Security: I Think We Can Win!

Bill Cheswick

Visiting Scholar, U Penn

ches@cheswick.com
http://www.cheswick.com/ches/talks/opt.pdf

Introduction: attribution and the involuntary collaborators

- IP laundering has been used for decades
- Botnets enlist the unsuspecting
    - “Dad, you are blowing blue smoke all over the Internet.”
- With weak clients tamed, we could use shunning

Introduction

- Thinking about security since the Nixon administration
- Starting to get a long view of things
- Generalizations, Grumbles, Hand-waving
- Not a grant proposal, No perfect solutions, No universal solutions
- References are on the slides, see my web page for a PDF of the talk

What is the current state of affairs in computing? Great!

- It's great!
- banking?
- retirement accounts
- shopping and commerce?


AAPL
600
500
450
400
350
300
250
200
180
160
140
120
100
90
80
70
60
50
45
40
35
30
25
20
18
16
14
12
10
9
8
7
6
5
4.5
4
3.5
3
2.5
1000
700
2
for
1
2
for
1


AMZN
340
290
240
190
170
150
130
110
90
80
70
60
50
45
40
35
30
25
20
18
16
14
12
10
9
8
7
6
5
4.5
4
3.5
3
2.5
2
1.75


GOOG
33%
200
40
200
200
120
110
100
90
80
70
60
50
45
40
35
30
25
20
18
16
14
12
10
90
80

What is the current state of affairs? Lousy!

- Spies are all in our business
- Huge advantage to the attackers
- Crappy client operating systems
  - leaky sandboxes
  - feature-driven
- A visit to grandma's house

What Isn’t Working?

“Security people are paid to think bad thoughts”

— Bob Morris

Security paranoia

- We live in a dark world.
- A lot of thoughts are dismissed as “theoretical”
- Here are some examples

Better than passwords

- Both are much better than passwords
- SNK-004 used symmetric key, known only to device and server
  - PIN known only to device
- SecurID's key known to device, server, RSA
- SNK was an ε better

ε had a large value

- RSA break-in caused major attacks on a government contractor and others
- RSA had to reissue fobs
- All of this was because they relied on a (successful) business model that had a security weakness.
- RSA is not a slouch in the security business.

“The best is the enemy of the good”

- A call for mediocrity in the name of getting something done.
    - Don’t flatter yourself that your efforts are “good”.
- Also, from *Soul of a New Machine*, “Not all jobs are worth doing right.”
- This leads to…

Cool, it works, let's use it

- Ethernet, telnet, ftp, rlogin, rsh
- NFS, smb, tftp, finger, sendmail, rlogin
- First generation nuclear reactors
- aircraft, etc., etc.

Shared libraries seem like a bad security idea

- You can change a program after it is installed
- A checksum of a binary does not ensure that it is the same program
- Makes installation in chroot(8) environment more difficult, and requires extra crap in that envinroment.

Not working: shared and dynamic libraries

- “sshd day zero bug” in 2013 was shared library replacement attack.
- Long history of similar attacks
- implemented to save memory and load time back in the days of small memory and the X window system
- *not* worth it
- Make all your binaries static!
- Ditto DLLs

Not working: checklists and audits

- Checklists certainly will catch oversights, but you are not secure when you are done
- PCI audits have missed major, embarrassing intrusions.
- Alas, these are often the response to our endemic problems.

Laws, General and Specific

- General: nice guidelines, but exactly how much protection does HIPAA demand

- Specific: see *Checklists*, above

- Liability: who will be left to write any software if you demand full liability?

Not working: user education

- They don’t (can’t!) understand the complexities of the computer and making the right decisions.

- Even the experts generally lack all the information needed to make the fully-informed choice.

- Even if you do know what you are doing, we all use computers when a little tired sometimes.

Not working: strong passwords

- Forty years of research and experience show that people can not select and remember a passphrase that is resistant to a full-blown dictionary attack; and especially not different ones for dozens of different sites.

- More poor engineering: it just doesn't work by itself, and isn't needed when used with the right authentication tools.

Not working: strong passwords

- Virus checking: it helps, but it will never be a win
  - It solves the wrong problem, and ultimately requires solving the halting problem.
- Strong user passwords
  - More poor engineering: it just doesn't work by itself, and isn't needed when used with the right authentication tools.

Massive data spills

- Credit cards
  - TJX, Target, many others
- Passwords
  - Rockyou, Facebook, Twitter, Linkedin, Google, Adobe, SnapChatDB, EverNote, Stratfor, …

Not working: PKI

- The trusted CA list is way out of hand
- Major attacks find ways around this. Stuxnet, others.
- Try CertPatrol on Firefox to see what is going on
    - (Actually, this is a cesspool. Certificate Transparency or similar efforts?)

SMM

- Been around since the Intel 386. A separate, protected “maintenance mode”.
- It has always worried me.
- A major player in the the list of specific attacks mentioned in the Snowden releases.
- The star of several security papers.

Pentium complexity

- Rings 3 and 0
- System Management Mode*
- Virtual machine interface
- Microcode?!
- How bad can a compromised CPU be?

* Duflot, Loïc, Daniel Etiemble, and Olivier Grumelard.
Using CPU system management mode to circumvent operating system security functions.
CanSecWest/core06 (2006). http://cs.usfca.edu/~cruse/cs630f06/duflot.pdf

Aspects of Virtual Machines worry me

- The kernel/hardware interface is not a natural security perimeter
- The trusted kernel (DOM0) is generally huge
- Co-resident VMs may leak data, and there are papers demonstrating this

Aspects of Virtual Machines worry me (cont)

- It seems very hard or impossible to hide the VM's activities from the supplier

- Homomorphic encryption is a rat-hole:

- never efficient if even possible

  - opens algorithms to a whole new field of attacks similar to traffic analysis

  - The virus guys are already doing this, a bit.

When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge of it is of a meager and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced it to the stage of science.

— Lord Kelvin

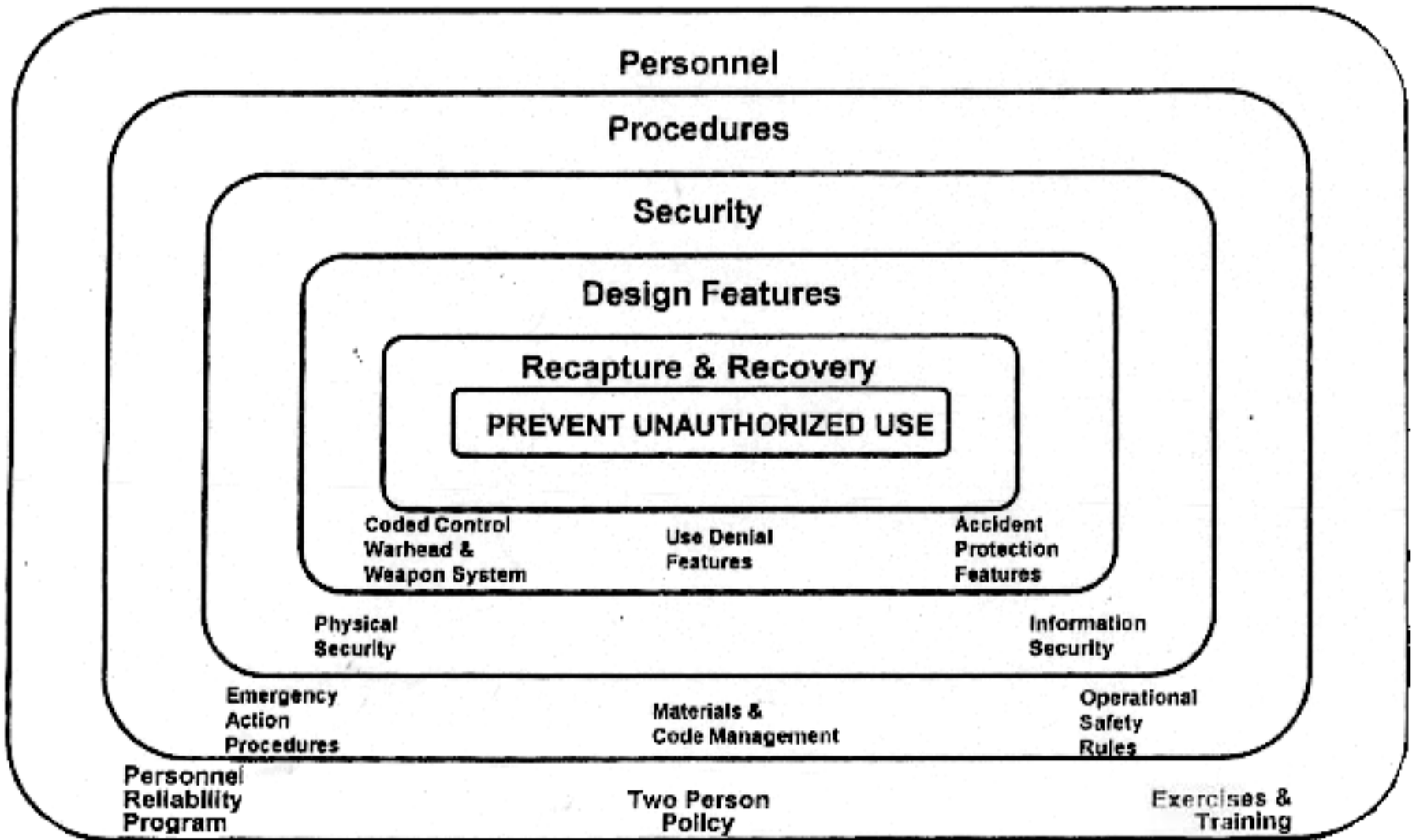
Measuring computer security

- Long-sought, and we aren't even close
- This is still engineering, not science
- What should the answer look light? Something from graph theory? Percolation theory? Chaos theory? An Erdős paper?

Where to measure?

256

Layered Positive Measures to Assure Against Unauthorized Use



SECRET UNCLASSIFIED

Introduction

- I *love* living in the future
- Velcro, 12-hour nasal spray, surgical “lasers”, routine rockets to LEO, astonishing computers
- Sick and tired of computer and network security problems
- Hacked for CPU seconds!
- Already a lot of good security work done
  - Time sharing, Multics
  - Spooks

I think we can win

- Meaning build an affordable computing platform that can't be compromised by any user error not involving a screw driver
- Its our hardware, our software, and our network connection. We ought to be able to control it, dammit!
- Winning doesn't mean that your machine can't misbehave on the Internet

Sick and Tired

- APT are not Advanced, but certainly Persistent and Threats

- Most of the attacks are on the same kinds of weaknesses: we are not making much progress

- Consarn it, I am becoming an old timer!

Long view: it is still early in the computer revolution

- I know, I know, we aren't talking UNIVAC or "minicomputers" any more.

- The order of things: make it work, then worry about security: **(It Works!)**

  - (Very bad prognosis for Obamacare data handling)

- rlogin, NFS, X windows, MSFT before 2001.

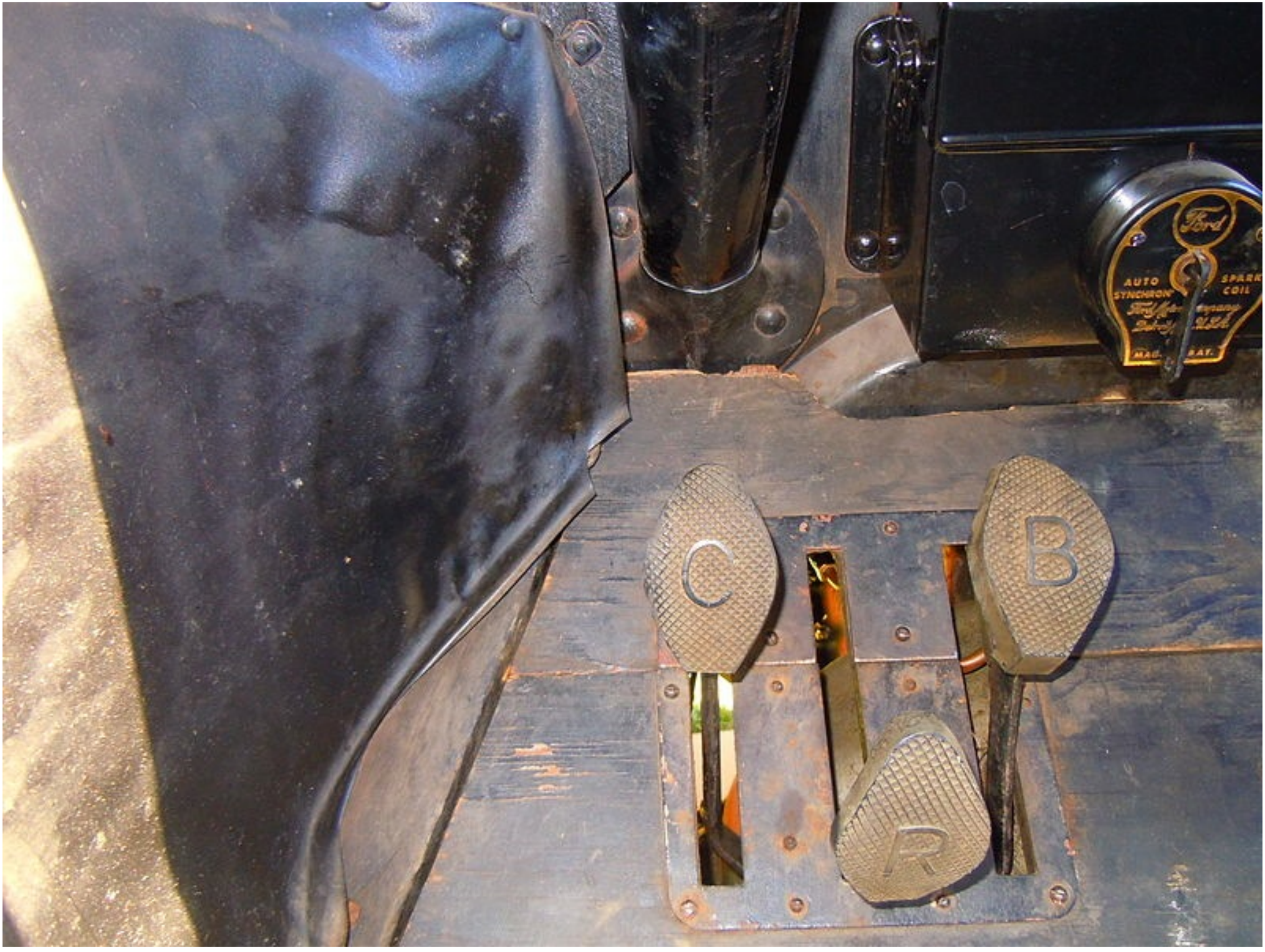
- But look where we are in UIs: I thought we might get stuck with MSFT menus, like the QWERTY keyboard

The car metaphor

- I didn’t like it: apples and oranges
- Now I do: grapes and raisins
- Consider the Model T:

Ford Model T (1913)

- 20 hp
- runs on gasoline, kerosine, and ethanol
- rear wheel drive
- two speeds, plus reverse
- grey, green, blue, and red (1909 - 1913)
- 1913 model (shown) was $550 (four months pay for an assembly line worker.
- Electric start!


Ford
AUTO
SYNCHRON
C
B
R

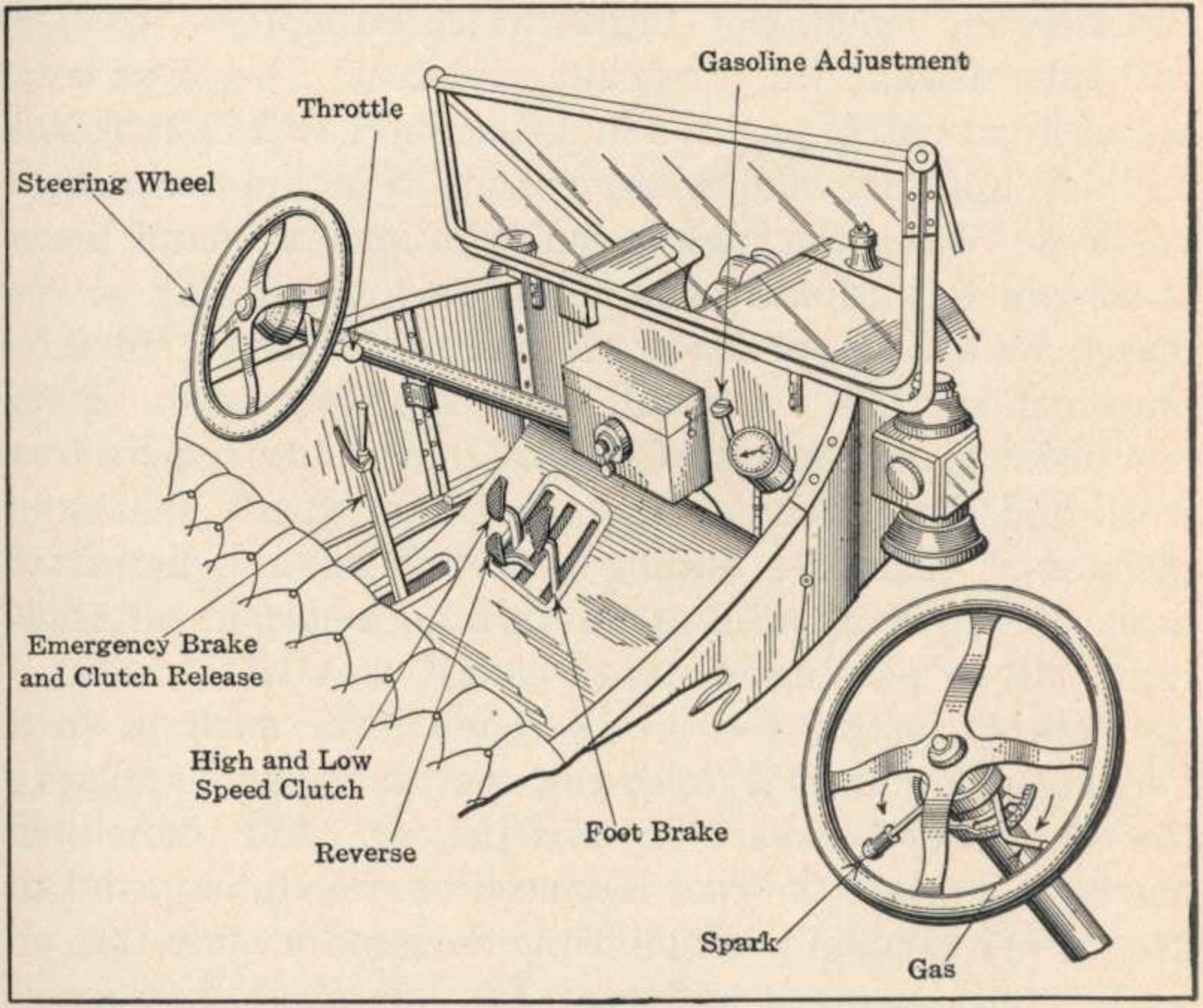


Fig. 42.—The Control System of the Ford Model T Car.

Some old-timey auto stuff

- Fading terms: choke, flood the engine, friction point, vapor lock, double-clutch
- My mother had a car you had to back up steep hills because there wasn't a fuel pump
- First seat belts (two-point) common in mid-1960s

You don’t have to be a mechanic to drive your car, and you shouldn’t have to be a programmer or security expert to use your computer safely.

It’s not the driver’s fault if the engine catches fire

- This is an engineering problem.

- We don’t accept most company claims that it is the driver’s fault.

New car troubles

- Note: cars now need the second kind of firewall
- Attacks on the CANBUS (It Works!)
  - attacks through Bluetooth, evil mp3 files, etc.
  - web search for *“CANBUS security”*
    - *Tiffany Rad*
- Here we go again

Various industries

- Computing
  - Awkward teenage stage
- Aircraft/flight
  - Mature
- Medicine/Health
  - Very early

Measuring security

Adobe Reader	$5,000 - $30,000
MAC OSX	$20,000 - $50,000
Android	$30,000 - $60,000
Flash or Java Browser	$40,000 - $100,000
Microsoft Word	$50,000 - $120,000
Windows	$60,000 - $120,000
Firefox or Safari	$60,000 - $150,000
Chrome or Internet	$80,000 - $200,000
iOS	$100,000 - $250,000

Andy Greenberg, *Shopping For Zero-Days: A Price List For Hacker's Secret Software Exploits*. Forbes, 23 March 2012. http://www.forbes.com/sites/andygreenberg/2012/03/23/shopping-for-zero-days-an-price-list-for-hackers-secret-software-exploits/

Unsolved issues

- Desktop or client/server?
- Programming languages
- Programming is hard
- User interfaces

Programming languages

- ALGOL-60, Pascal, …?
  - I am surprised that strong type checking hasn't won
- PL360, C, ANSI C…
- C++, Java

Programming languages

- Why didn't Modula 3 or Oberon win?
- Why do we accept languages with undefined properties. (I am looking at you, C)
- Why Perl, which looks like TECO input, which looks like TTY communications line noise?

Java

- Why didn't Java win in the terminal and handheld?
- Bush 41-era solution, proof assurances
- Was it just weaknesses in the sandbox
  - Native methods?
- Why did Javascript win?

Cloud computing

- Clearly there is a use for bulk computing
- Netflix is the best example: high volume, low security
- Security is going to remain an issue
- See VMs (above)

Wrong track: virus checkers

- Virus checkers
  - Forget the halting problem (solution: ^C)
    - Like running background checks on indigents living in your house
- StackGuard and similar technologies
  - hobo-resistant rugs and furnishings
- Don’t get me wrong: we need these, for now.

Legacy problems

Inventing a New Internet: Lea

Dewayne Hendri

Tetherless Acce

About the talk:

>From a future historical perspective, are we descendants of Icarus
ciphers and codes, brilliant capabilities built on immature engineerir
taking us to great heights, but systematically flawed? For a brief his
a vulnerable first generation Internet platform. Which as been used
science, commerce, and machines. Promising brilliant futures with t
personalized services and immersive media. But, now our first gene

The tyranny of legacy systems

- We can’t rewrite this, it’s our whole business, and our customers rely on it and want enhancements.
  - (this started as a good system)
- Case in point: Cisco IOS. You can name a bunch more.
- Successes….

Legacy Problems

- Baudot, EBCDIC, ASCII, Latin-1
  - UTF-8, UNICODE
- Microsoft Word document formats
  - newer formats
- Cisco IOS code
  - Still running Tony Li code from Reagan days. No grammar!
- Original Macintosh code
  - rewrite using BSD. iOS?

Legacy-shedding successes

- Macintosh rewrite, using FreeBSD as a start
- Not perfect, but easier to manage than Windows
- iOS and iPhone, rejecting old UIs
    - the security model was that apps couldn’t touch other apps, or the OS. (but see below)
- Many of these efforts fail, or try to do too much.

Still early in the computing game: terminal or desktop?

- Mainframes (Roosevelt)
- Timesharing (Kennedy)
- Minicomputers (Kennedy)
- Workstations (Reagan)
- Client/server
- X terminals and Plan 9 (Reagan)
- Palmtop (Clinton)
- Cloud computing (Bush 43)

UI?

- Tired of listing them, but pinching/tapping/sliding is only about 10 years old
    - Microsoft is migrating away from their awful drop down menus!
- Good UIs are part of the solution

What Works

Lessons and Suspicions
(you may disagree)

Small is better: software

- It is harder to design, build, understand, debug, document, and audit complex systems
- In current open software environment, there is ongoing pressure to add features
- Norman Wilson's IAG
  - removed 2,000 lines of code from the Unix kernel

Small is better

- Plan 9/Inferno operating system compiled in under 20 seconds.
- Very few system calls
- Very few graphics calls
- For a taste of the approach, check out the *go* language from the same folks, at Google
    - A smart phone written in *go* would be very interesting

Small is better: simpler hardware?

- Most people have extremely modest computation and feature requirements, most of the time
  - Wordstar ran on computers 30 years ago

What winning looks like

- Spiral dives and the artificial horizon
- Rinderpest vaccine
- Analog phone cloning
- Hotel keys
- Automobile keys

What winning looks like

- “Getting out of the game” - Fred Grampp

- “Best block is no be there.” — Mr. Kesuke Miyagi (Pat Morita), Karate Kid (1984).

What winning looks like

- You must be present to win.
- No more need for training about clicking on bad things
- More non-IT time with grandma.

A note on Grandma

CPU speed

- No 100GHz Intel Octium processor
- Plenty of power for client crypto
- We could eschew a lot of CPU complexity for audit-ability and reliability
- We could use cores as separate machines, instead of coprocessors. Separate cache and memory, too.
- Simple processor is a project for a grad student

What works: personal responsibility for the code

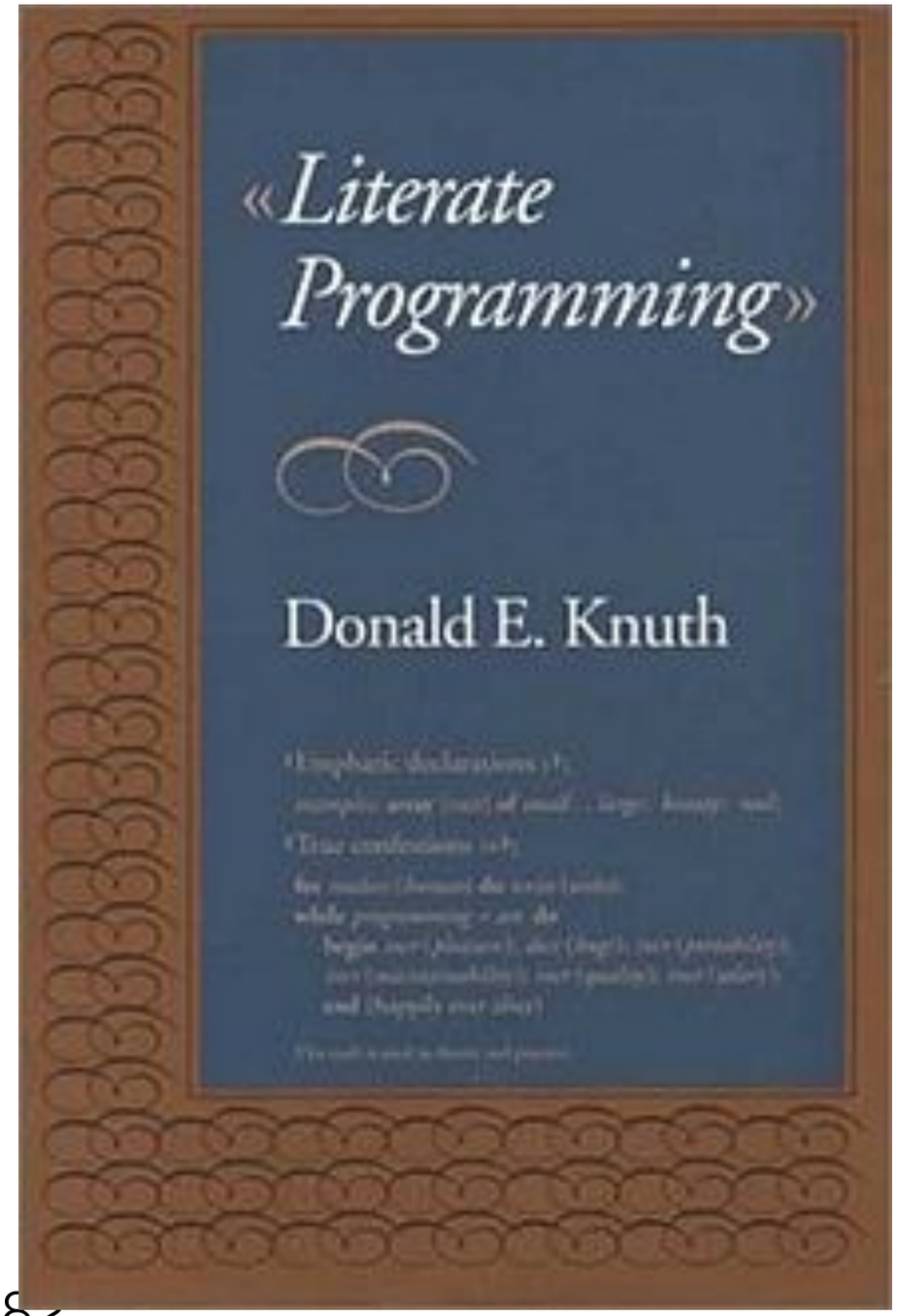
- Knuth's personal checks
- Dockmaster: if someone breaks it, you are fired

Works: Literate programming

- You write a document that explains the program, algorithms, etc., with code embedded in an order natural to the description, not what the compiler wants.

- *weave* and *tangle* generate a document and a program

- Imagine a kernel lovingly described and written in this form.



What works: software “annealing”

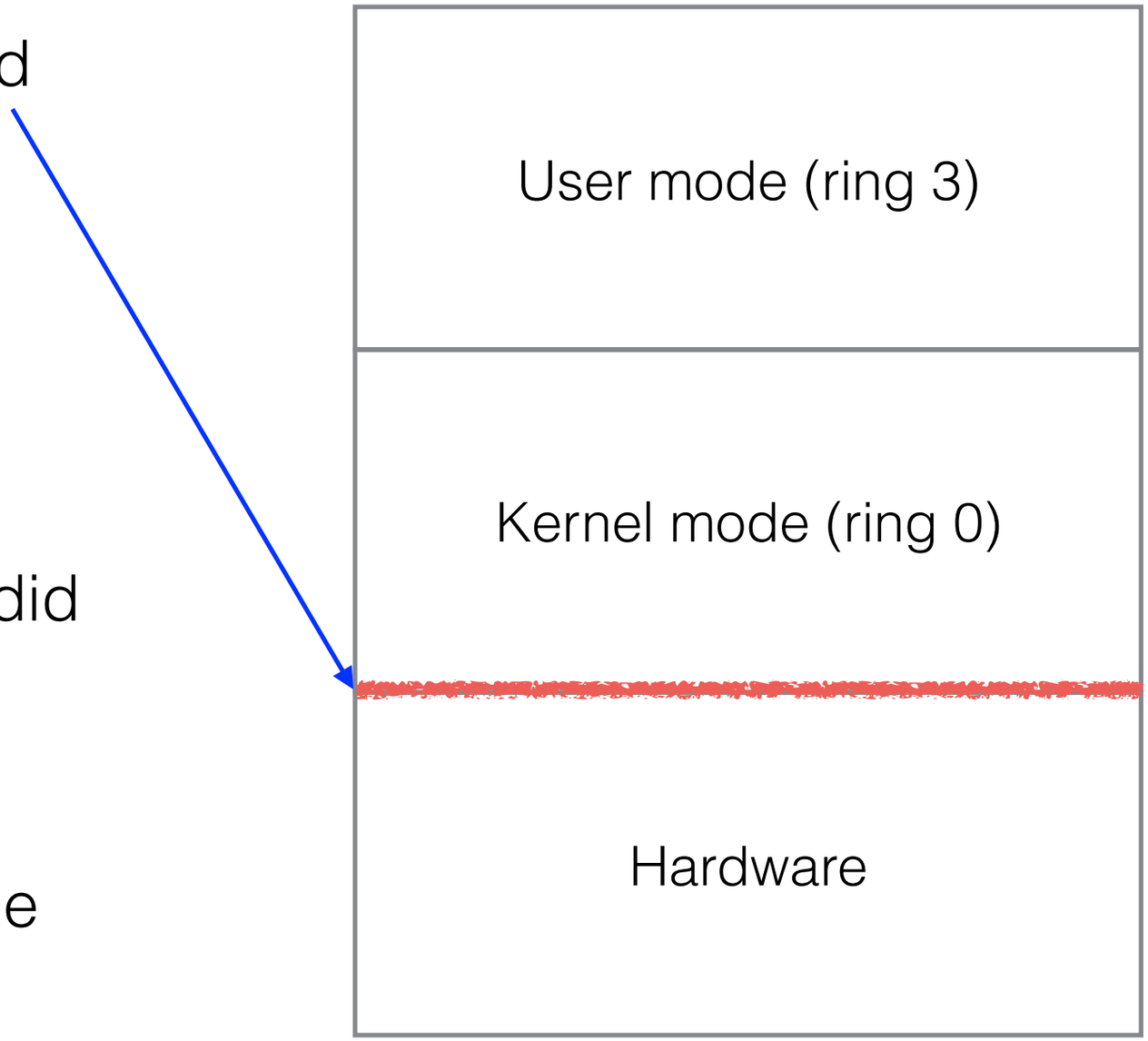
- Sendmail
- Postfix, in beta for a year
- ssh and its protocol
- why was openssl changed?

Strong type checking

- My experience with BASIC, FORTRAN
  - Dykstra, then Pascal
- Too bad C won: my choice was Modula 3 or Oberon, perhaps
- Small is still beautiful.

Are VMs okay?

- Yes, but there is a very weird security line there
- Kernels and the hardware have always been intimate pals
- If we throw away that trust, did we find all the hardware weaknesses?
- Also, DOM0 is an awful large entity to trust.



What works: 4 digit PINS!

- Why? Limited tries
- Robust history of success
- Only a few PINS need to be illegal

What works: end-to-end crypto

- Johnny still can't encrypt, and there is no excuse for it
- There is plenty of compute power
- The algorithms are fine
- It solves a lot of problems

What works: trusted path

- how do you know you are talking to the trusted operating system?
- ctrl-alt-delete was an example
- out-of-band PIN
- make standard screens slightly taller than movie aspect ratio (16:9), and dedicate a section to trusted system messages

Formal methods

- These have been known for a long time (e.g. see the Orange Book.)

- We are making much better tools now.
    - Jon Anderson's work on TESLA (Temporally Enhanced System Login Assertions.)

- They are expensive and require unusual skills, but

- Once run, we can all share the results. I think they are part of the answer.

What Might A Secure World Look Like?

Design goals for Grandma's computer

- There's nothing she can type, tap, swipe, or click on that will change the software she is running, or hurt her computer.

- There is nothing a remote attacker can do to her computer without having physical access to the hardware.

Design goals for Grandma's computer (cont.)

- The software she runs can be reliably ascribed to a particular vendor, and that vendor can be confident enough to be willing to assume significant liability for misbehavior of that software.

- Alien software can be ably and reliably contained and run in a sandbox that preserves all of the above guarantees.

- Grandma has clear indications when she is surfing the web off of well-defined paths on the Internet.

Target users for this computer

- Grandmas, for large values of grandma
- Most employees and regular computer users
- Most military clients. Grandma could run Milspec.
- Maybe 70% of the market?
- Not gamers.

Building a computer from scratch

Goal: be like a wise man who built his house on the rock

- Trusted hardware
- Trusted firmware
- Trusted OS
  - trustable sandbox

The hardware is a problem

- Relies on the trustability of the design and fabrication
- Changes to circuits by malefactors or National Security Letters
- Confident auditing of the final chips is worthwhile, but is very hard
    - We will never be able to detect subtle hardware modifications. A small dose of molybdenum?
- Good news: CPUs could be quite cheap

Software layers

- Proved correct: BIOS, kernel, compiler, libraries, sandboxes

- Peter Neumann and others have been working on this since at least the 1970s.

- Expensive, but cheap when amortized over the whole user community.

Sandboxes have to be rock-solid

- Data may be need to be saved in a specific way between instantiations
  - Browser cache, history, cookies, etc. This is a tough problem
- Applications that want to break the sandbox will not work on the machine
- Such a machine is not for every one, but you probably don’t want to do banking on another one

Some special purpose systems already try to do this

- aerospace and aircraft
- medical devices, but many use ancient Windows software as a trusted computing base **(It Works!)**
- Controller hardware, esp. since Stuxnet.

Other solutions, if your hardware is ok

- Live CDs and thumb drives.
- Bank with a CD/thumb drive from the bank
- Provenance is an obvious attack

Where Might These Solutions Come From?

Microsoft?

- They certainly turned around in 2001
- Vista and Win7 appear to be vastly more secure than Windows XP
- This was a *huge* job. I don't know how much of the legacy problem they solved.

Windows OK

- There is nothing you can click, tap, or say that will corrupt your computer.
- It should be intuitively obvious when you are not visiting a Fortune 500 web site, or a place you have never searched before.
- Offers standard services
- It could meet the specs for this dream system.

Do we have this already?

- Jeff Jones (MSFT) said Win 7 was much safer than corresponding Linux
- Maybe Win 8, too
- Seems like an awfully large hunk of software to declare victory, and maybe they haven't.

Apple?

- Macintosh redesigned in late 1990s, on FreeBSD
    - Vastly improved, big market success. Does have legacy software that lagged for a while.

Maybe iOS…

- Certainly Apple tried hard to design security into iOS, and they had a fresh start, sort of

- App isolation and app walled garden were key security goals.

- How can we tell? Measure security…

Apple security?

larization is obtained by integrating along the *unperturbed* line of sight,

$$\psi(\hat{n}) = (1/2)\varepsilon^{ij}{}_k n^k \int^{\chi_s} d\chi \left(\partial_i B_j - n^l \partial_i h_{jl}\right). \qquad (4)$$

Here ε_{ijk} is th or in three dimensions, and m the observer parameterizes n-zero ψ is not a coordinate a nation modifies Eq. (4) by only boundary terms. Those correspond to Lorentz transformations of the source frame and the observer frame, since Eq. (3) defines different tetrads in different gauges.

Unlike Faraday rotation, the rotation due to metric perturbations is achromatic. Scalar metric perturbations, namely the

Malware Alert

Install now?

Yes | Yes

Measuring security

Adobe Reader	$5,000 - $30,000
MAC OSX	$20,000 - $50,000
Android	$30,000 - $60,000
Flash or Java Browser	$40,000 - $100,000
Microsoft Word	$50,000 - $120,000
Windows	$60,000 - $120,000
Firefox or Safari	$60,000 - $150,000
Chrome or Internet	$80,000 - $200,000
iOS	$100,000 - $250,000

Andy Greenberg, *Shopping For Zero-Days: A Price List For Hacker's Secret Software Exploits*. Forbes, 23 March 2012. http://www.forbes.com/sites/andygreenberg/2012/03/23/shopping-for-zero-days-an-price-list-for-hackers-secret-software-exploits/

Apple security?

- I love these devices, so I learned Rejective C and usually follow their UI advice slavishly.
- NextStep is from the late 1980s, which is okay in itself, but
  - retain count stuff went away (mostly) only a couple years ago when ARC came
  - It’s not just my software that crashes

Apple security?

- I don't see how anyone can have confidence that their non-trivial program is correct in this system.

- **AND**…they get jailbroken as soon as there is a new release. This is not a good sign.

- My best bet for the most secure clients at the moment, but it is scary

Google

- A lot of efforts in important areas, with security on their mind:
  - Android
  - Chrome
  - Chromium
- and go (a nice language)

Android

- Android is the regular and systematic target of security research papers, probably because it is much more accessible than iOS.

- As for the apps: “the problem with folk songs is that they are written by the people.” — Tom Lehrer

- It is also the basis for some brand new attempts at secure clients, like Boeing Black.

Other players

- Any of these companies could start over, and maybe some should
- A basic operating system has approximately a $0 billion startup cost.

Who Are You Gonna Call?

- Hyper-careful industrialists
- Dean Kamen (insulin pumps, wheelchairs)
- Elon Musk (rockets, cars)

Academic and other research groups

- Small teams have produced very interesting operating systems, and I bet small is going to be an important part of the answer. Some examples:

  - Plan 9, Minix 3

    - Peter Neumann, DARPA CRASH program: clean slate redesign from hardware on up.

- The military has a strong interest in this, and even in disseminating the solution

  - *c.f.* Linux SE

Yeah but

- People make buggy code
- Programming bugs imply security bugs
- There is no evidence that our code is getting less buggy
- General computing has many requirements, and they change too often
- Karger/Thompson: On Trusting Trust

Yeah but

- Governance is a big concern
- Did your hardware provider get a National Security Letter?
- National debate and resultant policy, enforced

Yeah but

- Still have DDoS
- People can still be fooled
  - phishing

I think we can win

- It is our hardware, and our software
- There is no law of physics that says this can't be done, and
- We have engineered reliable systems out of unreliable parts before.
- We have the home-field advantage
- Correct software can be implemented, if we are very careful

I won’t live to see all this happen

- And there will still be plenty of security problems
- You can always fool people somehow
- And every public service can be flooded by the public (DDoS)

Security: I Think We Can Win!

Bill Cheswick

Visiting Scholar, U Penn

ches@cheswick.com,
http://www.cheswick.com/ches/talks/APPsec2013.pdf

107of about 106