# Network Security 1: Threats, Assets, Risks

i.g.batten@bham.ac.uk

# Why Network Security?

- Special case of security in general, and in principle 27001 (or similar) picks it up the same as it picks up door locks and disk disposal.
- But has a much longer history: I published in a technical journal about firewalling in (I think) 1993, when security as a specialism was restricted to banks and was mostly about locks.
- We were worrying about securing networks before we were worrying about security more general, because of the Morris Worm (November 1988).

# It’s the Internet

- Network security’s history is mostly about the area around, and fanning out from, your internet connection(s).
    - There were security issues around inbound dialup modems, but they were largely ignored.
    - JANET security, pre-Internet, was very weak.
- A lot of the discussion was, and often still is, about the design of the area around the router that the connection arrives on: firewalls and DMZs.
- In 2018, we need to look deeper and wider.

# What are we protecting?

- Confidentiality
  - We don’t want unauthorised people reading our data
- Integrity
  - Or changing/deleting our date
- Availability
  - And we want the computers to keep working

# What are the assets?

- Information assets are both data and services.
- So that’s files in filestores, email in email repositories, private webpages in intranet servers…
- And also the fileservice, the email service, the web service, the routers that glue it all together, the DNS servers, the DHCP servers…

# Who are the threat actors?

- When considering security, and information security, in the large, they are almost all **insiders**. We have the advantage of physical security.
- But for network security, the threat actors will be in large part **outsiders**.  They may have help from inside, but internal attackers are a smaller part of the problem.
- This course focuses on dealing with outsider risk, but good practice helps in all scenarios.

# Types of actors

- “Script Kiddies”
- “Anonymous” and their equivalent
- Well-funded state actors
  - We need to consider motivation, skill and resources.

# Types of threat

- Service disruption
- Information theft
- Service theft
- Fraudulent changes to data
- Dot, dot, dot.

# Types of risk

- DDoS
- Theft of credentials
- Malware and Phishing attacks; Viruses/Trojans
- Bribery, blackmail
- Cryptographic attacks
- Service compromise
- Again, dot, dot, dot.

# What is our objective?

- To secure data so that it can be **trusted** by authorised users.
  - Data that is not confidential doesn't need to be confidential, but probably does need to be unchanged and available.
  - In general, integrity and availability are often more important.

# Identifying Assets

- Ideally, we start from the 27001 asset register that has been constructed as part of our fully accredited ISMS.
- But more often, we have to be pragmatic: “we haven’t done 27001 so can do nothing” is not a good position to try.
- Naively, we are concerned with assets that are:
    - networked
    - potentially exposed to the outside world
- But we also need internal controls, both to limit insider threats and to prevent escalation of one vulnerability into a wider attack.

# Identifying Assets

- Typical sites may have a mail server, a web server, some sort of file sharing (Sharepoint, etc).
- But larger sites may have a large range of applications including databases of various flavours, test equipment, production lines equipment, ovens, ***centrifuges***…
- The first question is “why am I exposing this to the network?”

# Threats and Risks

- Who might want access to an asset?
  - Topic for another course, but often confidentiality is much less of a problem than integrity or availability.
- What would they do with it?
- What would it cost you if they did it?
- What are the legal implications?
- How might they attack it?

# Assets aren’t just information

- Routers and switches: if you can break into them, you might be able reconfigure the network to bypass security (a network design which survives attackers owning the switches is hard)
- Identification servers (active directory, etc): if you can break into them, you can add new users and give them privileges, or upgrade existing users.
- DNS Servers: a future lecture for the full horror story, but you can do Really Bad Things if you can reconfigure them or otherwise attack them.
- And see also centrifuges.

# Slack for School of Computer Science (Thanks to Jonathan Duffy).

Please join here or look on the front page of the Network Security Canvas module.
https://goo.gl/Tz4v5d

There is a channel for #network_security

Great for collaborating, discussing problems and ideas etc

Feel free to create and join as many channels as you like

Used in many work places

Please join and install the apps

The more people using it the more useful it becomes

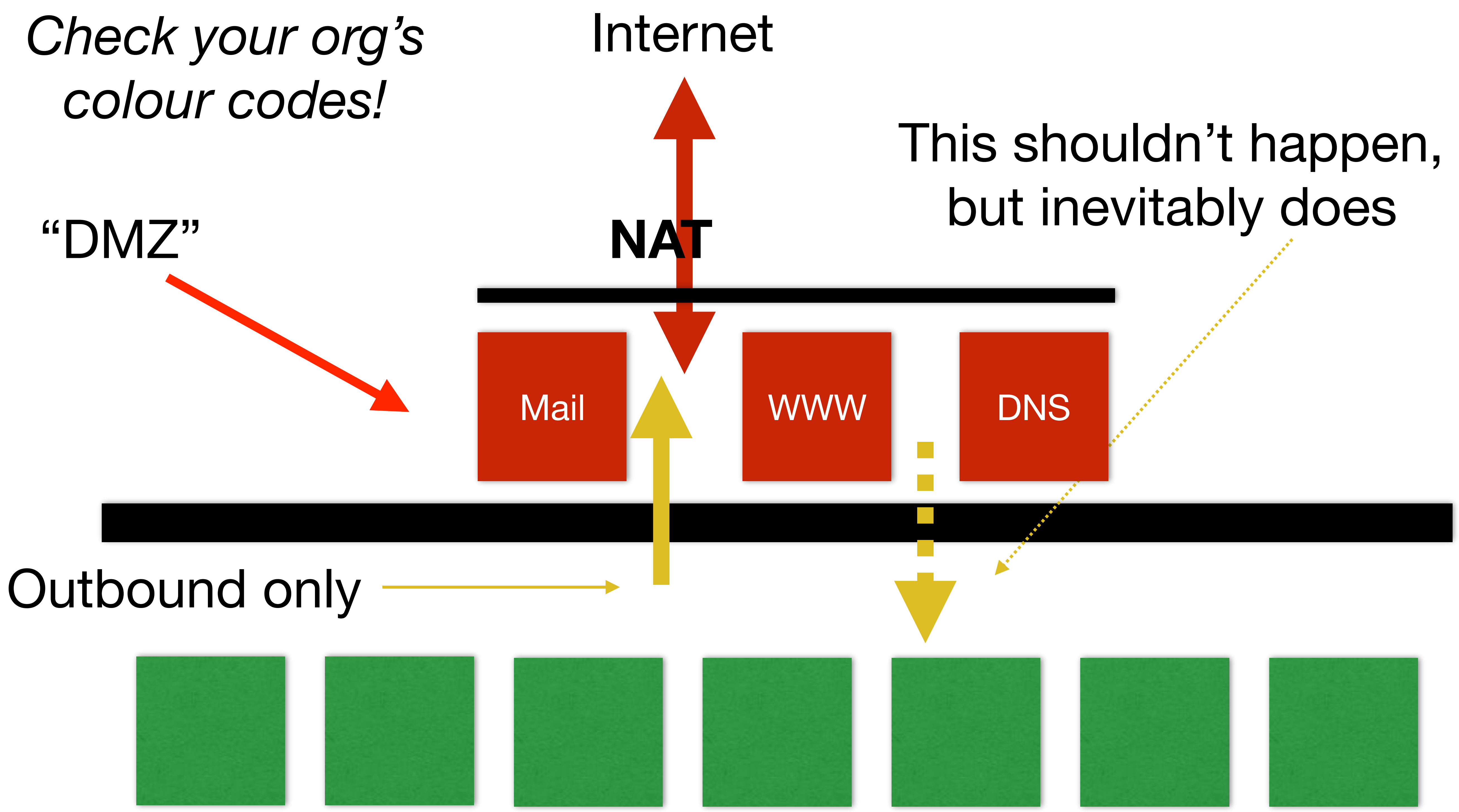
# The two extremes of protecting a network of computer

- Default Deny: pass all connections through a restrictive set of filters and proxies, essentially isolating the internal systems, while providing a small set of machines which communicate with the outside world.
- Default Permit: all internal systems have access to the internet, and are responsible for looking after their own security.

# Approach 1: Default Deny

- One approach is to completely separate everything from the Internet, and then expose a very limited subset of things that need to be available.
- Extremely difficult to do well, and operationally very painful.
- Data crosses the gap in all sorts of ways anyway.
- Companies try to do this and usually give up; much more common in government.  Open question as to how well it works.  Design pattern from the Clinton presidency.

# Default Deny



*Check your org's colour codes!*

Internet

This shouldn't happen, but inevitably does

"DMZ"

NAT

Mail

WWW

DNS

Outbound only

# Approach 1: Default Deny

- Idea is that outside world and inside world can contact machines sat in the DMZ, and the machines in the DMZ are hardened enough to guard temporary data and not be repurposed.
- Crucially, the internal firewall is one-way: it only permits connections initiated from the inside.
  - This is often forgotten, and also very hard to do in a real operational network. For example, both putting an IMAP server in the DMZ and allowing SMTP through are risky temptations.

# Approach 1: Default Deny

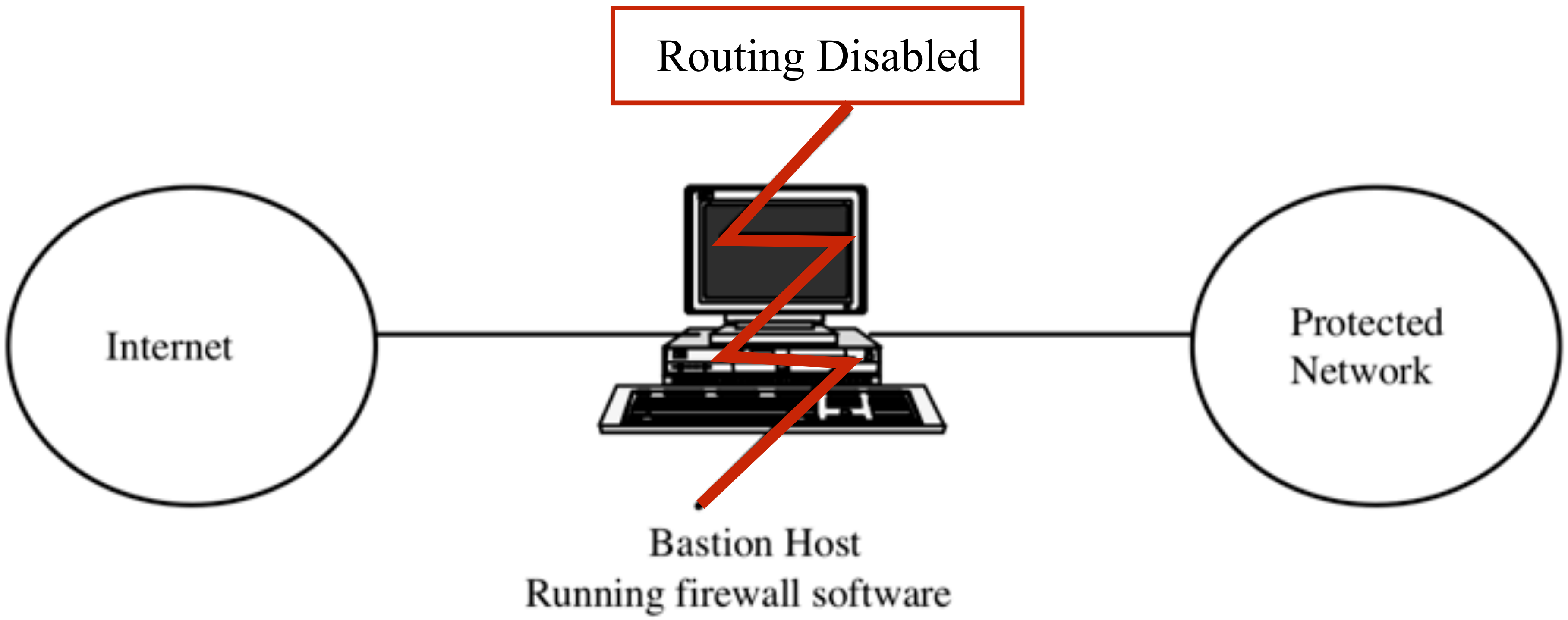
- Formalised by first internet touch point for White House circa 1992, done by Marcus Ranum of DEC (?), then “Trusted Information Systems”, presumably with support from NSA/spooks.
- The “TIS Firewall Toolkit” is still worth looking at, 25 years later. http://www.fwtk.org/fwtk/docs/overview.pdf (following diagrams come from that paper).
- Router-level policies **much** less capable then (no connection tracking, **no NAT**)
- Controlled outbound connections (in that era, ftp, mail and telnet) as well as inbound.
- Also, POP3 still main means for fetching email from servers, because no mobility (primary use-case for email fixed desktop device, GSM several years in the future, laptops several years in the future, Blackberry etc 10+ years in future).
- The world has changed a lot. The design pattern hasn’t, or at least hasn’t enough.

# Note interesting change in terminology

- Today, “firewall” has come to mean a box which sits between two networks, filtering packets (and by implication connections) which pass between them.
- When design patterns were first arising, “firewall” meant the whole area of one or more filtering (also “screening”) routers — with less flexible rules than today — and machines operating as proxies/bastions/etc, or it meant the proxies.
- Some of the slippery weakening of the design comes from this change in terminology: when people installed firewalls in the past they were much more complex and capable because of the computers involved.  Now they’re just packet filters.  Fancy packet filters, as we’ll see.  But they only understand protocols insofar as they need to for NAT, and that is completely defeated by encryption.

# Note historic use of “firewall”

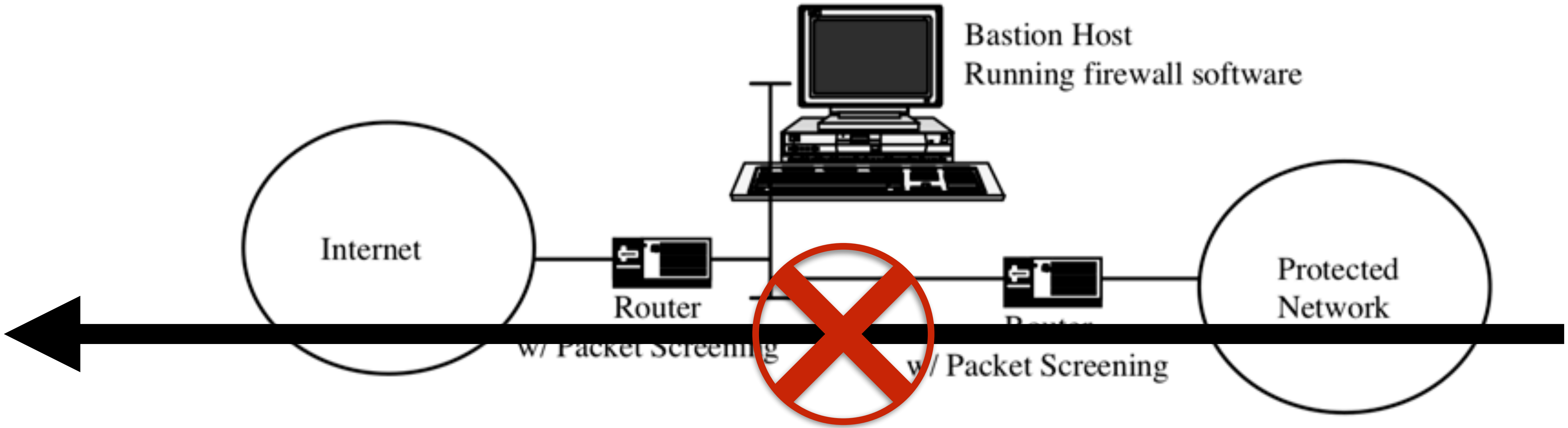
***Figure 1: A dual-homed gateway***



- Firewall element is the “bastion host”.
- No routing, no NAT. Log in to bastion, connect out from there, and (perhaps) vice versa.

# Modern setups were envisaged, but different

*Figure 3: A Screened Subnet Gateway*



- Again, firewall refers to the whole thing, or to the proxy software.
- Absence of NAT means no path In->Out for packets, everything is via the bastion host.

# UKUUG Winter Conference, 1996: many of you not born

- “The second day opened with **“Network security without firewalls” by Ian Batten.** This presentation documented Ian's experiences and opinions of firewalls, and why he believes they give no advantage for the system set-up he uses. He overviewed the advantages (firewalls do work, and can be very beneficial if administered properly) and disadvantages (complex to administer, can breed complacency) of firewall systems, and the alternatives to them. The disadvantage of the complacency issue was stressed. **There is a general attitude of “we've got a firewall, so we're OK as regards security”. This is terribly naive. For most systems, the greatest threat is from within. The percentage of adept penetrations from outside the system is very low in comparison with the percentage of data loss from users taking software and information off the system. The alternative system proposed was that of a screening router coupled with host security. So, all packets that are anomalous (eg claiming to be from that machine) or unnecessary (eg ICMP redirects) are denied, and internal measures are taken to help counter any external attack.** These include using encrypted ident as an auditing tool, using ssh for remote access to the system, and extensive system logging among others. The latter was stressed extensively, though care must be taken to ensure log files are resistant to tampering.”

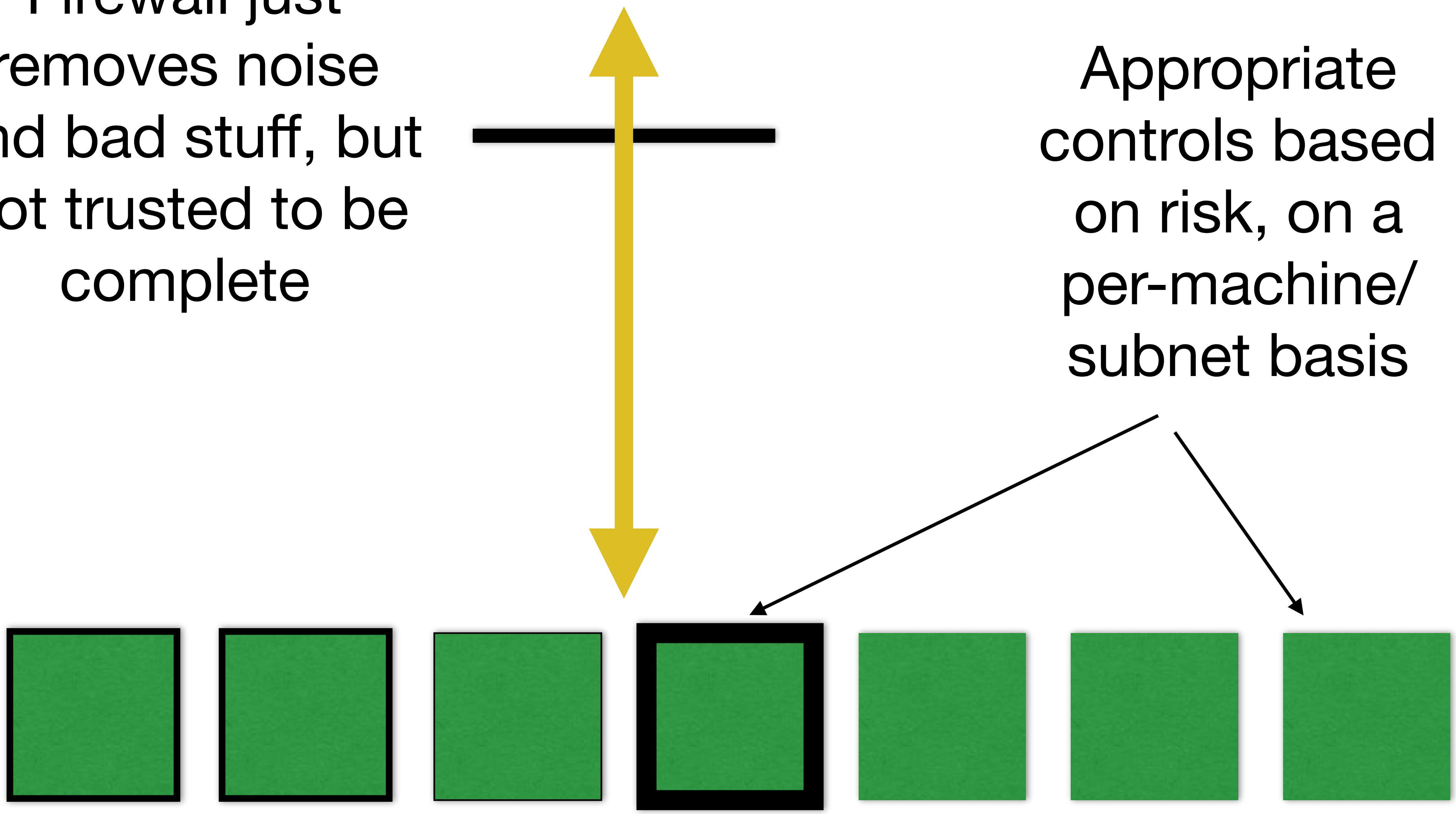
# 22 years later

- I was wrong about identd, and history has proven me wrong.
    - Idea was “connect to port on machine, provide a remote address and a local port, be told owner of that socket”. Relies on assumptions that were with hindsight broken even in 1996 (and probably in about 1993 when I started using it).
- I stand by the rest of the presentation. I have rescued a copy of the accompanying article from backups, so you can see the debates in this lecture are not new:
- https://www.batten.eu.org/~igb/security.pdf
    - also **https://goo.gl/bY3347**
    - Written in LaTeX 2.09, which handily modern xelatex processed without demur.
    - You might find it worth reading to get an overview of some of the issues, even at this distance in time.

# Approach 2: Default Permit

- The whole network is exposed to the outside world, with individual assets protected on a case-by-case basis.  Network-wide protection relies on security policy, ideally enforced by Group Policy etc, often not enforced or enforced only by (or on!) paper.
- University networks would be an obvious example, but any business that is moving towards bring your own device is implicitly doing this.

# Default Permit



Internet

Firewall just removes noise and bad stuff, but not trusted to be complete

Appropriate controls based on risk, on a per-machine/ subnet basis

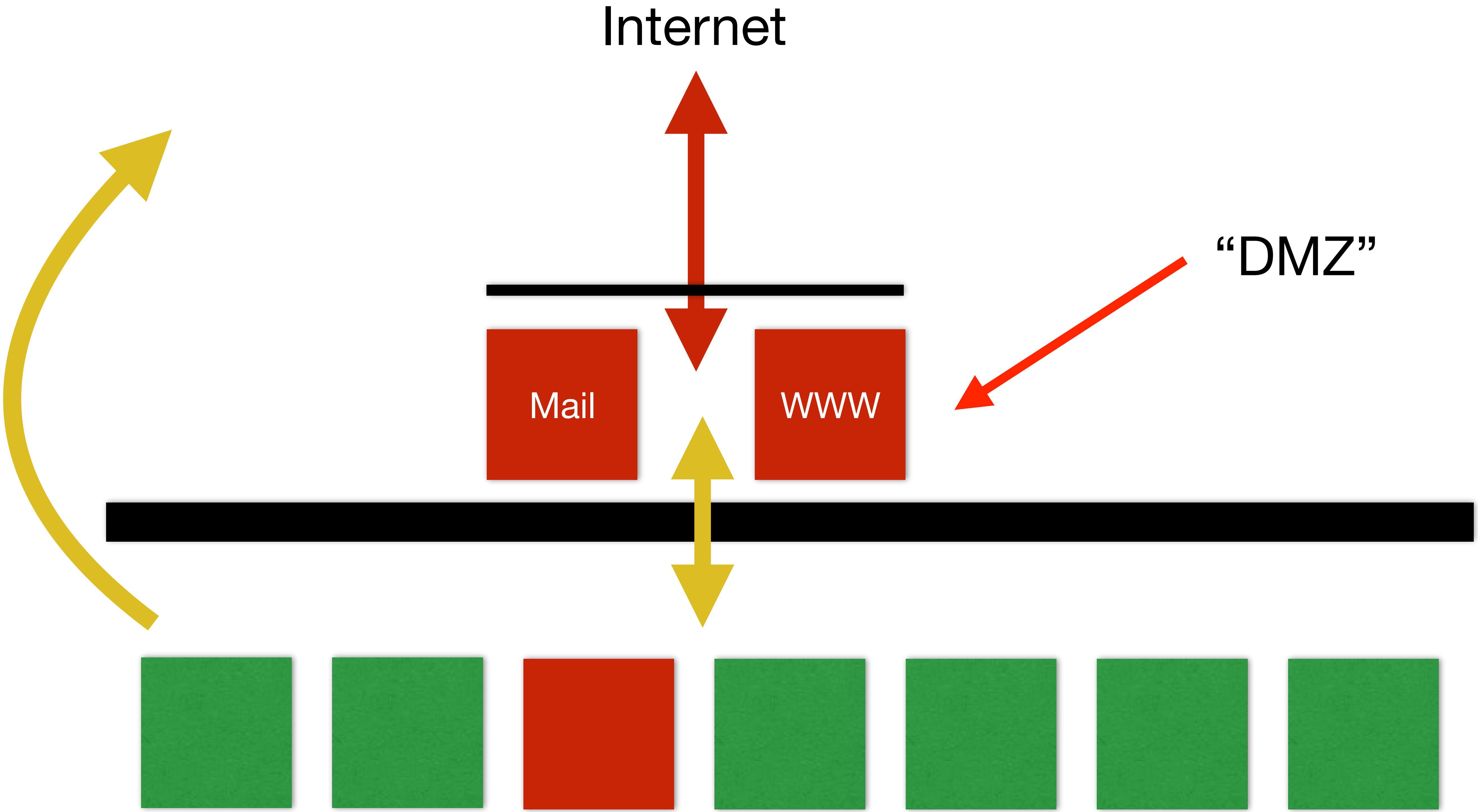
# Benefits of Default Deny

- You don’t need to inventory all your assets, just the stuff that is exposed to the outside world.
- You can rely on the inside firewall (or whatever) to protect everything that isn’t exposed.
- Users can’t (in general) expose stuff by accident.

# Problems with Default Deny

- It stops business moving quickly
- Some activities are almost impossible, or result in perverse architectures that expose more than would otherwise be at risk (large databases).  And it doesn’t work for IoT applications (hence the messy use of brokers for Hive, Nest, etc).
- It can be very expensive to implement well, and has very nasty failure modes.
- In 2018, it is an open question how much protection the “inside” router actually provides anyway.  It gets treated as an airgap, and it simply isn’t.  The cult of the firewall is strong.

# How Default Deny Ends Up



# Benefits of Default Permit

- It’s much more flexible, and new applications can brought into service very quickly
- Because all sensitive systems have to protect themselves, it doesn’t matter (as much) if a nearby system is compromised.
- Administrative load is spread out over departments

# Problems with Default Permit

- All machines are exposed to substantial threats
- Lack of central policy means “weakest link in chain” is a problem
- Auditors and other assessors will be very, very nervous
- But in reality, many networks are becoming like this

# So increasingly…

- We use network-level security to reduce the level of “simple” attacks, perhaps without having a formal DMZ (either by intent or by accident).
    - The formal DMZ pattern is used to protect small islands of sensitive systems, running defined, relatively slow-changing workloads.
- We use more appropriate host-based security to protect information assets elsewhere.
- We use policy to make sure that every host is protected (in the way that we use talking to our children to make sure that they never misbehave). *Note: This is sarcasm on many levels.*
- On campus, you have islands of protected networking in amongst a sea of what is effectively the open Internet.
- It is instructive if you have the opportunity to observe the University’s contortions on this topic, 20 years after the fact.
- Think: “how sensitive is the stuff on the CEO’s laptop when he’s travelling? What firewall is it behind?”