CSE 484 / CSE M 584: Computer Security and Privacy

Autumn 2019

Tadayoshi (Yoshi) Kohno
yoshi@cs.Washington.edu

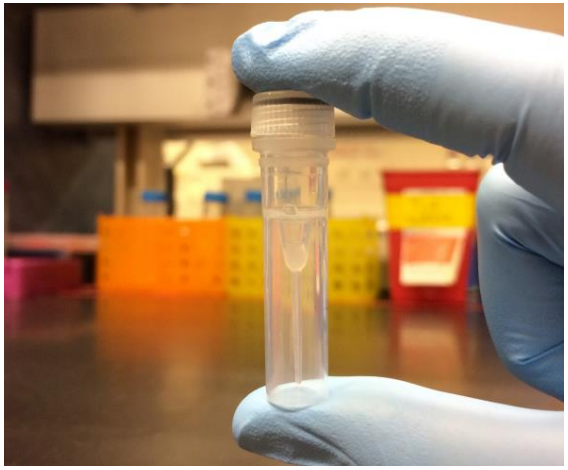
Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, John Manferdelli, John Mitchell, Franzi Roesner, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Announcements

- Quiz Section Next Week: Try Target 5 in Advance
- Next Week My Office Hours: W 12:30, starting in CSE1 403, then to CSE2 307 at 1pm
- (TA office hours as usual, and great place for lab discussions)

Research Discussions

- Monday (10/14): Peter Ney on Bio-Cyber Security and Cell Site Simulators
- Following Monday (10/21): Karl Koscher on Automotive Cyber Security
- Following Wednesday (10/23): Ivan Evtimov on Adversarial Machine Learning
- Following Monday (10/28): Emily McReynolds on Law and Policy



Broad Classes of Security Research

- Measurement
- Analysis / attack exploration
- Building secure systems
- Human-computer interaction

- Guest lectures connected to threat modeling and to buffer overflows as well

Software Security: More on What to Do

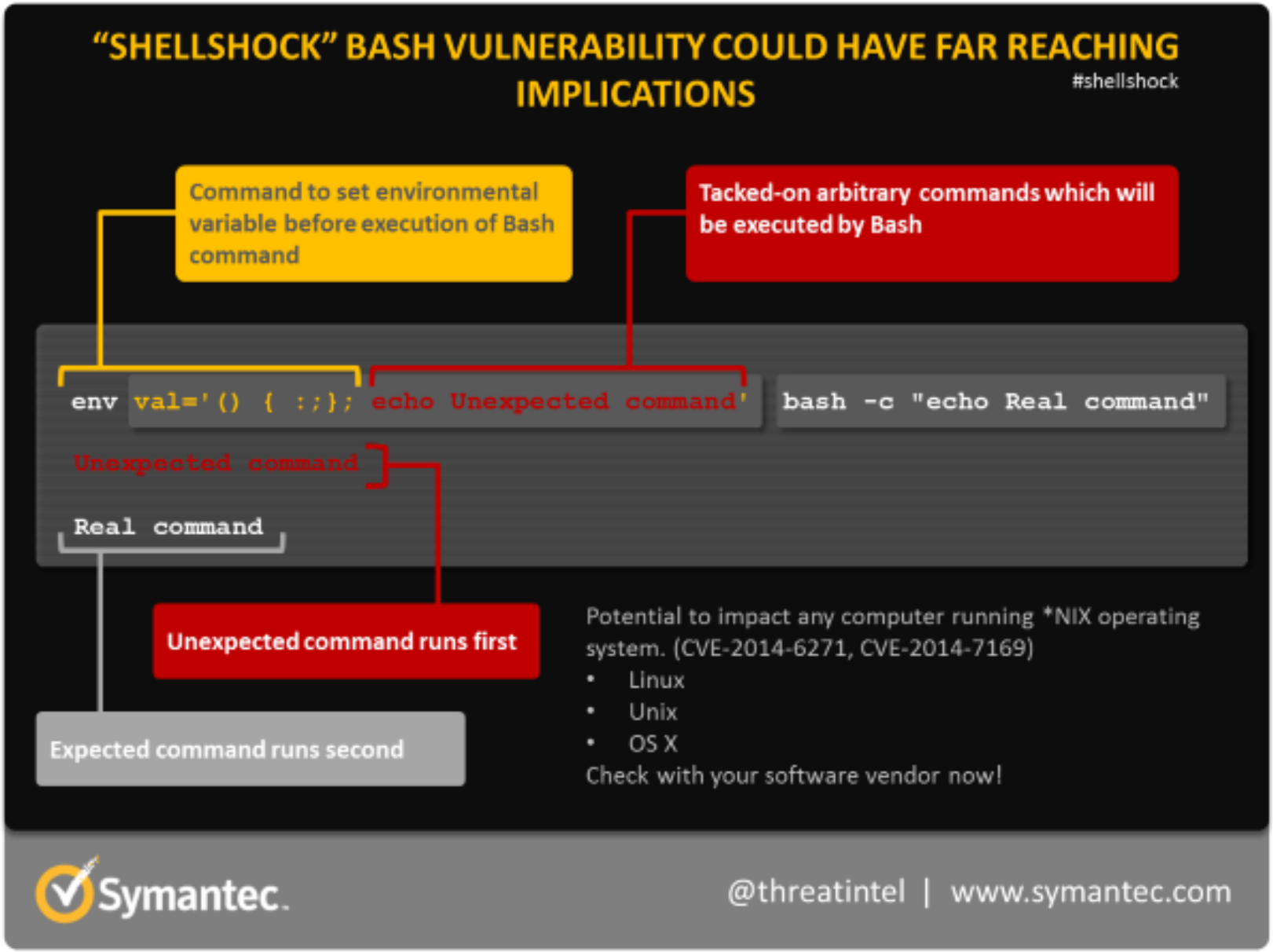
General Principles

- Check inputs

Shellshock

- Check inputs: not just to prevent buffer overflows
- Example: Shellshock (September 2014)
  - Vulnerable servers processed input from web requests
  - Passed (user-provided) environment variables (like user agent, cookies…) to CGI scripts
  - Maliciously crafted environment variables exploited a bug in bash to execute arbitrary code

```
env x='() { :;}; echo Vulnerable'
    bash -c "echo Real Command"
```



https://www.symantec.com/connect/blogs/shellshock-all-you-need-know-about-bash-bug-vulnerability

General Principles

- Check inputs
- Check all return values
- Least privilege
- Securely clear memory (passwords, keys, etc.)
- Failsafe defaults
- Defense in depth
  - Also: prevent, detect, respond

- NOT: security through obscurity

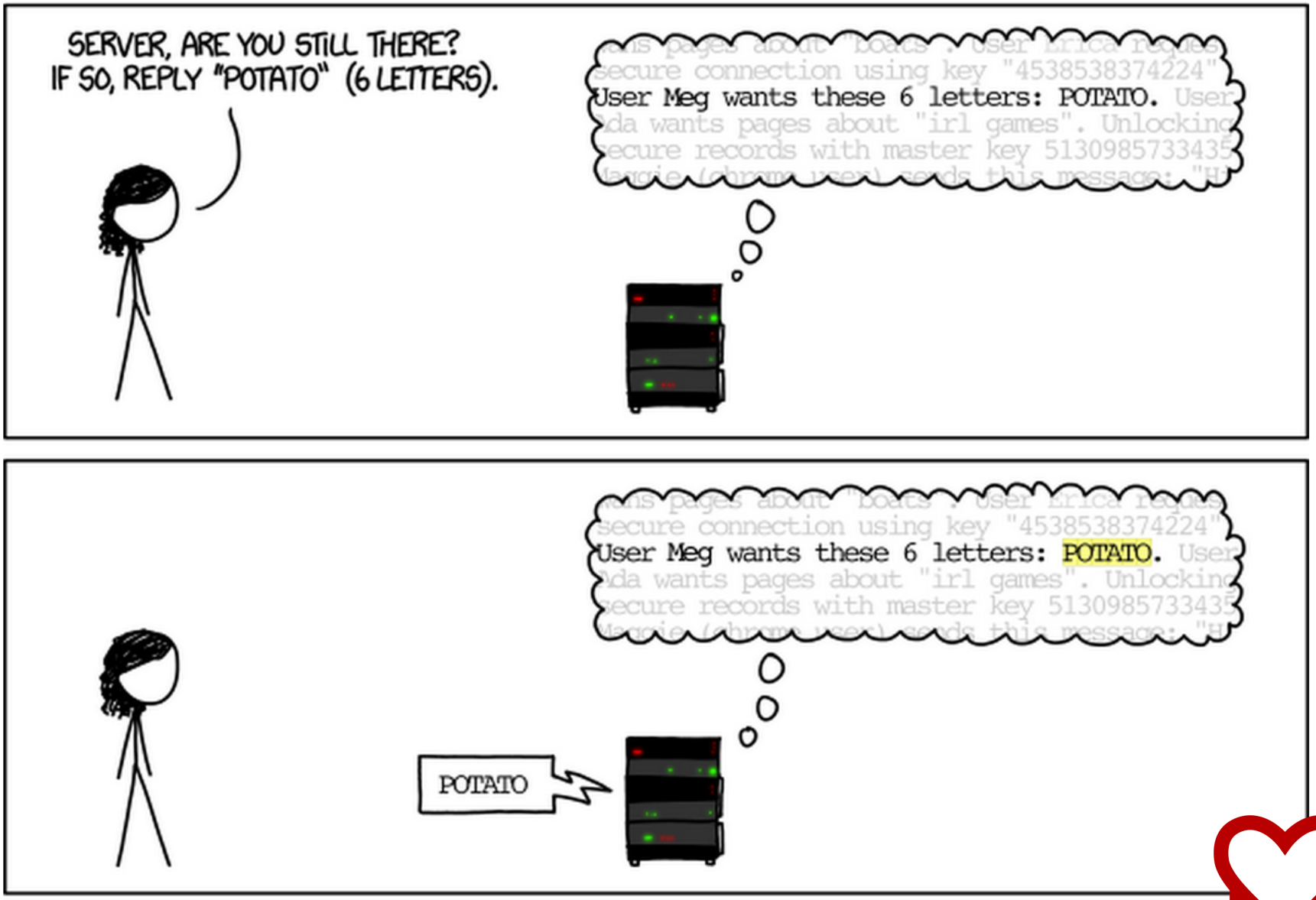
General Principles

- Reduce size of trusted computing base (TCB)
- Simplicity, modularity
  - But: Be careful at interface boundaries!
- Minimize attack surface
  - What does this mean?
- Use vetted component
- Security by design
  - But: tension between security and other goals
- Open design? Open source? Closed source?
  - Different perspectives

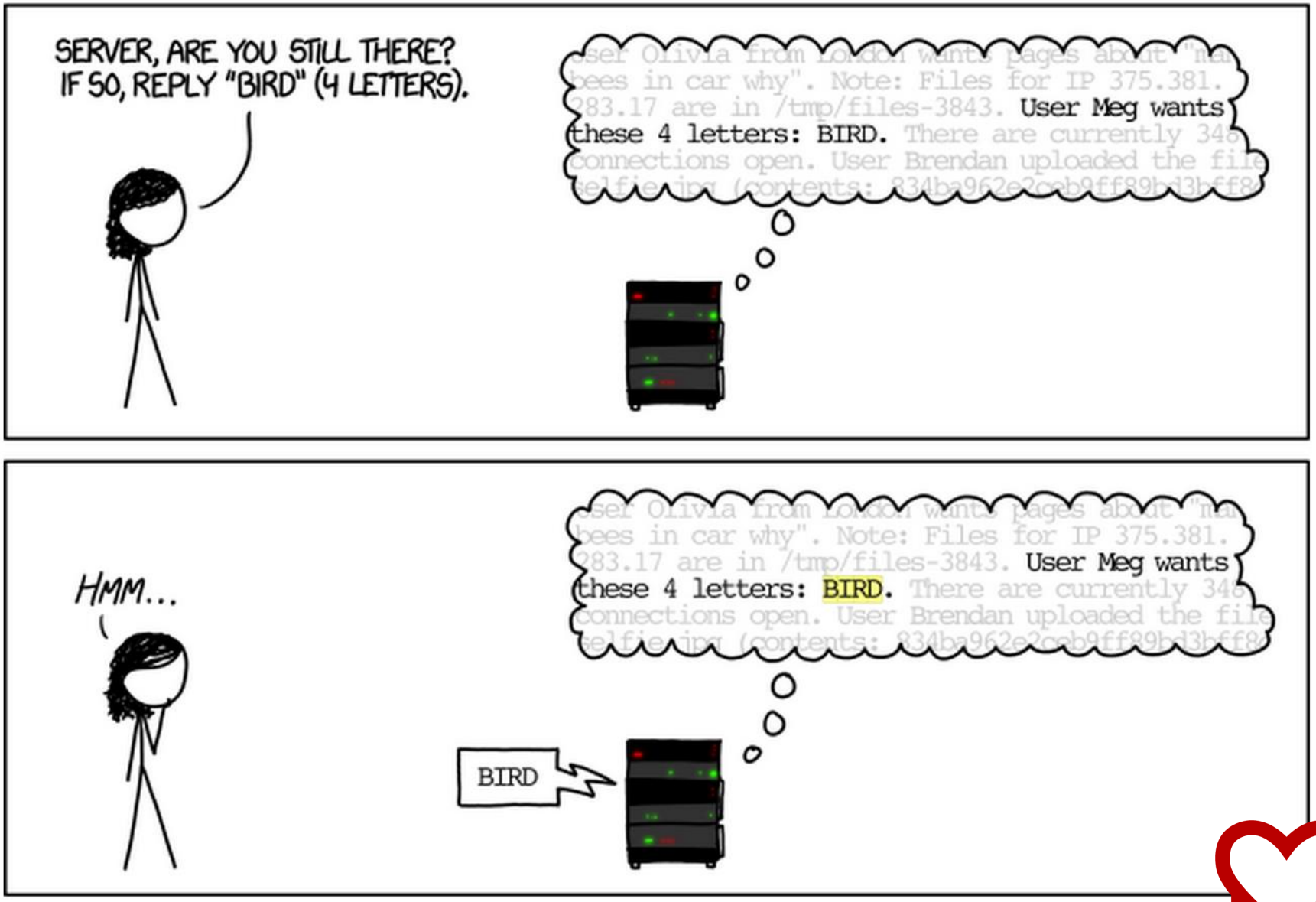
Does Open Source Help?

- Different perspectives…
- Maybe? Maybe not?
  - Linux kernel backdoor attempt thwarted (2003)
    (http://www.freedom-to-tinker.com/?p=472)
- Maybe not? Maybe?
  - Heartbleed (2014)
    - Vulnerability in OpenSSL that allowed attackers to read arbitrary memory from vulnerable servers (including private keys)

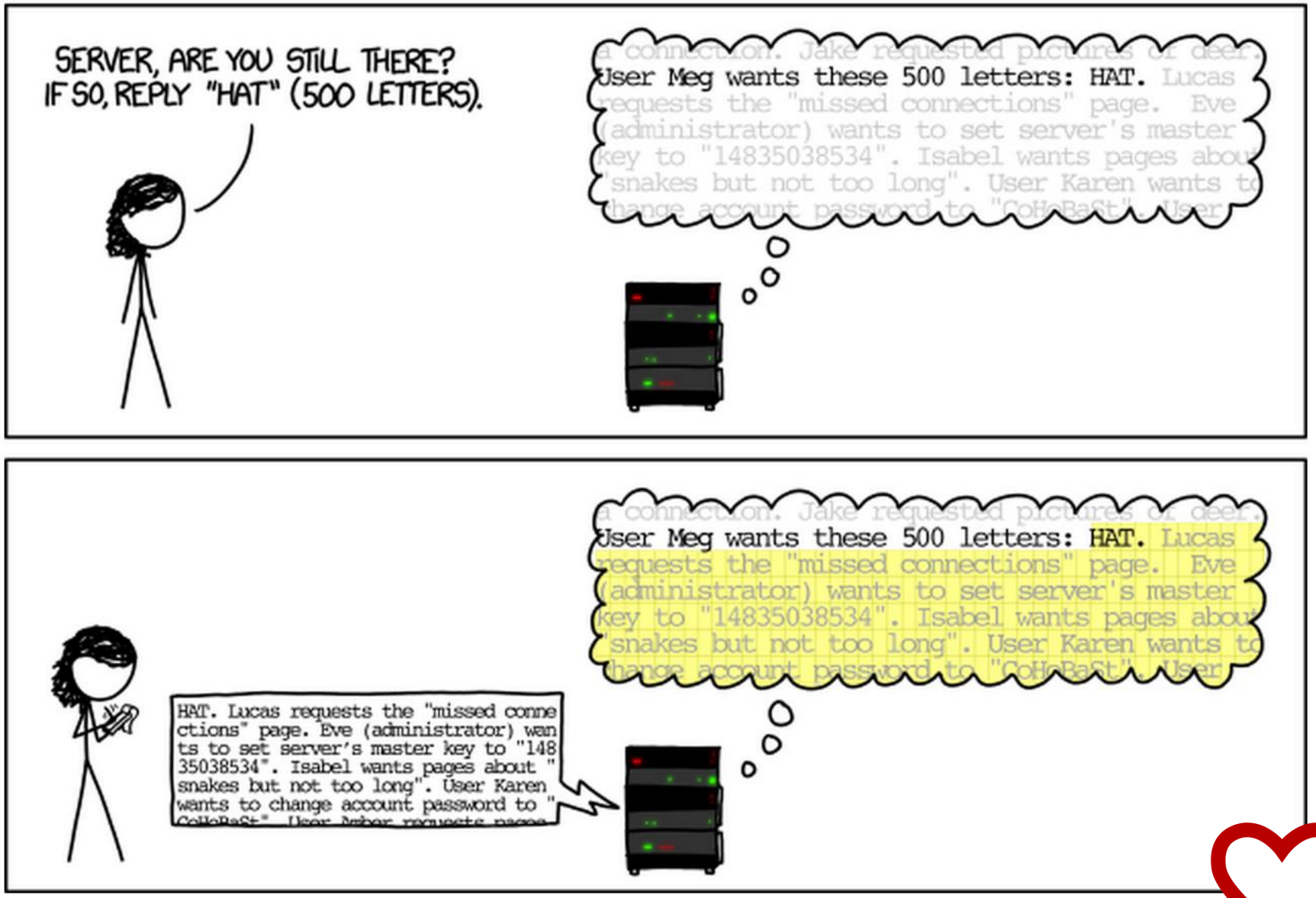
http://xkcd.com/1354/



http://xkcd.com/1354/



http://xkcd.com/1354/



Vulnerability Analysis and Disclosure

- What do you do if you've found a security problem in a real system?
- Say
  - A commercial website?
  - UW grade database?
  - Boeing 787?
  - TSA procedures?

Vulnerability Analysis and Disclosure

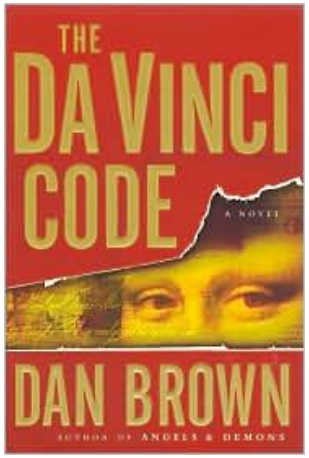
- Suppose companies A, B, and C all have a vulnerability, but have not made the existence of that vulnerability public
  - Company A has a software update prepared and ready to go that, once shipped, will fix the vulnerability; but B and C are still working on developing a patch for the vulnerability
  - Company A learns that attackers are exploiting this vulnerability in the wild
  - Should Company A release their patch, even if doing so means that the vulnerability now becomes public and other actors can start exploiting Companies B and C?
  - Should Company A wait until Companies B and C have patches?

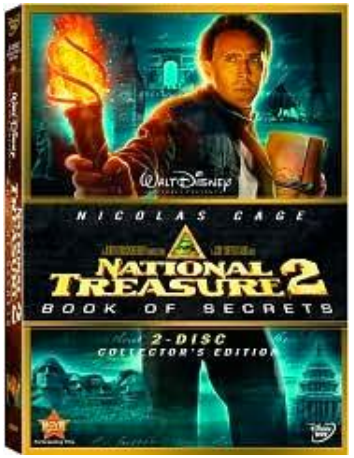
Next: Cryptography

V NZ FBEEL GUVF FRPERG PBQR VF ABG ZBER RKPVGVAT. JRYY, ZNLOR V PNA ZNXR VG ZBER HFRSHY. SBE CNEG BS YNO 1, LBH ZVTUG JNAG GB QVFNFFRZOYR GUR RKVG SHAPGVBA NAQ GUVAX NOBHG JUNG VF PNYYRQ GUR TYBONY BSSFRG GNOYR.

Cryptography and Security

- Art and science of *protecting* our *information*.
  - Keeping it **confidential**, if we want privacy.
  - Protecting its **integrity**, if we want to avoid forgeries.





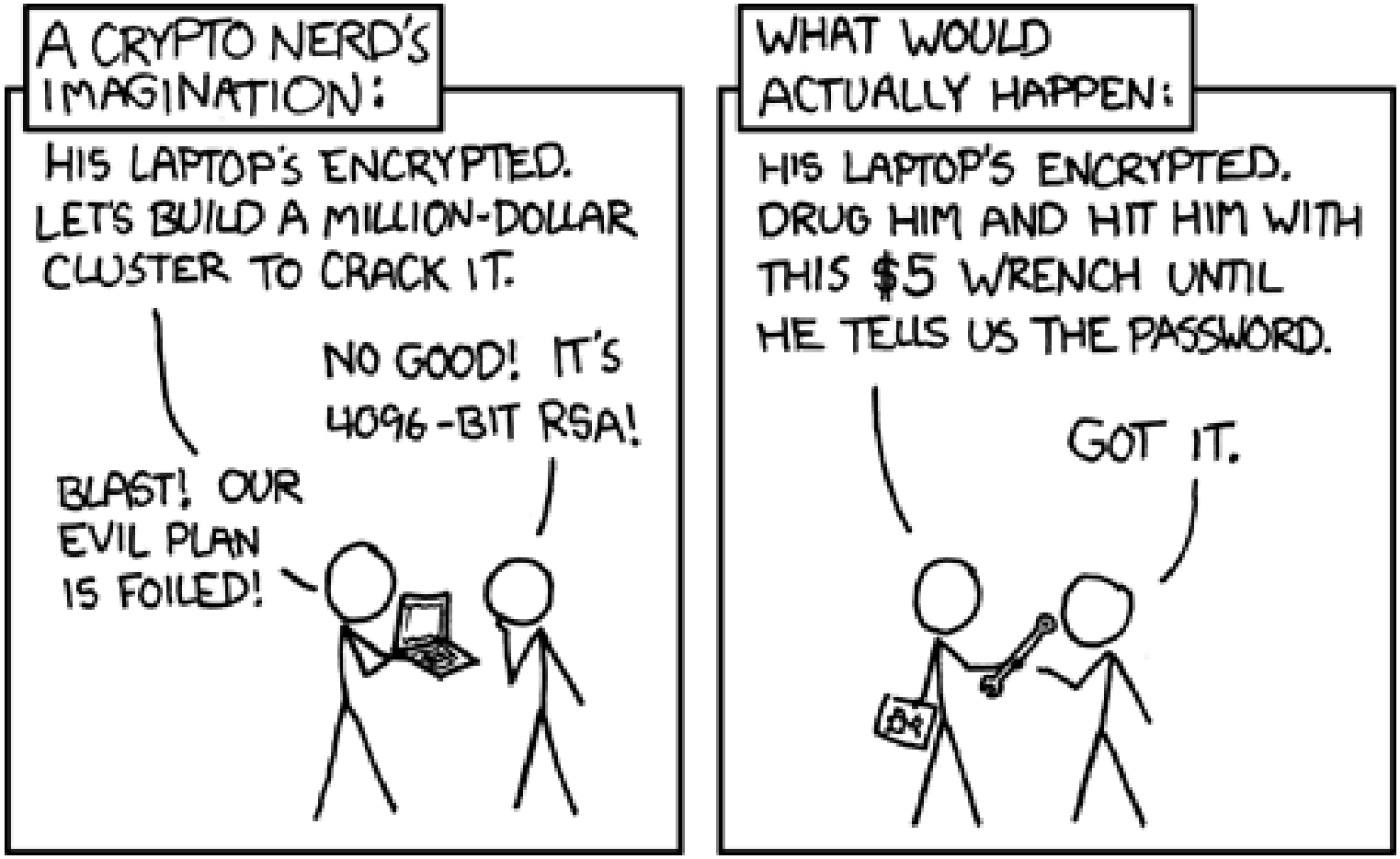
Images from Wikipedia and Barnes & Noble

Some Thoughts About Cryptography

- Cryptography only one small piece of a larger system
- Must protect entire system
  - Physical security
  - Operating system security
  - Network security
  - Users
  - Cryptography (following slides)

- Recall the weakest link
- Famous quote: "Those who think that cryptography can solve their problems don't understand cryptography and don't understand their problems."

XKCD: http://xkcd.com/538/



History

- Substitution Ciphers
  - Caesar Cipher
- Transposition Ciphers
- Codebooks
- Machines

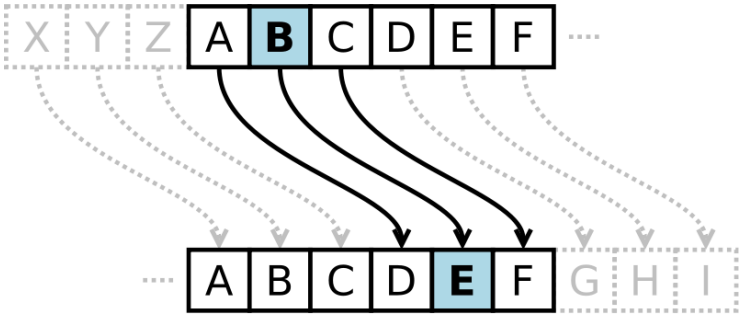
- Recommended Reading: **The Codebreakers** by David Kahn and **The Code Book** by Simon Singh.

History: Caesar Cipher (Shift Cipher)

- Plaintext letters are replaced with letters a fixed shift away in the alphabet.
- Example:
  - Plaintext: `The quick brown fox jumps over the lazy dog`
  - Key: Shift 3

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC
```

  - Ciphertext: `WKHTX LFNEU RZQIR AMXPS VRYHU WKHOD CBGRJ`



History: Caesar Cipher (Shift Cipher)

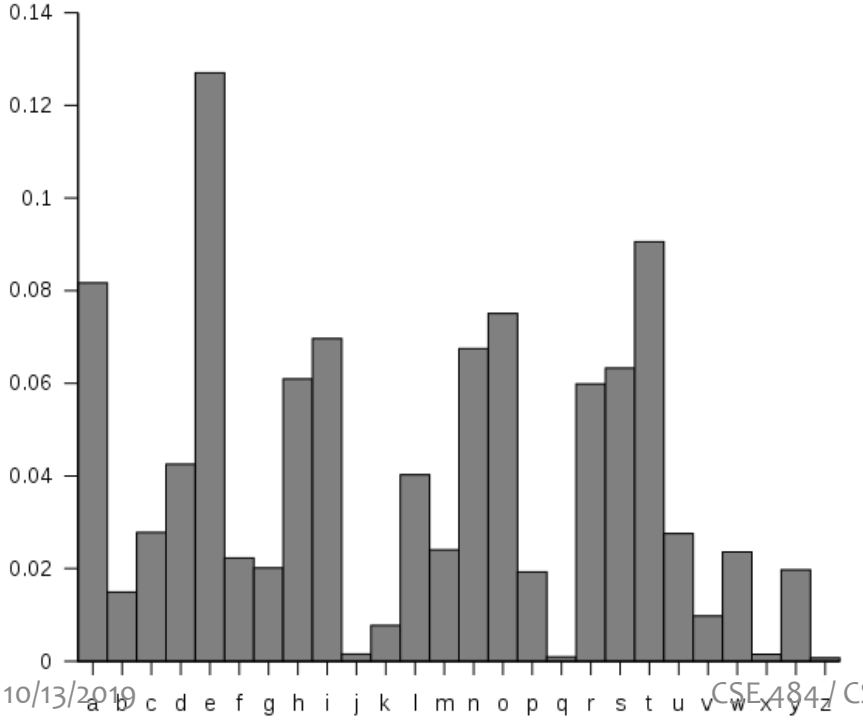
- ROT13: shift 13 (encryption and decryption are symmetric)
- What is the key space?
  - 26 possible shifts.
- How to attack shift ciphers?
  - Brute force.

History: Substitution Cipher

- Superset of shift ciphers: each letter is substituted for another one.
- Add a secret key
- Example:
  - Plaintext: ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - Cipher: ZEBRASCDFGHIJKLMNOPQTUVWXY
- "State of the art" for thousands of years

History: Substitution Cipher

- What is the key space? 26! ~= 2^88
- How to attack?
  - Frequency analysis.



Bigrams:

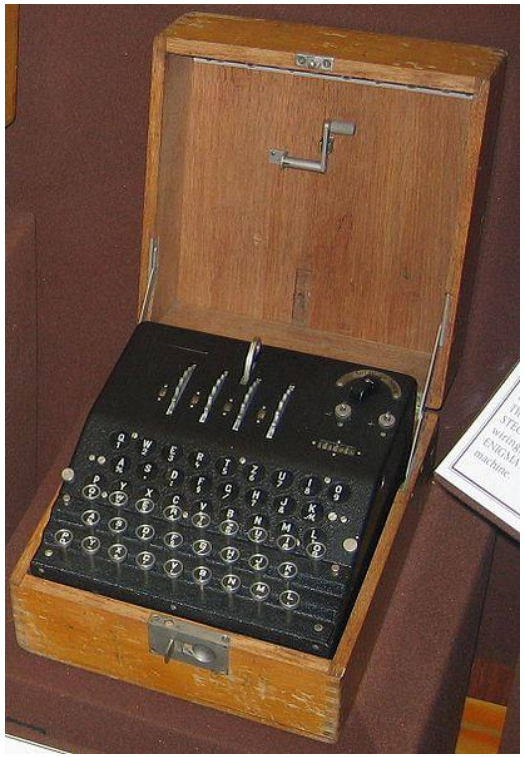
th 1.52%	en 0.55%	ng 0.18%
he 1.28%	ed 0.53%	of 0.16%
in 0.94%	to 0.52%	al 0.09%
er 0.94%	it 0.50%	de 0.09%
an 0.82%	ou 0.50%	se 0.08%
re 0.68%	ea 0.47%	le 0.08%
nd 0.63%	hi 0.46%	sa 0.06%
at 0.59%	is 0.46%	si 0.05%
on 0.57%	or 0.43%	ar 0.04%
nt 0.56%	ti 0.34%	ve 0.04%
ha 0.56%	as 0.33%	ra 0.04%
es 0.56%	te 0.27%	ld 0.02%
st 0.55%	et 0.19%	ur 0.02%

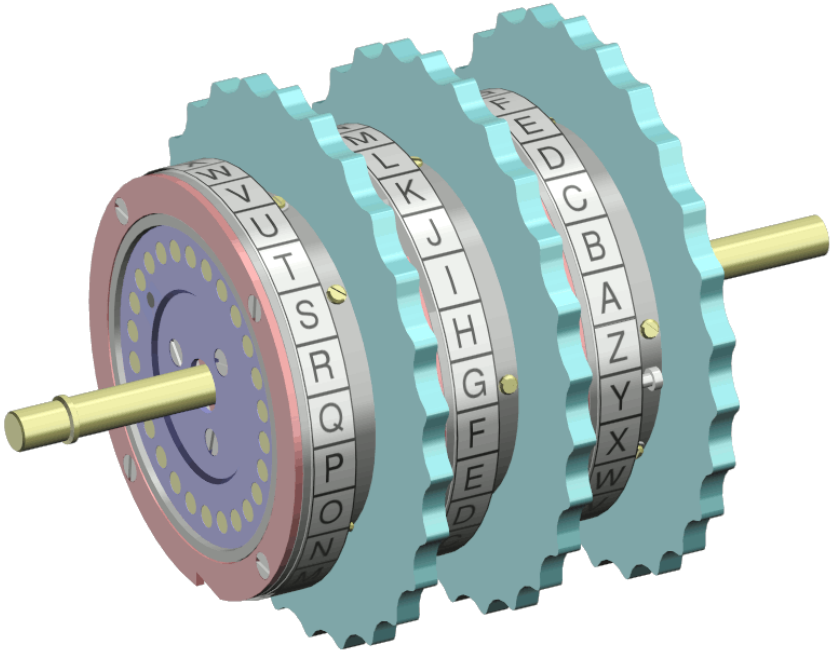
Trigrams:

1. the	6. ion	11. nce
2. and	7. tio	12. edt
3. tha	8. for	13. tis
4. ent	9. nde	14. oft
5. ing	10. has	15. sth

History: Enigma Machine

Uses rotors (substitution cipher) that change position after each key.





Key = initial setting of rotors

Key space?

26^n for n rotors