Giant bags of mostly water

Securing your IT infrastructure by expecting human error

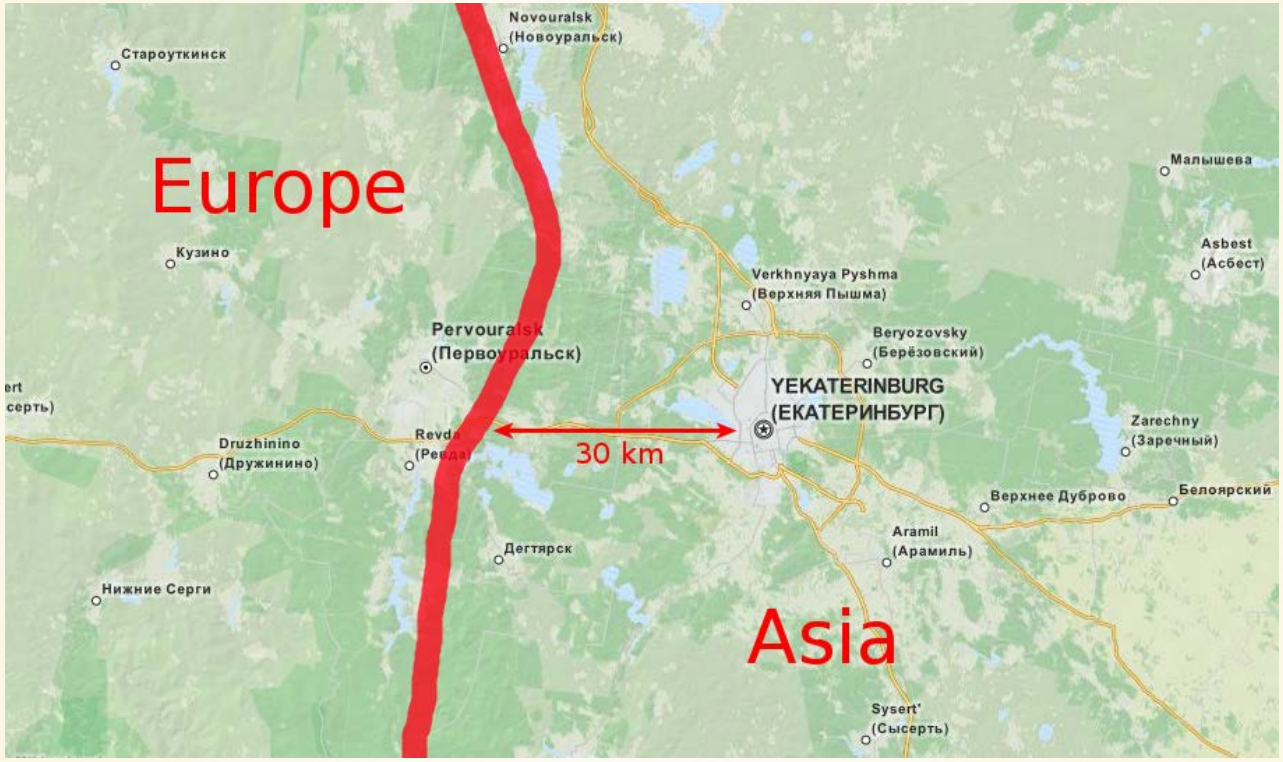
Konstantin Ryabitsev

Korea Linux Forum 2015

Who am I

- Programmer since 1989
- Linux administrator since 1998
- InfoSec since 2008
- Linux Foundation since 2011
- I'm from Asia, too!

See?



(they won't let me use "Asian Canadian" on government forms, though)

image source

I'm a giant bag of mostly water

(and so are you)

image source

This is a 1957 Chevy Bel Air

1957 Chevy Bel Air

- 283 HP "Super Turbo Fire V8" engine
- TurboGlide automatic transmission
- Air Conditioning
- Roomy seats
- Smooth ride

Raw power and comfort

1957 Chevy Bel Air (contd)

Safety features:

- Seat belts with separate shoulder harnesses
- Triple-locking door latches
- Padded dashboard
- Steering wheel with a "safety-styled recessed hub"

"Vehicle interiors are so poorly constructed from a safety standpoint that it is surprising that anyone escapes from an automobile accident without serious injury."

- Journal of the American Medical Association, 1955

Unsafe at any speed

Ralph Nader's 1965 book famously blamed car manufactures for "designed-in dangers of the American automobile."

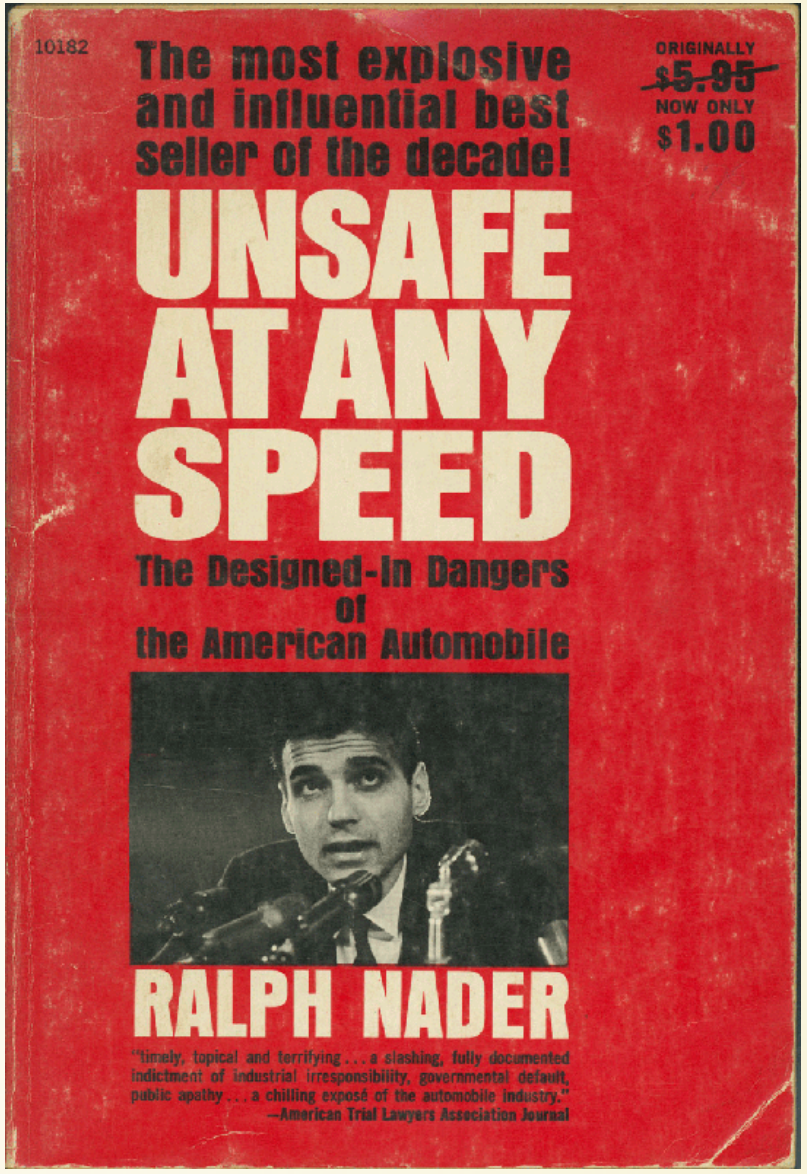


image source

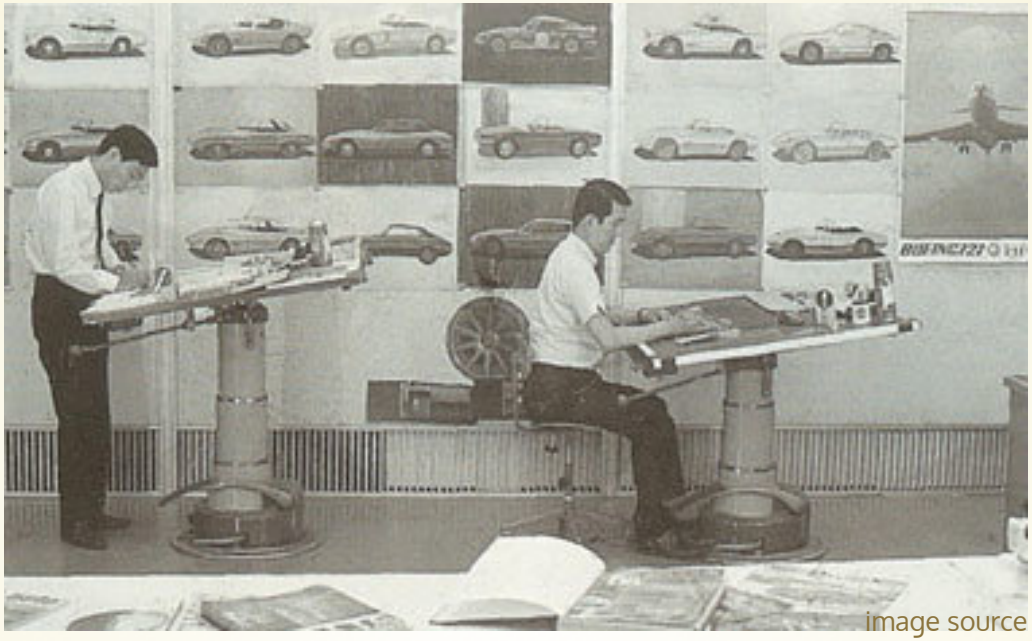


image source

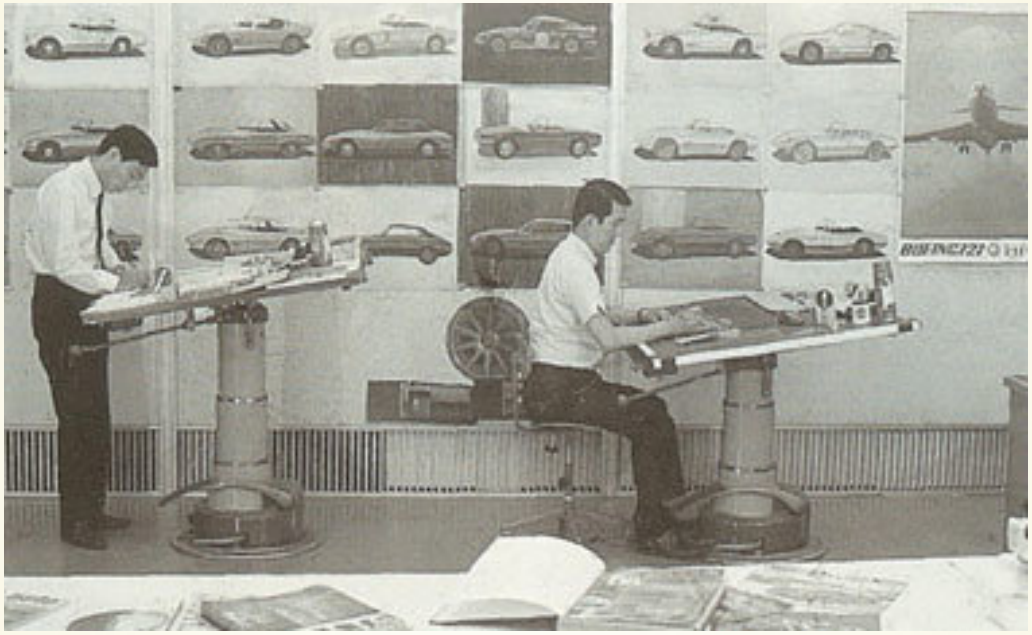
1960's engineers response:

What the hell are you talking about?

1960's car engineers:

- We have designed the components to run safely!
- Nothing leaks, explodes, or jabs you in the face
- Nothing catches on fire under most conditions
- Harmful fumes are kept well away from occupants
- Components fail safely by stalling the engine and letting the vehicle coast to a stop

The car is designed to drive, not to crash!



Most crashes are due to bad drivers!

(sounds familiar?)

Car company responses:

- Protecting drivers against crashes is expensive
- Adding safety features sacrifices style and comfort
- Problem much better solved by driver education
- Customers are just not asking for it!

Fast-forward 50 years

Image source

This is a 2015 BMW i3

(sorry, I unfortunately don't own a Kia Soul!)

2015 BMW i3

- 168 HP all-electric drive
- Single speed fixed transmission
- 0-60 in 7 seconds

Cute and green
but still plenty of torque

2015 BMW i3 (contd)

Safety features:

- Carbon fibre roll cage ("life module")
- Driver, passenger, side air bags
- Entire front is crumple space
- Seat belts with pre-tensioners
- Collision sensors stop the car

Only 4-star safety rating!

Modern cars are all about driver errors!

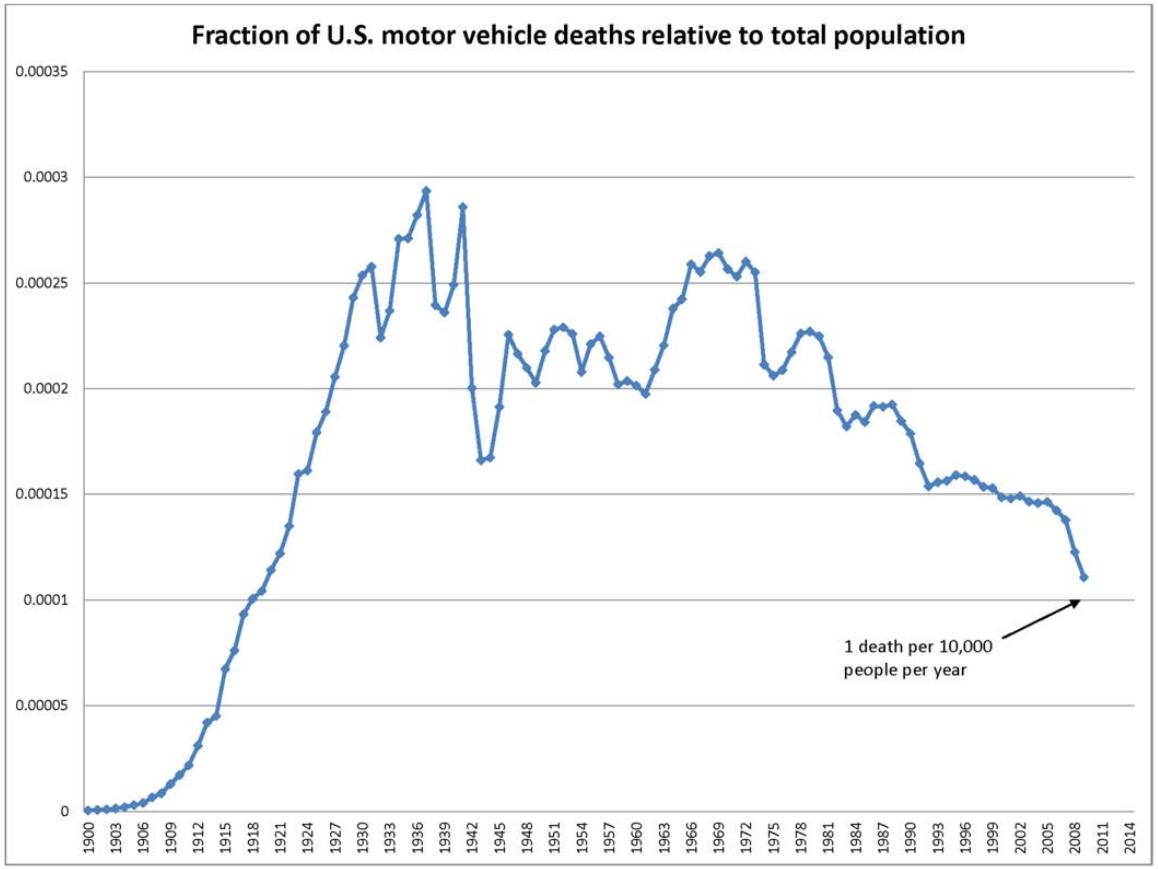


image source

See the results?

Wait, I thought we were going to talk about Linux security?

We still design IT infrastructure like we designed cars in the 1960s.

Raw power and comfort

- More HZs!
- More RAMs!
- More cores!
- Larger, faster disks!
- Faster, lower-latency networks!
- One-click deployment!
- Containers, so anyone can deploy!

As an industry, we don't care about user security. We will gladly ship products with known security failings and no plans to update them.

- Matthew Garrett, 2015

source

This is as damning of a quote as the one from the American Medical Association

2015 IT engineers:

- We have designed the components to run safely!
- Nothing leaks, explodes, or jabs you in the face
- Nothing catches on fire under most conditions
- Harmful traffic is kept well away from users
- Components fail safely by saving state and dumping core before crashing

Yes, but what about dumb things humans do?

IT company responses:

- Protecting users against their own mistakes is expensive
- Adding safety features sacrifices usability
- Problem much better solved by user education
- Customers are just not asking for it!

We are not responsible for user errors!

Right?

So we create an IT Security post and hire a smart person to help protect our infrastructure.

(that's you!)

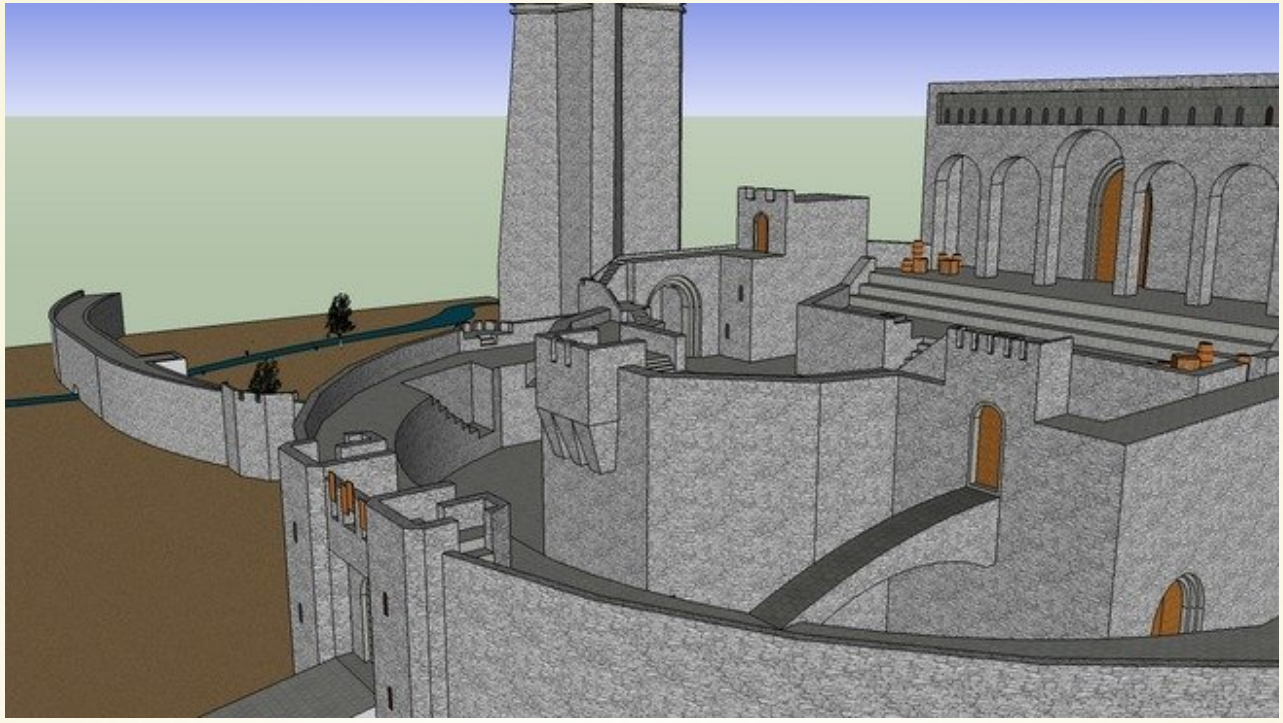


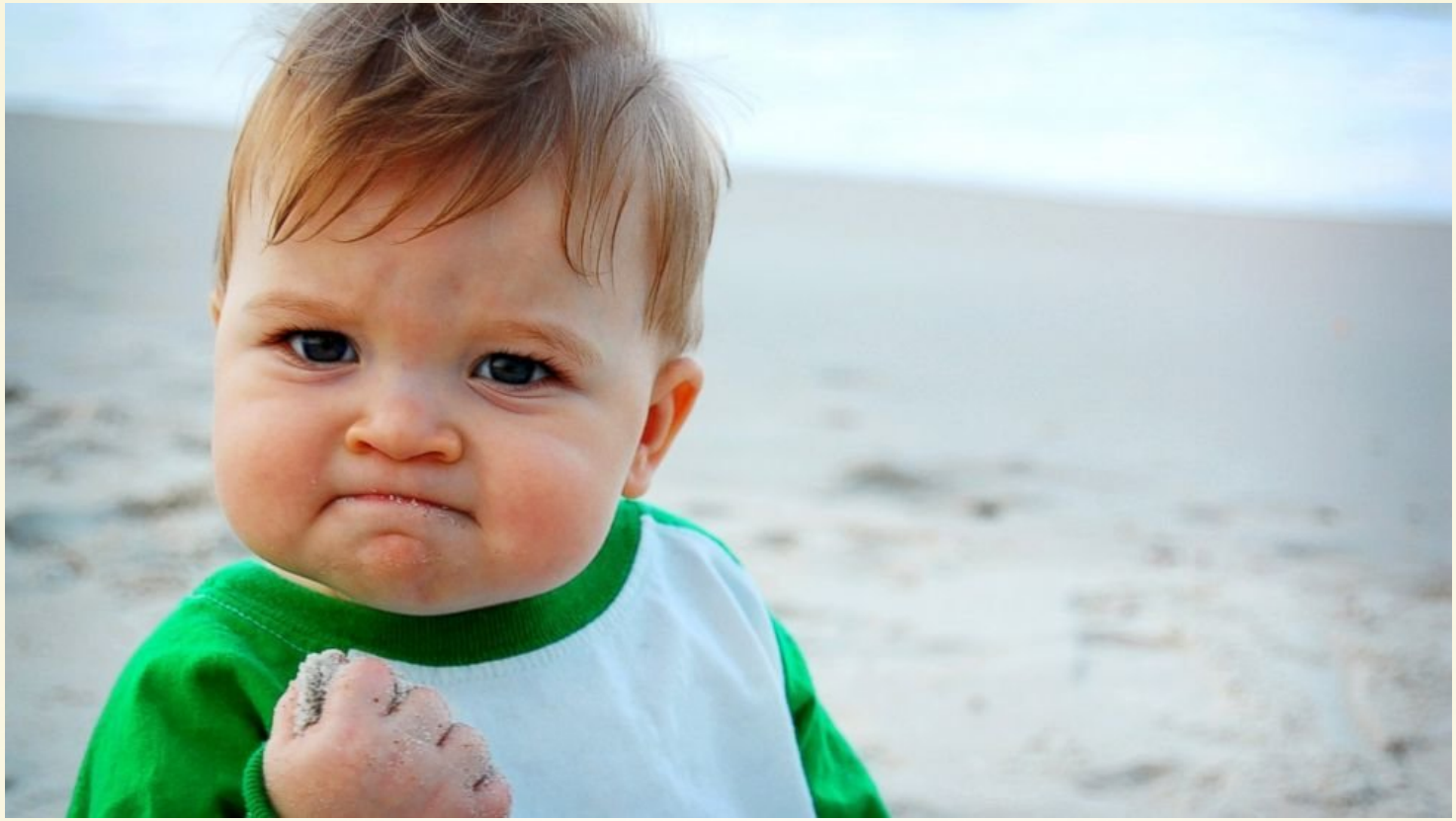
image source

You design the infrastructure like it's a medieval fortress

(and you should!)

Helm's Deep

- First line of defence (the Big Wall)
- Second line of defence (Archers)
- Third line of defence (Gimli the Dwarf)
- Fourth line of defence (the Inner Keep)
- Final line of defence (two old white guys)
- Disaster Recovery (Gandalf the wizard)

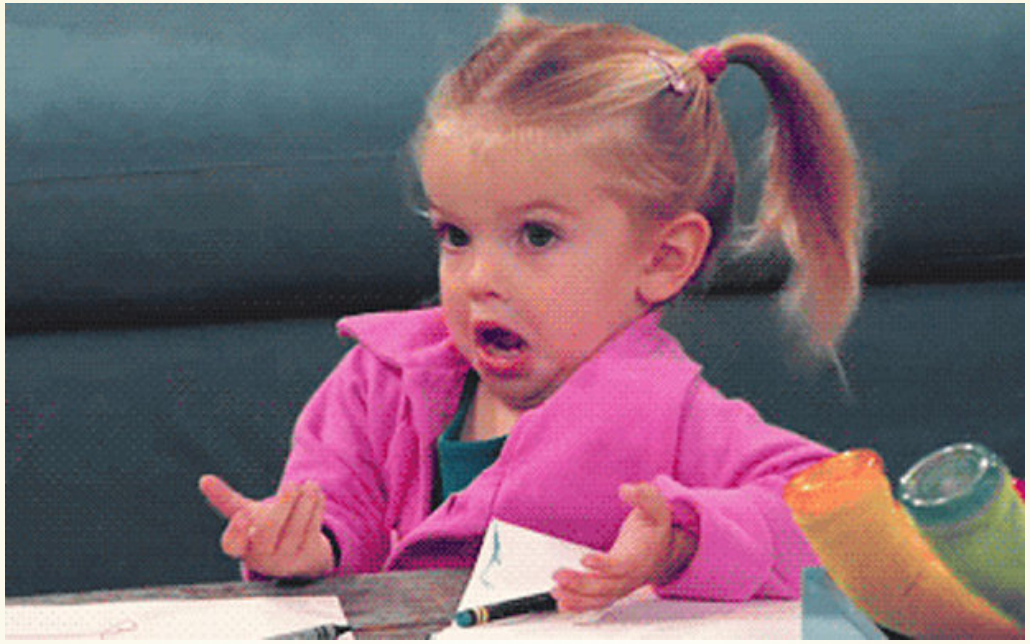


Success!

But then one day...

And your boss is like...

"How did they get in?"



And you're like...

(I sometimes get carried away.)

So, the forensic consultants are called and they find that:

- The "backdoor" has always been there
- In fact, this is how your admins get in
- It's called, you know, "the VPN"
- The attacker got in with one of the admin's credentials
- And you can't just turn the VPN off
- Because then your admins can't get in

- Or they got in via an internal PHP app
- ...which shouldn't have been remotely accessible
- ...but was exposed because of a firewall rule
- ...with a comment "temporary for Bob"
- ...made 2 years ago
- ...and nobody remembers who Bob was any more

You laugh, but I know your darkest fears.

Because they are also my own.

So, in the end PR will put out a statement making it sound like this happened:

Source: DUH

But you still have a problem.

How to prevent human mistakes from becoming security issues in the future.

Let's talk airbags

How can we reduce damage when
something has already gone
terribly wrong?

We'll talk about:

- Airbags for your networks
- Airbags for your servers
- Airbags for your admins
- Seat belts for your team

Networks

What you're probably already doing:

- Stateful firewalls
- VLANs and zoning
- VPNs
- Routine NMAP scans

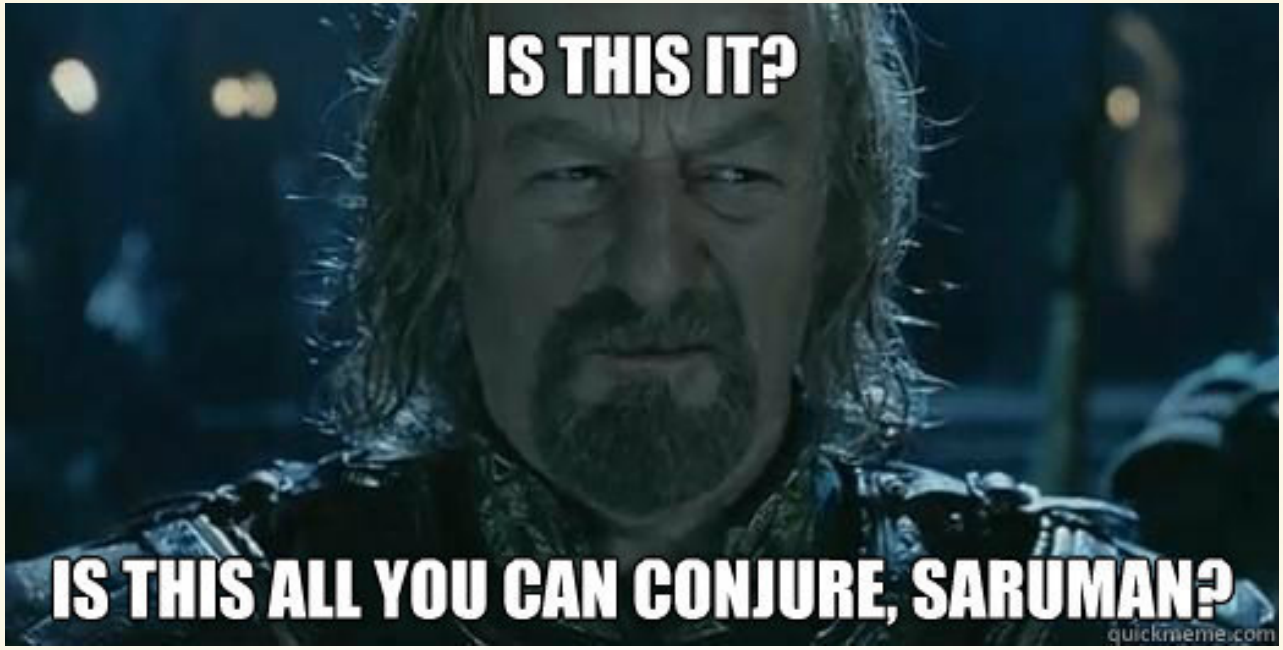
What you're probably NOT doing:

- Payload inspection
- IP reputation
- Actually bothering with DNSSec
- Actually checking TLS certs
- Heck, actually using TLS

Servers

What you're probably already doing:

- Virtualization for service isolation
- Routine security updates
- Centralized logging and monitoring
- ...
- SSH keys vs. just passwords?



We can do better than this!

SELinux

- Yes, SELinux! (or GrSec/AppArmor, pick one)
- It's for when something has gone wrong
- That PHP app has no business connecting to other sites
  - Or looking in /proc
  - Or in /etc/passwd
  - Or scanning your network
  - Or sending mail (spam)
- If you could learn IPTables, you can learn SELinux
- At least use it on your hypervisors!

Stop turning it off!

Use 2-factor auth

- Make your admins enter an OATH token when performing sudo
  - Best to use hardware tokens, not soft-tokens running on phones
  - Look up Yubikeys, they are awesome
- Put their ssh keys on smartcards
  - Yubikey NEO can do this, too
  - We publish detailed docs

Embrace containers

Yes, I did just say that.

image source

All SysAdmins just made that face

Containers are not going to:

- Make sysadmins a thing of the past
- Let you "run any app safely"
- Make security updates unnecessary
- Replace VMs

They help to confine and isolate complex software stacks

Containers are like whole-OS static linking

Containers as airbags:

- Bundle away complex stacks
- Confine them with SELinux
  - You can even write detailed policies using MCS, but that's hardcore
- Add extra network isolation via SDN

Containers as crash test dummies:

- Easy to set up test/production environments
- Easy to write a battery of pre-deployment tests
  - Check what ports it listens on
  - Check that it writes to the right log files
  - Check that it's (un)able to send mail
  - Check that it correctly rejects malicious input
- Deploy to production if all tests pass

This is what DevOps is about: running Ops like you're Developing an app, not letting your devs run your ops.

That's when you end up with
curl http://coolapp.io | sudo sh

Workstations

Woefully inadequate safety

- Confining workstation apps is really hard
- SELinux is near-useless on the desktop
- X is the manifestation of pre-safety design
  - Security only partitions between users
  - Any app gets full access to all of user's X
  - Any app gets all keystrokes
  - Any app gets access to the whole screen
- Any app gets access to microphone, camera

Workstations are notoriously bad at protecting us from harm when things go wrong.

A single vulnerable X app spells complete compromise of a user's desktop (and all their files).

Attackers don't even need root access.

X must die

Yay Wayland, amiright?

X won't die

For quite some time

Q: What do you call software written with the sole purpose of downloading and executing arbitrary code on a user's system, without their explicit consent?

A: Web browser

Web browsers:

- Fantastically complex applications
- ... with huge attack surfaces
- ... that are extended with proprietary plugins
- ... and add-ons that run with minimal protection
- ... from un-signed sources

And we can't live without them.

CVE-2015-4495

The PDF reader in Mozilla Firefox allows remote attackers to read arbitrary files or gain privileges, as exploited in the wild in August 2015.

Don't get cocky, Webkit users, this could have happened to you.

Things that are not safe enough on a user's workstation:

- Keystrokes (passwords)
- Files (private keys)

The most unprotected system on your network is your sysadmin's workstation.

Sysadmin's workstations:

- Are probably on the VPN, with full access
- Probably have access to the password vault
- Have privileged ssh keys on them
  - (never allow to ssh in as root, even with a pubkey)
- Have a list of systems they can ssh to
- Are awash in dangerous keystrokes
  - sudo passwords
  - root passwords
  - password vault master passwords

Mitigation

- Require One-Time Passwords to elevate privileges
  - Best if not a phone app, but better than nothing
- Keep private keys on smartcards
  - this includes ssh private keys
  - we publish a detailed guide on how to do it
- Create policies and educate

Qubes-OS

- The only serious attempt at workstation security
- Works by creating virtualized "AppVMs"
  - Apps are isolated from others via Xen
  - Sidesteps most X vulnerabilities
  - Minimizes impact of when things go wrong
- Is the Volvo of Linux Distros

Teams

We still have to largely rely on user education.

So prepare for failures.

Establish secure comms

- Set up and maintain a web of trust
- Or use an exclusively trusted CA
- Set up and require trusted email exchange
- Set up and require trusted IM exchange
- Establish a policy on what can never go into cleartext
- Enforce that policy to show that you mean business
  - Less "hey, try not to send passwords in an email"
  - More "stick with the security policy, copy attached"

Establish workstation security guidelines

- Create a checklist for workstation hardening
- Provide examples of secure workflows, e.g.:
  - PGP and SSH keys on smartcards
  - Use a dedicated browser for work
  - Run the other browser in a VM
- Have a recommended configuration
  - For hardware
  - For distro
  - For applications

Use code review

- Use dedicated software or just pull requests
- Require sign-off (for later shaming)
- Do the same for sysadmins
  - Use config management backed by git

Write tests and use CI

- These are your crash test dummies
- You can use CI with sysadmins, too
  - This is where containers are great
  - But can be used with VMs, too
  - (this won't be easy!)

FYI, sysadmins will hate you for this.

Create and use checklists

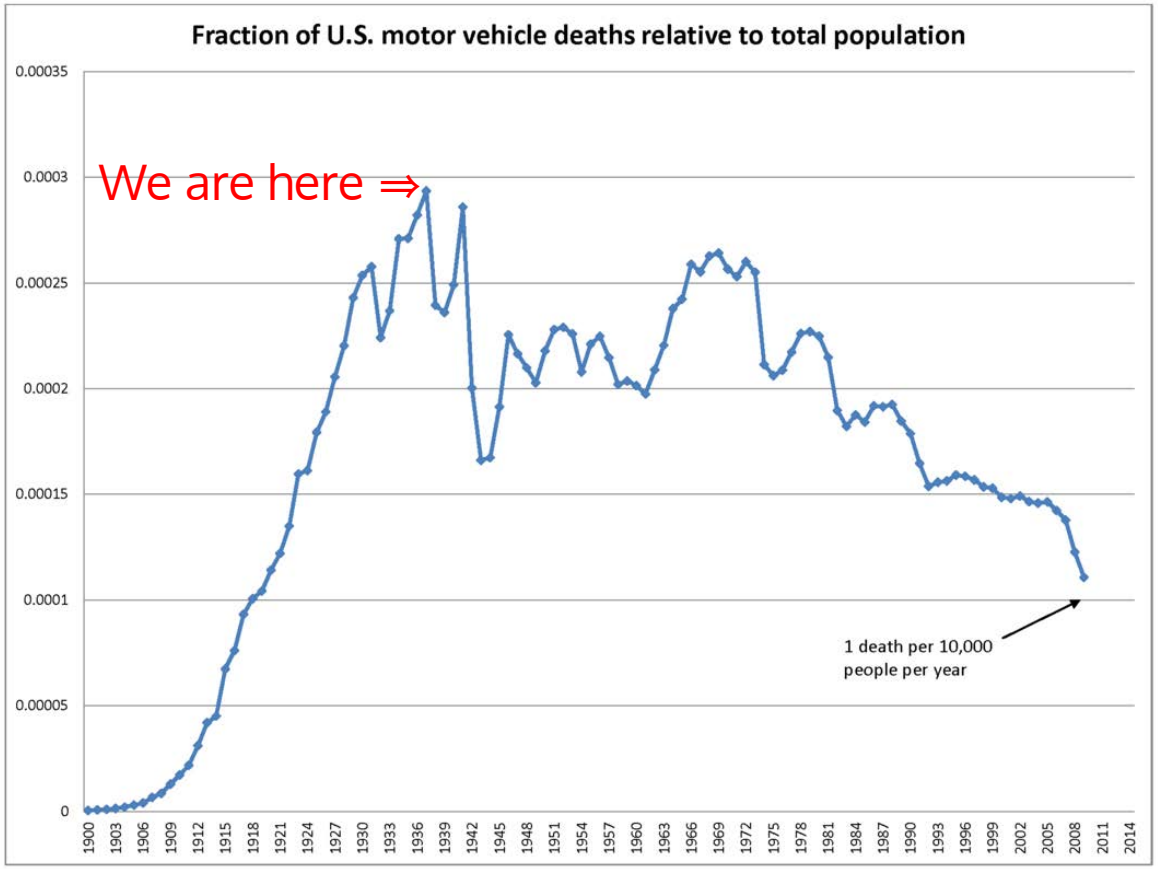
- For system deployment
- For code review
- For staff on-boarding
- For staff close-out
- For rapid admin lockout

Checklists are your most powerful tool to avoid the "oh sh*t moments"

Mistakes will happen

It is our job to make sure they are not fatal

In conclusion



We are at a fork in the road

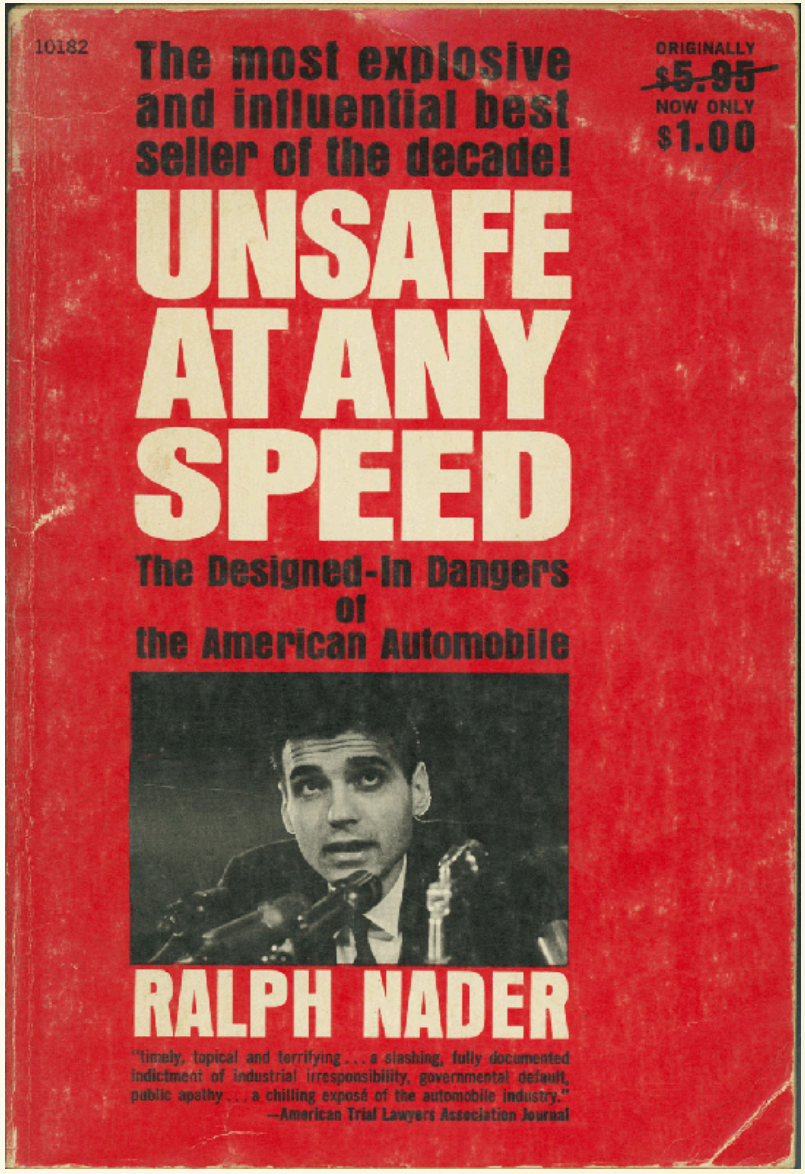
We build systems that are great to drive

But that do not forgive mistakes

We are overdue

For our industry's own "unsafe at any speed" moment.

(And with in-vehicle computing taking off, it's not going to be a metaphor for much longer.)



Our current approach:

- Padded dashboards
- Optional seatbelts
- "Safety-styled recessed hubs"

And a large dose of victim blaming.

The technology is there!

- Seccomp
- Namespaces
- Safer languages
- SELinux/AppArmor/PaX
- Decent crypto

Use them in your stuff!

Kernel devs:

We need more airbags!

Distro devs:

Put in those airbags!

Sysadmins:

Stop turning them off!

Developers:

Stop asking sysadmins to turn them off!

Managers:

Foster secure team practices

And learn how encrypted email works,
it won't kill you.

github.com/lfit/itpol

We are opening up and sharing the policies
the Linux Foundation IT team uses.

Help us improve them.

Let's not take 50 years to get to the point where computing is fun, powerful, and is a lot less likely to maim you when you make a mistake.

Thank you!

@mricon