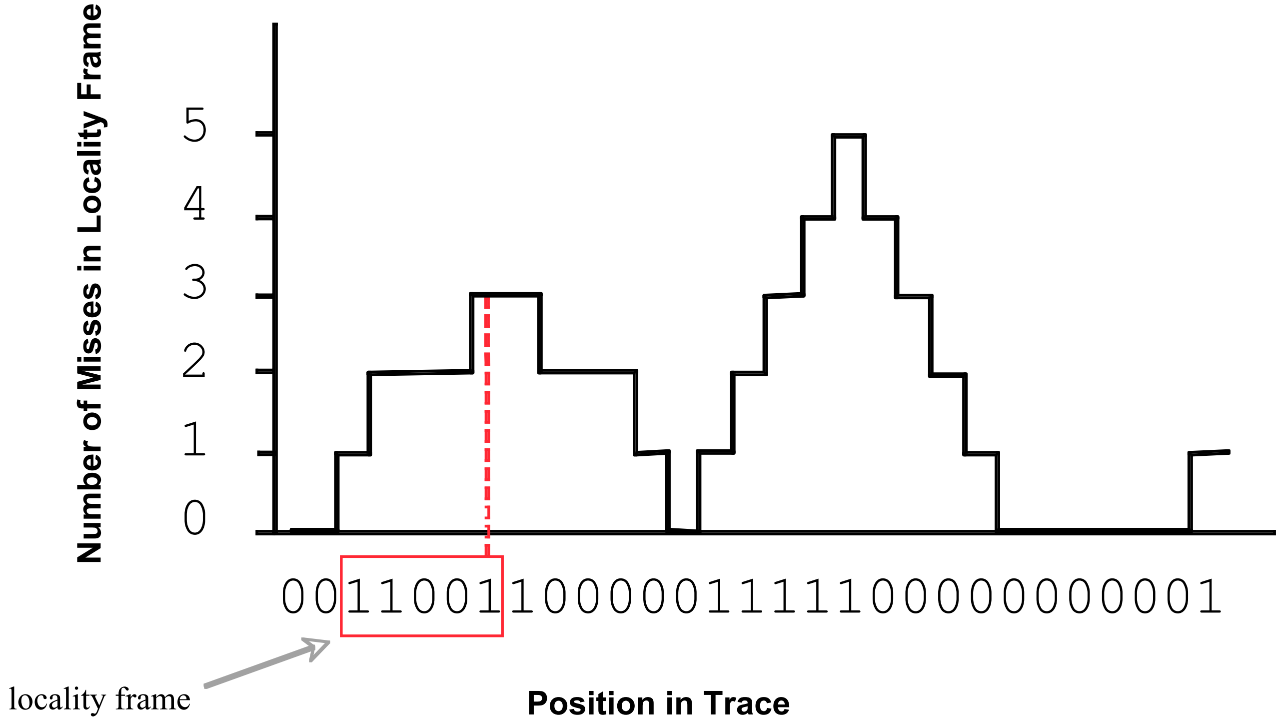
open, read, mmap, mmap, open, open, getrlimit, mmap

Anomalies:

open, open
open, *, getrlimit

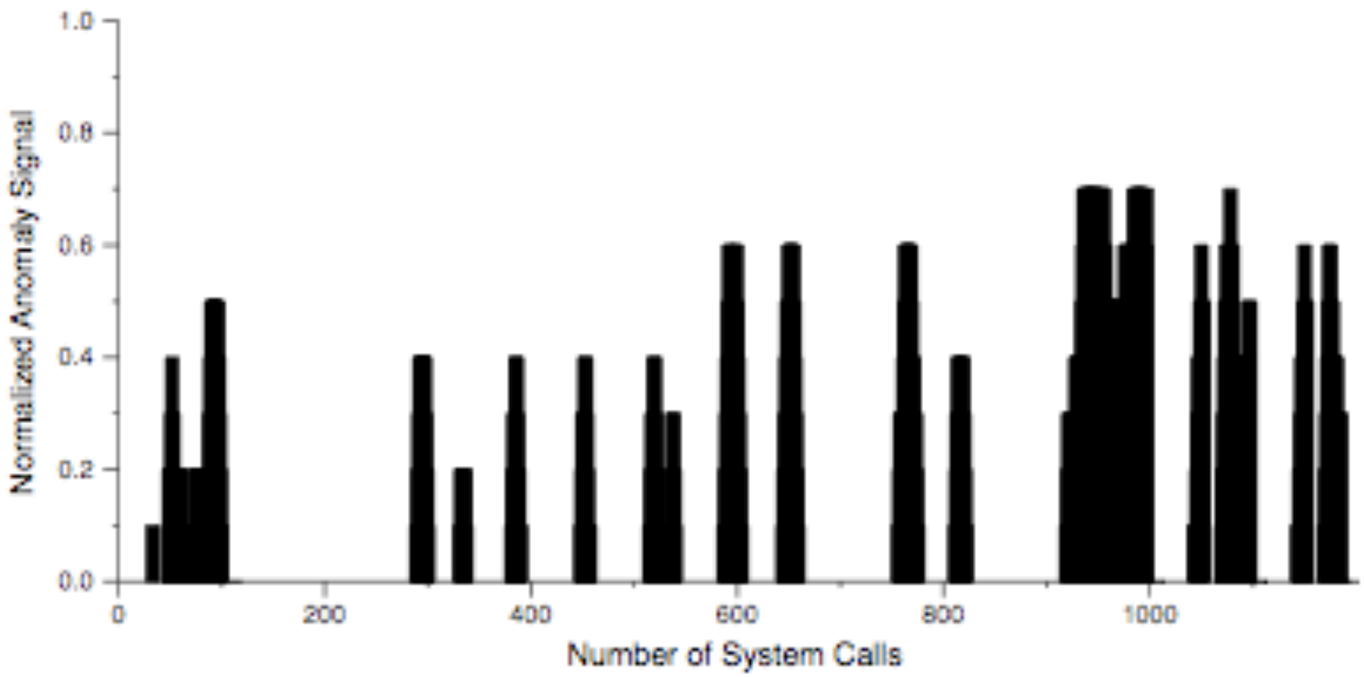
THE UNIVERSITY of NEW MEXICO

Measuring Anomalies



THE UNIVERSITY of NEW MEXICO

Example: syslogd intrusion



Automated Response

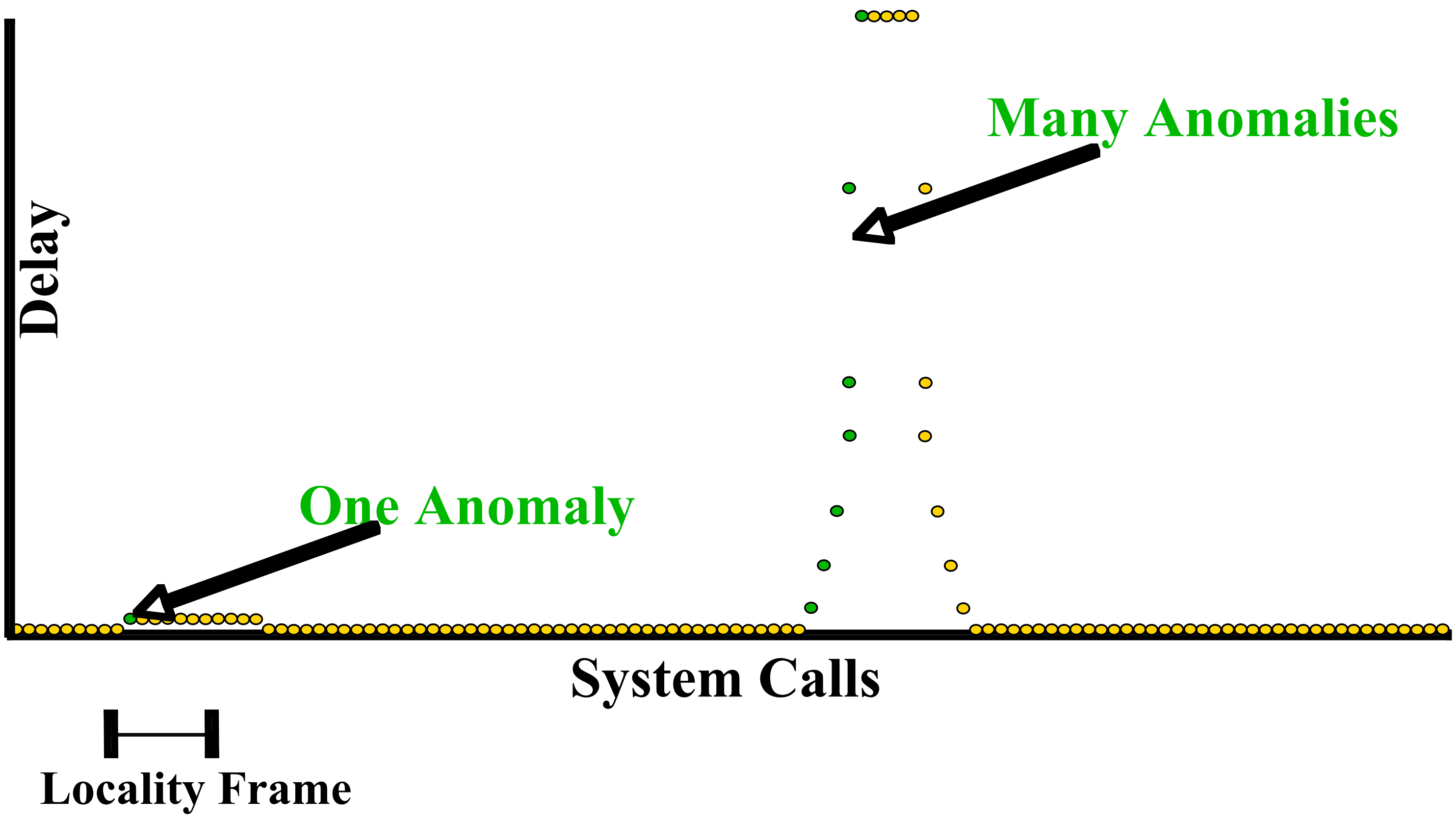
- Intrusion detection incurs a cost of persistent false positives
  - Perpetual novelty
  - Legitimate normal behavior evolves over time
  - Inherent ambiguity between normal and intrusive
- Automated response often ignored because false-positives are expensive
  - Must reduce systems administration burden (rather than increasing it)
  - Must be tolerant of some false-positives

Graduated response

- Process Homeostasis (pH):
  - Computer autonomously monitors its own activities
  - Continually makes small corrections to maintain itself in a “normal” state
- Anomalous sequences trigger system-call delays
  - Exponentially increasing delay
  - Small delays imperceptible to users
  - Long delays trigger timeout mechanisms at network and application level
- HP’s ProCurve network Immunity Manager

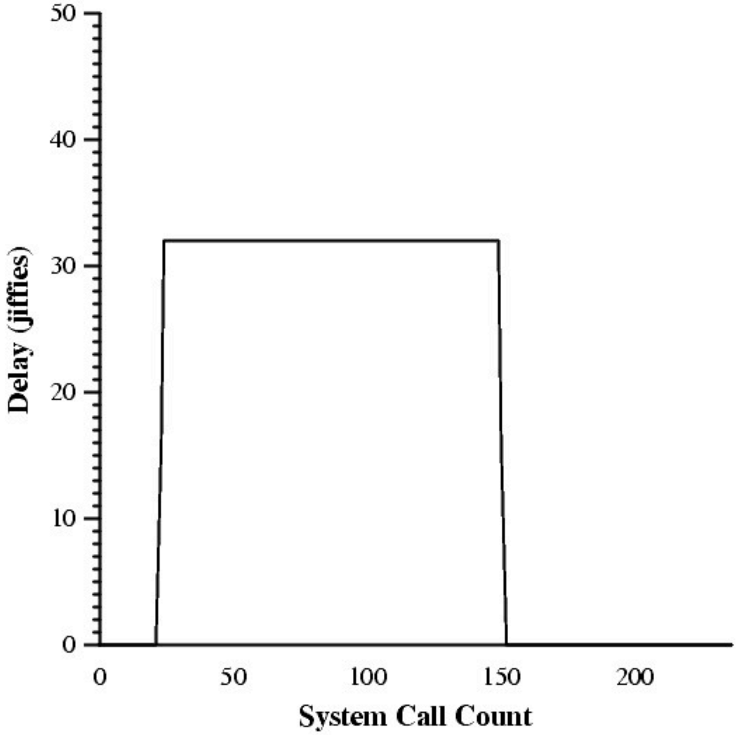
process Homeostasis (pH)

Somayaji and Forrest Usenix, 2000

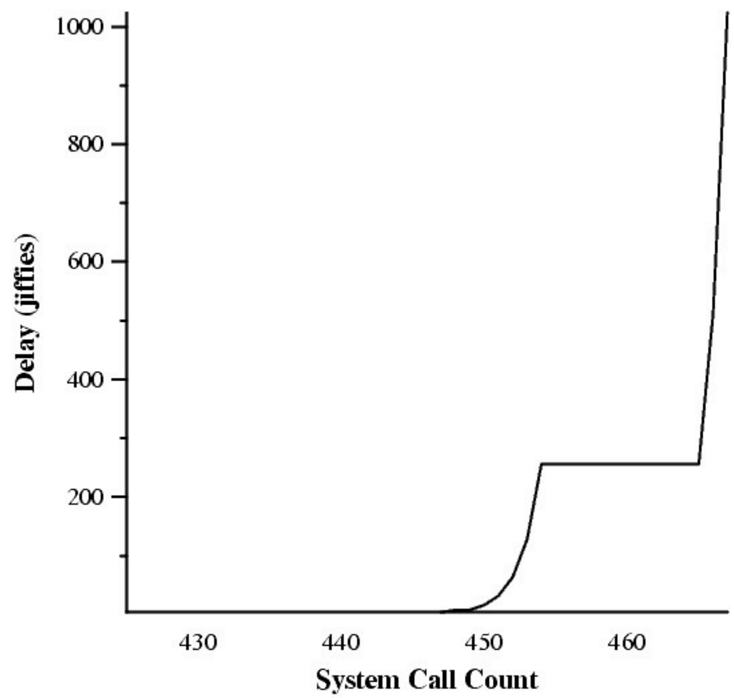


THE UNIVERSITY *of* NEW MEXICO

Stopping attacks in real-time



ssh Trojan program (buffer overflow)



Linux capabilities bug (via sendmail)

Note: Other ssh and sendmail processes unaffected

THE UNIVERSITY of NEW MEXICO

Mimicry Attacks

- Sequences of system calls that exploit a vulnerability but appear normal
  - Relies on successful code injection
  - Code bloat from nullified calls
  - Mimicry has to persist as long as the attacker exploits the process
  - Diversity of normal profiles is a potential barrier
- Also, non control flow attacks

```
read() write() close() munmap() sigprocmask() wait4()
sigprocmask() sigaction() alarm() time() stat() read()
alarm() sigprocmask() setreuid() fstat() getpid()
time() write() time() getpid() sigaction() socketcall()
sigaction() close() flock() getpid() lseek() read()
kill() lseek() flock() sigaction() alarm() time()
stat() write() open() fstat() mmap() read() open()
fstat() mmap() read() close() munmap() brk() fcntl()
setregid() open() fcntl() chroot() chdir() setreuid()
lstat() lstat() lstat() lstat() open() fcntl() fstat()
lseek() getdents() fcntl() fstat() lseek() getdents()
close() write() time() open() fstat() mmap() read()
close() munmap() brk() fcntl() setregid() open() fcntl()
chroot() chdir() setreuid() lstat() lstat() lstat()
lstat() open() fcntl() brk() fstat() lseek() getdents()
lseek() getdents() time() stat() write() time() open()
getpid() sigaction() socketcall() sigaction() umask()
sigaction() alarm() time() stat() read() alarm()
getrlimit() pipe() fork() fcntl() fstat() mmap() lseek()
close() brk() time() getpid() sigaction() socketcall()
sigaction() chdir() sigaction() sigaction() write()
munmap() munmap() munmap() exit()
```

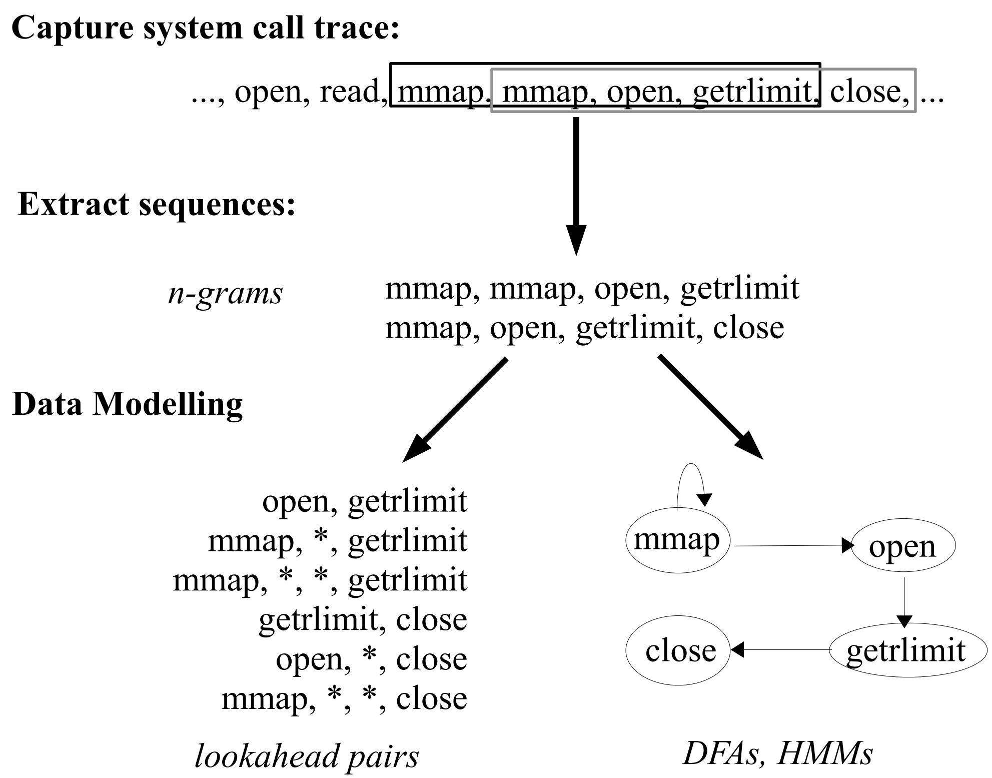
Wagner and Dean CCS 2002

THE UNIVERSITY of NEW MEXICO

Evolutionary Innovations

Many authors (see paper)

- Data modeling methods
- Extensions
  - Data flow (sys call arguments)
  - Execution context (PC)
  - Static analysis
- Other observables
  - Library calls, JVM, HTTP requests, ...



THE UNIVERSITY *of* NEW MEXICO

The biological analogy led to a set of general principles

- Generic
  - Universal weak methods are applicable to many problems
  - Do not require specialized domain knowledge
  - Coverage of a broad range of attacks, but not 100% provably secure

The biological analogy led to a set of General principles

- Generic
- Adaptable
  - To changes in the environment and self
  - Simple learning to construct models and update over time

The biological analogy led to a set of General principles

- Generic
- Adaptable
- Autonomy
  - Graduated response
  - Need for speed dictated simplicity

The biological analogy led to a set of General principles

- Generic
- Adaptable
- Autonomy
- Diversity
  - Each profile is unique, making it difficult for the attacker to predict the profile
  - Led to automated diversity project

THE UNIVERSITY of NEW MEXICO

Lessons Learned

- Designed repeatable experiments
  - Open source code and data
  - Comprehensible system design that focused on one hypothesis
- Careful comparison between methods is difficult
  - Environments are complex and systems difficult to replicate
  - Metrics emphasize breadth of coverage and corner cases
  - Results depend heavily on data set choice; methods might not matter

Conclusion
Engineering practices based on biology

- Why do we need them?
  - Evolution of the software ecosystem (software rot, malware)
  - Dynamic, mobile, complex, and hostile environments
  - Moore's Law won't rescue us
- Hallmarks
  - Simple and generic
  - Computationally and memory efficient
  - Automatically self-tuning, distributable, diverse, and autonomous

THE UNIVERSITY of NEW MEXICO

What I’m doing now

- Autonomous security for autonomous systems (BGP), privacy enhancing data representations (Negative Databases)
- A scaling theory for the rest of computer science
- Using GP to fix bugs in software automatically

THE UNIVERSITY of NEW MEXICO

Biological defense mechanisms
Applied to computation

- Immunology:
  - Protect an individual (single host or a network) against network epidemics and other forms of attack.
  - Antivirus programs, intrusion-detection systems
  - Sana Security *Primary Response*
- Autonomic responses, e.g., homeostasis:
  - Tightly coupled low-level detection/response phases.
  - pH and network (virus) throttling.
  - *HP's Virus Throttle*

THE UNIVERSITY *of* NEW MEXICO

Biological defense mechanisms Applied to computation cont.

- Diversity:
  - Genetic diversity leads to population-level robustness.
  - Disrupt software monoculture using randomization and/or evolution.
  - *Microsoft Vista Address Space Randomization*
- Epidemiology:
  - Network-based control of viruses/worms.
  - Focus on network topology (the epidemic threshold).
  - Survivability and attack resistance (PGBGP---work in progress)

Other biological defense mechanisms
Still to be tapped

- The innate immune system
- Ecological interactions and evolutionary biology
  - Malware ecology: Malware interactions, indicator species, etc.
  - Automated bug repair using evolutionary methods
  - Optimal levels of defense in depth
- Intracellular defenses and repair mechanisms
  - RNA_i
  - Restriction enzymes

THE UNIVERSITY *of* NEW MEXICO

Significance

- Early successful example of anomaly intrusion detection
- On-line, real-time, adaptive, automated response
  - Stops attacks in real-time
- Diversity of protection
- Sana Security started by former UNM student, Steven Hofmeyr

S. Forrest et al. “A sense of self for Unix processes” *IEEE S&P* (1996)

A. Somayaji and S. Forrest ``Automated response using system-call delays.'' *Usenix* (2000)

A. Somayaji ``Operating system stability and security through process homeostasis''
PhD Dissertation (2002)

THE UNIVERSITY *of* NEW MEXICO

Mantra

- The only code that can hurt you is code that actually runs
- Keep it simple stupid (KISS)
- Never let the geeks forget there is a bigger picture
- Nothing says it won't work