Lecture 3: Threat Models

CS 181S

September 12, 2018

Idea 1: Eliminate Vulnerabilities

Bases for Trust

Synthetic Trust

Analytic Trust

Axiomatic Trust

- **Synthetic Trust:** Trust derived from modification of the system. Trust in the whole derives from how components are combined. Examples: *OS* isolation, reference monitors, firewalls
-
-

Language Support

```
int a[20];
for(int i=0; i<max; i++){
    a[i]=0;
}
```

```
int a[20];
for(int i=0;i<max;i++){
    if(i<0) signal error;
    if(i>=20) signal error;
    a[i]=0;
}
```

Bases for Trust

Synthetic Trust

Analytic Trust

Axiomatic Trust

- **Synthetic Trust:** Trust derived from modification of the system. Trust in the whole derives from how components are combined. Examples: *OS* isolation, reference monitors, firewalls
- **Analytic Trust:** Trust derived from testing and/or reasoning to justify conclusions about what a component or system will and/or will not do. Trust in an artifact is justified by trust in some method of analysis.
-

Testing

- Goal is to expose existence of faults, so that they can be fixed
- **Unit testing:** isolated components
- **Integration testing:** combined components
- **System testing:** functionality, performance, acceptance

Testing

When do you stop testing?

- **Bad answer:** when time is up
- **Bad answer:** what all tests pass
- **Better answer:** when methodology is complete (code coverage, paths, boundary cases, etc.)
- **Future answer:** statistical estimation says Pr[undetected faults] is low enough (active research)

Testing for security?

Penetration testing

- Experts attempt to attack
  - Internal vs. external
  - Overt vs. covert
- Typical vulnerabilities exploited:
  - Passwords (cracking)
  - Buffer overflows
  - Bad input validation
  - Race conditions / TOCTOU
  - Filesystem misconfiguration
  - Kernel flaws

Fuzz testing

[Barton Miller, 1989, 2000, 2006]

- Generate random inputs and feed them to programs:
  - Crash? hang? terminate normally?
  - Of ~90 utilities in '89, crashed about 25-33% in various Unixes
  - Crash implies buffer overflow potential
- Since then, "fuzzing" has become a standard practice for security testing
- Results have been repeated for X-windows system, Windows NT, Mac OS X
  - Results keep getting **worse** in GUIs but better on command line

Fuzz testing

Testing strategy:

- Purely random no longer so good, just gets low-hanging fruit
- Better:
  - Use grammar to generate inputs
  - Or randomly mutate good inputs in small ways
    - especially for testing of network protocols
  - Research: use analysis of source code to guide mutation of inputs

FindBugs

- Looks for *patterns* in code that are likely faults and that are likely to cause failures
- Categorizes and prioritizes bugs for presentation to developer
- Watch video of Prof. Bill Pugh, developer of FindBugs, present it to a Google audience:

  https://www.youtube.com/watch?v=8eZ8YWVl-2s

Formal Verification

- prove program is correct with respect to some formal specification
- Examples: seL4, CompCert
- Problems: correctness of specification, scale

Bases for Trust

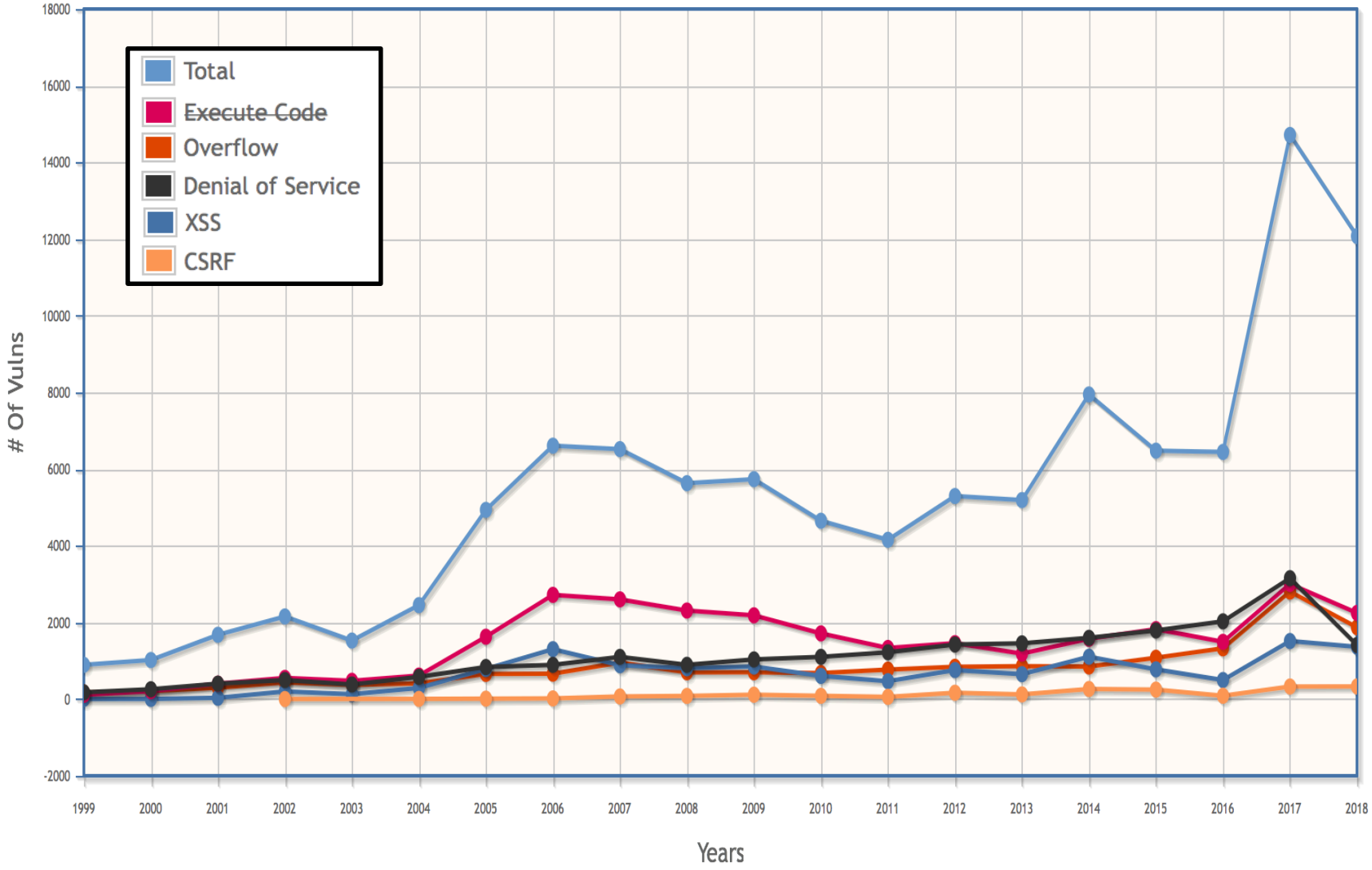
Synthetic Trust

Analytic Trust

Axiomatic Trust

- **Synthetic Trust:** Trust derived from modification of the system. Trust in the whole derives from how components are combined. Examples: *OS* isolation, reference monitors, firewalls
- **Analytic Trust:** Trust derived from testing and/or reasoning to justify conclusions about what a component or system will and/or will not do. Trust in an artifact is justified by trust in some method of analysis.
- **Axiomatic Trust:** Trust derived from beliefs that we accept on faith. We might trust some hardware or software, for example, because it is built or sold by a given company. We are putting our faith in the company's reputation.

Vulnerabilities by Year



Idea 2: Engineer Countermeasures

Attacks
are perpetrated by
threats
that inflict
harm
by exploiting
vulnerabilities
which are controlled by
countermeasures.

Engineering methodology

1. Threat analysis
2. Functional requirements
3. Harm analysis
4. Security goals
5. Feasibility analysis
6. Security requirements

Threats

A principal that has potential to cause harm to assets

- Adversary or attacker: a human threat, motivated and capable
- Sometimes humans aren't malicious: accidents happen
- Sometimes non-humans cause harm: floods, earthquakes, power outage, hardware failure

Threat Models

- Identify threats of concern to system
  - Especially malicious, human threats
  - What kinds of attackers will system resist?
  - What are their **motivations**, **resources**, and **capabilities**?
- Best if analysis is specific to system and its functionality
- Non threats?
  - Trusted hardware
  - Trusted environment
  - e.g., physically secured machine room reachable only by trustworthy system operators

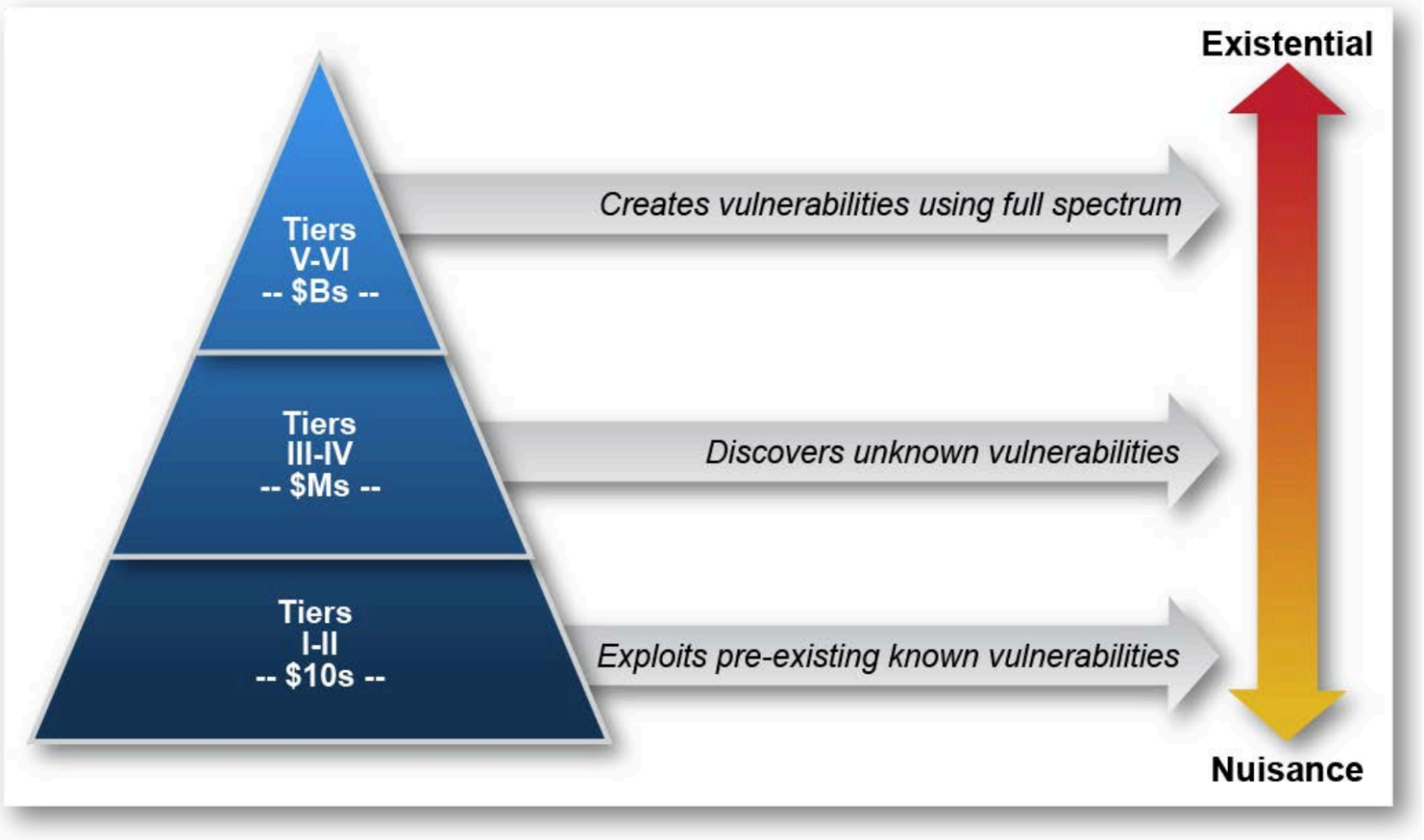
Threats

- Inquisitive people, unintentional blunders
- Hackers driven by technical challenges
- Disgruntled employees or customers seeking revenge
- Criminals interested in personal financial gain, stealing services, or industrial espionage
- Organized crime with the intent of hiding something or financial gain
- Organized terrorist groups attempting to influence policy by isolated attacks
- Foreign espionage agents seeking to exploit information for economic, political, or military purposes
- Tactical countermeasures intended to disrupt specific weapons or command structures
- Multifaceted tactical information warfare applied in a broad orchestrated manner to disrupt major military missions
- Large organized groups or nation states intent on overthrowing a government

Threats (DoD)

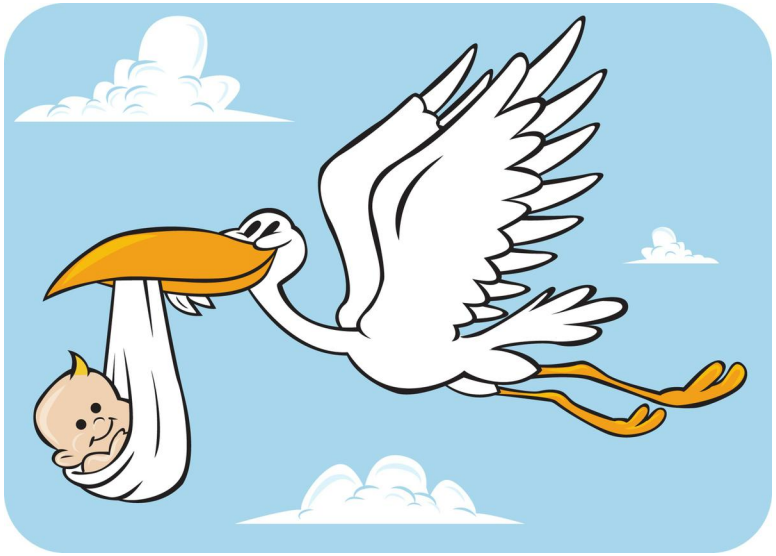
Tier	Description
I	Practitioners who rely on others to develop the malicious code, delivery mechanisms, and execution strategy (use known exploits).
II	Practitioners with a greater depth of experience, with the ability to develop their own tools (from publically known vulnerabilities).
III	Practitioners who focus on the discovery and use of unknown malicious code, are adept at installing user and kernel mode root kits[10], frequently use data mining tools, target corporate executives and key users (government and industry) for the purpose of stealing personal and corporate data with the expressed purpose of selling the information to other criminal elements.
IV	Criminal or state actors who are organized, highly technical, proficient, well funded professionals working in teams to discover new vulnerabilities and develop exploits.
V	State actors who create vulnerabilities through an active program to "influence" commercial products and services during design, development or manufacturing, or with the ability to impact products while in the supply chain to enable exploitation of networks and systems of interest.
VI	States with the ability to successfully execute full spectrum (cyber capabilities in combination with all of their military and intelligence capabilities) operations to achieve a specific outcome in political, military, economic, etc. domains and apply at scale.

Threats (DoD)


Existential
Tiers
V-VI
-- $Bs --
Creates vulnerabilities using full spectrum
Tiers
III-IV
-- $Ms --
Discovers unknown vulnerabilities
Tiers
I-II
-- $10s --
Exploits pre-existing known vulnerabilities
Nuisance

Threat Model = Capabilities

Threat model: The adversary desires to prevent baby deliveries. The adversary has access to radio equipment that transmits and receives on the same frequencies that providence uses for communication with a stork. The adversary also controls weapons systems that can destroy a stork in flight.



Threat Model = Capabilities

- privilege levels
- disk access
- memory access
- physical access
- key access
- network access

Threat Model = Capabilities

- privilege levels
- disk access

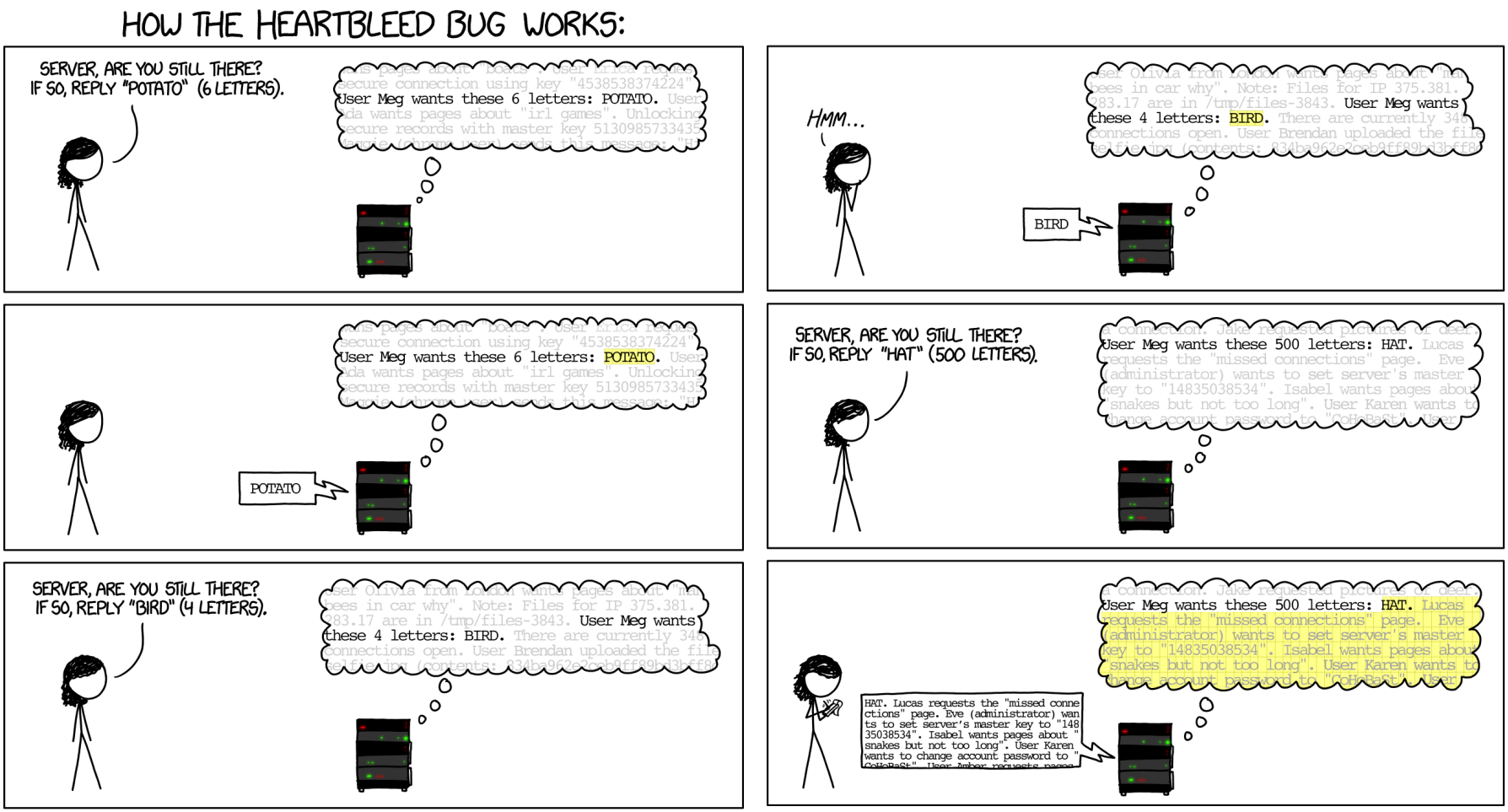

DIRTY COW

Threat Model = Capabilities

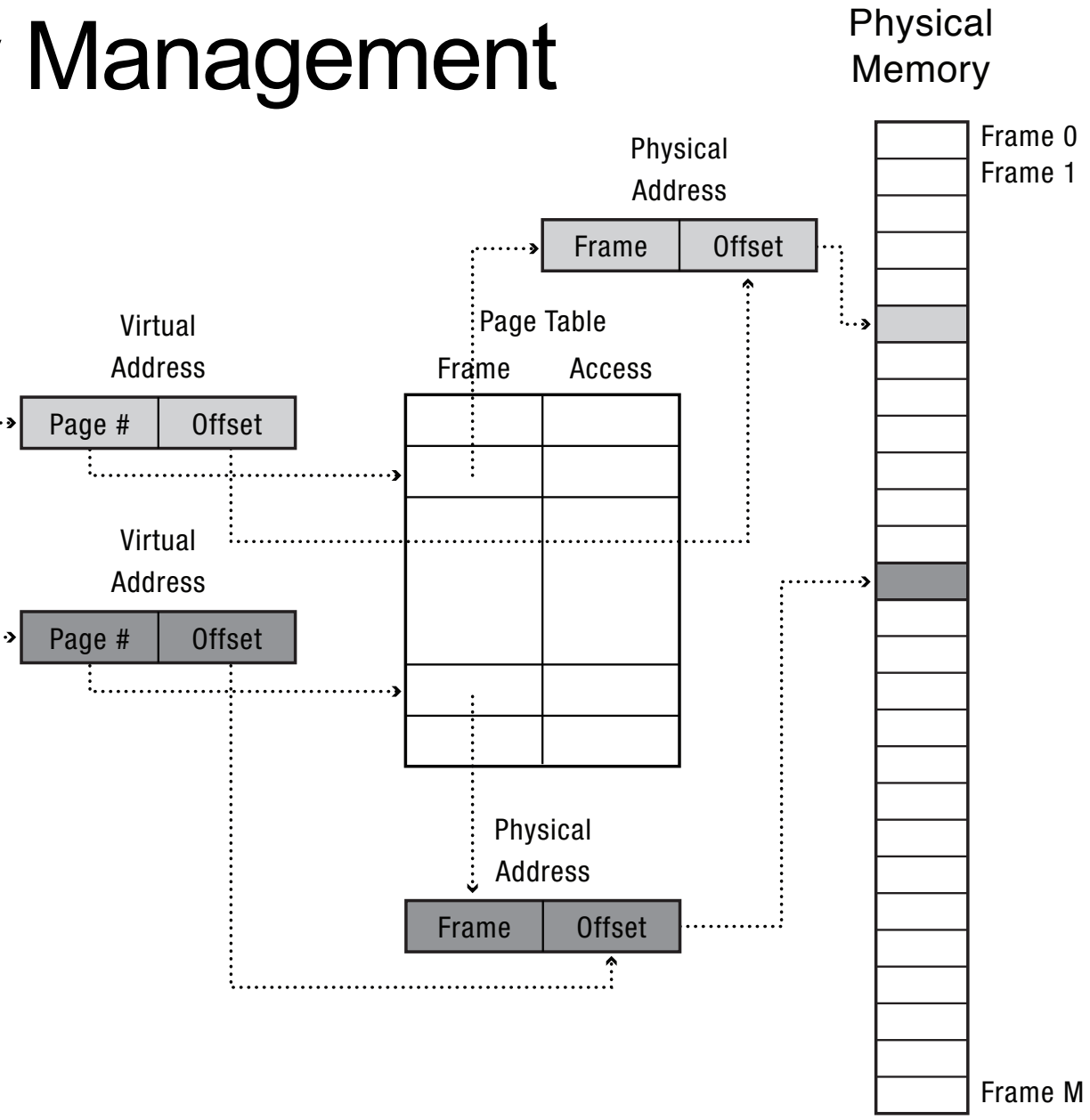
- privilege levels
- disk access
- memory access

Heartbleed

Heartbleed


HOW THE HEARTBLEED BUG WORKS:
SERVER, ARE YOU STILL THERE? IF SO, REPLY "POTATO" (6 LETTERS).
User Meg wants these 6 letters: POTATO.
HMM...
BIRD
User Meg wants these 4 letters: BIRD.
POTATO
SERVER, ARE YOU STILL THERE? IF SO, REPLY "HAT" (500 LETTERS).
User Meg wants these 500 letters: HAT.
SERVER, ARE YOU STILL THERE? IF SO, REPLY "BIRD" (4 LETTERS).
HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to

Memory Management



Speculative Execution

```
int i1, i2;
boolean b1,b2;
boolean[] a1,a2;


if (i1 < a1.length()) {
    boolean bval= a1[i1];
    if(bval){i2= 1;} else{i2= 0;}
    if(i2 < a2.length()){
        b2 = a2[i2];
    }
}
```

Timing


MODE
SPLIT RESET
START STOP
ACCUSPLIT
STOPWATCH
10:5323
MIN. SEC. 1/100
PRO SURVIVOR
601X
3V.1

Threat Model = Capabilities

- privilege levels
- disk access
- memory access
- physical access

Stuxnet

Threat Model = Capabilities

- privilege levels
- disk access
- memory access
- physical access
- key access

FileVault

Threat Model = Capabilities

- privilege levels
- disk access
- memory access
- physical access
- key access
- network access

Network Adversaries

Attacker Properties			
Membership	insider		outsider
Method	active		passive
Adaptability	dynamic		static
Organization	cooperative		individual
Scope	global	extended	local
Motivation	malicious	rational	opportunistic

Dyn DDoS

Threat Models