Providing Security in Routing Infrastructures

Charlie Kaufman

Security Architect, Windows Azure

Microsoft

Network Security

- Traditional View: Security is an endpoint problem. Endpoints should use cryptography such that their security does not depend on the network

“Every packet will be delivered zero or more times… now there is a guarantee you can count on.”

Network Security Challenges

- End to End challenges that can be offloaded to a network
  - Firewalls
  - Intrusion detection
  - Crypto Acceleration
- Things a network might be able to do better
  - Assurance of identity of the other end of a connection
- Can only be solved in the network
  - The network has to actually deliver packets to the right places some of the time
  - QoS / DoS by bandwidth exhaustion

Network Security Challenges

- Within a datacenter
  - Network can make guarantees based on common administration
  - May be able to assume few attackers
- On the Internet
  - Whole sections can be hostile
  - Must assume attackers are ubiquitous

Difficult Decisions with High Performance Networking

- Offloading security features into the network has costs (in either performance or price)
- What checks are best done in the network?

Typical Configurations

- Corporate Intranet
  - User workstations
  - Shared (scattered) Servers / Printers
  - Gateway (outbound) to Internet
- Internet Server
  - Centrally managed data center
  - Gateway (inbound) from Internet
  - Gateway (inbound) from administrative workstations
- Cloud Service Provider
  - Centrally managed data center hosting untrusted services
  - Gateway (inbound and outbound) to Internet

Where are the Bad Guys

- Corporate Intranet
  - Out on the Internet
  - On compromised employee workstations
- Internet Server
  - Out on the Internet
  - Possibly on compromised servers
- Cloud Service Provider
  - Out on the Internet
  - In the hosted services

(And then of course, it could always be an insider)
(And once they get it, their code could be anywhere)

Agenda

- **Cloud Data Center Networks**
- Quick Introduction to Cryptography
- Quick Introduction to Public Key Infrastructures
- What security features do people want from a datacenter network hub (and why)
- Security in Large Networks (no central administration)
- Is security against Byzantine failures possible?
- Lessons learned in Cloud Security

Cloud Data Center Networks

- What’s different about them?
- What are the design criteria?

  (What do cloud builders really want?)
- What do they look like today?

What's Different About Cloud Data Center Networks

- Network load patterns are highly variable and unpredictable
  - Need high bandwidth and low latency from anywhere to anywhere (no hardware changes to adapt to application)
- Servers usually run Virtual Machines with a trusted Host that can act as intelligent network adaptor
- Scale tends to be large

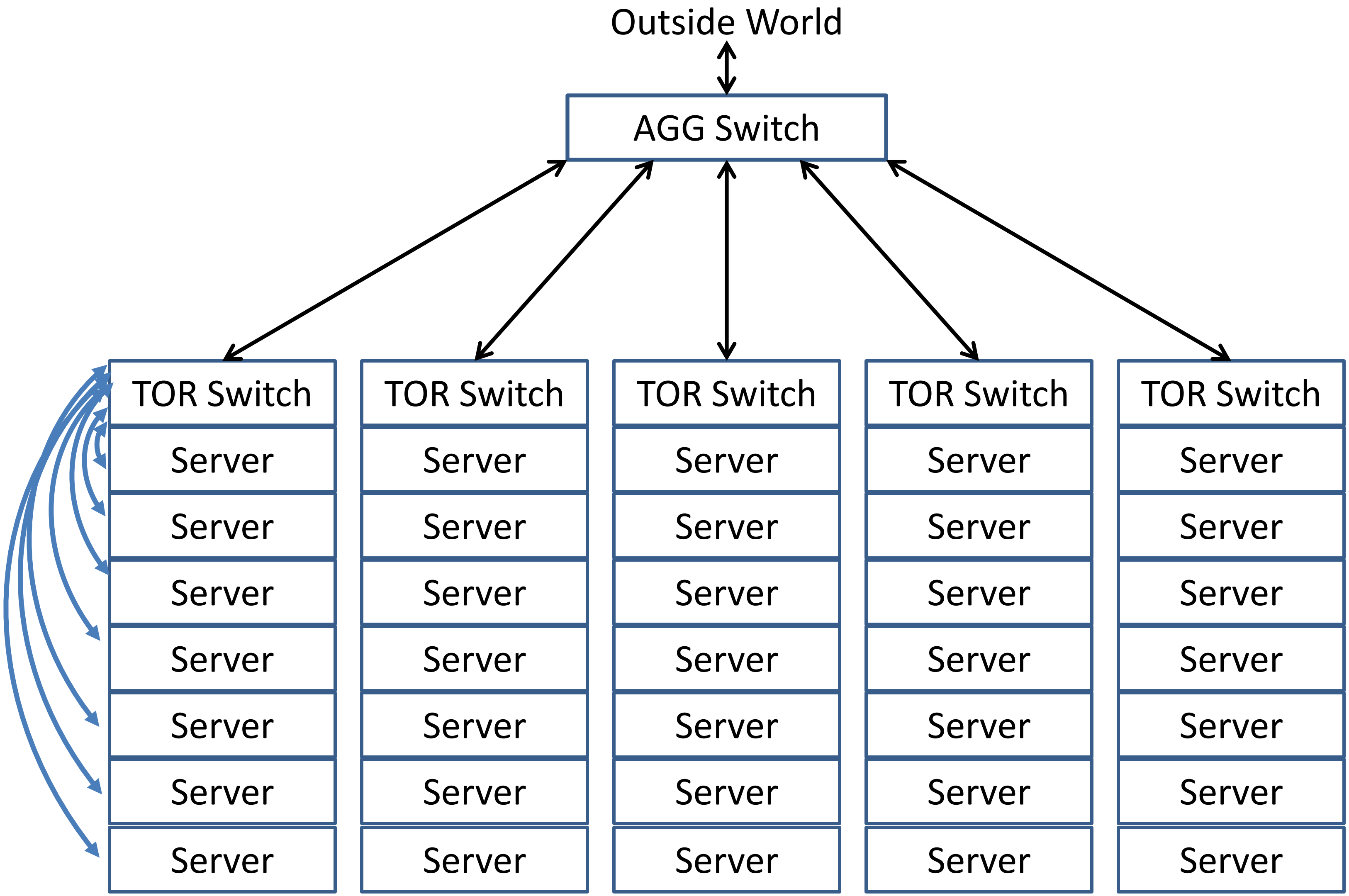
What are the Design Criteria?

What do cloud designers really want?

- High bandwidth
- Low latency
- High Availability
- Low cost
- Off the shelf – proven – technology

This usually means commodity products even when something custom might make more sense technically

Consider This Design



Outside World

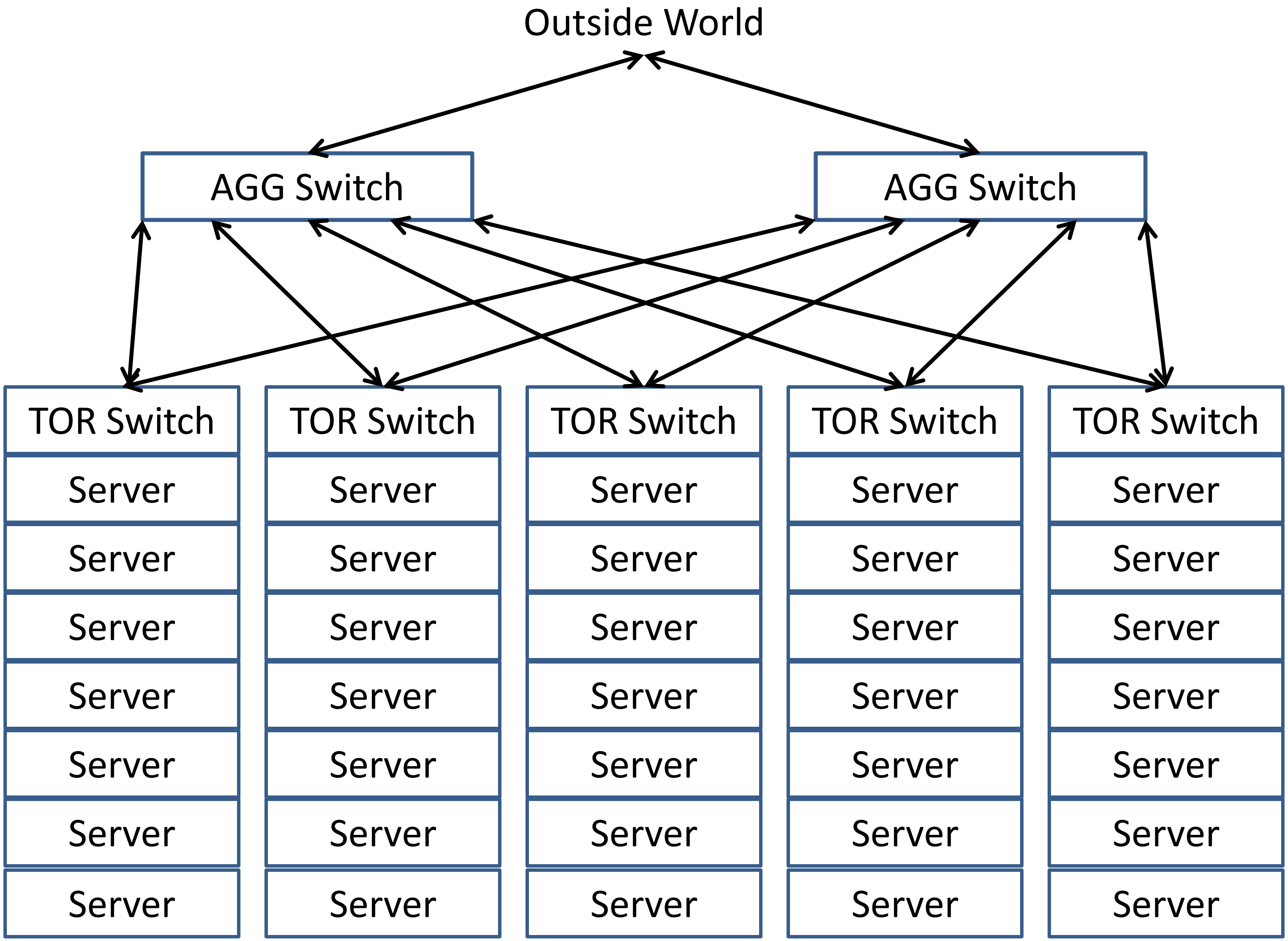
AGG Switch

TOR Switch	TOR Switch	TOR Switch	TOR Switch	TOR Switch
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server

Performance Properties

- Within a rack, bandwidth & latency optimal
- Between racks, bandwidth limited
- Single point of failure

Avoiding Single Point of Failure



Outside World

AGG Switch AGG Switch

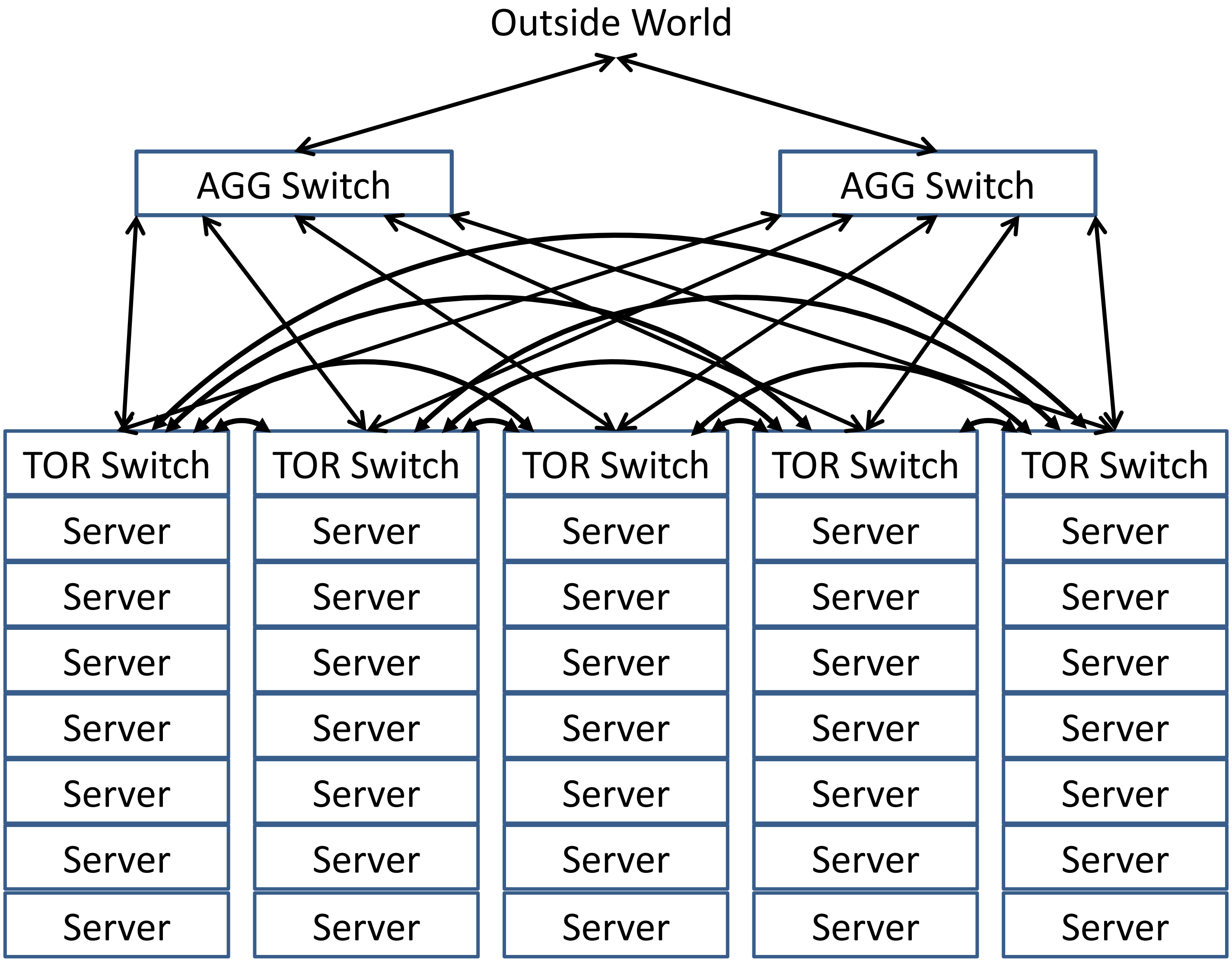
TOR Switch TOR Switch TOR Switch TOR Switch TOR Switch

TOR Switch	TOR Switch	TOR Switch	TOR Switch	TOR Switch
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server

Performance Properties

- Between racks, bandwidth still limited
- Using faster links between racks problematic
  - At any given time, not a big speed range for commodity links
  - Hops at different speeds hurt latency because of cut-through
- With L2 switches, spanning tree can’t use half the bandwidth (without tweaking)
  - Commodity L3 switches available, so now use that
- Could put two TORs per rack to avoid that failure mode, but racks usually have other points of failure (e.g. power) so systems have to be designed to deal with their failure anyway

You might think this would help…



Outside World

AGG Switch

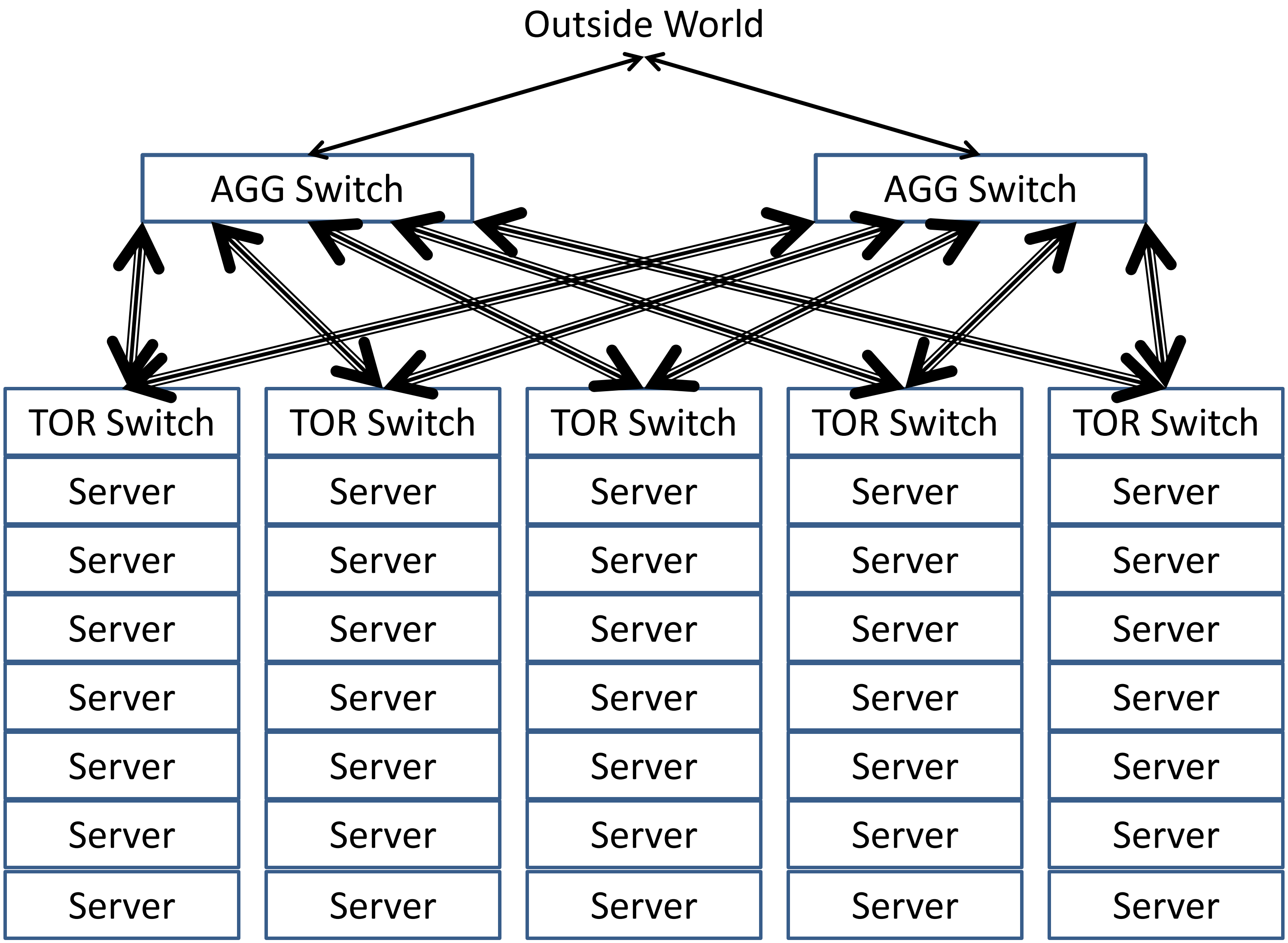
AGG Switch

TOR Switch	TOR Switch	TOR Switch	TOR Switch	TOR Switch
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server

Performance Properties

- If communications load was uniform, this would work well
- If communications load was light, it would improve latency (one less hop)
- But… shortest path routing means only one link’s worth of bandwidth between two racks

It turns out, this helps more



Outside World

AGG Switch

AGG Switch

TOR Switch	TOR Switch	TOR Switch	TOR Switch	TOR Switch
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server
Server	Server	Server	Server	Server

Performance Properties

- Many paths have equal costs
- Commodity switches do load balancing over equal cost paths
- If as many uplinks from TORs as blades below, network above TORs is never a bottleneck
- Other strategies would be more cost effective if routing could adapt based on load, but commodity switches can’t

Agenda

- Cloud Data Center Networks
- **Quick Introduction to Cryptography**
- Quick Introduction to Public Key Infrastructures
- What security features do people want from a datacenter network hub (and why)
- Security in Large Networks (no central administration)
- Is security against Byzantine failures possible?
- Lessons learned in Cloud Security

Quick Introduction to Cryptography

- Five kinds of cryptographic functions:
  - Message Digest
  - Secret Key Encryption
  - Secret Key Message Integrity Code
  - Public Key Encryption
  - Public Key Signatures

Message Digest

- Function that computes a fixed size checksum of arbitrary length data
- Make it a convoluted function like squaring the blocks and taking the middle bits to make it hard to reverse
- Cryptographic requirements
  - Given a checksum, can’t find a string with that checksum (except by guessing)
  - Can’t find two strings with the same checksum

Secret Key Encryption

- Transform a message into ciphertext using a key such that only someone knowing the key can compute the original message
- Think multiplication and division modulo a big number
- Cryptographic requirements:
  - Given message and ciphertext, can’t figure out the key
  - Without the key, ciphertext gives no clues about message (except its length)

Secret Key Message Integrity Code

- Think of as a keyed checksum
- Given a message and a key, I can compute the checksum
- Seeing lots of messages and checksums, I can’t figure out the key or the checksum on any other messages
- Someone with the key can confirm that the message is genuine

Public Key Encryption

- Suppose division had not been invented, but we knew about multiplication and inverses
- To encrypt a message, multiply by K (e.g. 5)
- To decrypt a message, multiply by K^{-1} (e.g. 1/5)
- Suppose we don’t know how to compute inverses
- My public key is 5; My private key is 1/5
- I tell everyone my public key, and they can send me encrypted messages, but only I can decrypt them

But wait…

- If we can’t compute inverses, how did we come up with 5 and 1/5?
- I’ll explain how RSA actually works…

RSA Public Key Encryption

- Instead of multiplication, we’ll use exponentiation modulo a large number
- Ciphertext = Plaintext^{e} mod n
- It turns out e has an “exponentitive inverse” d such that:
- Plaintext = Ciphertext^{d} mod n
- And it’s easy to compute d from e if you can factor n

RSA Public Key Encryption

- Factoring large numbers (say 600 digits) is hard
- Figuring out whether a large number is prime is relatively easy
- Pick 300 digit random numbers and test them for primality until you find two primes
- Multiply them together and use that for your ‘n’
- You can factor ‘n’ (because you constructed it), but no one else can
- You can pick e, compute $d = e^{-1}$, tell everyone e & n, and never tell anyone d

RSA Public Key Signatures

- Compute Sig = hash(msg)^d mod n
- Anyone knowing your public key can compute: Sig^e mod n
- If that matches hash(msg) they know you signed it

Key Management

- The hard part of cryptography is getting keys to the right people. The rest is just math.
- Public Key cryptography sounds easy, since you can tell everyone your public key. So just publish it in a directory or something.
- But… that would give the directory too much power. It could lie about people’s public keys and impersonate them

Agenda

- Cloud Data Center Networks
- Quick Introduction to Cryptography
- **Quick Introduction to Public Key Infrastructures**
- What security features do people want from a datacenter network hub (and why)
- Security in Large Networks (no central administration)
- Is security against Byzantine failures possible?
- Lessons learned in Cloud Security

Public Key Infrastructure

"Any problem in computer science can be solved with one more level of indirection."

--David Wheeler

- A *certificate* is a signed statement concerning someone's name and public key
- If you trust the signer and know his public key, you can believe that the named entity has the contained public key
- But do you trust the signer? Who should be trusted to sign certificates?

Certificate Chains

- If you know some small set of public keys of trusted people, and they sign public keys for more trusted people, you can construct certificate chains leading to the person whose key you want to know.

Strategies for CA Hierarchies

- Monopoly
- Oligarchy
- Anarchy
- Bottom-up

Monopoly

- Choose one universally trusted organization
- Embed their public key in everything
- Give them universal monopoly to issue certificates
- Make everyone get certificates from them
- Simple to understand and implement

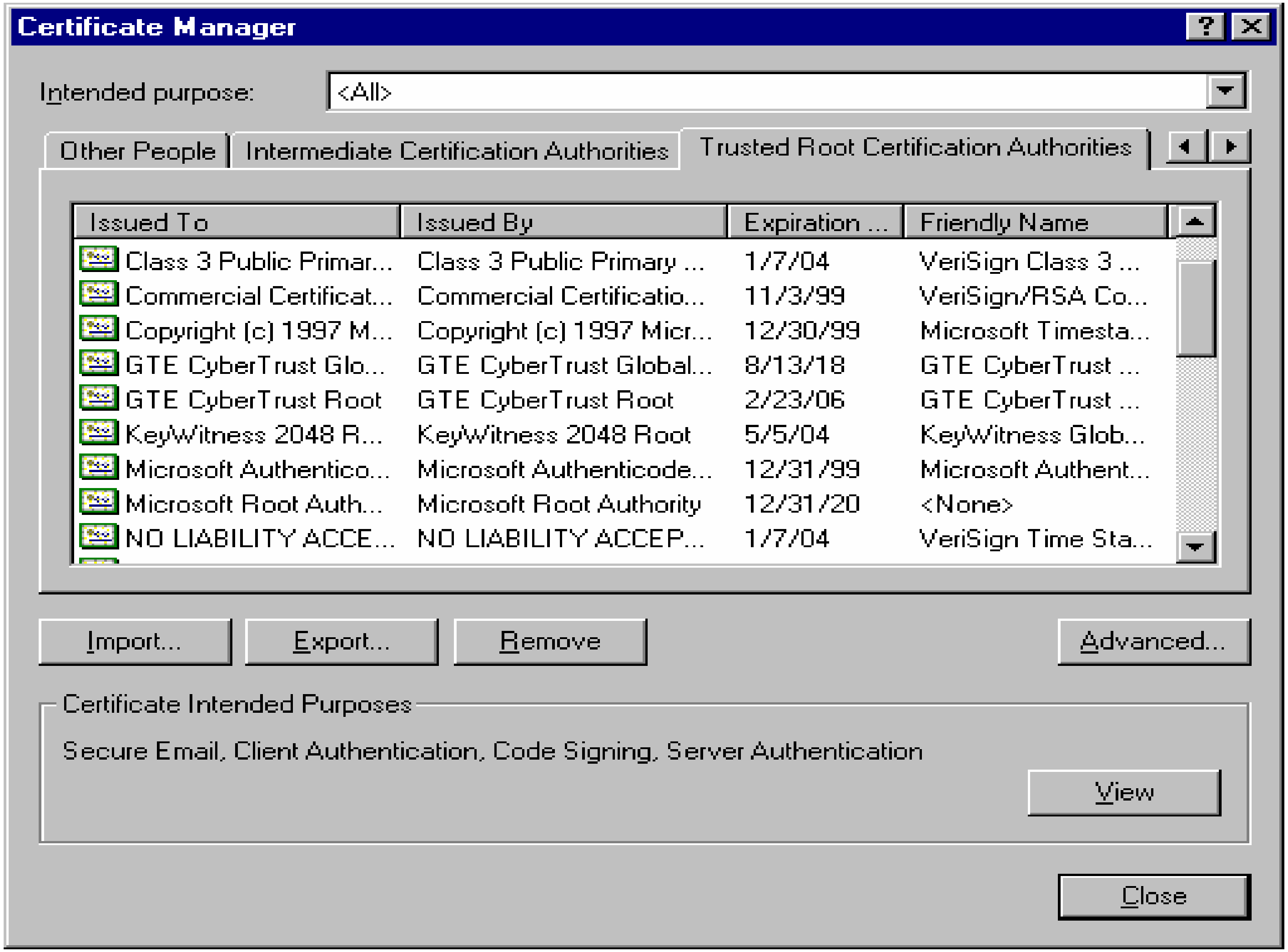
What’s wrong with this model?

- Monopoly pricing
- Getting certificate from remote organization will be insecure or expensive (or both)
- That key can never be changed
- Security of the world depends on honesty and competence of that one organization, forever

Oligarchy of CAs

- Come configured with 80 or so trusted CA public keys (in form of “self-signed” certificates!)
- Usually, can add or delete from that set
- Eliminates monopoly pricing

Default Trusted Roots in IE



Certificate Manager

Intended purpose: <All>

Other People
Intermediate Certification Authorities
Trusted Root Certification Authorities

Issued To	Issued By	Expiration ...	Friendly Name
Class 3 Public Primar...	Class 3 Public Primary ...	1/7/04	VeriSign Class 3 ...
Commercial Certificat...	Commercial Certificatio...	11/3/99	VeriSign/RSA Co...
Copyright (c) 1997 M...	Copyright (c) 1997 Micr...	12/30/99	Microsoft Timesta...
GTE CyberTrust Glo...	GTE CyberTrust Global...	8/13/18	GTE CyberTrust ...
GTE CyberTrust Root	GTE CyberTrust Root	2/23/06	GTE CyberTrust ...
KeyWitness 2048 R...	KeyWitness 2048 Root	5/5/04	KeyWitness Glob...
Microsoft Authentico...	Microsoft Authenticode...	12/31/99	Microsoft Authent...
Microsoft Root Auth...	Microsoft Root Authority	12/31/20	<None>
NO LIABILITY ACCE...	NO LIABILITY ACCEP...	1/7/04	VeriSign Time Sta...

Certificate Intended Purposes

Secure Email, Client Authentication, Code Signing, Server Authentication

What’s wrong with oligarchy?

- Less secure!
  - security depends on ALL configured keys
  - naïve users can be tricked into using platform with bogus keys, or adding bogus ones (easier to do this than install malicious software)
  - impractical for anyone to check trust anchors
- Although not monopoly, still favor certain organizations. Why should these be trusted?

Anarchy

- Anyone signs certificate for anyone else
- Like configured+delegated, but user consciously configures starting keys
- Problems
  - won’t scale (too many certs, computationally too difficult to find path)
  - no practical way to tell if path should be trusted
  - too much work and too many decisions for user

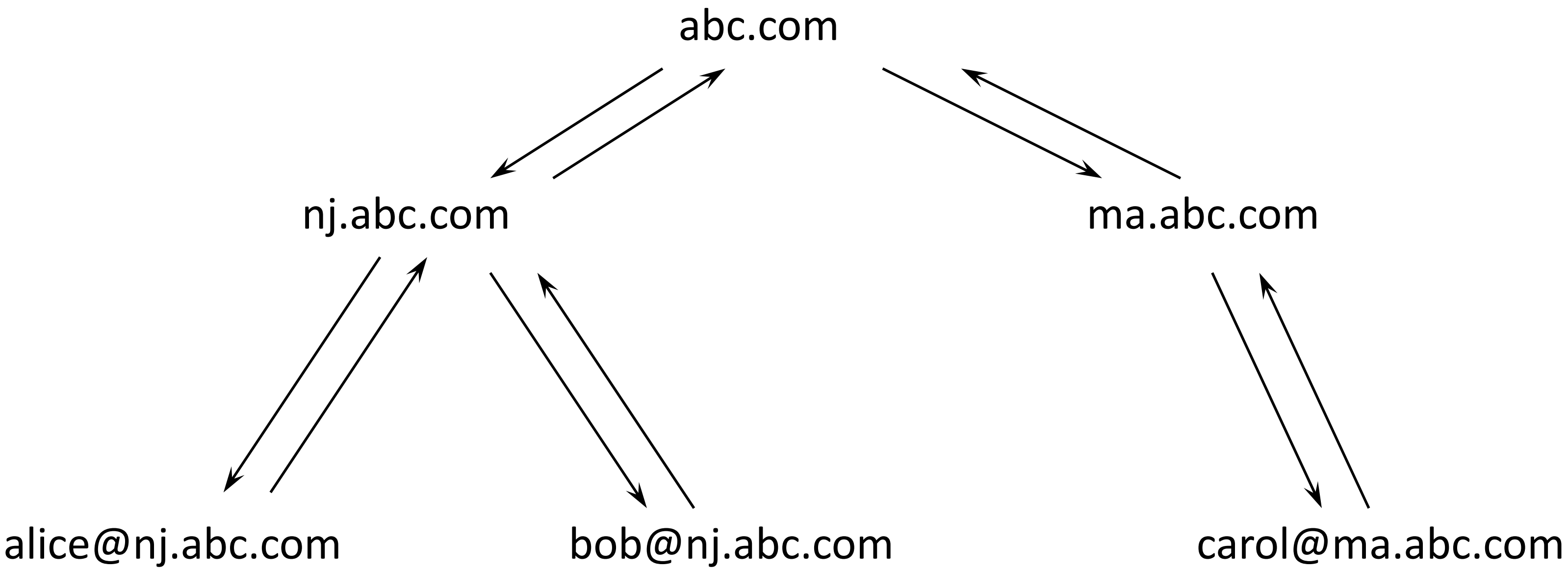
Top Down with Name Subordination

- Assumes hierarchical names
- Each CA only trusted for the part of the namespace rooted at its name
- Can apply to delegated CAs or RAs
- Easier to find appropriate chain
- More secure in practice (this is a sensible policy that users don’t have to think about)

Bottom-Up Model

- Each arc in name tree has parent certificate (up) and child certificate (down)
- Name space has CA for each node
- “Name Subordination” means CA trusted only for a portion of the namespace
- Cross Links to connect Intranets, or to increase security
- Start with your public key, navigate up, cross, and down

Intranet



abc.com

nj.abc.com

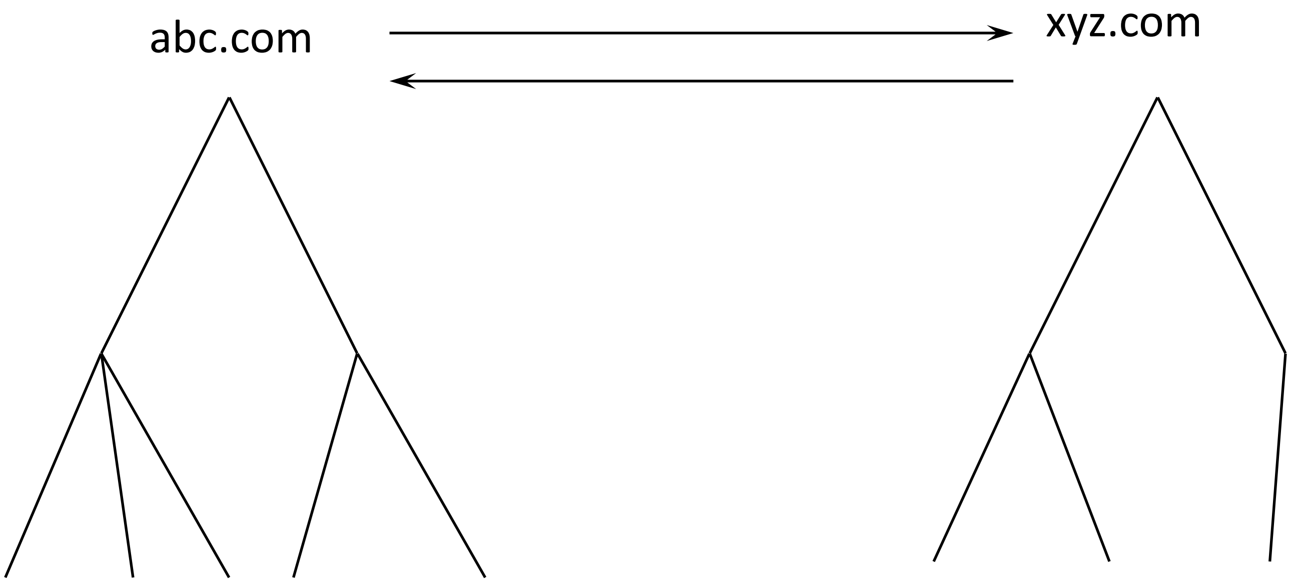
ma.abc.com

alice@nj.abc.com

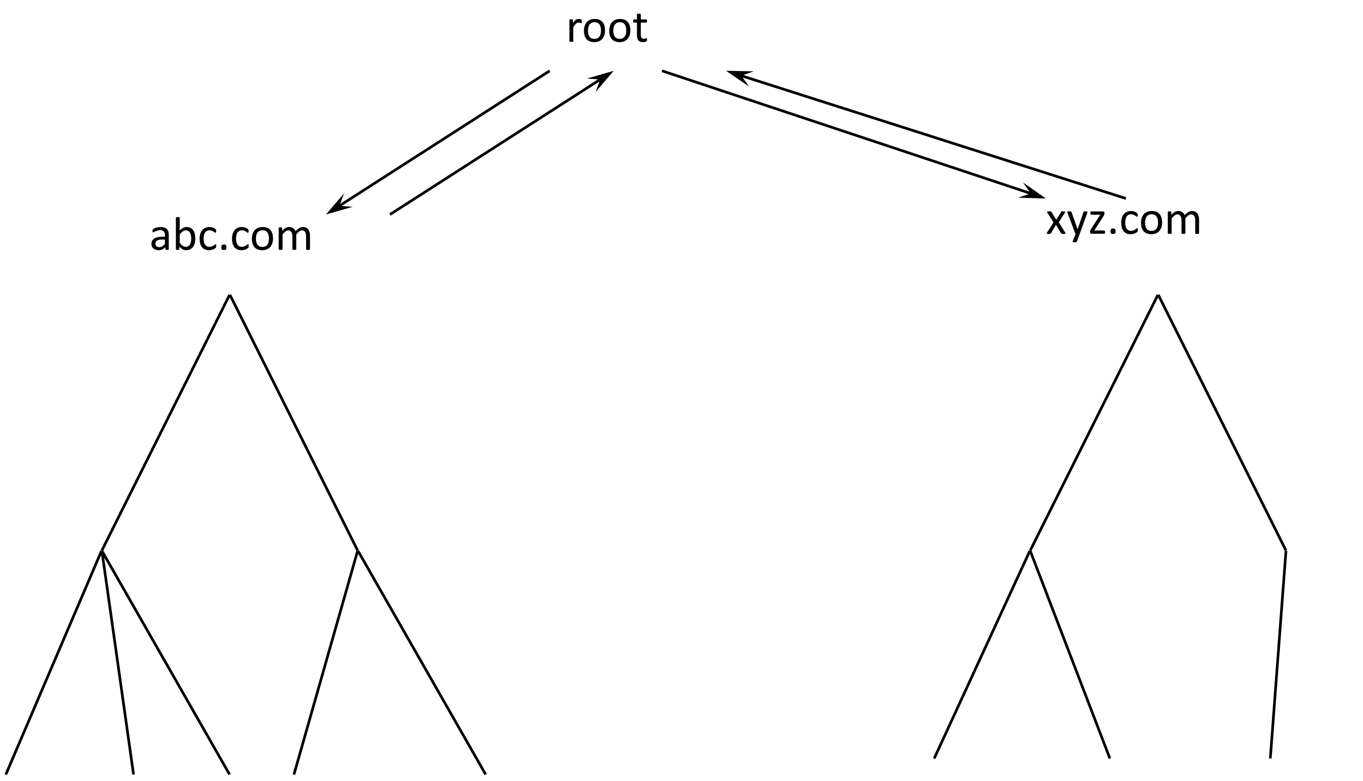
bob@nj.abc.com

carol@ma.abc.com

Extranets: Crosslinks



Extranets: Adding Roots



Advantages of Bottom-Up

- For intranet, no need for outside organization
- Security within your organization is controlled by your organization
- No single compromised key requires massive reconfiguration
- Easy configuration: public key you start with is your own

Bridge CA Model

- Similar to bottom-up, in that each organization controls its destiny, but top-down within organization
- Trust anchor is the root CA for your org
- Your org's root points to the bridge CA, which points to other orgs' roots

Agenda

- Cloud Data Center Networks
- Quick Introduction to Cryptography
- Quick Introduction to Public Key Infrastructures
- **What security features do people want from a datacenter network hub (and why)**
- Security in Large Networks (no central administration)
- Is security against Byzantine failures possible?
- Lessons learned in Cloud Security

What security features do people want from a datacenter network hub (and why)

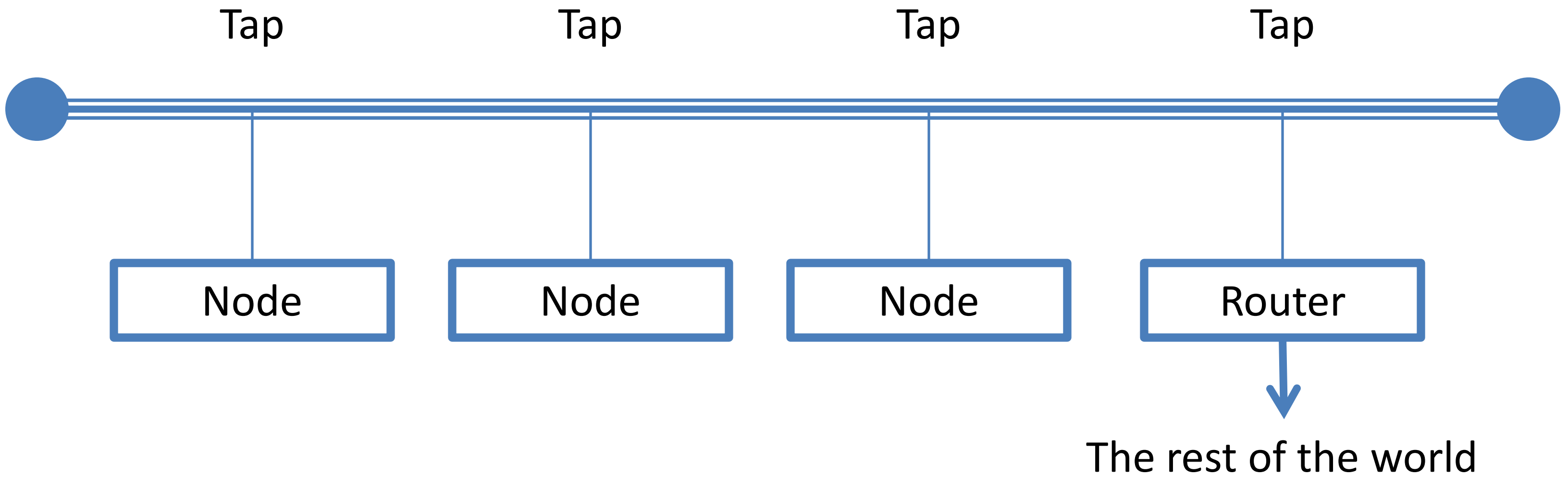
- Preventing IP address spoofing in a datacenter
- Address/Port filtering
- NATP
- Load Balancing
- HTTP proxy/cache
- Crypto Offload / Tunnel Endpoints
- DoS/DDoS diagnosis, blocking
- QoS / Bandwidth reservation

IP Address Spoofing

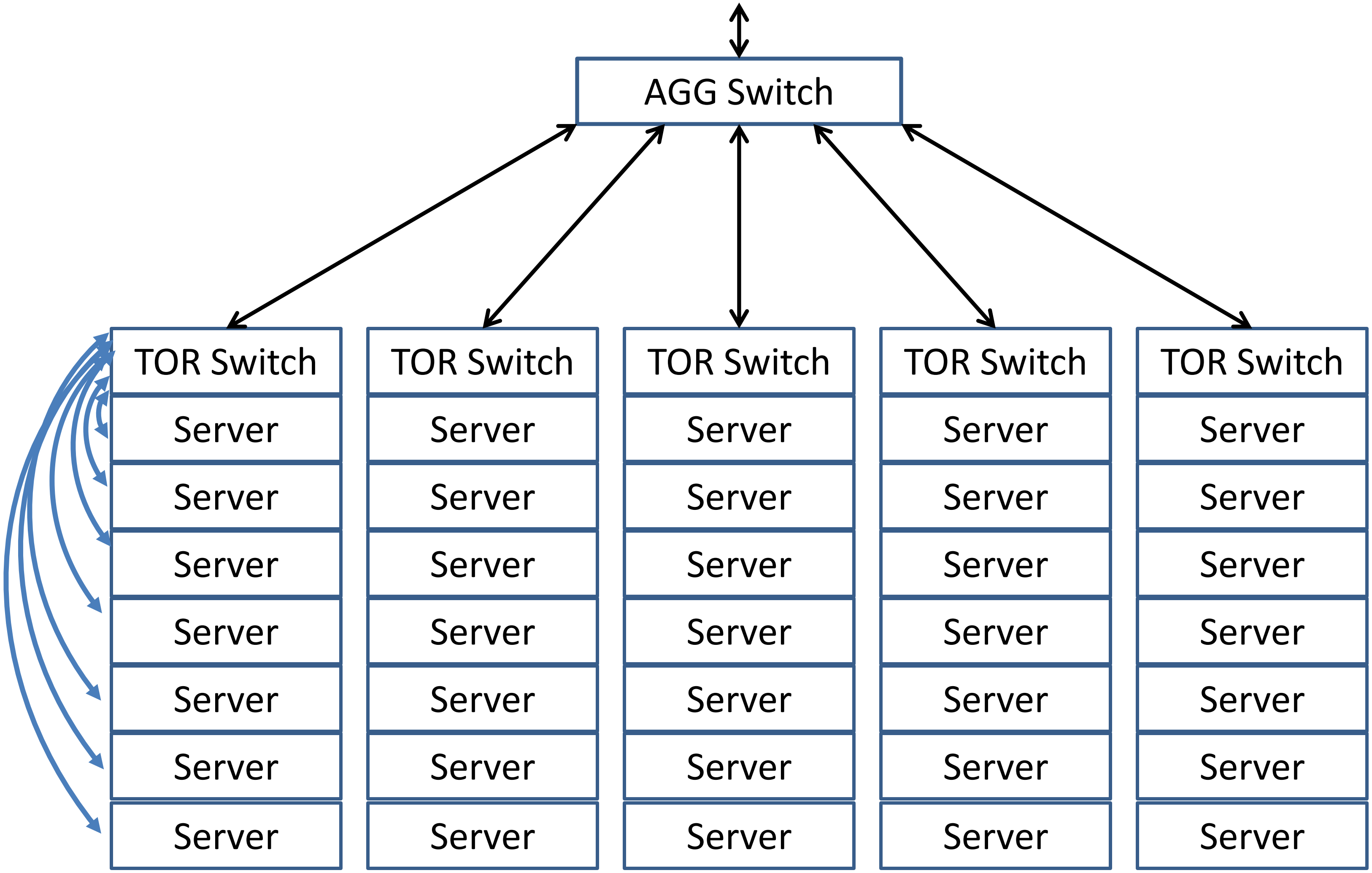
- If the network can prevent eavesdropping and assure the accuracy of the IP address from which communications come, you don’t need to cryptographically protect the connection!
  - With lots of caveats, including a secure translation from name to IP address
- Why is this hard?

IP Address Spoofing

- IP was designed of CSMA/CD Ethernet
- Links were inherently broadcast
- No way to tell who was broadcasting any given message
- They don’t exist anymore, but we live with their legacy



We Build Pseudo-Broadcast LANs for Compatibility



We Build Pseudo-Broadcast LANs for Compatibility

- Networks are built out of point-to-point links
- L2 switches imitate broadcast links, with some filtering to improve performance
- But they are technologically capable to verifying the source addresses of packets they receive and checking the destination addresses of packets they send
- Properly configured, then can prevent eavesdropping and prevent address spoofing

Packets on the Wire

- Ranges of IP addresses are assigned to pseudo-broadcast LANs

MAC Dest	MAC Src	IP Dest	IP Src	Payload

Auto-configuration and DHCP Option 82

- Nodes “boot from the network” with very little preconfiguration
- Nodes usually deployed with wired in 48 bit MAC address
- They learn their IP address using DHCP and download code
- How to assure download from “the right” server
- Answer: Configure switch with port leading to DHCP server. DHCP Option 82 allows to switch to reliably label packet with physical port information

Source IP address checking and autoconfiguration

- Switch can watch DHCP messages going by and only allow nodes to send from IP addresses that were assigned to them

Alternately, only connect trusted entities to the IP subnet

- Have all untrusted nodes on the network be VMs running under a trusted hypervisor
- Have the hypervisor do IP address filtering

Defense in Depth

- Independent security mechanisms, each secure by itself, e.g.
  - IP address authentication in the network
  - IP address authentication in the hypervisor
  - End-to-end cryptographic protection using SSL
- There is a danger of becoming overconfident

What security features do people want from a datacenter network hub (and why)

- Preventing IP address spoofing in a datacenter
- **Address/Port filtering**
- NATP
- Load Balancing
- HTTP proxy/cache
- Crypto Offload / Tunnel Endpoints
- DoS/DDoS diagnosis, blocking
- QoS / Bandwidth reservation

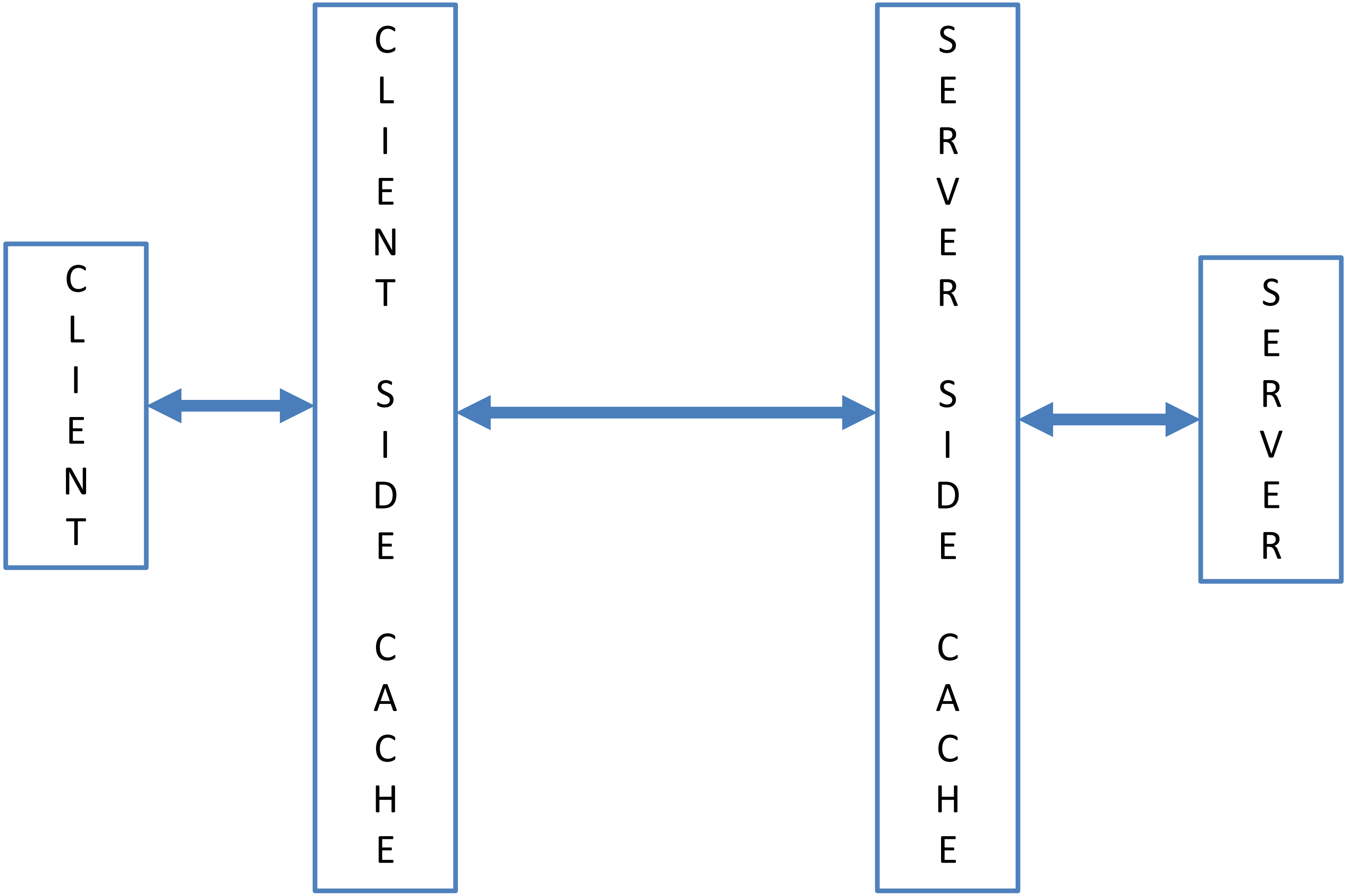
Address/Port filtering

- Duplicate the functionality in endnode firewalls
- Why both?
  - Defense in Depth
  - Different administrators want control
  - Firewalls on an endnode can be modified if software compromises the endnode

NATP / Load Balancer

- Network Address & Port Translation invented because of a shortage of IPv4 addresses
- Have come to have security functions
  - Allow nodes to make outbound connections but not accept inbound connections
  - Direct inbound connections to multiple nodes to balance the load

HTTP Proxy / Cache



CLIENT

CLIENT SIDE CACHE

SERVER SIDE CACHE

SERVER

HTTP Proxy/Cache

- Client Side Proxy/Cache
  - Speed up access and reduce network bandwidth if multiple clients in same organization accessing the same data
  - Often combined with NAT
  - Implement Browsing Restrictions
- Server Side Proxy/Cache
  - Reduce load on server
  - Position lots in locations near clients / speed access
  - Prevent DoS attacks from reaching server

Crypto Offload

- Cryptography can be expensive in software on the server
- Can offload SSL / IPsec processing onto a server proxy – possibly with hardware acceleration
- With encrypted tunnels, a link or series of links can be encrypted transparently to endnodes
  - For example, to link two company sites using the Internet
- Transparent vs. non-transparent offload

Network DDoS Filtering

- Any service can be overwhelmed by sufficient traffic
  - If it has a 1 Gb input link and an attacker sends it 10Gb of nuisance packets, 90% of legitimate traffic is not going to reach it
  - If the network can identify the nuisance packets (by source IP or other pattern) and discard them upstream, the server will recover
  - Analysis may be based on a random sampling of packets, so router must be capable to gathering them

QoS / Bandwidth Reservation

- If a network is overloaded, whose packets should be dropped
- TCP adapts to lost packets and shares bandwidth equally among connections
- Sometimes equality is not what you want
  - Some nodes may use more than their fair share and hurt performance
  - Some customers may have paid more for preference
  - Some protocols may use little bandwidth but react badly to lost packets (e.g. VoIP)

Agenda

- Cloud Data Center Networks
- Quick Introduction to Cryptography
- Quick Introduction to Public Key Infrastructures
- What security features do people want from a datacenter network hub (and why)
- **Security in Large Networks (no central administration)**
- Is security against Byzantine failures possible?
- Lessons learned in Cloud Security

Security in Large Networks

- It’s remarkable that the Internet works
- Security challenges are only apparent when something goes wrong
- Pakistan tried to block access to YouTube from Pakistan and *accidentally* blocked access from everywhere
- What could someone do if their actions were intentional?

How does the Internet Work?

- Who gets paid to deliver packets between you and someone on the other side of the world?
- You sign up with an Internet Service Provider (ISP) and pay them for “connectivity to the mythical middle”
  - Payment usually based on average and maximum bandwidth, independent of destination

How does your ISP deliver your data?

- If you are connecting to another of the ISP’s customers, the ISP handles it
  - This is rare
- The ISP can make deals with other ISPs where they split the cost of a connection between them
  - But there are too many to deal with all of them
- So the ISP buys transit service from other ISPs to connect to some or all other ISPs

What assures they are all connected?

- There are a dozen or so “core Internet Providers”, each of which is connected to all of the others, who can see connectivity to everywhere
- Every ISP buys connectivity (directly or indirectly) from one of them
- There are charges for “transit service” based on average and maximum bandwidth
- ISPs send data over the cheapest / fastest path that’s working at the moment

How does the ISP know which paths will work?

- BGP – Border Gateway Protocol
- ISPs announce to one another the sets of IP addresses they can reach
  - But only announce the ones they are willing to transit packets for
  - If we are peers not paying one another, we will only announce addresses of our customers
- Upstream data from BGP is reflected in downstream announcements
- Link failures cause rerouting to new shortest paths

An Ideal System would do much better

- Would like to send traffic over cheapest links until they become busy, then load balance to more expensive ones
- Would like to reserve bandwidth for things like video, but today it requires cooperation of too many parties
  - And they can’t figure out how to charge each other for it
  - If they can’t make money on it, they won’t do it

Other Problems

- Distributed Denial of Service – an attacker who understands the network topology can intentionally overload certain links causing traffic disruption

What happened with Pakistan?

- They announced a path to YouTube that didn’t really exist
- To deal with partitioned networks, an announcement of a path to a smaller range of IP addresses has priority over a path to a larger range even if the cost is higher
- All the YouTube connections in the world were redirected to Pakistan, where they were dropped

Proposed Solution: S-BGP

- Secure BGP – standard approved, but deployment is lacking
- Based on digital signatures
- A global authority signs statement concerning which ISP owns which addresses
- That ISP signs a statement of what networks it is connected to
- Find an route that goes through ISPs with signed evidence of connectivity

This is only a partial solution

- Pakistan could still block traffic that happened to transit Pakistan
- Panama for a few weeks blocked VoIP traffic to protect its telephone revenues
- When connections are reported as working, but traffic is selectively dropped, it’s difficult to figure out why, much less route around the problem

Agenda

- Cloud Data Center Networks
- Quick Introduction to Cryptography
- Quick Introduction to Public Key Infrastructures
- What security features do people want from a datacenter network hub (and why)
- Security in Large Networks (no central administration)
- **Is security against Byzantine failures possible?**
- Lessons learned in Cloud Security

Assuming some routers want to break the network, is it still possible to function securely?

- Generally not with today’s protocols
- Radia Perlman’s 1988 PhD Thesis proved it was possible
- Network Protocols with Byzantine Robustness (or NPBR)

Byzantine faults

- Not “fail-stop”
- Continue to operate, but incorrectly
  - Lie about who you’re connected to
  - Corrupt other routers’ packets
  - Flood the net with garbage
  - Do routing algorithm perfectly, but don’t forward data properly
  - Try to people think some other router is misbehaving

What are we trying to do?

- It’s not sufficient to secure the routing protocol
- We want the data to be delivered – reliably and with some “fair share” of bandwidth
- We’ll assume all data is encrypted and digitally signed end-to-end, so the network just has to deliver it; it’s OK if it also delivers fake packets or discloses real ones

NPBR’s Guarantee

- As long as there is any path between S and D consisting of properly behaving links and routers, data will be delivered, with some fair share of bandwidth
- (Strongest guarantee possible. If there is no such path, S and D can’t reliably communicate)
- We’ll derive the protocol by successive refinement

Attempt #1: Flooding

- Transmit each packet to each neighbor except the one from which it was received
- Have a hop count (or log, like source route bridging) so packets don’t loop infinitely
- This works! Pkts between A and B flow, if there is at least one nonfaulty path…

Attempt #1: Flooding

- Transmit each packet to each neighbor except the one from which it was received
- Have a hop count (or log, like source route bridging) so packets don’t loop infinitely
- This works! Pkts between A and B flow, if there is at least one nonfaulty path…
- *If there is infinite bandwidth….whoops!*

So, just a resource allocation problem

- The finite resources are
  - computation in switches
    - assume we can engineer boxes to keep up with wire speeds
  - memory in switches
  - bandwidth on links

Byzantinely Robust Flooding

- Flood newest pkt from each source to each nbr
- Do careful resource allocation
  - Computation (just engineer box for wire speed)
  - Memory (one buffer for each source)
  - Bandwidth (give all buffers a share)
- Source cryptographically signs packet so any node can verify signature
- And puts in a sequence number big enough to never “wrap” (e.g. 64 bits)
- Careful coordination between neighbors so packet delivery is reliable

Configuration

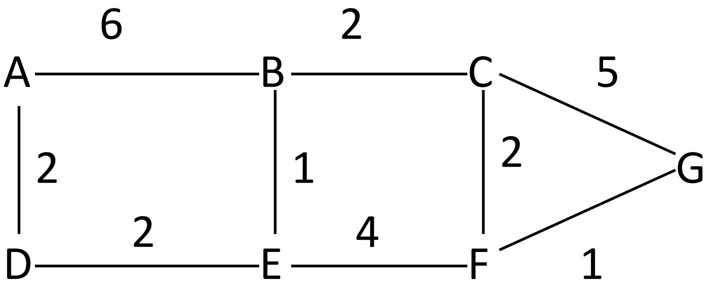
- Every node needs other nodes' public keys; would be a lot of configuration
- So instead have "trusted node" TN
  - TN knows all other nodes' public keys
  - All other nodes need their own private key, and the trusted node public key
- To simplify, assume every node is an endpoint and a router

Flooding

- First flood configuration information from the TN
- Use that configuration information to flood packets from others
- Coordinate with neighbor to make sure your packet databases are the same (exchange sequence numbers, get ack’s)
- If see new packet from source S, (sequence number bigger), send to other nbrs
- Cycle through sources, so all packets eventually sent

This design works!
…but is extremely inefficient

- So, we’ll do something else for unicast
- But we will use robust flooding for two things
  - easing configuration (advertising public keys)
  - distributing link state information



A	B	C	D	E	F	G
B/6	A/6	B/2	A/2	B/1	C/2	C/5
D/2	C/2	F/2	E/2	D/2	E/4	F/1
	E/1	G/5		F/4	G/1	

Configuration

- Trusted node (TN) configured with all public keys
- Everyone else (say R) configured with
  - public key of TN
  - R’s own private key
  - “n”: max # of nodes (so can allocate storage for TN)
- New node R1: configure R1 and TN with each other’s public keys, TN refloods

TN can flood

- Since everyone knows TN’s public key, TN can flood

Obvious question: What if TN faulty?

- Can’t be subtly faulty (will get quickly caught)
- Could require 3, and vote
- Or, can allocate resources for any public key advertised by TN
- At most, bad TN can use up 1/2 resources
- And it will get quickly detected

Data Packets (Unicast)

- “Traditional” per-destination forwarding won’t work
  - Bad guy can keep network in flux by flipping state of a link
  - What do you do about a path that works for everyone but S?
- Conclusion: Source chooses and sets up its own path
- Has link state database from flooding, so it’s easy…

Choosing a path

- Just because a path exists in the link state database doesn’t mean it will work
- If path doesn’t work, who is the bad guy?
- Could try to isolate bad guy, but:
  - Can’t be reliable… can never distinguish between two routers at ends of a link who each claim the other is misbehaving
  - Routers could start behaving when they detect testing
  - **We don’t need to in order to make the system work**

Simple heuristics

- If path to D works (end-to-end acks), then have more trust in routers along that path
- If path doesn't work, be suspicious of the routers on that path
- Try to eliminate routers one at a time, but if lots of bad guys, can be really expensive

Data packets (unicast)

- Source chooses a path
- Sets it up with a cryptographically signed setup packet, specifying the path
- Routers along the path have to keep per (S,D) pair
  - Input port
  - Output port
  - Buffers for data packet fwd’ed on this flow
- Forwarding data packets is efficient (no crypto)
- If source sends too fast, overwrites *its own* buffers

Overwriting Buffers

- For link state info and distributing public keys, only want latest packet
- For data packets, it may be a performance issue to allow buffers to be overwritten
  - TCP can cope but doesn't like lost pkts
  - Things like video could be encoded optimally if bandwidth available known in advance
- But it's not a correctness issue (for NPBR goals)

And how many buffers do you really need?

- Ideally enough, at each switch, for each flow, to keep each link full*, if that one flow is the only thing going on in the net*

Why doesn’t the Internet implement this?

- Large amount of buffering required in routers – at least one buffer per connection flowing through it
- For very large networks, finding a path could take a long time
- We would need to introduce hierarchy, which is not as easy as it sounds

The Real Reason

- So far, there haven’t been participants trying to sabotage the network
- There is little motivation to invest in a secure solution so long as the insecure one works well enough in practice
- To the extent endnodes use end-to-end encryption, network sabotage “only” causes denial of service
  - Attackers have better things to do, like using the network to attack the endnodes

Agenda

- Cloud Data Center Networks
- Quick Introduction to Cryptography
- Quick Introduction to Public Key Infrastructures
- What security features do people want from a datacenter network hub (and why)
- Security in Large Networks (no central administration)
- Is security against Byzantine failures possible?
- **Lessons learned in Cloud Security**

What's Different about Security in a Public Cloud?

- The stakes are higher
- The customers are less trusted…
  - Must be treated as hostile
- The customers' data must be protected from system operators
  - What's good practice within an enterprise is a contractual guarantee in a public cloud

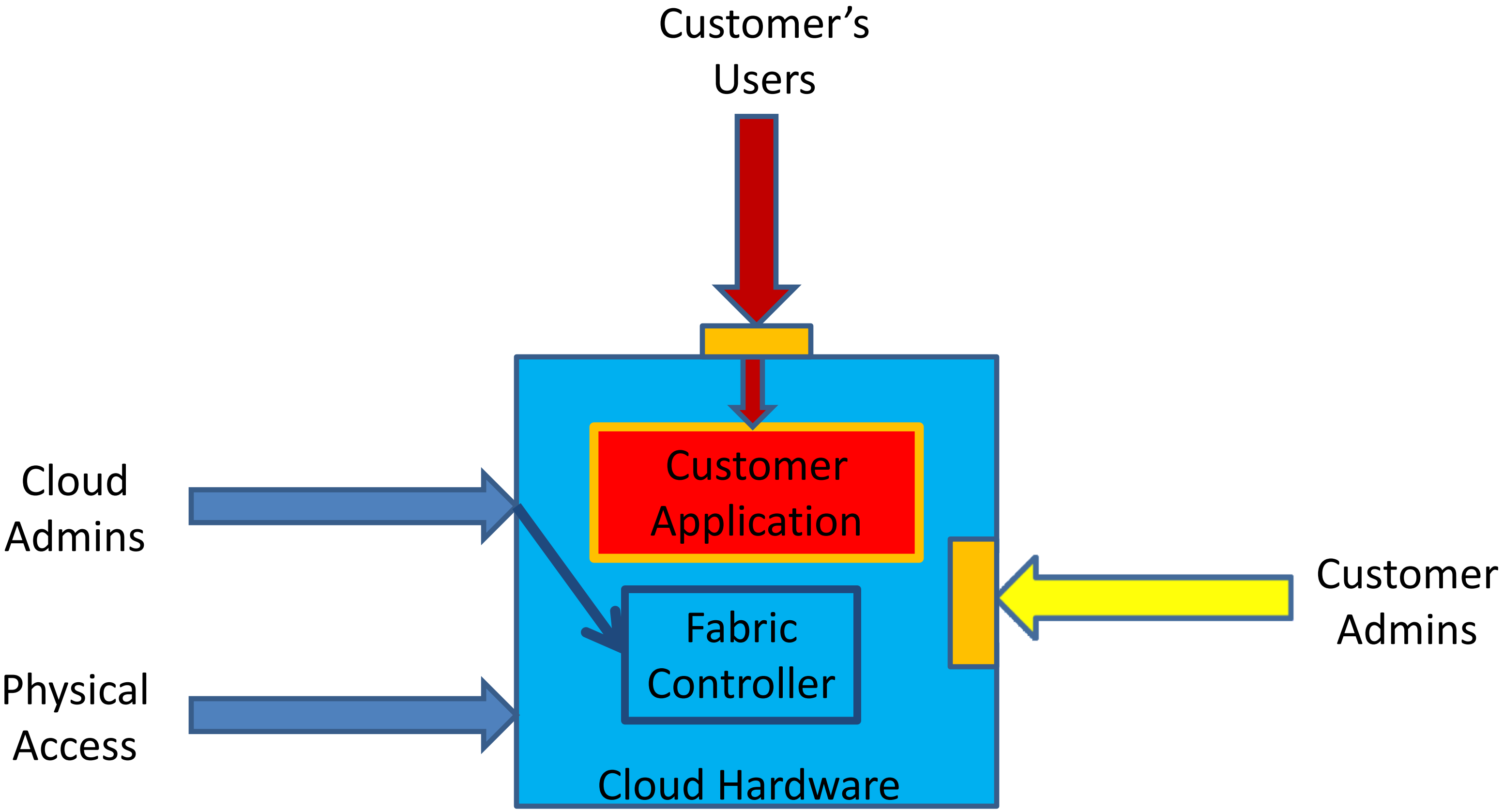
What’s the Same?

- Detecting and preventing intrusions
- Mitigating DDoS attacks
- Protecting services from one another
  - Including fair allocation of shared resources
- Keeping patches up to date
- Focus on minimizing the attack surface

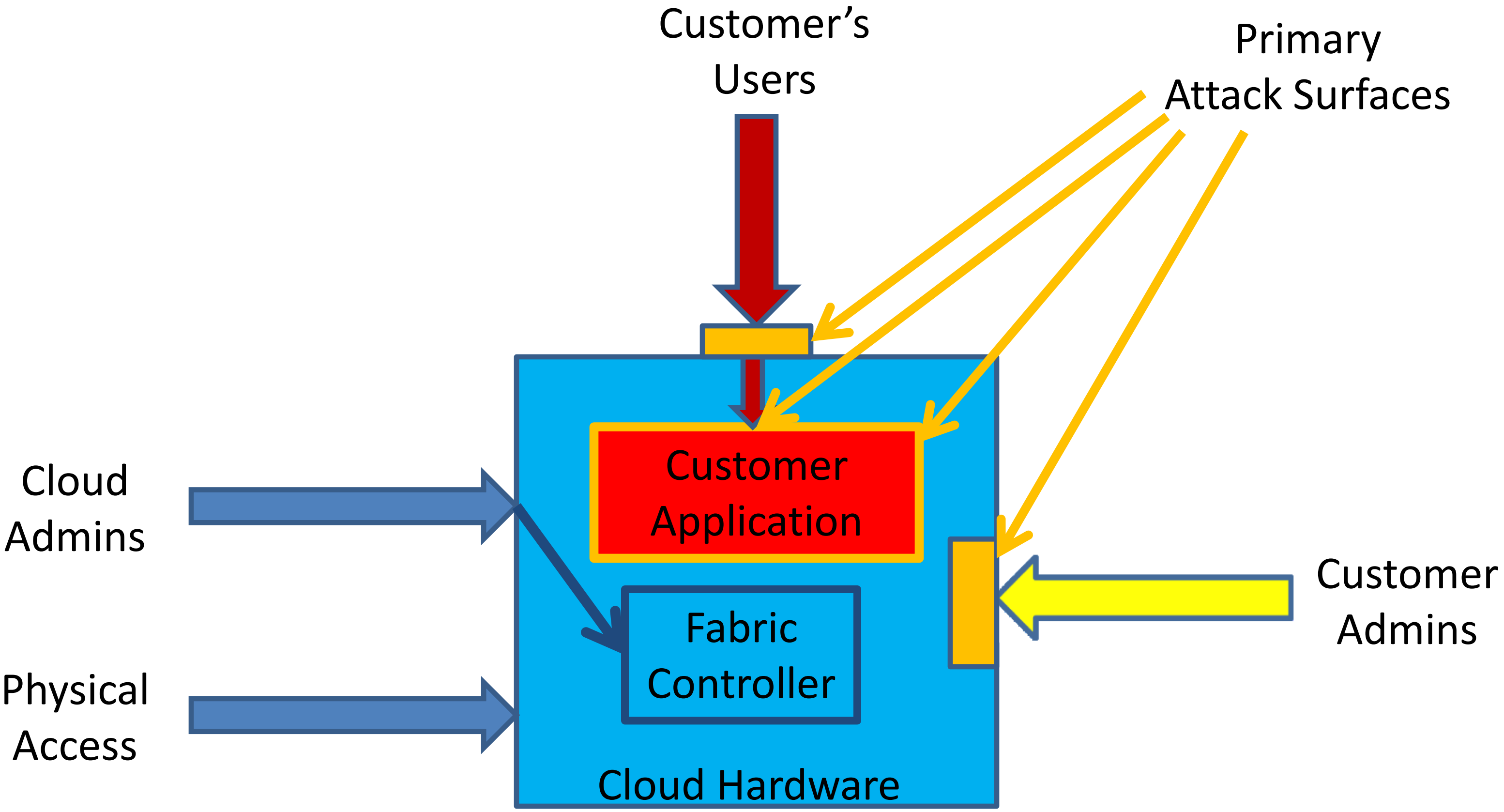
Division of Responsibilities

- Protection of a service requires use of a variety of tools
  - Some can be used by the cloud provider
  - Some can be used by the customer
  - *Some can't be provided easily by either, and these require some workaround*

Generic Cloud Computing Engine



Generic Cloud Computing Engine



Protecting the Infrastructure from Customer Admins

- Many systems delegate limited administrator privileges
- …but they typically don’t assume the limited administrators are actually hostile
- In a public cloud, you must assume they are

Protecting the Infrastructure from Customer Applications

- Within a corporate data center, it is not unusual for some server to be compromised by some bug
- Designers therefore should assume that these applications might be hostile
- But most don’t take the threat seriously; in a public cloud, we must
- If you mess up in your own data center, you’re less likely to be sued

What to use for an application sandbox?

- We chose to use VMs over a hypervisor
- Could have used processes within an OS
- Could have used managed code isolation within a process (Java, C#)
- Could have used machines isolated by VLANs
- VMs have the advantage of being new, without a lot of time to introduce performance features that weaken security

Helping Customers to protect themselves from their users

- Typical datacenters don’t expose their servers to the full onslaught of the Internet
  - Datacenter firewalls
  - Intrusion detection hardware/software
  - DDoS mitigation systems
  - SSL accelerators
- Often these require considerable expertise to configure optimally

So those were the attacks we prepared for…

What did we actually see?

So those were the attacks we prepared for…

What did we actually see?

1. Bots establishing accounts with stolen credit cards

So those were the attacks we prepared for…

What did we actually see?

1. Bots establishing accounts with stolen credit cards
2. A new challenge that requires some innovative thinking…

Protecting the Internet from our Customers

A Cloud provider acts as – among other things – an Internet Service Provider

- Provides greater anonymity than most ISPs
- Provides more bandwidth than most ISPs
- Rents out resources for a much shorter period of time

What kinds of behavior are acceptable?

Bad Behavior

- Acting as a rendezvous point for a bot army
- Impersonating another site in a phishing attack
- Sending out Spam!
- Posting malware for download
- Conducting DoS attacks (AaaS)
- Probing systems for vulnerabilities

The Internet has developed an immune system

- IP addresses that are the source of spam or malware get blacklisted
- IP addresses that are the source of DoS or probing attacks are blocked and reported to their owners for corrective actions
- If someone rents an IP address and a gigabit of bandwidth for 15 minutes, the reaction hurts the next tenant

How do you define bad behavior?

- How do you distinguish a spam engine from a mail agent relay distributing mail to a mailing list?
- How many failed DNS queries are allowed before it constitutes an exhaustive search through a namespace?
- What looks like an attack could be someone testing the security of their own system

How do you handle complaints?

- Forward them to the customer responsible?
- Forward customer contact information to the complainant?
- The complainant could be complaining as a form of DoS attack on the customer

Some Sad Realities of Security

- You won’t be attacked until it really matters
  - You can be lulled into a false sense of security by getting away with sloppy practices for a long time
  - Even if you aren’t, your management will be
- You won’t be attacked at the interface you focused so hard on securing
- DDoS is the last attack you’ll think about and the first attack you’ll see

Quote from Akamai Employee

…when asked about their experience with Denial of Service attacks.

- The good news is that we have only experienced one serious denial of service attack.
- The bad news is that is began the day we enabled production traffic and continues to this day.

Some Sad Realities of Security

- The fun part of security is coming up with clever solutions to hard problems
- The hard part is knowing when something is secure enough – there are two ways to fail:
  - Deploying something that will lead to disaster
  - Not deploying anything until it is provably secure against any conceivable threat

A Happy Reality of Security

Si spy net work,

big fedjaw iog link kyxogy

Dedication in the book:

Network Security: Private Communication in a Public World

To the bad guys,

for making our jobs secure

Questions?