A Protection Architecture for Enterprise Networks

Martin Casado *(Stanford)*
Tal Garfinkel *(Stanford)*
Aditya Akella *(CMU/Stanford)*
Michael Freedman *(NYU)*
Dan Boneh *(Stanford)*
Nick McKeown *(Stanford)*
Scott Shenker *(ICSI/Berkeley)*

Talk Outline

- Example: Breaking into an Enterprise network
- Problems with enterprise security today
- SANE: rethinking the network architecture

Enterprise Threat Environment

- Incidental attacks (phishing, spam, worms, viruses, kiddies)
- External, Targeted Attacks
  - Competitors (e.g. getloaded.com vs. truckstop.com)
  - Idealists (e.g. SCO)
- Insiders (29% of all attacks?)

Enterprise Threat Environment

- Incidental attacks (worms, viruses, kiddies)
- External Targeted Attacks
  - More access to resources
  - Ability to hire skilled attacker
- Insiders (29% of all attacks?)
  - Locality (access to internal network)
  - Knowledge of internal workings

Example: External Targeted Attack

- **Target:** Large company (Bank.com)
- **Attacker Profile:** Skill-level equivalent to a B.S. in computer science
- **Rules of Engagement:**
  - No physical access
  - Cannot limit availability of network resources
- **Goals:**
  - Map out operations
  - Gain access to sensitive information
  - Ability to disrupt internal communications if needed

Step 1: Reconnaissance

Netcraft search: bank (find all relevant domains)
Google/groups: @bank.com
“*at*bank*com”
“*bank*com”
“at*bank*”

```
frufru at media dot bank dot com
lilo [at sign] shingle [dot] bank [dot] com
laura@rapnet.something.bank.com
Dr. HeL Lo at <dhello@bank.com >
Gin ( dot ) H ( dot ) Polka ( at ) bank ( dot ) COM
Car Mc Kubrik · kubik AT NOSPAM bank dot com
Chris Finkledine at chrisfink@bank.com
David Spade at spadea@bank.com
Alicebob@bank.com
```

Step 1.5: Profiling

Google/groups: **“*Alicebob*”**
“*alicebob*bank*”

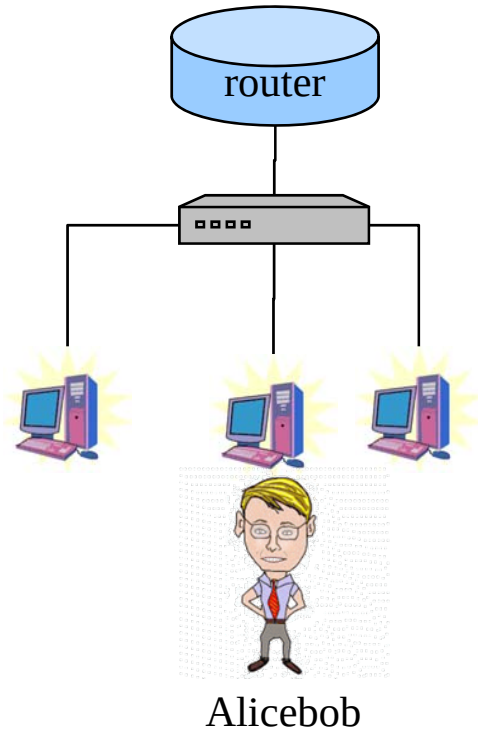
"someone please email me and tell me how to lose the weight? im trying the atkins but its sooo hard! catie how did you lose 67 lbs? what did you eat?? please email me at alicebob@bank.com and tell me ok??"

“You are truly blessed!!! I wish you a happy and healthy 8 more months. If you don't mind me asking....when was your tr? lengths? Is this your first pregnancy since your TR? I go for my TR on 10/24/03 so I am just trying to get lots of info together. Again Congratulations and I will lift you, dh and little one in prayer!!!”

etc ...

Step 2: Contact

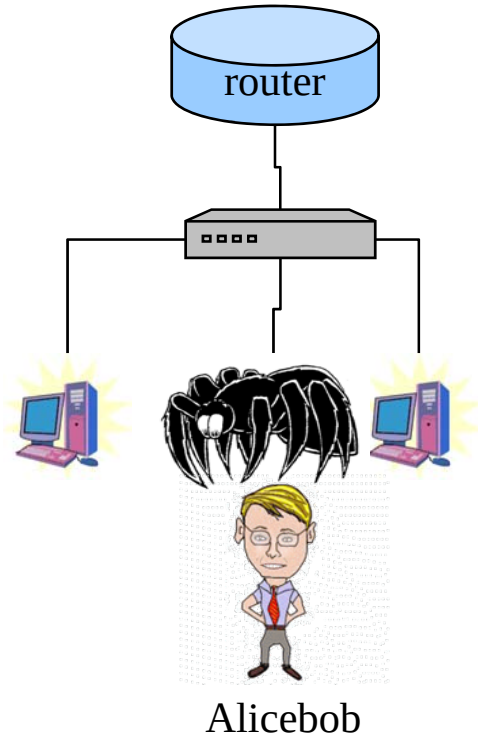
- Post to forum
- Establish rapport
- Get IM/email
- Write custom trojan
- Send infected file over IM, email, etc.



Step 3: Do Bad Stuff

- Gather local information
  - Local network parameters
  - Email addresses, documents etc.
- Gain access to traffic
  - Sniffing (switches)
  - Redirection (ARP, DHCP, DNS etc.)
- Further attack through binary injection
  - Redirect + proxy
  - Many vulnerable protocols

    (http, smtp, htp, nfs, SMB)
- Determine DoS attack channels



Properties of the Attack

- Does not require elite attacker
- Simple to launch by an insider
- Effective against traditional perimeter security models
  - Difficult to stop with signature detection
  - Weak internal protection allows propagation of attack once inside

IP vs. Security

- Overly permissive

  (e.g. broadcast on ARP request)
- Many heavily trusted components

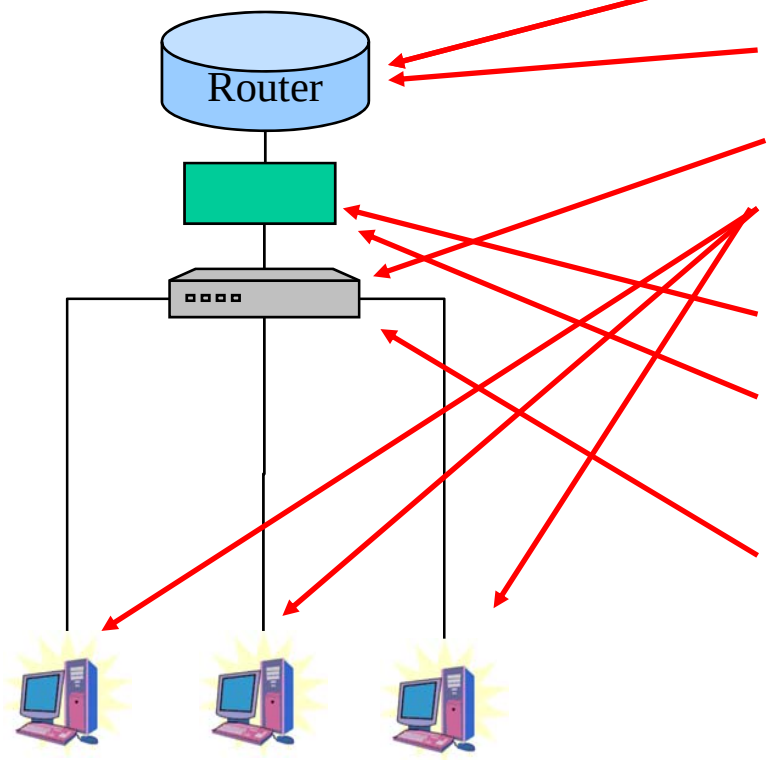
  (end-hosts, dhcp, dns, directory service, routers etc.)
- IP addresses are meaningless

  (can be forged, stolen, changed etc.) (NOTE: very weak notion of identity)
- No hiding of info

  (reconnaissance is easy)

No formal support for enforcing access controls within a network

Retrofitting Security onto IP

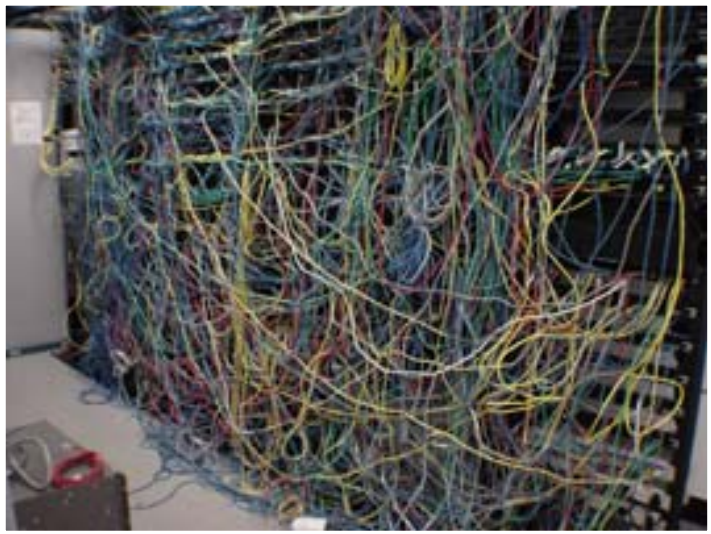


- ACLS or filtering rules at router
- Static ARP cache at router
- Port security on switch
- Static ARP cache on end-hosts
- Signature detection
- Proxies at choke points (full TCP termination)
- VLANs for isolation

Common Solutions = Crummy Networks (and not-great security)

- Inflexible
  - Hard to move a machine (yet difficult to know if someone has moved)
  - Really difficult to deploy a new protocol
- Brittle
  - Change a firewall rule, break security policy
  - Add a switch, break security policy
- Confusing
  - Many disparate point solutions
  - State = a bunch of soft state
  - Hard to state meaningful policies
- Lose redundancy
  - Introduce choke points
  - Can't migrate routes b/c of all the soft state

Strong coupling of topology and security policy



Argument Thus Far

- Targeted attacks can be quite effective
- IP not designed for attack resistance
  - permissive
  - Many trusted components
  - Unauthenticated end-points
  - No attempt to control access to information
- Attempts to retrofit access controls have resulted in less-than-ideal networks

Our Approach: Start from Scratch

- Secure by design
- Reduce trusted computing base
- Leverage characteristics unique to Enterprise
  - Centrally managed
  - Known users
  - Structured connectivity
- Simple policy declaration
- Retain flexibility and redundancy

  (decouple topology and security policy)

SANE

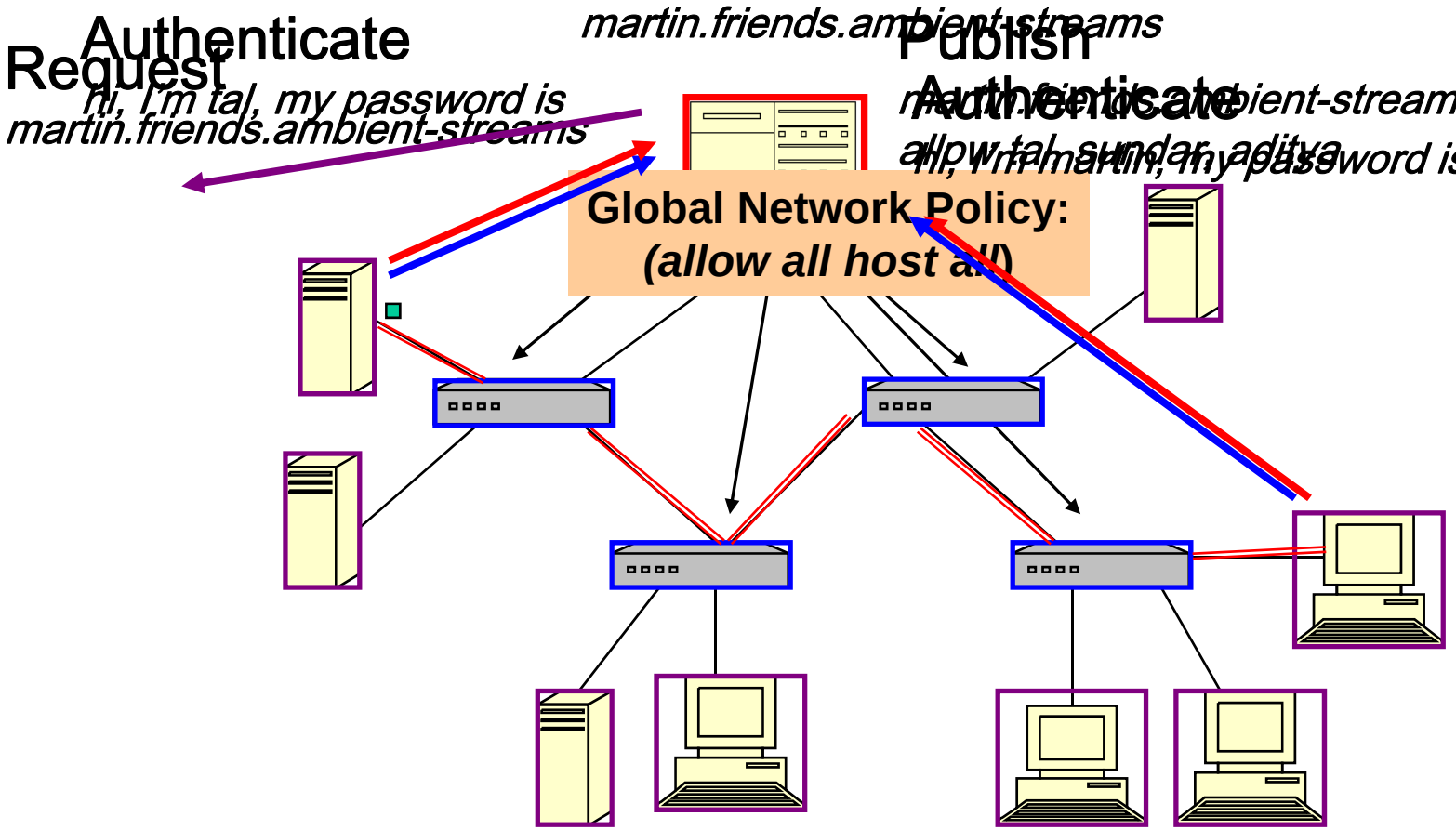
(Secure Architecture for the Networked Enterprise)

- Centrally declared policy defines all connectivity
- Policy declared over users, services, hosts

  (*e.g. Alice can access internal-web using http*)
- All communication requires permission (at the flow level)
- Users must authenticate before using network
- Network information is tightly controlled

SANE:

High-Level Operation

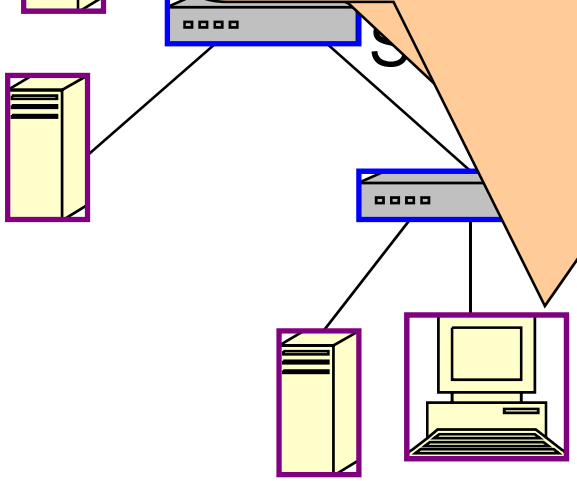


SANE:

nt Overview

•Send network topology information t
•Provide def
the DC
•Enforce pat
•Handle flow

•Publish services at the DC
•Specify access controls
(export streams.ambient allow tal)
•Request access to services

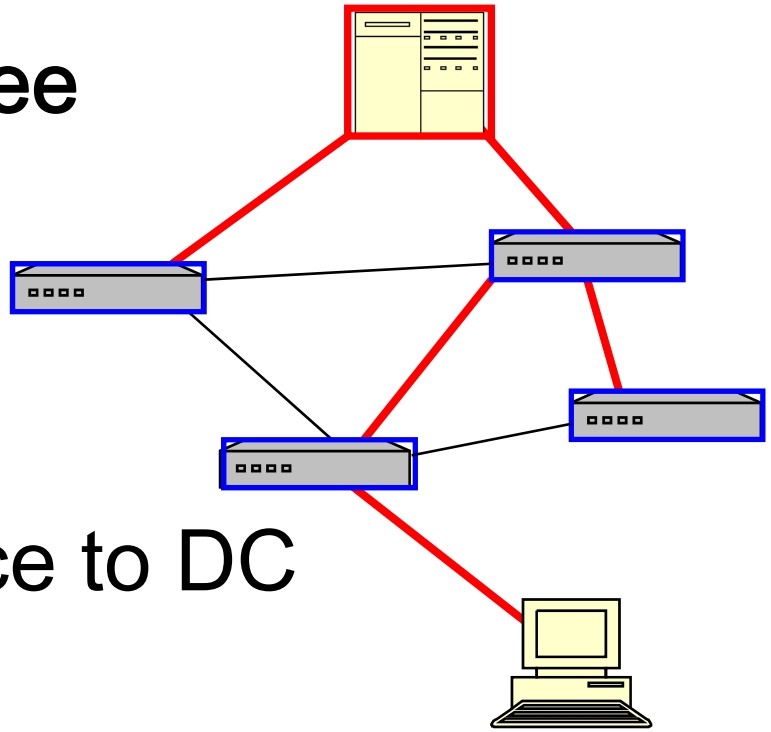


End-Hosts

•Authenticates switches/end-hosts
•Contains network topology
•Computes routes
•Handles permission checking for all flows

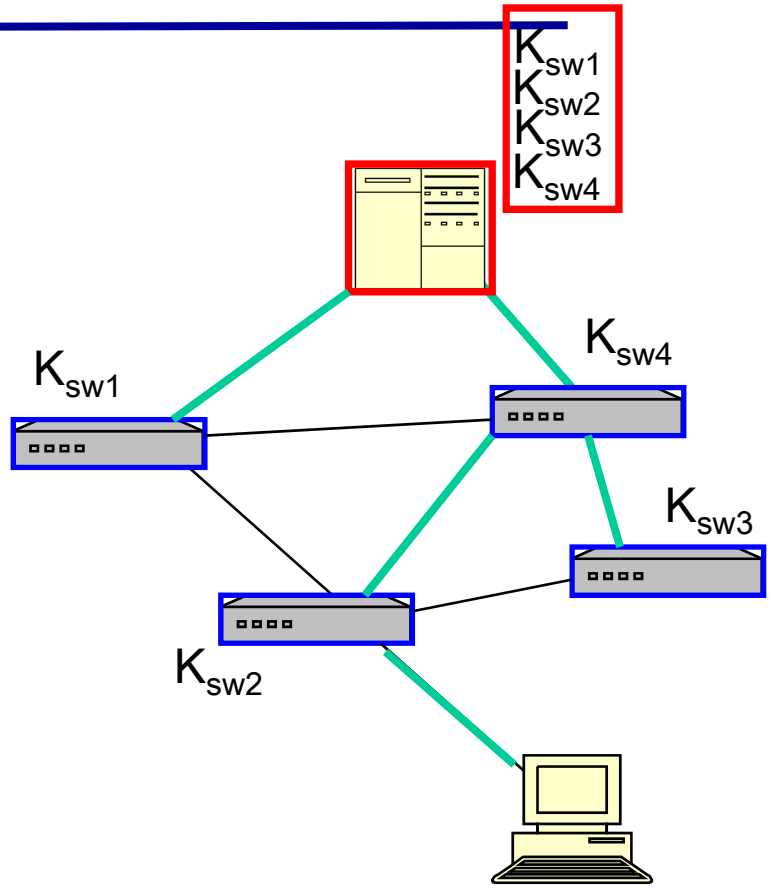
Connectivity to the DC

- Switches construct **spanning tree**
  Rooted at DC
- Switches don't learn topology
  (just neighbors)
- Provides basic datagram service to DC



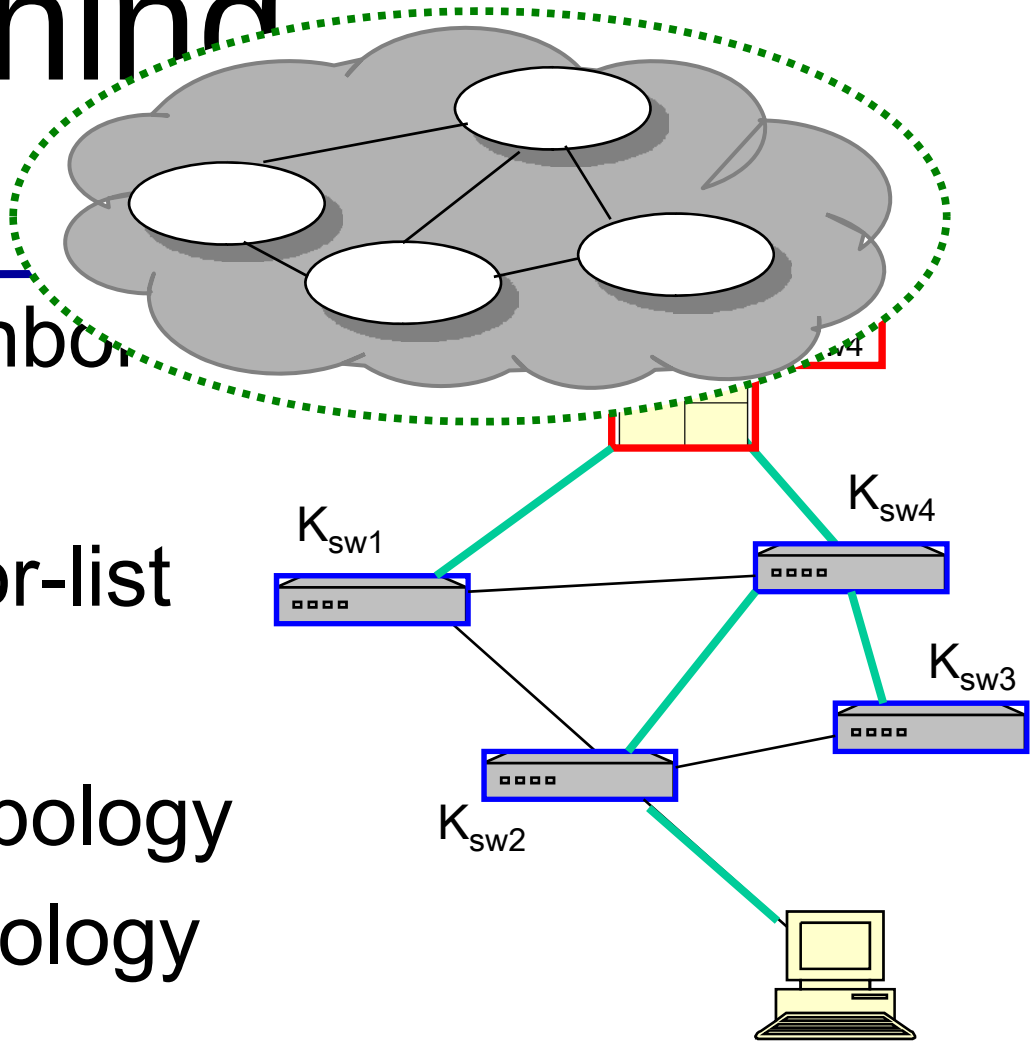
Establishing Shared Keys

- Switches authenticate with DC and establish symmetric key
- Ike2 for key establishment
- All subsequent packets to DC have “authentication header” (similar to ipsec *esp* header)



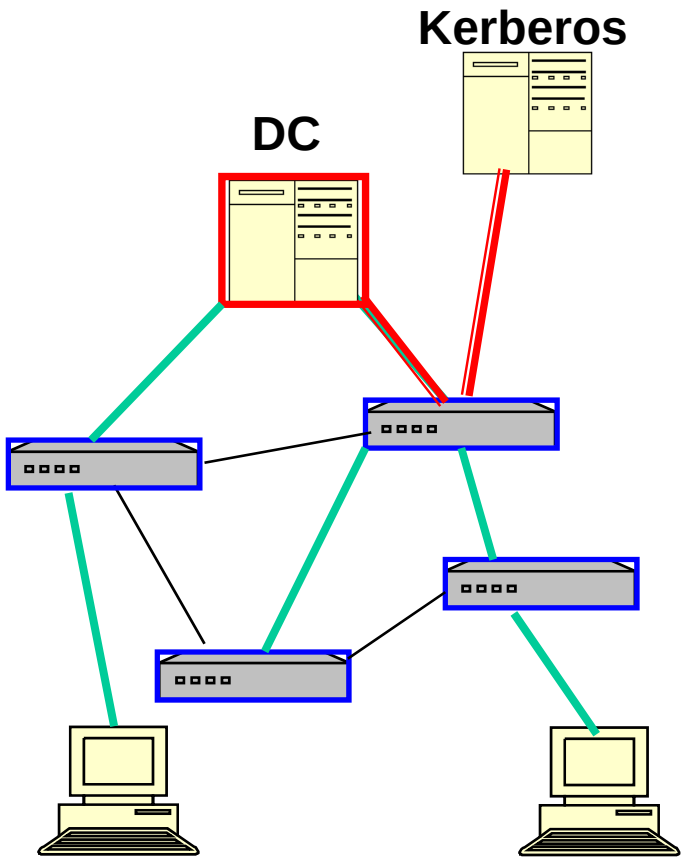
Establishing Topology

- Switches generate neighbor during MST algorithm
- Send encrypted neighbor-list to DC
- DC aggregates to full topology
- No switch knows full topology



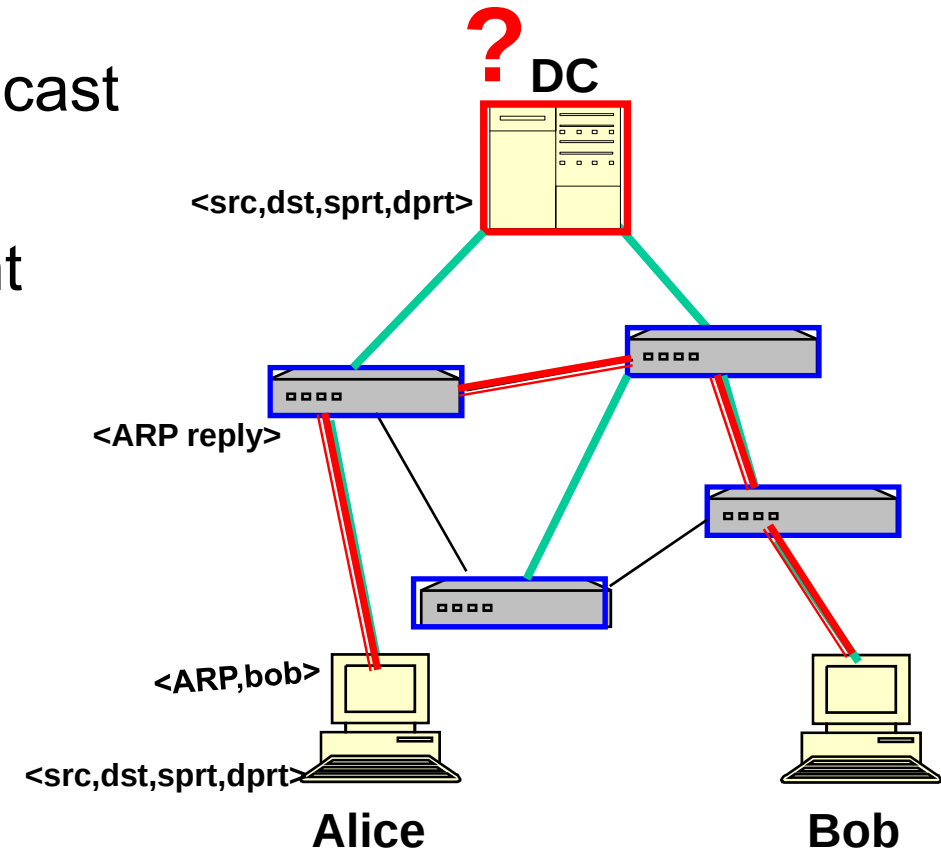
User Authentication

- DC creates route from itself to authentication server
- Use third-party mechanism for user authentication
  - Kerberos
  - Radius
  - AD
- DC places itself on-route for all authentication
- Snoops protocol to determine if authentication is successful
- Identifies user by location + network identifier (e.g. MAC address)



Connection Setup

- Switches disallow all Ethernet broadcast (and respond to ARP for all IPs)
- First packet of every new flow is sent to DC for permission check
- DC sets up flow at each switch
- Packets of established flows are forwarded using multi-layer switching



Security Properties (revisited)

- Permission check before connectivity
- Simple mechanism
- Users only access resources they have permission to
- Policy enforced at every switch
- Authenticated end hosts (bound to location)
- High level policy declaration
  (topology independent)
- Control information regarding
  packet path, topology

Other Nice Properties

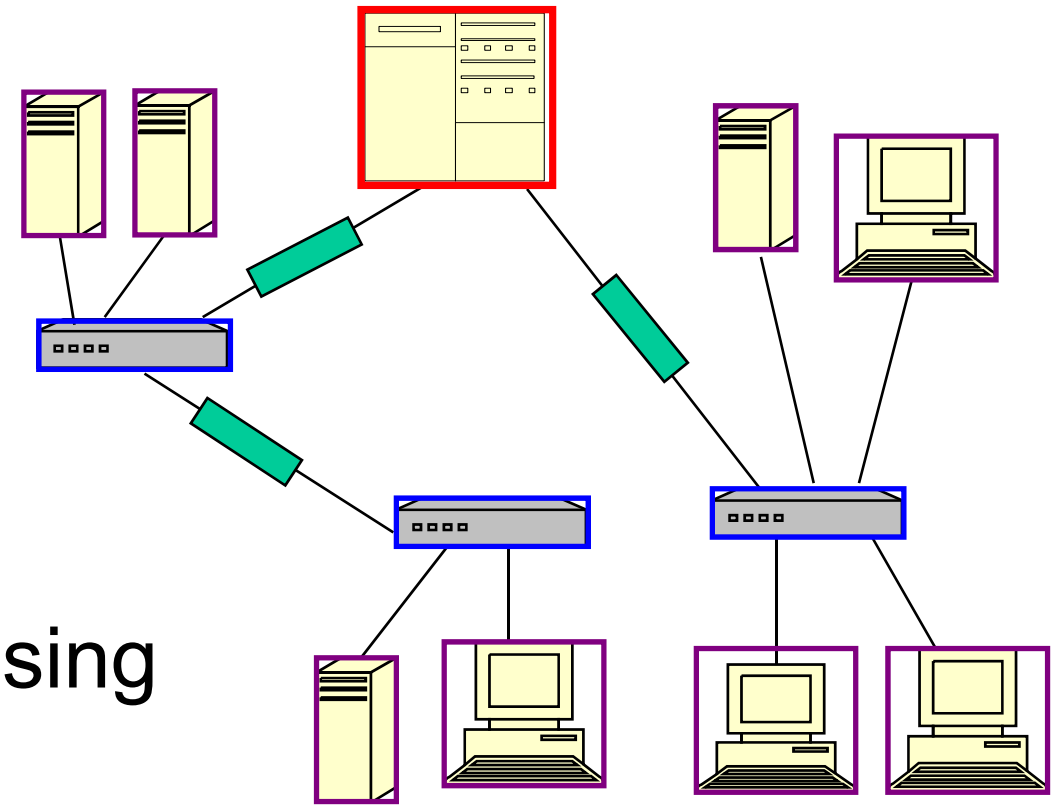
- Central point for connection logging (DC)
- Addition of switches (redundancy) does not undermine security policy
- Application-informed routing
- Anti-mobility

Extensions and Considerations

- Backwards compatibility
- Middlebox integration
- Performance
- Fault Tolerance
  - Managing the DC as a single point of failure
  - Adaptive routing

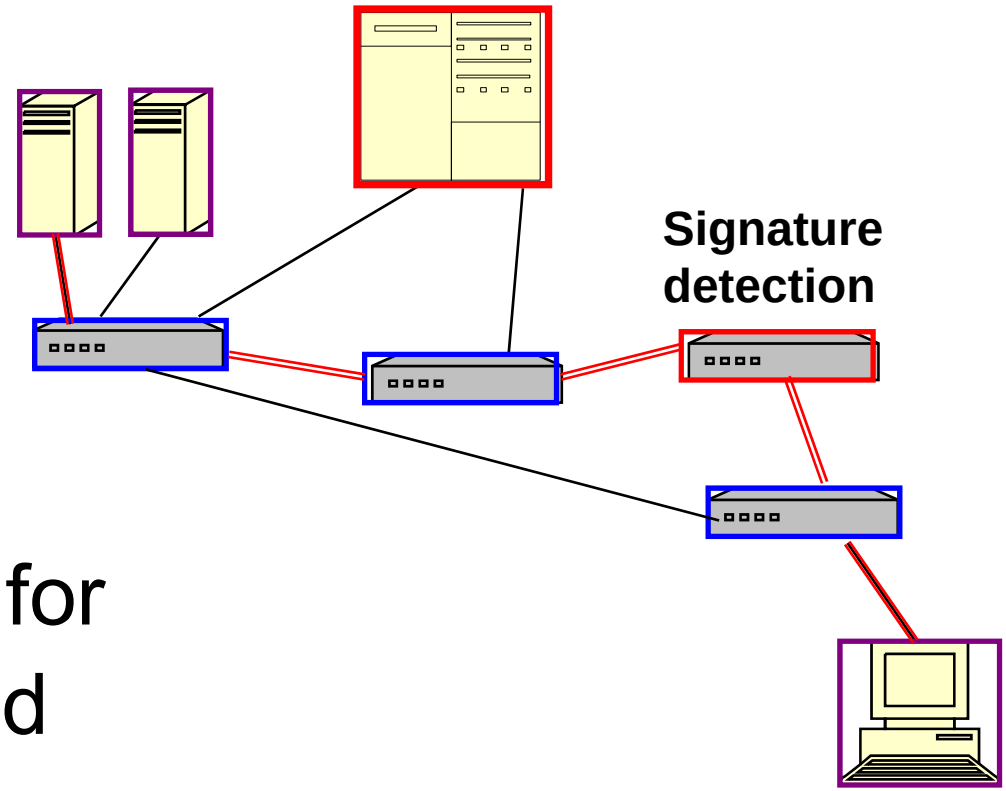
Easing Deployment

- Use trivial 2-port switches (bumps)
- On links between Ethernet switches
- Can be enhanced by using VLAN per port



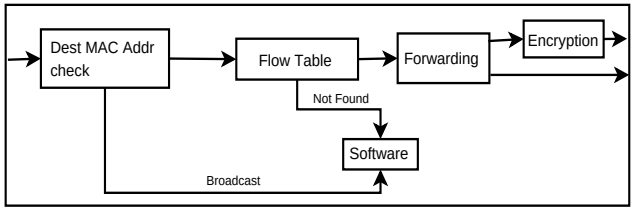
Middle Box Integration

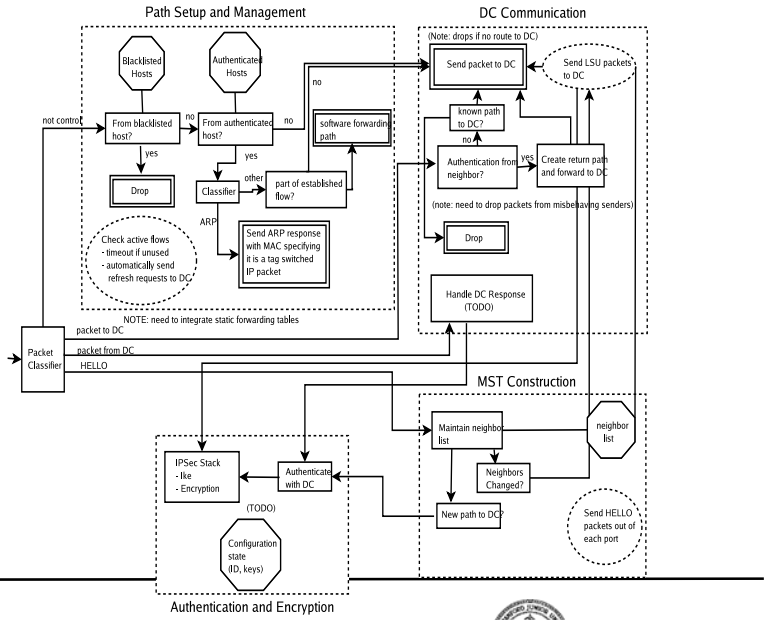
- Control of routes is powerful
- DC can force routes through middlebox based on policy
- E.g. signature detection for all flows from laptops and users in marketing



Performance

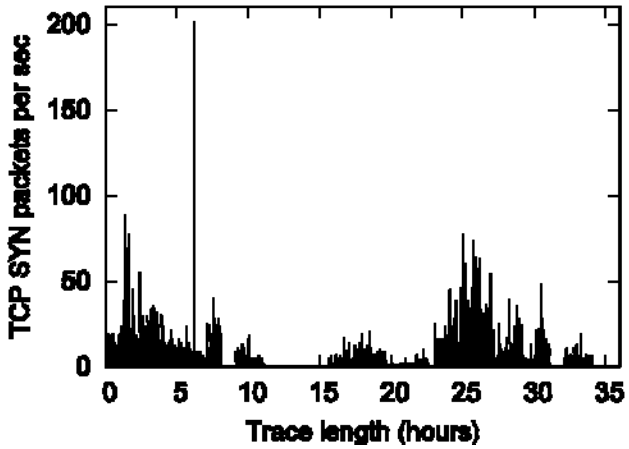
- Decouple control and data path in switches
- Software control path (connection setup) (slightly higher latency)
- Simple, fast, hardware forwarding path (Gigabits)

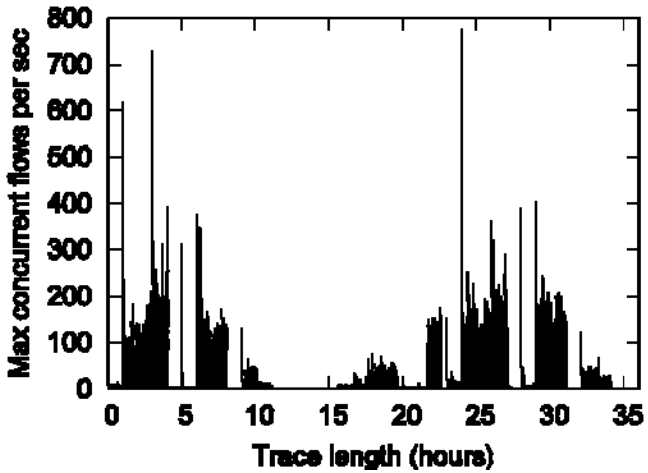




DC: Single Point of Failure?

- Exists today (DNS)
- Permission check is fast
- Replicate DC
  - Computationally (multiple servers)
  - Topologically (multiple servers in multiple places)





Status

- Built software version of similar system (using capabilities)
  - All components in software
  - Ran in group network (7 hosts) 1 month
- Currently in development of full system
  - Switches in hardware + software
  - DC using standard PC

Questions?