Security through Diversity

Adam/Javaman
adam@philtered.net/javaman@ghetto.org

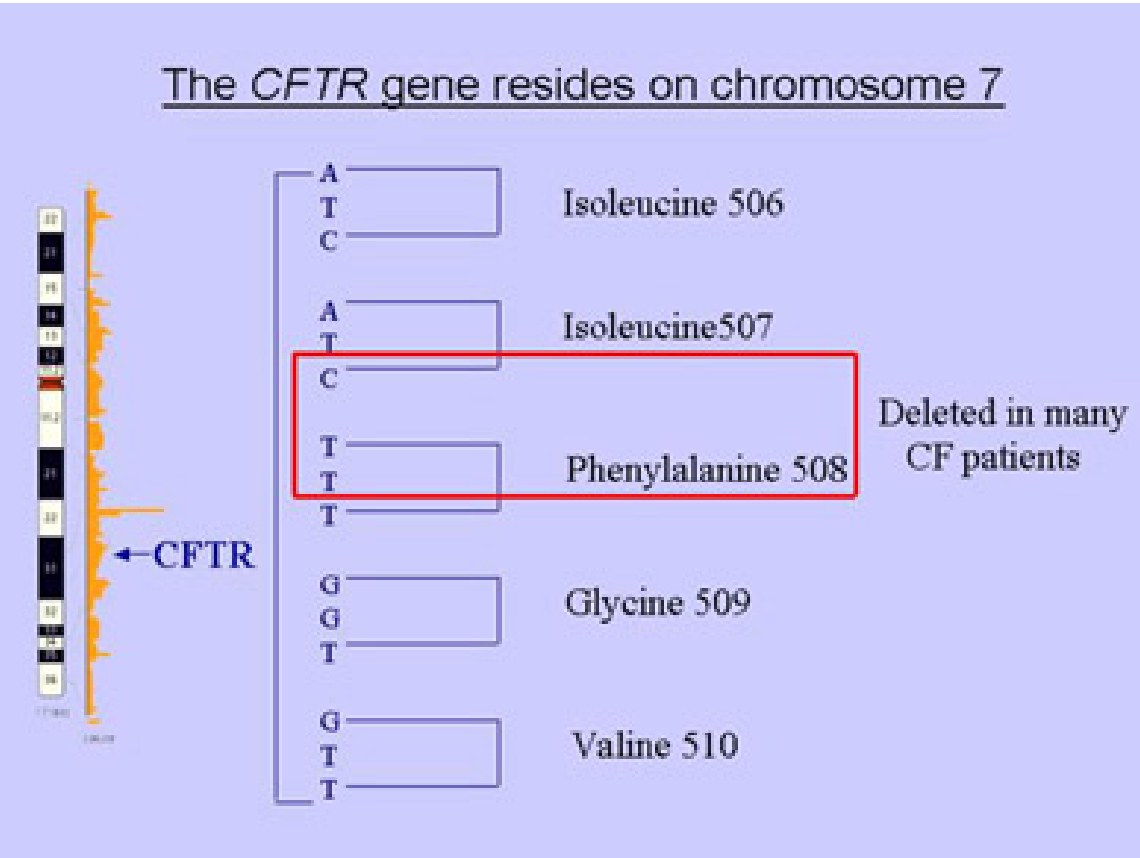
Biological Perspectives

- Biological systems survive not as individuals but as populations.
  - If you were not aware, the world will not stop turning if you die
- From a genetic standpoint, individuals are highly variable, but are similar enough to one another so that we can interact and procreate

Biological Perspectives

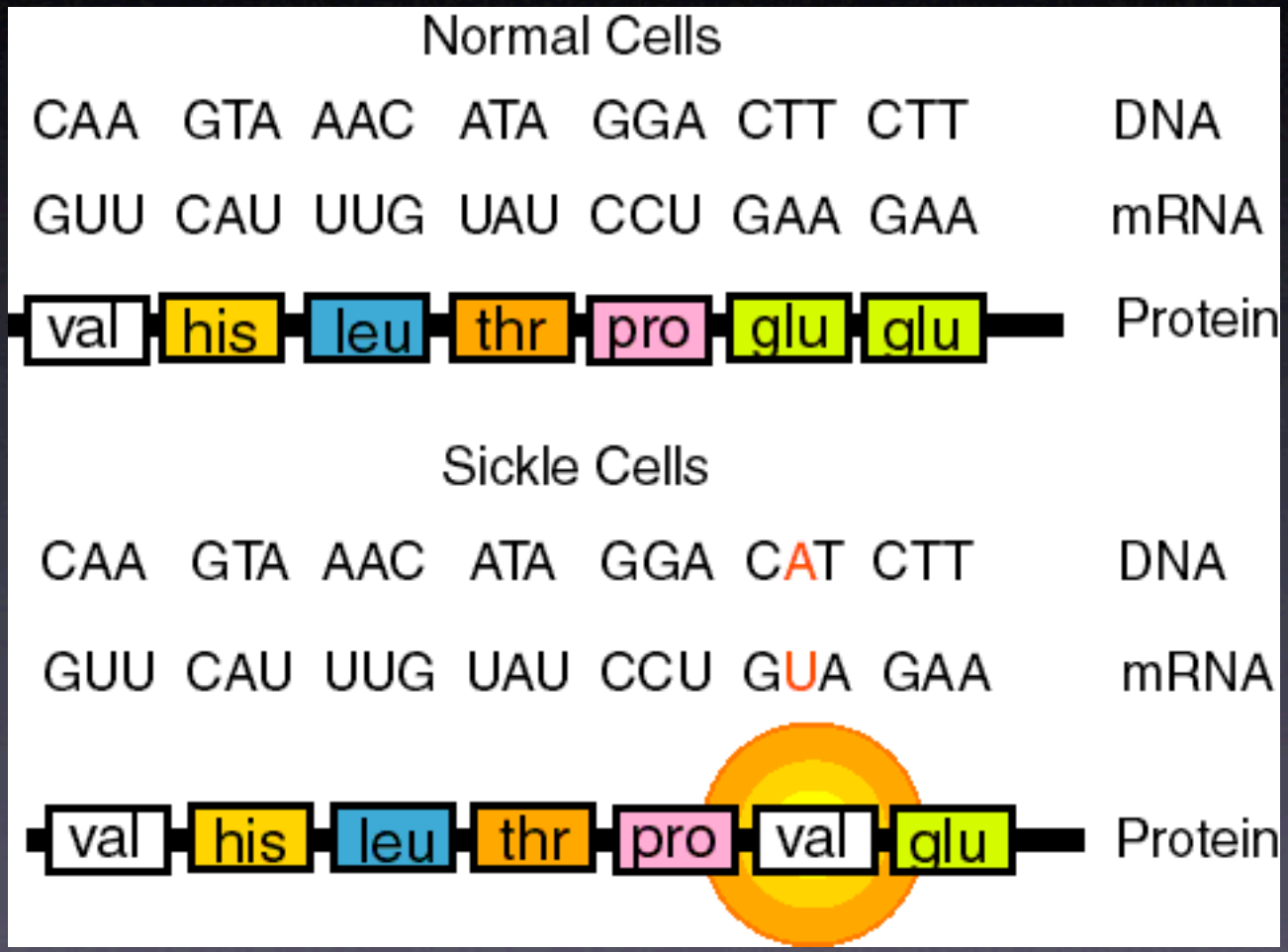
- Minor changes in our genome can give rise to rather devastating illnesses
- Rather than being naturally selected out, these variations have remained with our species... but why?

Base Pair Deletion

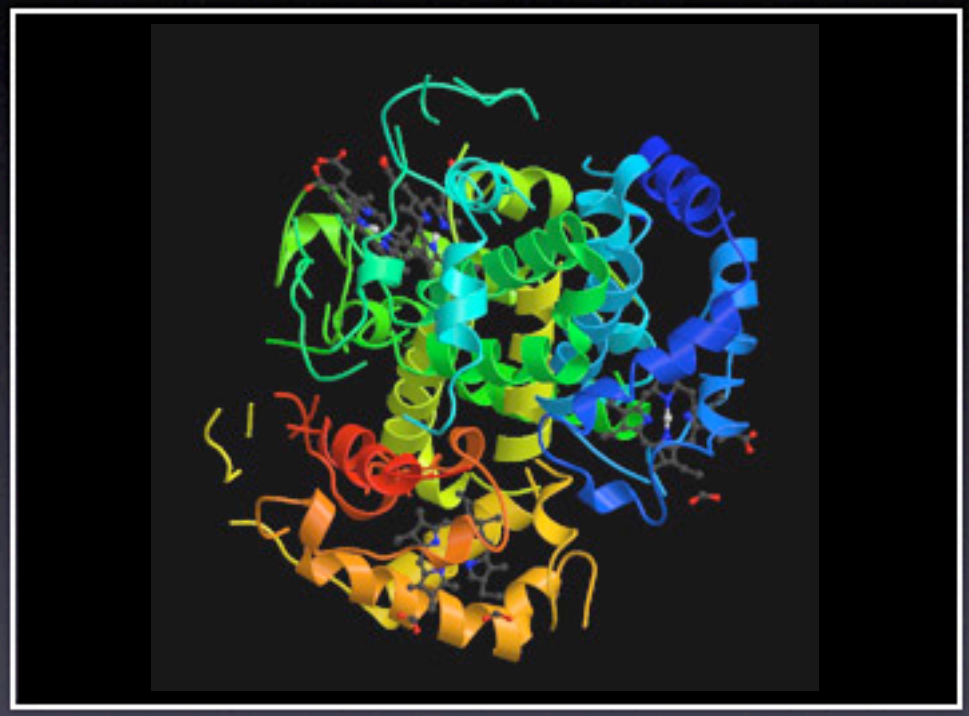


Effect

Single Base Pair Errors

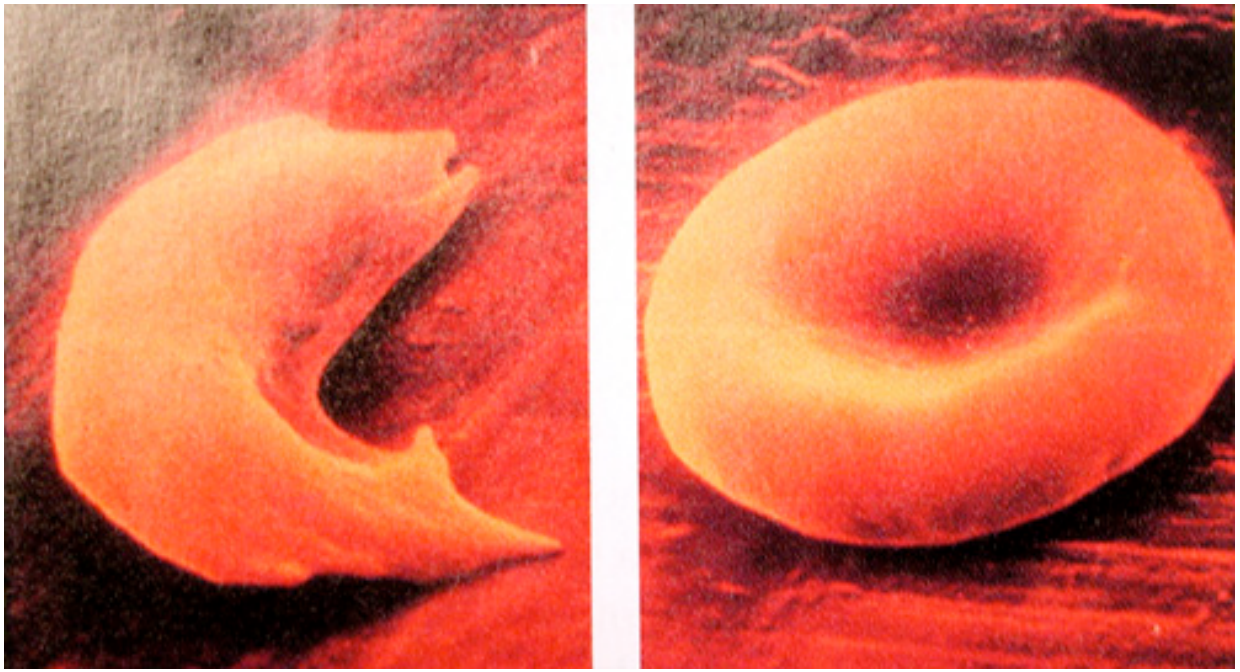


HbA



HbB

Effect



Biological Perspectives

- Single chromosome CTFR gene mutations protect against cholera
- Likewise individuals with HbB show increased resistance to malaria
- Research points to similar genetic resistance against HIV and SARS

Biological Perspectives

- Genetic variations may cause hereditary diseases, but also give us resistance to plagues
- Pathogens that are highly lethal to one set of individuals may cause no sickness in another.
- The diversity of the genetic code of the population leads to resistance in some individuals but not in others.

Biological Perspectives

- The survival of biological systems, including humanity, in response to environmental influences of all sorts has depended upon our genetic variations

Biological Perspectives

- When genetic diversity decreases, however, susceptibility to disease increases
- Correspondingly, systems with little or no diversity suffer catastrophic plagues
- Agricultural practices provide several examples

A Six Pack and a Potato

- The Irish are in America because of poor farming practices involving their food staple
- ... not Guinness...

Irish Potato Famine

- Caused when *Phytophthora infestans* fungus ravaged the Irish potato crop.
- Originated from South America
- Local farmers kept the infection in check by planting a variety of potato crops

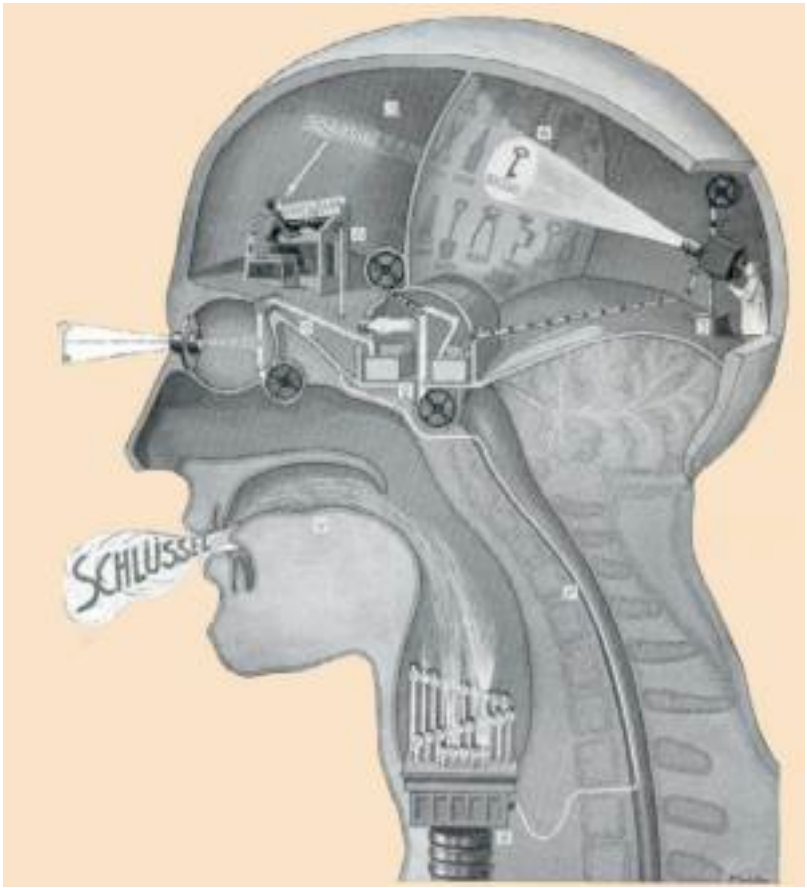
Learning from Mistakes

- Given the effect of the Great Potato Famine, global farming practices were irrevocably changed for the better, with farmers planting a variety of strains of standard food items

Southern Corn Blight

- Another fungus, *Bipolaris Maydis*, ravaged U.S. high yield corn in the 1970's
- Over 15%, or $\$1*10^9$, worth of the crop was lost [Horsfall72]

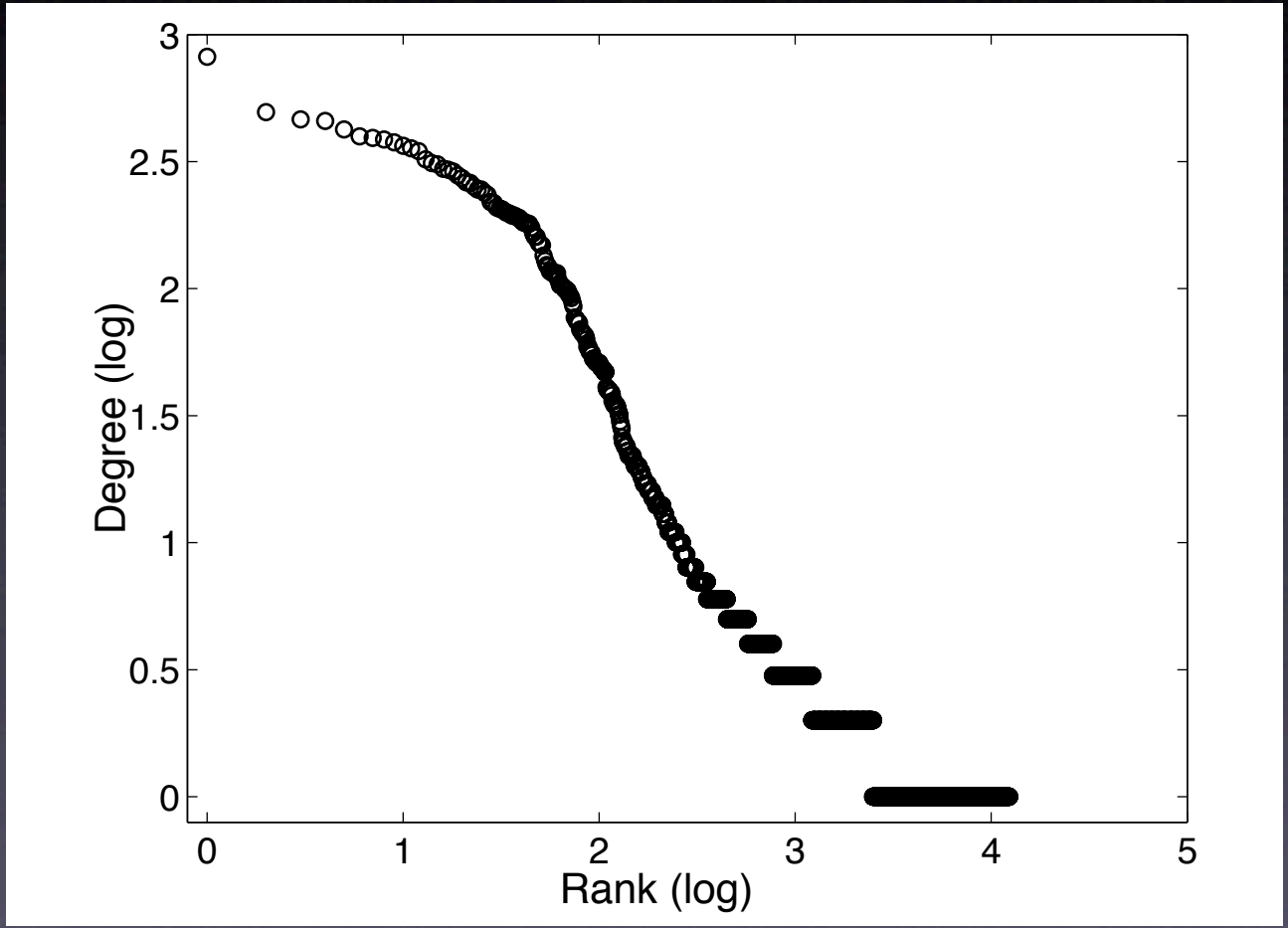
Are computers that different?



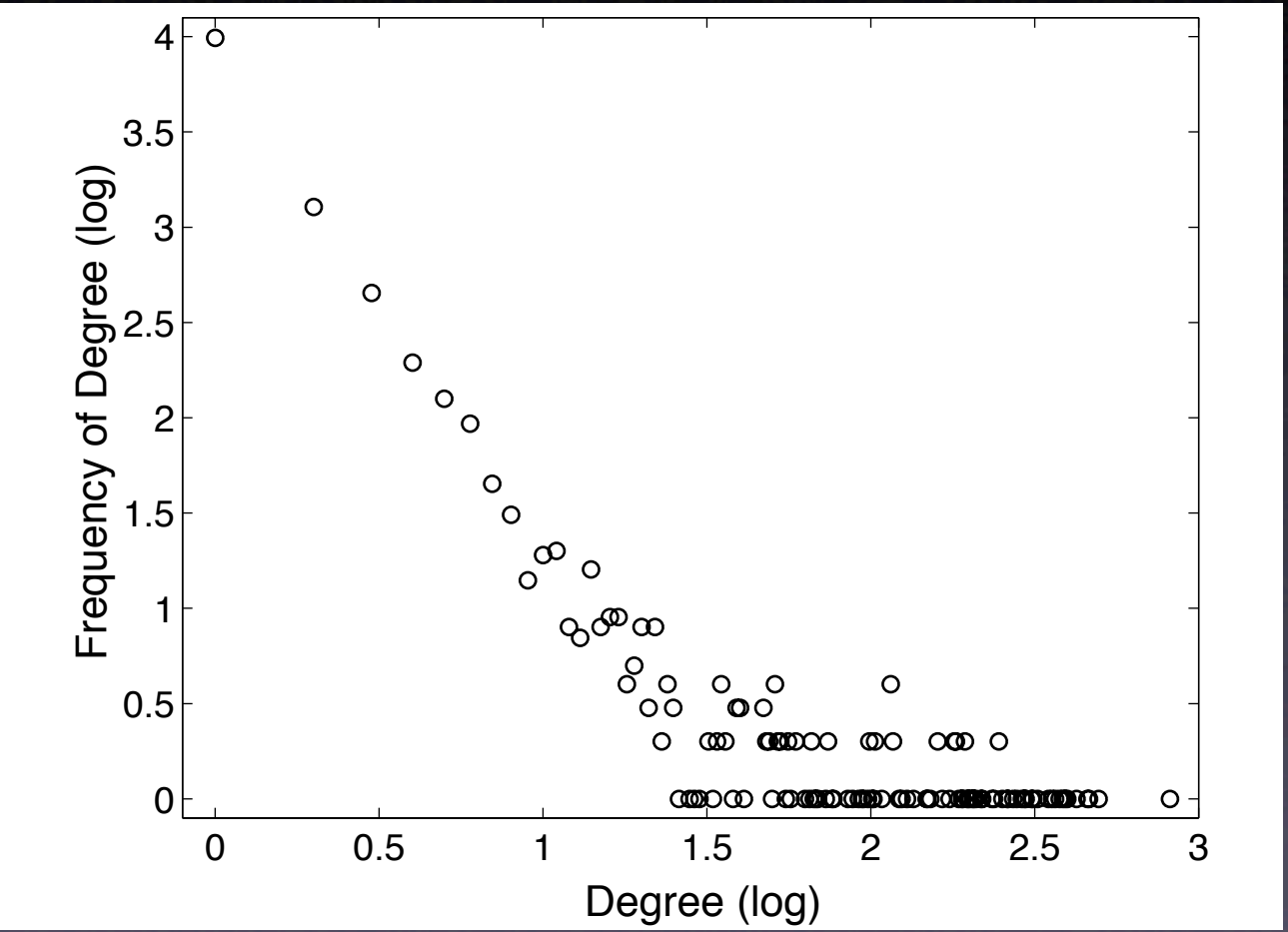
Biological vs. Computational

- Individuals, both silicon and carbon based, are complex systems
- Each composed of millions of lines or amino acids of instructions
- Populations interact in complex networks, which are extremely statistically similar [Faloutsos99, Ebel02, Schroeder92]

Statistics of Networks



Statistics of Networks



Biological vs. Computational

- Individuals from both groups get "sick" because of flaws in their construction, and both suffer from similar epidemics. [Kephart91, Pastor-Satorras01]
- Repairing flaws in construction ranges from nearly impossible in the case of biological systems to nearly impossible in the case of computational systems

The Principle of Security Through Diversity

- Computing systems should emulate biological systems to become survivable in the face of attack and adversity

Emulate?

- Survivability is only achieved for the species and not the individual, with the loss of a single individual being tolerated and expected
- Emulation of biological systems should not stop at organic immune systems and self/ non-self recognition
  - Are these systems true biological emulators?

Emulate?

- Epidemic manifestation closely resembles biological systems
  - Biological systems evolve in tune to viruses
  - Populations are wiped out by their lack of variation
- Populations which should have been wiped out are kept in place by flawed market economics

The Principle of Security Through Diversity

- Described in several position papers [Zhang01, Geer03, Stamp04],
- For good reason, the topic has been extremely controversial

Published Works

- Y. Zhang, et al, "Heterogeneous Networking: A New Survivability Paradigm", 2001
- D. Geer, et al, "Cyberinsecurity: The Cost of Monopoly", 2003
- A. Stamp, "Risks of Monoculture", 2004

Go Forth and Diversify!

- It should be the job of the security community to encourage/enforce diversity by any means necessary
- Implementing any form of diversity on a large scale remains challenge

Diversity Schemes

- Manual and Automatic systems for introducing heterogeneous behavior must be considered

Manual Diversity

- Manual replication of functionality in an uncommon way.

Manual Diversity

- Obvious example: Web Browsers
  - Third party browsers
  - Non-standard plug-ins

Manual Diversity

- Replicate libraries that have already been written in a totally new and more secure way using totally new and more secure programming languages
  - ... for the love of all that is holy, replace OpenSSL

Manual Diversity

- Implement new parsers for old grammars in new ways
  - ASN.1 parser in Lisp
  - Great idea for a senior project

Manual Diversity

- Make it easy to run different operating systems on all types of hardware, with only a few basic requirements
  - Functional out-of-the-box, like knoppix
  - Easy to use for a beginner, (again, like knoppix)

Manual Diversity

- Don’t stop at reinventing the wheel, do things that are totally new!
- Run crypto streams through the GPU, design reconfigurable computing back-ends to handle parser state machines, pass out computations over MPI to terminals in cubicles staffed by illegal immigrants

Manual Diversity

- Caveat: Use existing API interface schemas, or create thin API calls to your libraries, otherwise they may never be used
  - People need to be able to rapidly swap out one library for another in light of a security event
  - Programmers of new applications like familiar interfaces

Manual Diversity

- Idea described by Joseph and Avižienis, "A Fault Tolerance Approach to Computer Viruses", 1988

Automatic Diversity

- Algorithms exist which can automatically introduce variability into multiple levels of system behavior

Current Body of Work

- Introducing randomization on a system-by-system basis has been explored
- Manipulating instruction sets [Barrantes03, Kc03], general stochastic configuration manipulation [Linger99], source code manipulation [Etoh04]
- These techniques often require source code access, and don't take into account the state of the network

Stochastic Structural Manipulation

- Proposed by Linger in "Systematic Generation of Stochastic Diversity as an Intrusion Barrier in Survivable Systems Software", 1999
- Determines program flow and randomly generates a functionally equivalent code flow
- Works at the source level, but can be implemented at the binary level

Randomized Stack Protection

- One method involves insertion of a randomized variable between a targeted buffer and the old frame
- Attacker must correctly "guess" the value held in this canary variable for a targeted function to return and subsequently execute the arbitrary code
- General idea behind the StackGuard project

Randomized Stack/Heap Protection

- Simpler methods that don't require as much code transformation exist
- Randomize the relative location of the stack/heap
- While attacks are still COMPLETELY FEASIBLE, guessing the location of the offset takes time

Address Space Randomization

- Stack Randomization techniques don't protect against return-to-libc style attacks
- Targets to shared libraries can be randomized [think PaX]

Instruction Set Randomization

- Akin to running a processor with an alien instruction set
- Scrambles binaries using XOR encryption, decryption is done during the instruction decode stage of the processor

Instruction Set Randomization

- Can be easily done using runtime emulation schemes; think Bochs
- Including the technology in hardware would incur little to no speed penalty
  - Even if the critical path lies in the IF/ID stage, pipelining is already implemented in most processors at this stage

Instruction Set Randomization

- See:
  - Kc, Keromytis, and Prevelakis, "Countering Code-injection Attacks with Instruction Set Randomization", 2003
  - Barrantes, et al, "Randomized Instruction Set Emulation to Disrupt Binary Code Injection Attacks", 2003

F.A.C.

- Frequently Asked Criticisms

Criticism #1

- This is just Security through Obscurity!

Answer #1

- The difference between diversity and obscurity is subtle but significant
- Obscurity implies keeping an system closed or hidden inherently improves security
- Diversity implies that all systems will be broken, but utilizing an uncommon system will slow down the attacker

Criticism #2

- This is f---ing impractical! I can't manage this much software!

Answer #2a

- Ideally, diversity can be implemented in a completely transparent fashion, with multiple versions of libraries being continually updated by vendors.

Answer #2b

- Yeah, and redeploying software every day is practical?
  - HINT: patch management
- Also, how practical is every single Internet-wide virus?

Questions?