IT Security:
New Trends, Ancient Techniques

Alec Muffett

Principal Engineer, Security

Fin Svcs Adv Tech Proj

Sun microsystems

We make the net work.

Proposition

- There are many security “components” available today…
  - “Components” == Tools, Utilities, Appliances…
  - Available across many platforms…
  - Addressing many specific security risks…
  - Multi-billion dollar industry
  - But…

Proposition

- Many useful security components are available, but...
  - They are easily misassembled, misused or misconfigured
  - They are often used individually, where they would better used in combination.

Further...

- Regarding “components”
  - People tend to seek “homogeneity”
    (eg: using a single firewall vendor, everywhere)
  - ...whereas “diversity” yields robustness
    (at a cost of management complexity)

Further...

- Without proper design/architecture, you will be wasting money
  - It is perfectly possible to spend $1,000,000's and still have a terribly insecure network.
  - It is further possible to spend almost nothing, and yet improve your security enormously.

Therefore, today...

- We shall:
  - review what may go wrong.
  - review how it can go wrong.
  - suggest a strategy, a design philosophy, which can help to address the problems.

As a first illustration...

- First, consider this problem:
  - "The misassembly of security components"
  - In other words:

    "The Right Components, Put Together Wrongly"
  - Example:

    TopBox Video (2m 50s)

To analyse this, let's...

- Summarise the history of IT Security
- Compare it to the “State of the Art”
- Review individual security tools, and the issues they sought to address
- Determine whether we are still defending against the “older” security issues

IT Security –
An Approximate History

IT Security Eras

- 1955..65
  - Computers are
    - “too complex for ordinary people to comprehend”
  - yielding
    - “security through obscurity”

IT Security Eras

- 1965..75
  - Advisory separation of different “users” within a computer
  - Technology (mostly) not advanced enough to support mandatory separation.
    - Lack of “Virtual Memory”, etc

IT Security Eras

- 1975..85
  - Robust partitioning / brokering of file access via “file permissions”
  - Use of strong “virtual memory” to enforce system integrity
    - mandatory separation of user/program memory...
  - ...but not in all platforms
    - eg: Personal Computers, minicomputers, ...

IT Security Eras

- 1985..95
  - Password security extended to basic network services (eg: Telnet, HTTP)
  - Networking “too complex for ordinary people” yielding “security through obscurity” again
    - ...yet early “buffer-overflow” exploits still occur

IT Security Eras

- 1985..95 (...more...)
  - Personal Computers are virus-ridden from lack of hardware to implement “integrity”
  - Compartmented/Certified Systems considered “exotic”; Military & Banking only?

IT Security Eras

- 1995..now
  - Partitioning of service access via firewalls
    - Firewalling used as panacea, a cure for all ills
    - Impact upon network architecture and throughput
  - Personal Computers employ permissions, virtual memory, etc, to ensure integrity...
  - causing subsequent growth in “macroviruses” and “active-content” exploits.
    - Hackers using bugs in popular applications, to fill the void caused by more-secure operating systems.

IT Security Future?

- 2005+ ...
  - What is the next big, open resource that is fit to protect with mandatory controls?
    - Encapsulated Data Security & Per-Object Crypto ?
    - Proximity wireless, Bluetooth?
    - SMS-Firewalling / Antivirus?
  - This - perhaps - is a exciting business opportunity for those with foresight
    - ...but good media contacts would not go amiss.

Implementation Cycles

- Generalising:
  - New resource/tool becomes available

    Identity, Filestore, Network, E-mail...
  - Resource/tool grows in popularity
  - Access restrictions for the resource are layered-on afterwards!?!

    Identities,
    Passwords,
    Permissions,
    Firewalls,
    Virus/Content Scanners...

Security Deployment

- Problem:
  - Access controls which are designed "after the fact" are often sub-optimal.
    - Eg: Password protection on plaintext HTTP
    - Eg: Session-State Cookies in HTTP
    - Eg: 40-bit WEP in 802.11b
  - Arguably all of the above could have been forseen and implemented "properly"

Security Deployment

- In security, often only the latest "trendy" issues are managed...
- ...to the detriment of older issues.
  - Weak file permissions on a big server
  - Ignored because:

    "The firewall does all our security!!!"

    "We have faith in our firewall to protect us!!!"

Security Deployment

- Don’t believe me?
  - How many people here have checked the permissions upon, fixed, and hardened the security upon every server that they own?
  - How many people have checked their logfiles within the last 7 days?

So Why Do Security?

- Fundamental questions:
  - What are we protecting?

    Data can has value to us, and also to “others”.

    Data is valuable but intrinsically defenceless.

    Data exists in more places for shorter or longer periods of time – caches, routers; how many of these places do you actually own?
  - How shall we protect it?

    So what we actually do to protect that which we value?

Issues of Implementation

- We (usually) do not protect data!
- Instead, we actually protect the containers where data exists!
  - But: data exists in many places!
  - Hence the need to defend:

    Multiple data containers

    In multiple places

    At the same time.
  - This partly explains why security is “complicated”

Same Old Problems

- In protecting data containers, we face challenges such as:
  - Over-reliance upon one security technology
  - Issues of “blithe trust”
  - Reusable weak authentication
  - The right tools, put together wrongly

Same Old Problems

- Overreliance upon single technologies
  - Obscurity
  - Permissions
  - Passwords
  - Firewalls & IDS
  - Cryptography
  - In real life, these are not tenable / acceptable:

    Potato famine

    Antibiotic Resistance in Bacteria

    ...so why is over-reliance accepted in IT?

Same Old Problems

- Blithe Trust
  - Unauthenticated identity
    - Many protocols take identity on trust.
  - For instance:
    - Forged passports / identity papers
    - Social engineering
    - HTTP session cookies
    - Even “Buffer overflows”

Same Old Problems

- Reusable weak authentication
  - Plaintext passwords
  - Unencrypted Communications
  - Compare:

    Story of “Ali-Baba and the 40 Thieves”

    Reusable Password: “Open Sesame”

    Published circa 950AD

    A 1000-year-old IT security issue!

Same Old Problems

- Right Tools, Assembled Wrongly
  - Firewalls with far too many “holes”
  - Firewalls with too much complexity
  - Same firewall technology everywhere
  - Poor Design

    Example: Top Box video (previously…)

    Example: Simple SSL Accelerator (later…)

How to fix?

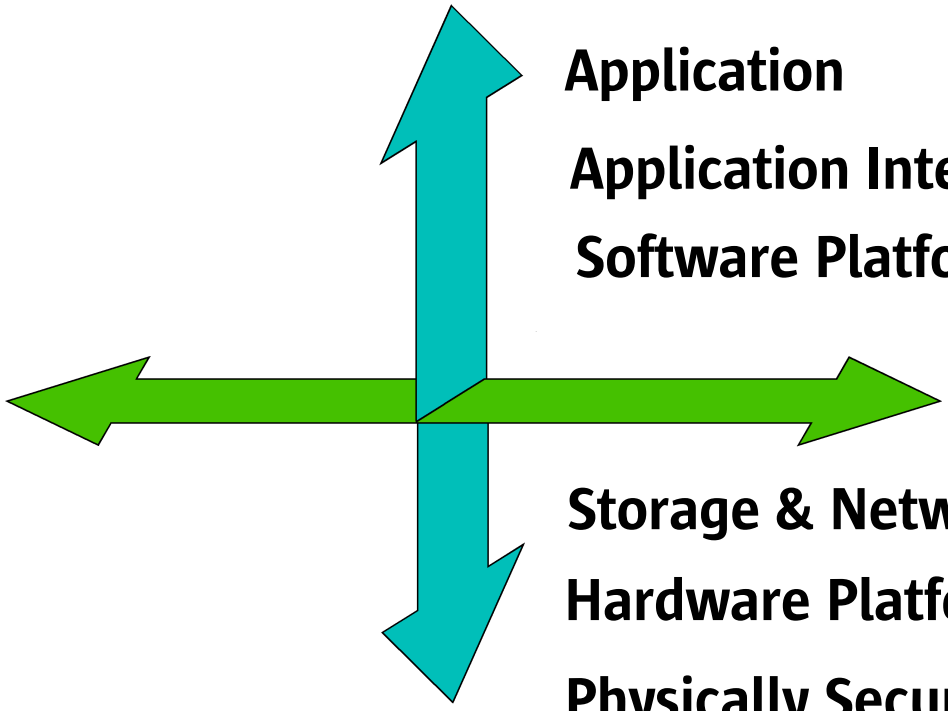
- If everyone else is implementing security wrongly, how can we implement it correctly?
- Do we actually understand security?
  - how big is the challenge of “security”?
  - what aspects of design must we address?
  - how can we be “end-to-end” secure?

Attempts to Address Security: “End To End” Security

End-to-End: Communication

Secure & Authenticated Communication

End-to-End: Integration



Application

Application Integration

Software Platform & OE

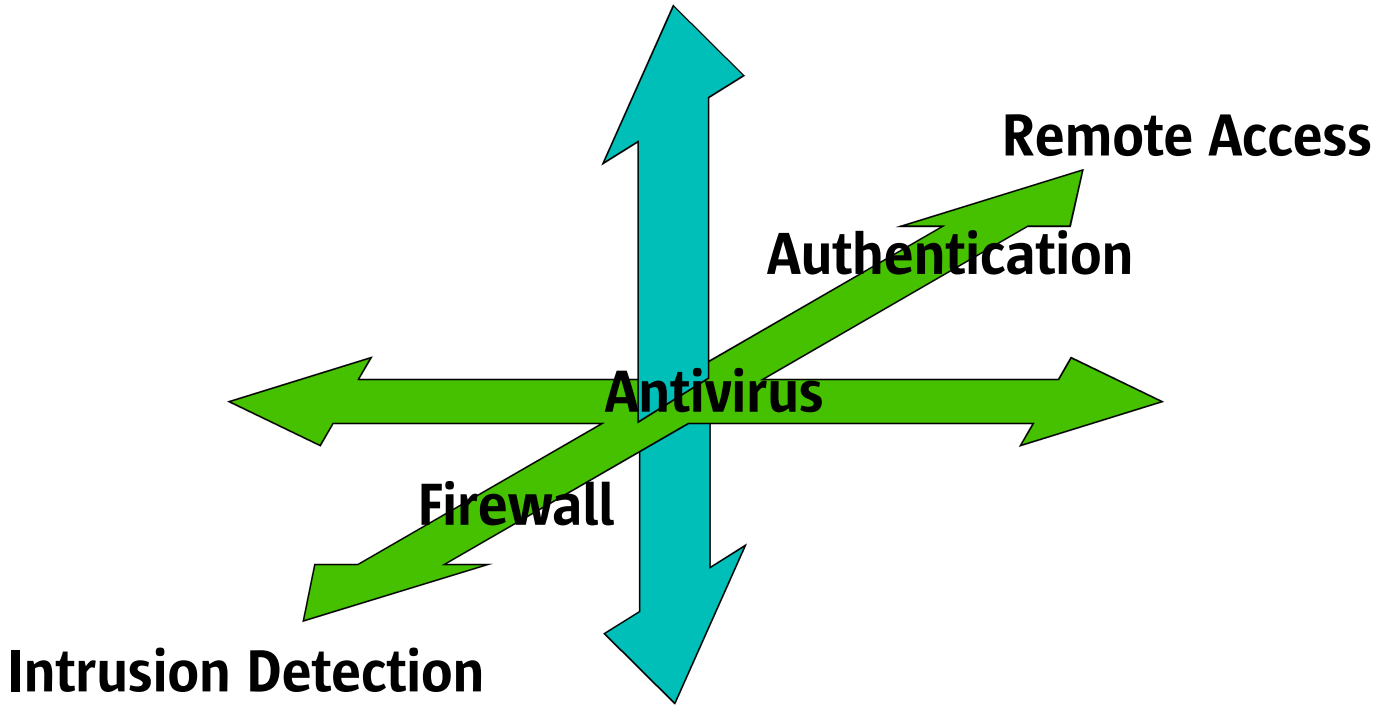
Storage & Network, Other HW

Hardware Platform, CPU/Mem

Physically Secure

Proper Integration of Components

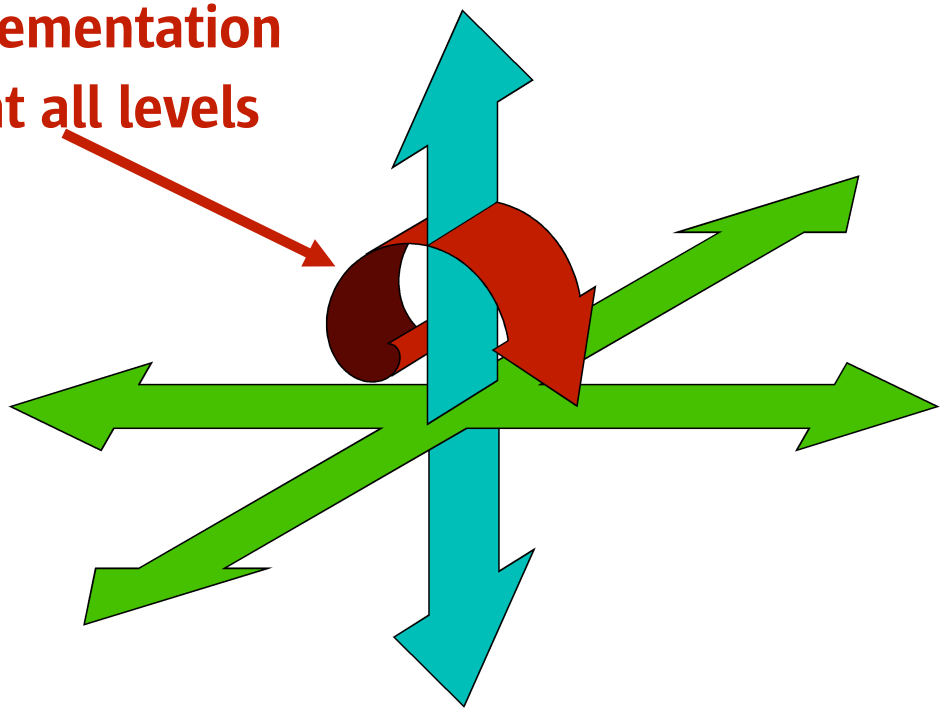
End-to-End: Functionality



Spectrum of Security Functionality

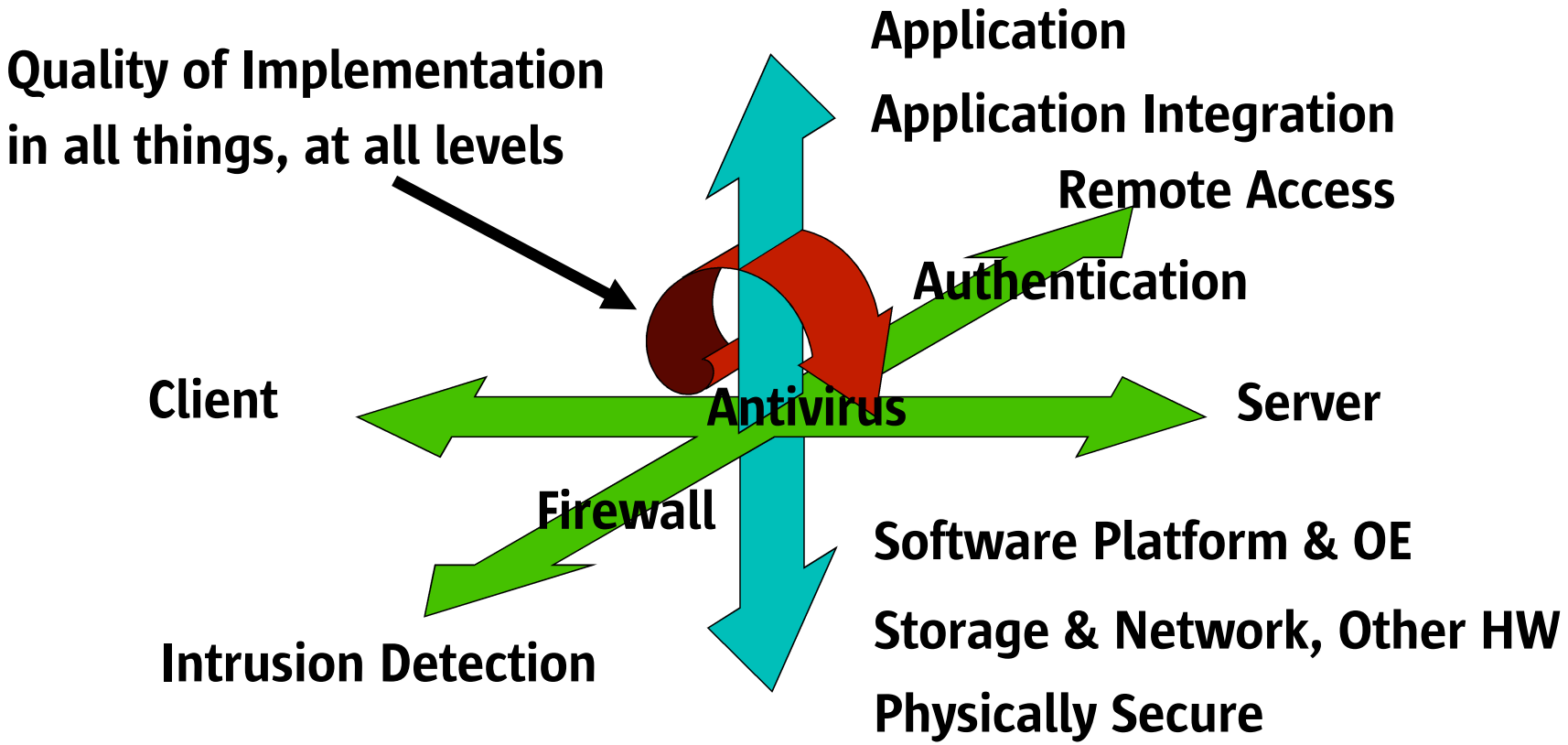
End-to-End: Quality

Quality of Implementation
in all things, at all levels



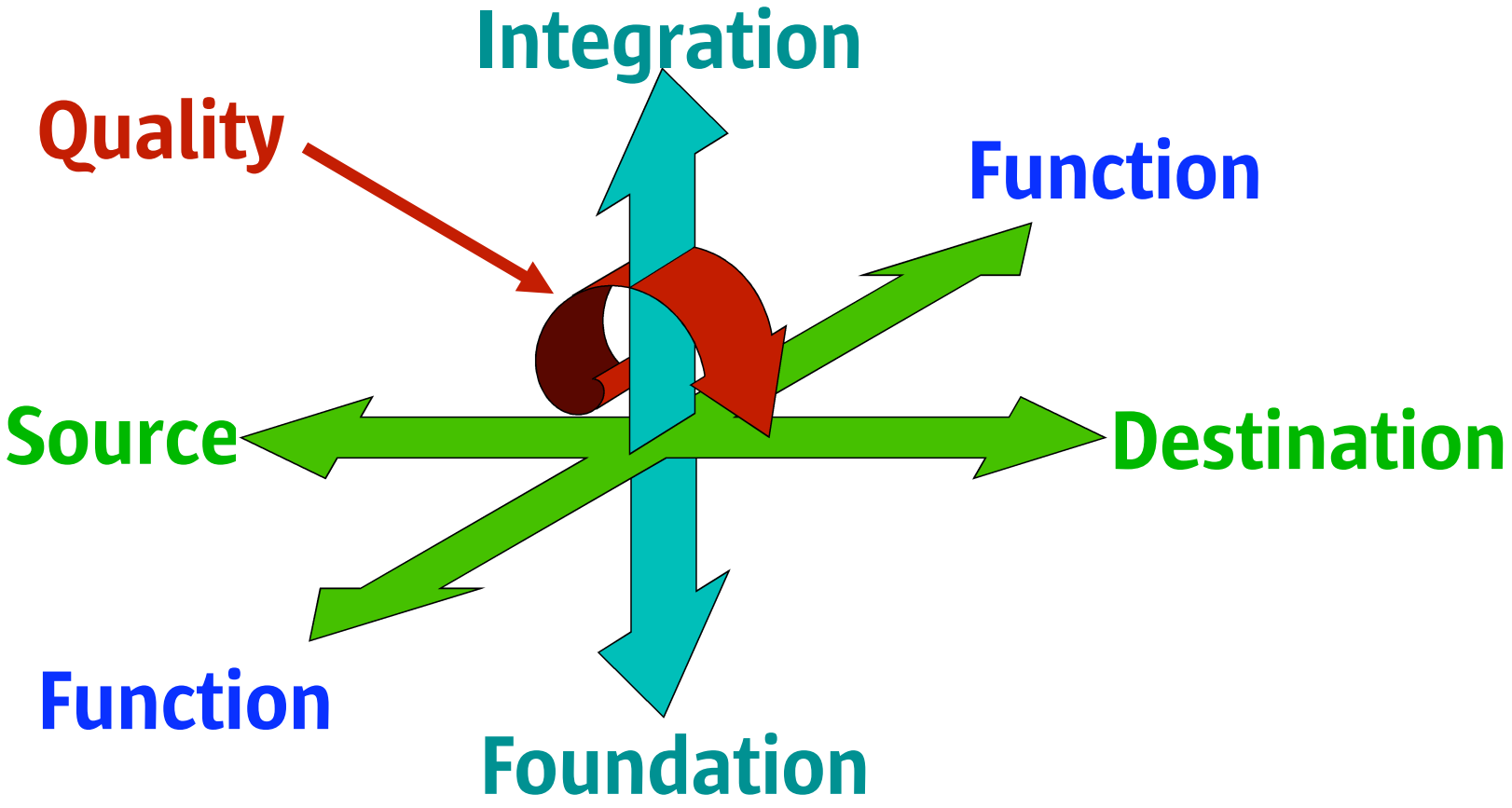
Quality of Components

End-to-End: 4-D Security



Apparently Confusing, but...

End-to-End: Simple 4-D Security



...actually rather simple.

When 2-D Drawings Fail...

There is even a fifth, “Human” dimension to security, that which pertains to having correct “People, Policy and Procedure” - there are also many others.

Now consider:

Does your security solution address all of these dimensions?

If not...

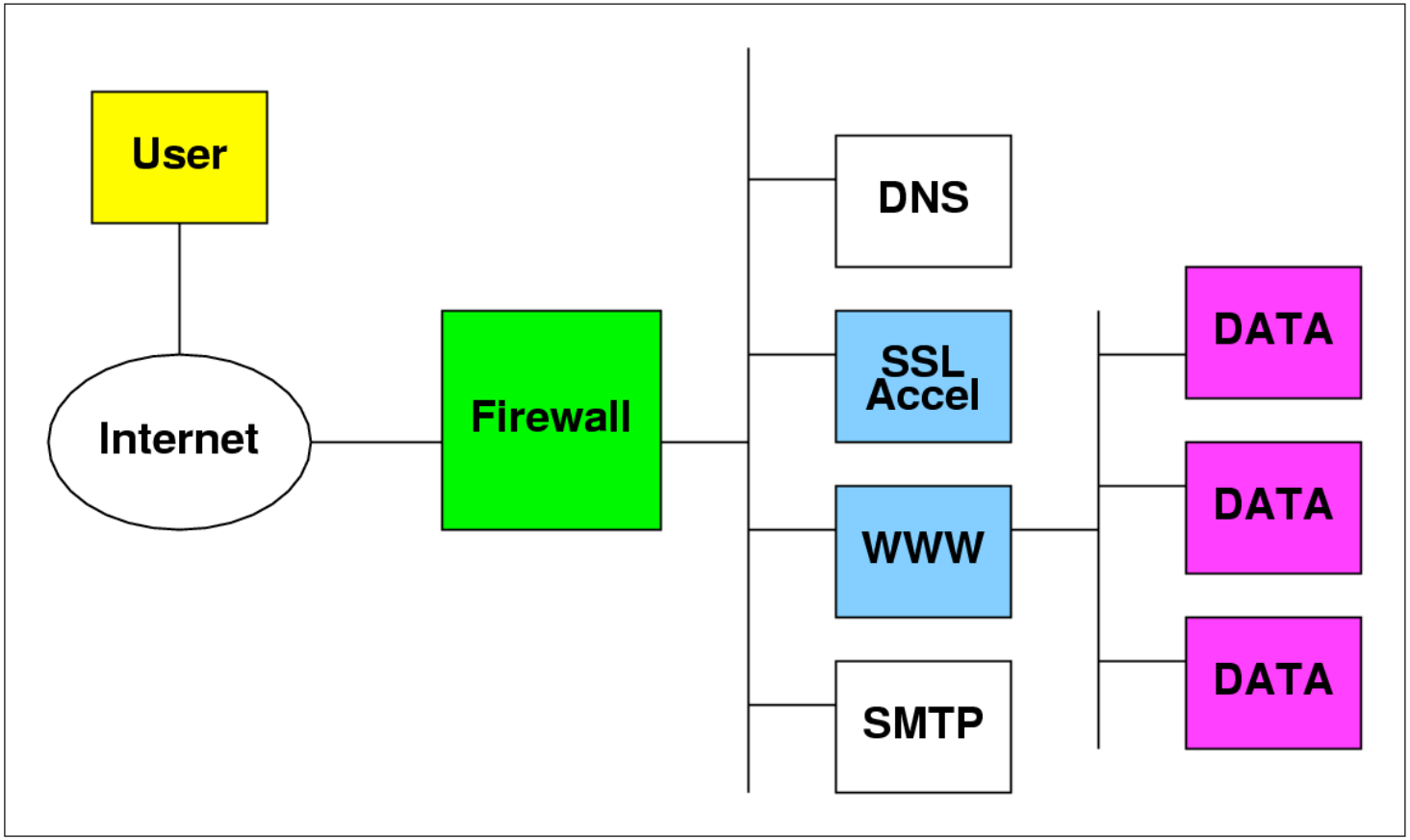
If not, perhaps it needs rethinking.

Better Security Through Better Design

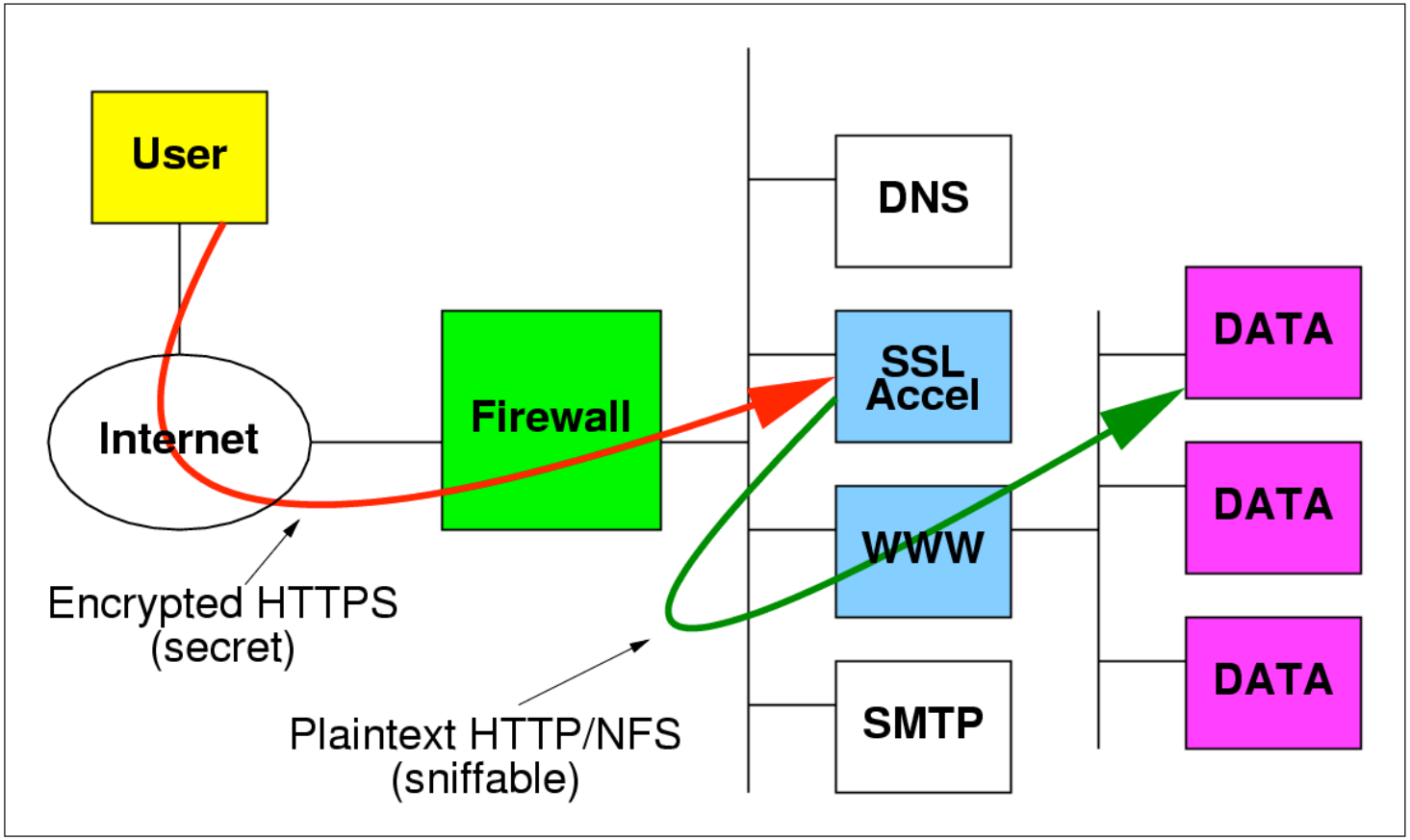
Design Example

- Wouldn’t it be nice to get some extra security for free?
- Here’s an example…

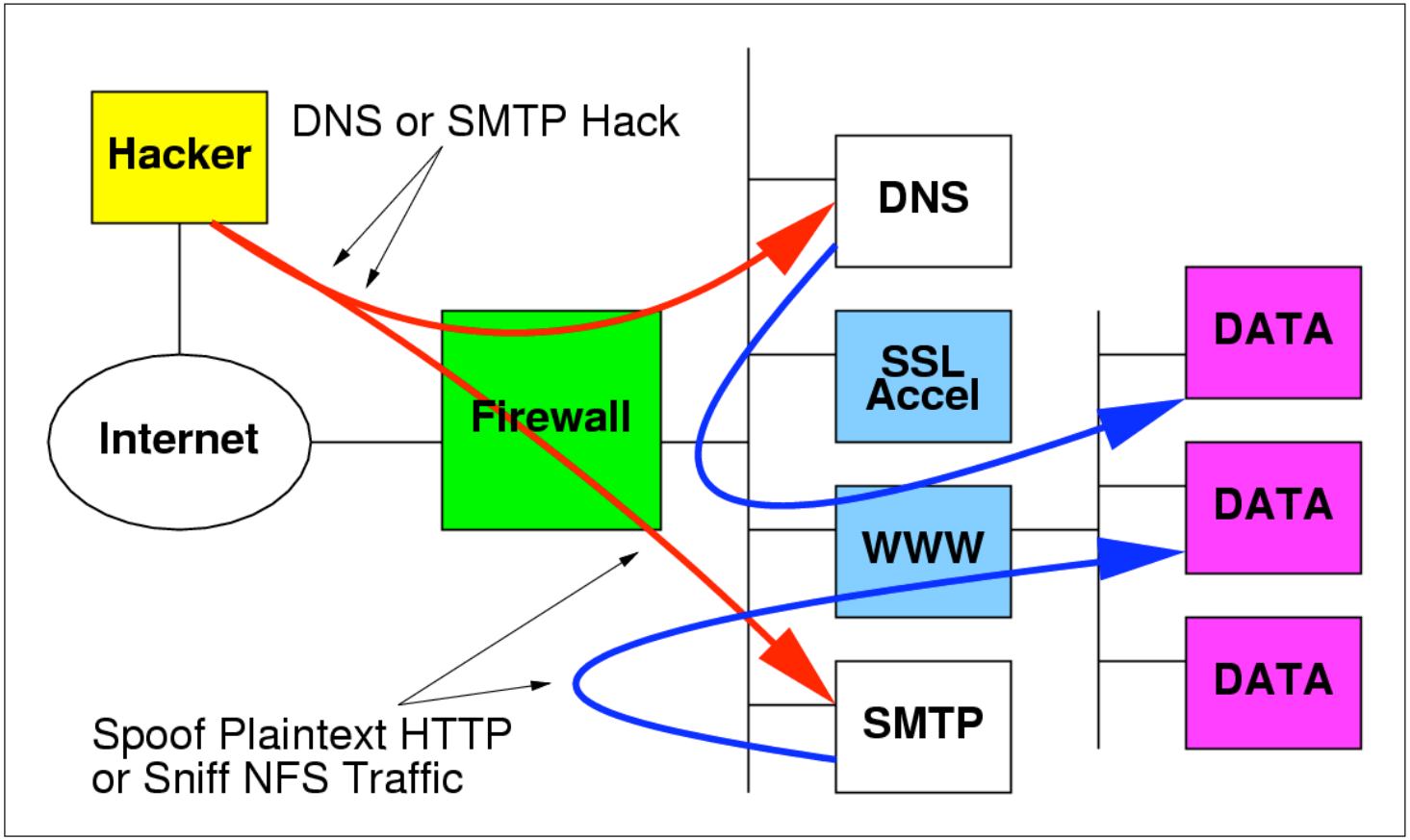
Design Example: SSL



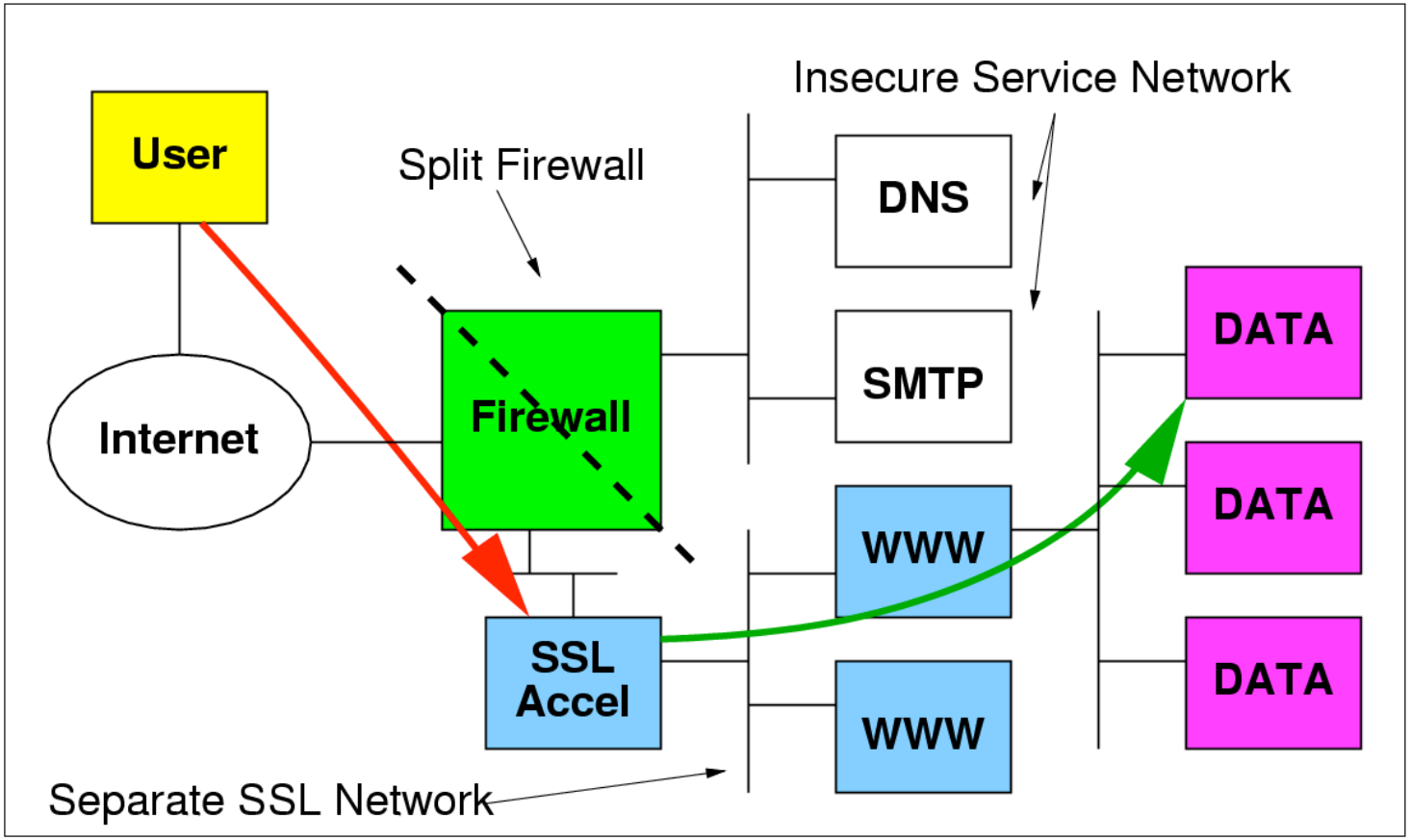
Design Example: Theory



Design Example: Oops!



Better Design



Solution?

- How do we get security for free?
  - Clever Network Design
    - Bear risks in mind when laying-out architecture
    - Design so that (some) problems never arise
  - Clever Host Design
    - Build computers so they are less-subject to attack
    - Design systems for extra robustness
  - Overall: apply “Defence In Depth” philosophy
    - So, what is “Defence in Depth” ?

The Philosophy of “Defence In Depth”

Defence in Depth

- Motto #1
  - Use multiple, independent, different, mutually-reinforcing security technologies
- Motto #2
  - Use whatever works, is manageable and available, and configure them sensibly and as simply as possible
- Motto #3
  - Employ a “default-deny” stance

    ie: “you can only see that which we publish”

Defence in Depth

- Typically benefits from use of:
  - Multiple
  - Independent
  - Different
  - Mutually-Reinforcing
  - ...Security Technologies
- Not a 100% solution...
  - ...but nothing ever is!

Defence in Depth

- The usual example for implementing "Defence in Depth" is a castle:
  - A castle is an excellent example of network defence using multiple technologies

    Normally at this point, your speaker would include pictures of towers, walls, parapets, and gates...

    I decided to do something slightly different...

    I decided to find a castle and attack it...

    With my video camera.

Summary of Approach

- A “Defence in Depth” design applies:
  - Use of different technologies with different failure modes
  - Multiple layers of security which work to reduce the visible profile available for attack
    - You only see 10% of 10% of 10% of attacks …
  - There may be some loss of some auditing information between layers, but…
    - Are you doing security research or…
    - Are you trying to defend yourself?

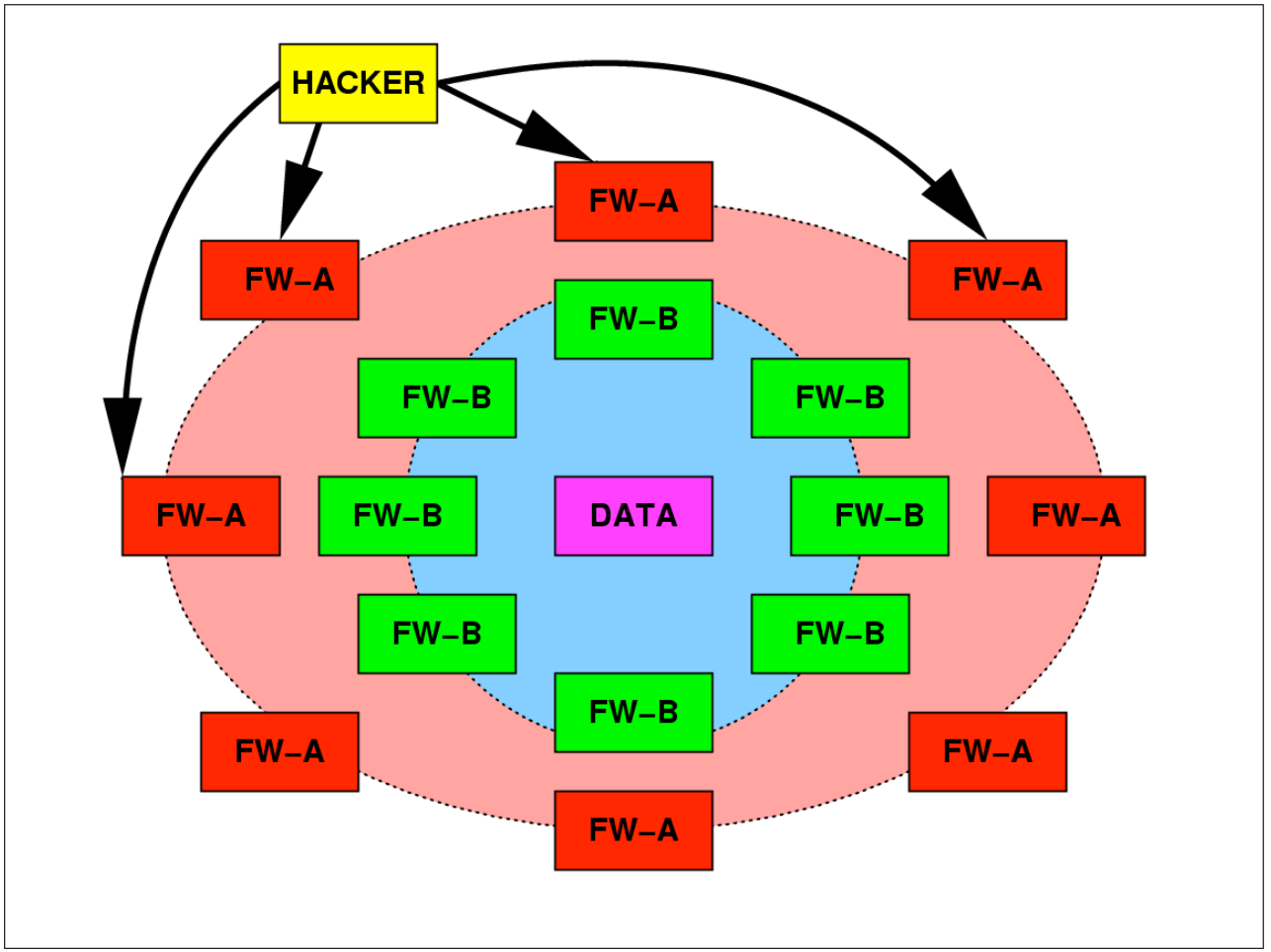
How Much Depth?

- When do I stop adding layers?
  - Good question!
  - Depends upon what you are trying to protect.
- Judgement call
  - Personally, I reckon when all major risks have been mitigated twice, in different, independent ways, that's the minimum.

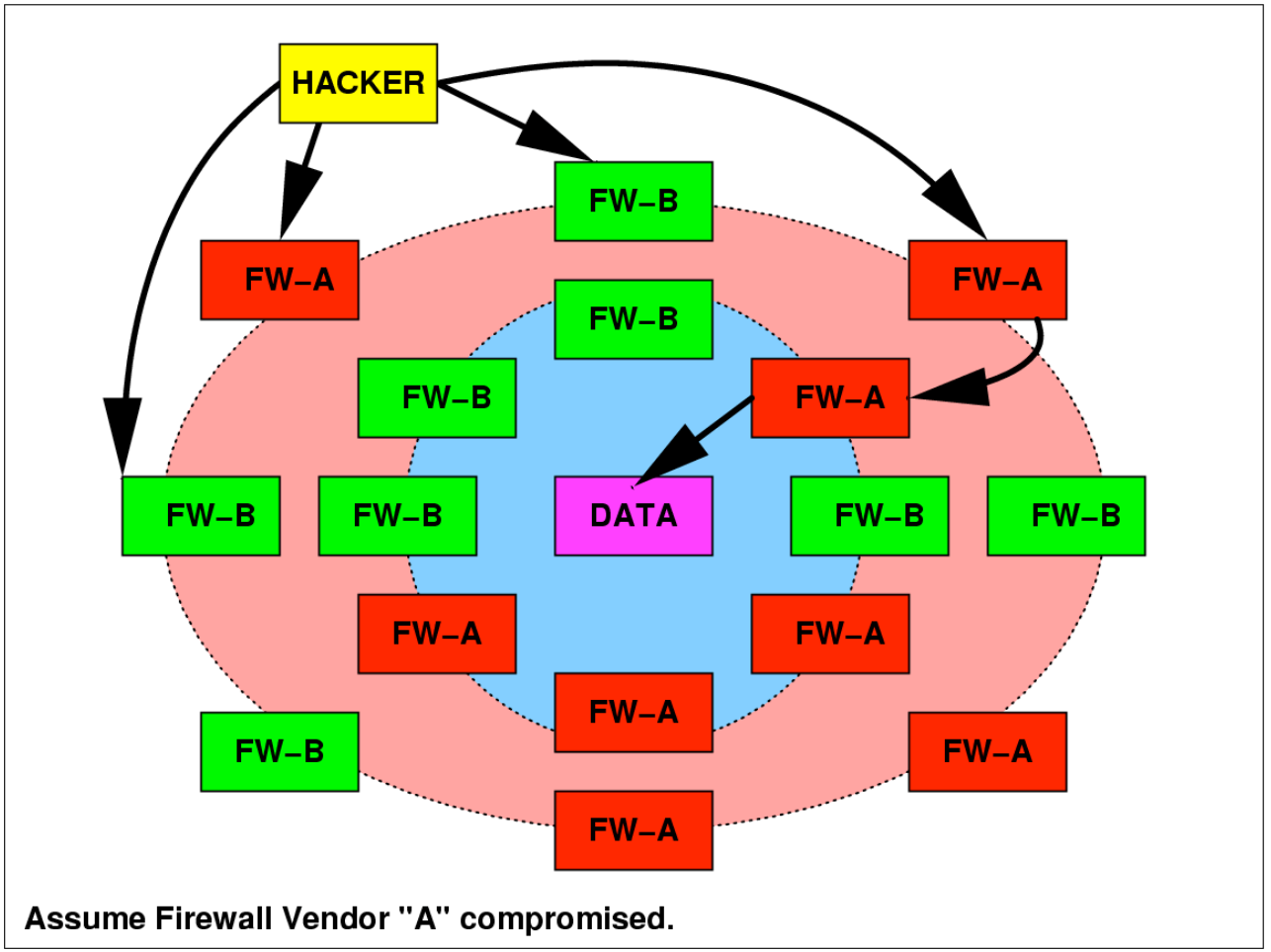
How robust is it?

- A very good approach to security...
  - although it is not a 100% solution...
  - the “ablative shield” approach yields better security than other “monolithic” solutions.
  - You will never get 100% security, anyway.
- But things can still be done wrongly...
  - For illustration...

If Implemented Well...

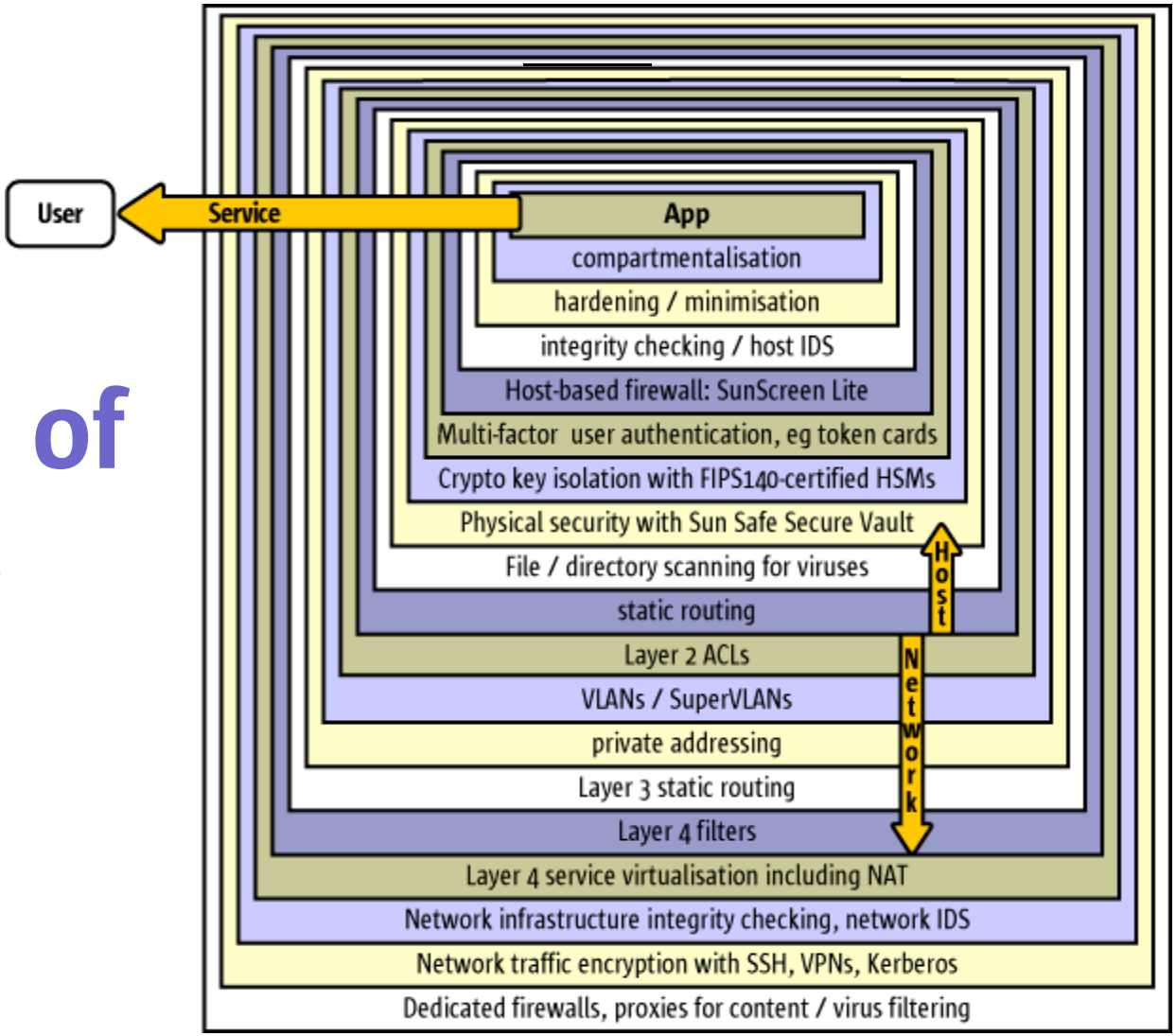


If Implemented Badly...



Assume Firewall Vendor "A" compromised.

Extremes of Available Choice



Summary

- Defence in Depth
  - Is a 7000-year-old approach to security that works really well
  - Avoids monolithic security issues and “monoculture syndrome”
  - Easy / inexpensive to build, but requires conscientious management and some forethought.
  - Investment in this methodology will last for a long time

Responsibility

- Security requires continual investment
  - Why audit, if you never read the logs?
  - Why have intrusion detection, if you don't want to wake up at 0300h?

    At best, these yield budget justification.
  - Why implement security, and yet fail to check its continued effectiveness over time?

    Healthchecks will yield “ROI” figures!
  - Why protect, if you do not value?

    If you do not value, do not protect!

Truisms

- “Security is not a product...
  - ...it is a process!”
  - ...or, personally speaking:

    “Security is not a process - it is a lifestyle!”

Alec.Muffett@Sun.COM

Sun microsystems
We make the net work.