Course Overview

CS 436: Spring 2018
Matthew Caesar
http://www.cs.illinois.edu/~caesar/cs436/
Motivating Scenario

• You are a VP Networking for fortune 500 company

• A data breach of critical assets occurs

• What do you do?
Example: Designing an Internet Service

• Imagine you are running Amazon, Yahoo, or Google.

• How do you design your software to handle massive load?

• Tens of thousands of clicks \textit{a second}

• Extremely bursty demand

• Service must be up and running 24/7 with no (discernible) outages

• This is a hard problem!
Suppose someone asks you to... (real examples)

• “The latest OS software update caused half of our CDN to go down! We are losing millions of dollars of contracts by the hour! How can we get things back up as fast as possible?”

• “Customers of our North American ISP backbone are noticing random outages – diagnose and fix what’s wrong”

• “Here’s a bunch of routers and ethernet cable. Can you put together a network for us?”

• “Our web service is being hit with a DoS attack – defend us”

• What would you do?
This stuff is harder than it looks
FCC blasts Verizon for 911 outages

Transport Airlines

United's global systems fail, grounding thousands

Russia hacked voting systems in 39 states before the 2016 election

Equifax's Massive Data Breach Has Cost the Company $4 Billion So Far

By Paul I. Lim September 12, 2017

While it's too soon to tell what the ultimate cost of Equifax's data breach will be, Wall Street has already rendered its initial verdict: $4 billion.

That's how much stock market value Equifax has lost since the credit bureau revealed last week that it was hacked, compromising the personal information of about 143 million people.

Trio of Cisco flaws may threaten networks

Power-Grid Cyber Attack Seen Millions in Dark for Months

Target: Hacking hit 70 million
Theory vs. Practice

• It’s important to understand the theory
  • Protocols, algorithms, techniques behind systems
• But it’s also important to know how to apply the theory
  • Where the real challenges are
  • Best practices and rules of thumb
  • Build breadth and see the “big picture”
• This class covers both, together
  • Learning practice along with theory makes learning the theory more fun
Why a “lab” class?

• Hands-on approach enables direct observation and experimentation
  • Builds deeper understanding of system design
  • Also makes learning the fundamentals more fun

• In this class you’ll work directly with real systems
  • We have an awesome lab environment for you
  • You’ll have direct access to code/systems widely used in many real-world production environments
So, how are we going to do this?

• Course is organized into a set of “cycles”
  • Each cycle composed of two lectures and two lab sessions

• **Complete** overview of modern network infrastructure
  • ISP and enterprise networking, data centers, clouds, streaming and online services, etc.

• Lab sessions are *not* MPs
  • Focus on **education**: goal is to “guide” you as much as possible, to walk you through the system’s design
  • Focus on **interactivity**: instructor and TAs will be present at the lab sessions to guide you
  • Focus on **experimentation**: we want you to explore—go off and have fun!
How do humans learn?
Cycle timeline

- There will be seven cycles, each comprising 2 weeks
- Lab released at start of cycle, due at end of cycle
- Labs designed to be started at **beginning** of cycle
  - Work on them throughout the cycle, not just lab days
Real-Time Communication: Physical Labs

- Work with real things
- Talk to me/each other
- Interactive lecturing
Real-Time Communication: *Slack*

---

**#lab4**

Wednesday, October 25th

usual. I see the typical warning: `ssh myserver` The authenticity of host `ec2-12-34-567-890.compute-1.amazonaws.com` (12.34.5...)

![Image: Slack chat](attachment:image.png)

**skim104** 12:40 PM
i thought aws had online console feature somewhere. i must be imagining things

**oscar** 12:42 PM
Yes, but that's actually a java instance running locally.

**skim104** 12:43 PM
nah. i was thinking of a console running on their side. but i think my memory is confusing it with vmware esxi (edited)

**jcz3** 12:44 PM
okay thanks, i already knew about the syslog approach described in your link which is pretty inconvenient i guess but enough in this scope

**oscar** 12:47 PM
http://d.pr/i/n3CZA0 the online console provided by AWS

---

d.pr

**Screen Shot 2017-10-25 at 12.47.39.png** • Droplr™

Shared with Droplr (197 kB)
Real-Time Communication: Living Labs

Lab 5: SDN and Mininet Lab

```python
def simpleTest():
    "Create and test a simple network"
    topo = SimpleSwitchTopo(n=4)
    net = Mininet(topo)
    net.start()
    print "Dumping host connections"
    dumpNodeConnections(net.hosts)
    print "Testing network connectivity"
    net.pingAll()
    net.stop()

if __name__ == '__main__':
    # Tell mininet to print useful information
    setLogLevel('info')
    simpleTest()
```

Explanation:

Above is a script that is provided by mininet as an example script of a custom topology and explanation of the important classes, methods, functions and variables in the above code is provided.

**Topo**: the base class for Mininet topologies

**build()**: The method to override in your topology class. Constructor parameters (n) will be passed
<table>
<thead>
<tr>
<th>Cycle</th>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Internet and ISP Networking</td>
<td>Routing protocols, ISP systems, how to run your own ISP</td>
</tr>
<tr>
<td>2.</td>
<td>Enterprise Networking</td>
<td>Network defense, troubleshooting, and design</td>
</tr>
<tr>
<td>3.</td>
<td>Equipment and Devices</td>
<td>Router, switch, server, and network architectures</td>
</tr>
<tr>
<td>4.</td>
<td>Cloud Networking</td>
<td>Amazon Web Services, Cloud and service infrastructure, network programming</td>
</tr>
<tr>
<td>5.</td>
<td>Emerging Tech: SDN and NFV</td>
<td>Software-defined networking, network virtualization</td>
</tr>
<tr>
<td>6.</td>
<td>OS Network Stacks</td>
<td>Linux kernel internals, network stacks, service-customized OS</td>
</tr>
<tr>
<td>7.</td>
<td>Devops and Deployment</td>
<td>Network management and configuration, common network protocols and practices</td>
</tr>
</tbody>
</table>
About me:
Matthew Caesar (Instructor)

- Expertise: networking, systems
- Faculty in CS department
  - PhD from UC Berkeley in 2007
- CSO at Veriflow Systems
- Industrial experience at AT&T Labs, Microsoft Research, HP, Nokia DSL; helped found a CDN company; ongoing partnerships/tech transfer with Cisco, DARPA, NSA, Boeing
- I like designing/building/deploying large-scale software systems that are grounded in strong theoretical principles
- Office: 3118 SC
- Email: caesar@Illinois.edu
This class

• Teaches networking/systems design through a hands-on approach
  • OS networking kernels, cloud services, firewall/router configuration, protocol stacks
• Uses modern implementations commonly used in industry
  • Cisco IOS, Linux networking stack, Amazon web services, etc.
• Leverages real-world scenarios commonly encountered in industry
  • Building networks, deploying cloud services, blocking an attack, implementing a new protocol
Grading Policy

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Lab Assignments</td>
<td>75%</td>
</tr>
<tr>
<td>Course Excellence</td>
<td>25%</td>
</tr>
</tbody>
</table>

• Goal: I want you to learn
  • Grade measures completion of tasks and also how much you learned
  • If you know a lab already, talk to me
Examples of “Excellence”

• Take on a promotion
• Help improve a future lab
• Work through a lab early
• Research project
• Complete a lab and then go “a step beyond”
• Ask questions, pose ideas, answer questions

• I am looking for things that help you and help others
• I will suggest examples on slack throughout the semester
Administrative Details
Prerequisites

- Networking / Operating Systems Concepts
  - CS 241 or equivalent

- Programming
  - Shell scripting (sed/bash)
  - Java/Python/C++
Homework and Projects

• Write a report proving you did the lab
  • All code, answers to questions
  • Organize it to make it readable

• Email to the TA

• Late policy: 2% off per hour late
Academic Honesty

• Your work in this class must be your own.
• All infractions reported to the department
• If students are found to have collaborated excessively or to have cheated (e.g., by copying or sharing answers during an examination or sharing code for the project), all involved will at a minimum receive grades of 0 for the first infraction.
  • We will run a similarity-checking system on code and binaries
• Further infractions will result in failure in the course and/or recommendation for dismissal from the university.
CS 436: EXTREME EDITION

• Want an added challenge?

• Take on an additional CHALLENGE PROJECT
  • Get four units instead of three
  • Open to both grad and undergrad

• Come talk to me if you are interested in this option
Class Communications

• Web site: http://www.cs.illinois.edu/~caesar/cs436
  • Assignments, lecture slides, announcements

• Email list: cs-436@lists.illinois.edu

• Slack group: https://cs-436.slack.com

• Please cc our staff on any non-private emails sent directly to me (caesar@illinois.edu)
Emergency Preparedness

• Learn different ways to leave building

• Severe weather – get to low-level in middle of building

• Active shooter – run > hide > fight
  • Lock/barricade door

• Sign up for emergency text messages at emergency.illinois.edu

• More info: police.illinois.edu/safe
Any questions?
Cycle 1:
Internet and ISP Networking

CS 436: Spring 2018
Matthew Caesar
Motivating Questions

• One of our enterprise customers is undergoing a DoS attack – how can we protect them?
• We are entering the European market and need to construct an international IP backbone – can you propose a design for us?
• Customers of our North American ISP backbone are noticing random outages – diagnose and fix what’s wrong.
• We have to peer our network with our biggest competitor – how do we protect ourselves from them?
How can Two Hosts Communicate?

- Encode information on modulated “Carrier signal”
  - Phase, frequency, and amplitude modulation, and combinations thereof
  - Ethernet: self-clocking Manchester coding ensures one transition per clock
  - Technologies: copper, optical, wireless

Data: 1 0 1 0 0 1 1 1 0 0 1

Voltages: 0.7 Volts and -0.7 Volts
How can many hosts communicate?

- Naïve approach: full mesh
- Problem:
  - Obviously doesn’t scale to the 570,937,778+ hosts in the Internet
How can many hosts communicate?

- Better approach: Multiplex traffic with routers
- Goals: make network robust to failures and attack, maintain spare capacity, reduce operational costs
  - Introduces new challenges: What topology to use? How to find and look up paths? How to identify destinations?
• Hosts assigned topology-dependent addresses
• Routers advertise address blocks ("prefixes")
• Routers compute “shortest” paths to prefixes
• Map IP addresses to names with DNS
Internet Routing

• Internet Routing works at two levels

• Each AS runs an intra-domain routing protocol that establishes routes within its domain
  • (AS -- region of network under a single administrative entity)
  • Link State, e.g., Open Shortest Path First (OSPF)
  • Distance Vector, e.g., Routing Information Protocol (RIP)

• ASes participate in an inter-domain routing protocol that establishes routes between domains
  • Path Vector, e.g., Border Gateway Protocol (BGP)
Intra- vs. Inter-domain routing

- Run “Interior Gateway Protocol” (IGP) within ISPs
  - OSPF, IS-IS, RIP
- Use “Border Gateway Protocol” (BGP) to connect ISPs
  - To reduce costs, peer at exchange points (AMS-IX, MAE-EAST)
Link-State Routing
Each node maintains a “topology database”

F tells all routers: there is a link between F and E

- How to prevent update loops: (seq numbers)
- How to bring up new node: (load TDB from neighbor)
Each router computes shortest path tree, rooted at that router
Determines next-hop to each dest, publish to forwarding table
Operators can assign link costs to control path selection
Link-state: packet forwarding

In practice: shortest path precomputed, next-hops stored in forwarding table.
Downsides of link-state:
  - Lesser control on policy (certain routes can’t be filtered), more cpu
  - Increased visibility (bad for privacy, but good for diagnostics)
Distance Vector Routing

- Each router knows the links to its neighbors
  - Does *not* flood this information to the whole network
- Each router has provisional “shortest path” to every other router
  - E.g.: Router A: “I can get to router B with cost 11”
- Routers exchange this distance vector information with their neighboring routers
  - Vector because one entry per destination
- Routers look over the set of options offered by their neighbors and select the best one
- Iterative process converges to set of shortest paths
**Distance vector:**

- **Update propagation:**

  - **D tells B:** I am D, and I can reach F via 1 hop

<table>
<thead>
<tr>
<th>Dest</th>
<th>NextHop</th>
<th>Dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>D</td>
<td>2</td>
</tr>
</tbody>
</table>

  - **F tells D:** I am F, and I can reach F via 0 hops

<table>
<thead>
<tr>
<th>Dest</th>
<th>NextHop</th>
<th>Dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>1</td>
</tr>
</tbody>
</table>

**B's forwarding table:**

<table>
<thead>
<tr>
<th>Dest</th>
<th>NextHop</th>
<th>Dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>D</td>
<td>2</td>
</tr>
</tbody>
</table>

**D's forwarding table:**

<table>
<thead>
<tr>
<th>Dest</th>
<th>NextHop</th>
<th>Dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>1</td>
</tr>
</tbody>
</table>

---

**Nodes:**

- A
- B
- C
- D
- E
- F

**Arrows:**

- Green arrows represent message paths:
  - A to B: (F,2)
  - B to C: (F,2)
  - C to B: (F,2)
  - B to D: (F,1)
  - D to F: (F,0)
  - F to D: (F,1)
  - D to E: (F,1)
  - E to D: (F,0)
  - D to B: (F,1)
  - B to A: (F,2)
  - A to C: (F,2)
  - C to A: (F,2)
  - A to D: (F,2)
  - D to A: (F,2)
  - C to D: (F,1)
  - D to C: (F,1)
  - E to B: (F,0)
  - B to E: (F,0)
  - E to A: (F,0)
  - A to E: (F,0)
  - D to F: (F,0)
  - F to A: (F,0)

**Legend:**

- **Source:** A
- **Destination:** F

---

**Notes:**

- The diagram shows the propagation of distance vector updates among nodes A to F with their respective forwarding tables and distance counts.
Distance vector: convergence

Updates received by A: \(0, 1, 2, 3, 4, 5, 7\)
Withdraw(H)

- How many updates would link-state require?
- Is link-state better or worse than distance vector?
- Which should be used for intra-domain routing?
- What about inter-domain routing?
Hierarchical routing

In regular link-state, routers maintain map of entire topology
Hierarchical routing

Routers summarize paths across areas as “virtual links”

Aggregate groups of routers into “areas”

“Border routers” generate “summary LSPs” to reach other border routers
IP Routing: Interdomain
Internet Routing

• So far, only considered routing within a domain.

• Many issues can be ignored in this setting because there is central administrative control over routers.
  • Issues such as autonomy, privacy, policy

• But the Internet is more than a single domain
Recall from Lecture 3:

- **“Autonomous System (AS)” or “Domain”**
  Region of a network under a single administrative entity

- **An “end-to-end” route**

- **“Border Routers”**

- **“Interior Routers”**
Autonomous Systems (AS)

• AS is a network under a single administrative control
  • currently over 30,000 ASes
  • Think AT&T, France Telecom, UCB, IBM, etc.

• ASes are sometimes called “domains”.
  • Hence, “interdomain routing”

• Each AS is assigned a unique identifier
  • 16 bit AS Number (ASN)
Administrative structure shapes Interdomain routing

• ASes want freedom to pick routes based on policy
  • “My traffic can’t be carried over my competitor’s network”
  • “I don’t want to carry A’s traffic through my network”
  • Not expressible as Internet-wide “shortest path”!

• ASes want autonomy
  • Want to choose their own internal routing protocol
  • Want to choose their own policy

• ASes want privacy
  • choice of network topology, routing policies, etc.
Choice of Routing Algorithm

Link State (LS) vs. Distance Vector (DV)?

• LS offers no privacy -- global sharing of all network information (neighbors, policies)

• LS limits autonomy -- need agreement on metric, algorithm

• DV is a decent starting point
  • per-destination advertisement gives providers a hook for finer-grained control over whether/which routes to advertise
  • but DV wasn’t designed to implement policy

The “Border Gateway Protocol” (BGP) extends distance-vector ideas to accommodate policy
Shortest-path forwarding isn’t enough

- In the real world, ISPs want to influence path selection
  - Load balance traffic, prefer cheaper paths, avoid untrusted routes, give preferential service, block reachability, limit external control over path selection decisions

- One trick: change the “cost” used to compute shortest paths
- Another trick: filter routes from being received from/advertised to certain neighbors
A few other inconvenient aspects

• What if we use a non-additive metric?
  • E.g., maximal capacity

• What if routers don’t use the same metric?
  • I want low delay, you want low loss rate?

• What happens if nodes lie?
Can You Use Any Metric?

• I said that we can pick any metric. Really?
• What about maximizing capacity?
ISP A prefers route through C over direct route

ISP C prefers route through B over direct route

ISP B prefers route through A over direct route

Advertise (D-p)
Prefix P
Advertise (A-D-p)

Advertise (A-C-D-p)
(link price: $100 per 1Gbps)

Advertise (B-D-p)
(link price: $5000 per 1Gbps)

Advertise (C-D-p)
(link price: $5000 per 1Gbps)

Withdraw
Withdraw
Withdraw
Withdraw
Withdraw
Withdraw
Policy disputes

ISP A prefers route through C over direct route

ISP B prefers route through A over direct route

ISP C prefers route through B over direct route

Advertise(D-p)
Advertise(A-D-p)
Advertise(A-D-p)
Advertise(C-D-p)
Advertise(B-D-p)
Advertise(B-C-D-p)
Withdraw
Withdraw
Withdraw
Withdraw
Withdraw
Withdraw
Policy disputes

ISP A prefers route through C over direct route
ISP B prefers route through A over direct route
ISP C prefers route through B over direct route

Advertise(D-p)
Advertise(A-D-p)
Advertise(A-D-p)
Advertise(C-D-p)
Advertise(B-D-p)
Advertise(C-D-p)
Withdraw
Withdraw
Withdraw
Withdraw
Withdraw
Withdraw
ISP A prefers route through C over direct route

ISP B prefers route through A over direct route

ISP C prefers route through B over direct route
No agreement on metrics?

• If the nodes choose their paths according to different criteria, then bad things might happen

• Example
  • Node A is minimizing latency
  • Node B is minimizing loss rate
  • Node C is minimizing price

• Any of those goals are fine, if globally adopted
  • Only a problem when nodes use different criteria

• Consider a routing algorithm where paths are described by delay, cost, loss
Must agree on loop-avoiding metric

• When all nodes minimize same metric
• And that metric increases around loops
• Then process is guaranteed to converge
Metrics (intradomain)

- Propagation delay
- Congestion
- Load balance
- Bandwidth (available, capacity, maximal, bbw)
- Price
- Reliability
- Loss rate
- Combinations of the above

In practice, operators set abstract “weights” (much like our costs); how exactly is a bit of a black art
What happens when routers lie?

• What if a router claims a 1-hop path to everywhere?

• All traffic from nearby routers gets sent there

• How can you tell if they are lying?

• Can this happen in real life?
  • It has, several times....
“Configuring” routers

- “Log in” to router (vty)
- Issue commands to view state, instill policy

```
R7(config)# route-map LOCALPREF permit 10
R7(config-route-map)# set local-preference 500
R7(config-route-map)# router bgp 67
R7(config-router)# neighbor 172.31.78.8 route-map LOCALPREF in
```
Changing the “cost” of paths

- ISPs have a lot of different kinds of policies
  - Could make cost a linear combination of different metrics
  - More expressive: have several “costs” per link

- Main idea: append “attributes” to updates

- Can set preferences (or filter the route) based on set of attributes contained in update
  - Hard-coded “decision process” orders importance of attributes
  - This process can be influenced by changing values of attributes

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
<th>Controlled by local or neighbor AS?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Highest LocalPref</td>
<td>local</td>
</tr>
<tr>
<td>2</td>
<td>Lowest AS path length</td>
<td>neighbor</td>
</tr>
<tr>
<td>3</td>
<td>Lowest origin type</td>
<td>neither</td>
</tr>
<tr>
<td>4</td>
<td>Lowest MED</td>
<td>neighbor</td>
</tr>
<tr>
<td>5</td>
<td>eBGP-learned over iBGP-learned</td>
<td>neither</td>
</tr>
<tr>
<td>6</td>
<td>Lowest IGP cost to border router</td>
<td>local</td>
</tr>
<tr>
<td>7</td>
<td>Lowest router ID (to break ties)</td>
<td>neither</td>
</tr>
</tbody>
</table>
Example: Using MED to balance traffic across ingresses

- MED: “multi-exit discriminator”
  - tell neighboring ISP which ingress peering points I prefer
  - Local ISP can choose to filter MED on import

I would like AT&T to route to me via PoP A

- MED=1
- MED=2

PoP A

PoP B

Sprint

AT&T

source

dest
Different peering points and advertisements

- Sprint can trick AT&T into routing over a longer distance!
- Consistent export: make sure your neighbor is advertising the same set of prefixes at all peering points
- ISPs sometimes sign SLAs with consistent export clause

AT&T isn’t listening to my MEDs, but I would REALLY like AT&T to route to me via PoP A
How inter- and intra-domain routing work together

1. Provide internal reachability (IGP)...
2. Learn routes to external destinations (eBGP)...
3. Distribute externally learned routes internally (iBGP)...
4. Select closest egress (IGP)....
Policies between ISPs: Types of ASes

- **Stub**: ISP with no customers
- **Multihomed**: ISP with more than one provider
- **Transit**: ISP that forward traffic between other ISPs
- **Tier-1**: ISP with no providers (core of Internet is clique of tier-1s)
- **Tier-1s** must be connected in a full mesh (Why? Who makes sure that happens?)
Policies between ISPs:
Types of AS relationships

Provider-customer: customer pays provider money to transit traffic

Peer link: ISPs form link out of mutual benefit, typically no money is exchanged
AS relationships influence routing policies

- Example policies: peer, provider/customer
- Also trust issues, security, scalability, traffic engineering
Problem: need to export routes only to certain neighbors

Solution: use “community attribute” tags to annotate routing advertisements
“Costing out” of equipment

- Increase cost of link to high value
  - Triggers immediate flooding of LSAs
- Leads to new shortest paths avoiding the link
  - While the link still exists to forward during convergence
- Then, can safely disconnect the link
  - New flooding of LSAs, but no influence on forwarding

Suppose we want to take down this link.
Delivery Models: Unicast vs Multicast
Delivery models

• Unicast
  • One source, one destination
  • Widely used (web, p2p, streaming, many other protocols)

• Broadcast
• Multicast
• Anycast
Delivery models

- Unicast
- Broadcast
  - One source, all destinations
  - Used to disseminate control information, perform service discovery
- Multicast
- Anycast
Delivery models

• Unicast
• Broadcast
• Multicast
  • One source, several (prespecified) destinations
  • Used within some ISP infrastructures for content delivery, overlay networks
• Anycast
Delivery models

- Unicast
- Broadcast
- Multicast
- Anycast
  - One source, route to “best” destination
  - Used in DNS, content distribution
Source-specific trees

• Each source is the root of its own tree
• One tree per source
• Tree consists of shortest paths to each receiver

Diagram:

- A
- B
- C
- D
- E
- F
- G
- H
- I
- J

Legend:

- Green circle: Member of multicast group
- Green dot: Sender to multicast group
Source-specific trees

- Each source is the root of its own tree
- One tree per source
- Tree consists of shortest paths to each receiver
Shared Tree

- One tree used by all members of a group
- Rooted at “rendezvous point” (RP)
- Less state to maintain, but hard to pick a tree that’s “good” for everybody!
Shared Tree

- Ideally, find a “Steiner tree” minimum-weighted tree connecting only the multicast members
  - Unfortunately, this is NP-hard
- Instead, use heuristics
  - E.g., find a minimum spanning tree (much easier)
What Happens Here?

Problem: “cost” does not change around loop

Additive measures avoid this problem!
What Happens Here?

- Cares about price, then loss
- Cares about delay, then price
- Cares about loss, then delay