By

# CSCI551 Midterm Review

Jiechang 2017.3.2

# Week 2

1. P5. [Clark88a] David D. Clark. The design philosophy of the DARPA Internet protocols. In Proceedings of the 1988 Symposium on Communications Architectures and Protocols, pages 106–114. ACM, August 1988. (Clark88a.pdf)
2. P6. [Saltzer82a] Jermome H. Saltzer. On the naming and binding of network destinations. In International Symposium on Local Computer Networks, pages 311–317, April 1982. (Saltzer82a.pdf)
3. P7. [Saltzer81a] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. Proceedings of the 2nd International Conference on Distributed Computing Systems, pages 509–512, April 1981. (Saltzer81a.pdf)

## Note For Saltzer81a

### Overview

1. End-to-End Argument:
    - A function can only be solve with the knowledge and help of the high-level(application level).
2. Proof that the low-level redundant for system crash recovery is little value.
    - particularly if the lower layers have a cost to do
    - if the app doesn't need it, it still need to pay for the cost
3. key Ideas:
    - where to put the function?
    - don't duplicate function at multiple level if it must be done at the end.
    - Why? the app is the only one that can guarantee it.
    - However putting it at lower level may improve the performance

### Example:Careful File Transfer

1. Defind the problem and transfer process(In Section 2.1)
2. 5 possible fault during the transfer process:
    1. hardware faults in the disk storage system
    2. mistake in copying and buffering
    3. transient error in processor or memory
    4. fault in communication system(Network)
    5. host crash during the process
3. Non end-to-end solution: Duplicate copies, unconomicial
4. End-to-End check and retry:
    - reduce the total time of retry(compare with non end-to-end solution)
    - The checksum is kind of must-do section due to the fault 1,2,3,5, even the network is fully garuantee
5. The low-level reliability is still improtant

    - the fault rate of a file transfer will exponentially increase with the file length increase
    - low-level duplication may hurt performance:
        - this check has a cost
    - it may help performance:
        - reduce the number of retransmissions?
6. But the low-level does not need to provide "perfect" reliability
7. In real-world problem:(Guildline)
    - Reliability measure is an engineering trade-off
    - The end-to-end check is not a need according to the reliable of network
    - **Better performance enhancement can be achieved at the high level**
    - **Every application need to pay for low-level function in a system, even it does not necessary need it**, so do some thing in low-level may not be cost efficiently.
    - Improvement of checksum is periodically compute and report the error earlier, however it seem to be a case to case function.
8. Defining the "End"(for WhatsApp)
    - left sending phone vs. device received vs. message read

## Other example

1. Delivery Guarantees
    - Why RFNM in ARPANET is not good?
    - End-To-End acknowledge is a need in some spectial case
2. Secure Transmission of Data(Encryption)
    - Why not in low-level?
        - securely manage the key
        - data is vulnerable between the interface of network and the application
        - authenticity still be checked by application
    - Why in Application level? More safety.
    - Lower-level encrption: Work as a firewall and can use a common key
3. Duplicate Message Suppression:
    - Why?
        - some duplicate message is originated by application itself rather than the network fault, so it can only recognize by application
        - not need to do again in lower-level
4. FIFO Delivery(Ordered Delivery):
    - Only independent mechanish at High-level can deal with some FIFO request
5. Transaction Management(SWALLOW SYSTEM, a remote data storage system):
    - not need to suppress duplicate message
    - not need to provide delivery acknowledgment, significant reduce the host and network load
6. WhaysApps:
    - dilivery confirmation:
        - multiple confirmations
        - when should you get the notificaiton?

        - when the message is read is more end-to-end
7. Online Shopping:
    - confirmation the order went through
    - payment accepted
    - shipped,arrived?

## Conclusion and history

1. How to use end-to-end method is based on the need of application
2. End-to-End arguments is really similar to :
    - RISC
    - kernelization OS
    - Occam's razor
3. End-to-End argument may be part of the layered system
4. Why End-to-End Now?
    - smarter end devices enable end-to-end processing

##Note For Saltzer82a

## Overview

1. A perspective on the subject of names in network.
2. Distinguish 4 kinds of destination
3. Use the binding(in Operating System Concept) to describe the relation among 4 destinations:
4. Real-World confusing properties
5. key Ideas:
    - How do we identify things we talk about in the network
    - Whay do we need to identify(not so important now.. beasue we identify lots and lots of thing)
    - part of naming

## Problem and Terminology

1. Problem:
    1. Association between objects and names is too tight.
    2. Too few well defined concepts at hand.
2. The concept of OS: Binding
    - A name is what you want
        - why use name?
        - to identify things
        - want different name for different
        - easy to remember
        - things can change(same name can map to the current place)
    - An address is where it is
    - An route is a way to get there
    - Binding: mapping name to an address

- context(by Neuman,1989): background about a binding

## Listing types of destination and binding:

1. 4 kinds of destinantion:
    - Service
    - Node
    - Network Attachment
    - Path(Route)
2. 2 obeservations: form and binding
3. Form Part:
    - More than 1 form of name for single type of objects
    - "name" broadly encompasses both forms(binary-digital and character)
    - not to associate some form with particular types of objects
4. Binding Part:
    - 3 kind of bindings:
        - Service to Node
        - Node to Attachment Point
        - Attachment Point to Route
    - Why binding?
        - need to resolve a name to an address
            - acctully, a string can both be a name or an address based on different context
        - for secure, give trust to go to the right place
        - Preserving identity when moving
    - Seend a data to a service need to discover 3 binding:
        - **Service name resolution**: a node run the service
        - **Node Name Location:** a attachment point that node is connected
        - **Route Service**: a path to the attachment point
5. The final binding choice may recorded outside the binding services within the network
6. What exactly one record means and which part of record can be changed

## Real-World Examples:

1. Ethernet: node and attachment point have the same name
    - pro:
        - node can physically move without any change in network record
        - one level of binding talbes is omitted
        - In two Ethernet, one node can present the same attachment point name
    - con:
        - one node have 2 attachment points and need to direct a message to one rather than the other
        - harder to accomplish
2. ARPANET: confusing about the name of node with name of service(due to mnemonics)
3. 3 conceptually distinct binding services may not be mechanically distinct:

- name service: input service name and return list of network attachment points(Combind binding 1st and 2nd)

## Correspondence with Name, Address and Routes

1. Any of the four kinds of object may have a name
2. The address of an obejct is a name of the object it is bound to
3. A route is a more sophisticated concept include a path to the attachment point and some identificaiton of which activity within that node

# Note for Clark88a

## Overview

1. Capture some early reasons which shaped the Internet protocols:
    - the idea of datagram(In Section 9)
    - layers seperation for TCP and IP(In Section 5)
    - original objectives of the Internet architecture(In Section 2)
    - the relation between goals and feature(In Section 3-7)
2. Some Basic Idea of Internet:
    - Datagram(In Section 9)
    - TCP(In Section 10)
    - Flow(Future Building Block, In Section 11)

## Fundamental Goal of Internet

1. effective technique for multiplexed utilization of existing interconnected network
2. Why package switching? Why no other technique, likes circuit switching?
3. benefits of using circuit switch for phone:
    - each circuit is not interfered with:
        - you get your bits and fixed latency
        - won't get terminated
    - phone companies can engineer capacity
    - get cheap handsets
    - easy for accounting
4. Packet Switching(Internet):
    - something worse than circuit:
        - more overhead in each packet headers
        - packets can be out of order
        - queueing can delay you, can it can vary(**jitter**)
        - can be out of capacity(in the middle of an interaction)
    - advantage:
        - statistical multiplexing:

        - we let people in, interleave their traffic, depend on statistics and idle time to make sure we're under capacity
        - implications: interference and possible overcapacity, congestion management
5. The fundamental structure:
    - A packet switched communications(gateways) which implement a store and forward packet forwarding algorithm

## The Goal and related feature

7 goals which were established for Internet:

1. Survivability:
    - What kinds of failures: host, link, routers, line noise
    - solution: fate-sharing approach, only check at the endpoint
    - pro: protect anainst any number of intermediate failure, easier to engineer(for network)
    - con: hard to engineer(for host), not cost effective for retransmission
    - other solution: checksums, redundancy
2. Multi-type of service:
    - Why?
        - different appilications need different services
        - different services have different requirments
            - quick startup
            - steady rate
            - real time
    - solution: IP and TCP layers seperation
    - use datagram to be the building block of services
3. Varieties of networks:
    - minimum set of assumptions:
        - reasonable size
        - reasonable reliability
        - suitable form of addressing
4. Other 4 Goals(Not that important)
    - Distributed management: by exchange routing tables
    - Effitive: Not solve but reasonable not that bad
    - Attaching of new host: Become harder because of the solution of survivabilitya
        - in 1980s: implement the protocols
        - 10 years ago: configuring addresses
        - today: authentication
    - Accountability: not that important in 1980s but important today

## Architecture and Implementation

1. The architecture tried hard not to constrain the range of service
2. Hard to guide the designer of realization of Internet:

    - Lack performance measure(Only on logic correct)
    - Lack Simulator
3. Whay is Architecture and Implementation:
    - Architecture: the components and protocols of the net(see the RFCs)
    - Implementation: consider real-world constaint(like memory, processor)
4. realization: an instance of the Internet class, different requirement for different class

## Datagram

1. Why datagram?
    - eliminate the need for connection state
    - a basic building block for all kinds of services
2. Using datagram **Does Not** equal to high level service should be datagram, usually it is more complex

## TCP

1. Some of the early reasoning for the parts of TCP:
    - flow control by bytes:
        - original ARPANET: flow control based on both bytes and packets
        - TCP only use bytes:
            - permit the insertion of control into sequence
            - permit TCP packet to broken up
            - permit gathered small packet together
        - However,
            - acknowledged the control as well as data make the idea complex in practice
            - packet borken was moved to IP, and this make hard for IP protocol
            - The small packet problem may not happen if control by packet.
        - Maybe the original ARPANET approach is better.
    - EOL(End-Of-Letter flag):
        - the data up to this point in the stream was one or more complete application-level elements
        - EOL should be a tool for mappint the byte stream to the buffer management of the host
2. Other protocols (than TCP/UDP/IP):
    1. ICMP:
        - why not TCP? don't want retranmissions
        - want it to be simple
    2. QUIC:
        - TCP replacement running over UDP, originally from google

## Conclusion and Future Network

1. Resource Management(QoS) and Accountability are difficult to achieve in context of datagrams
2. A better building block for Internet:
    - flow
    - gateway should have flow state to remember which flow are passing through them
    - flow state information would not be critical with service

- the service would be enforced by end point

## NABC

1. hook: what went into this "Internet" thing
2. need: understanding the Internet's history can help you improve it
3. approach: what are the design principles behind the Internet
4. benefits: when you build new things over the Internet, you can do them more efficiently
5. challenges:there were competitors to the Internet: ISO network stack, DECnet, ATM
6. close: whay should we do next to the Internet: resource management(QoS)

# Note For Clark02a

## Overview

1. internet landscape now
2. technical design principles for tussle environment of network
3. tussle space and related design
4. revisiting of some old network principles

## Internet Landscape

1. parts of the Internet: Users, ISP, Private sector network providers, Governments, Intellectual property right holders, Providers of content and higher level services
2. tussles span a broad scope

## Design Principle

1. Design for variation in outcome
2. Modularize:
    - Functions that are within a tussle space should be logically separated from functions outside of that space
3. Design for choice:
    - Protocols must permit all the parties to express choice.
4. Implications:
    - Open interface
    - Tussles often happen across interface
    - It matters if the **consequence** of choice is visible(The choice could be secret or not)
    - Tussles have different flavors
    - Tussles evolve over time
    - There is no such thing as **value-neutral design**.
    - You design the playing field, not the outcome.

## Tussle Space

1. Economics:
    - Our principle that one should design choice into mechanism is the building block of competition.

    - Provider lock-in from IP addressing
        - Tussle between addresses reflect topology to support efficient routing and desire of the customer to change providers easily
        - Address should reflect connectivity, not identity, to modularize tussle
    - Value pricing
        - due to there is no value-neutral design
        - User want to sidestep providers restriction of internet used.
    - Residential broadband access
        - the ISP in broadband genneration must be far less than the ISP in dialup gen
        - modularized along tussle space boundaries
    - Competitive wide area access
        - user want to choose which long distance provider to use
2. Trust:
    - The implies of firewall is opposite the basic idea of Internet(which is transparent packet carriage)
    - The role of identity and anonymously
3. Openness:(open interface and Vertical integration)

## Old principle

1. End to end arguments:
    - Still right for innovation and reliability but need a more complex articulation
    - Peeking is irresistible
2. Separation of policy and mechanism:
    - There is no value-neutral design
    - User Empowerment(User have the right to choose)
    - Doing nothing more

## Conclusion

1. Design a new enhancement for the Internet should analyze the tussles that it will trigger.
2. We(technical Designers) should not try to deny the reality of the tussle, but instead recognize our power to shape it

# Week 3

1. P8. [Caesar05a] Matthew Caesar and Jennifer Rexford. BGP routing policies in ISP networks. IEEE Network Magazine, 19(6):5–11, November 2005. (Caesar05a.pdf)
2. P9. [Shaikh00a] Aman Shaikh, Lampros Kalampoukas, Rohit Dube, and Anujan Varma. Routing stability in congested networks: Experimentation and analysis. In Proceedings of the ACM SIGCOMM Conference, pages 163–174, Stockholm, Sweeden, August 2000. ACM. (Shaikh00a.pdf)
3. P10. [Goldberg14a] Sharon Goldberg. Why is it taking so long to secure Internet routing? Communications of the ACM, 57(10):56–63, October 2014. (Goldberg14a.pdf)

# Note For Shaikh00a

## Overview

1. Background: former paper about congested network
    - Most of the studies assume reliable and loss-free delivery of routing control message
    - Losses of the routing protocol message lead to route flaps and instabilities
    - This paper is first attemp to analyze the dynamics of routing protocols
2. Build an analytical model of congest network
3. Analyze the model using markov chain
4. Experimental results and compare with section 3
5. Summary
6. Key Idea:
    - Evaluate OSPF and BGP under congestion
        - can fail unexpectedly under congestion
            - problem: control traffic no isolated from data traffic
        - two different protocols used in different places
        - they use different transport protocol: TCP and (UDP)
    - method:
        - emulate the network in testbeds(for experiments)
        - model the failure and recovery of the protocols
        - compare the results from models and the testbeds

## Model Setup

1. Network model:
    - 2-nodes model(source router to receiver router)
        - after failure, traffic goes over same link
    - 3-nodes model(contain 2 paths,a two-hop path(shorter path) and a three-hop path(secondary path))
        - after failure, traffic goes over the other link
    - congested **only** in **the source router's output**
2. Variable:
    - Independent variable:
        - **Overload Factor**
        - Buffer memory(only in 3-nodes model)
        - Buffer drop protocol(Drop-from-Front and Drop-tail, relative to **queueing delay**)
        - **Router Protocol: OSPF and BGP(TCP)**
        - ~~Package Size~~(Final it prove to be **Unrelated** variables)
    - Control variable:
        - Queueeing and Scheduling: FIFO
    - Result:
        - U2D time, the time it takes for a routing flap
        - D2U time, the time it takes for recovering

## Analyze the model

1. Assumptions:

- The overload factor remains constant
- Every packet has the same probability p of being dropped

### OSPF model

1. How to occur a route failure(~~flap~~):
    - source will provide a hello mesage in TH1 (HelloInterval,10s in paper and plus 1s as error rate)
    - receiver will acknowledge hello message using a timer in TRD1(RouterDeadInterval, 40s in paper without error rate)
    - if the interval between 2 hello message more than TRD1 then the rout flap happen
    - note that OSPF control messages are sending using IP datagrams directly
2. Using a 5 states Markov chain(Figure 2) to describe the U2D cycle
3. The Eq. (1) show the expected flap time in this Markov chain
4. How to recover:
    - receive a hello message
5. D2U time is easy to calculate in OSPF protocol, it is **TH1/(1-p)**

### BGP model

1. It's **hard** to deriving an accurate model for BGP, because BGP use TCP to transmit and retransmit the routing message
    - TCP is adaptive about the retransmit time
    - the number of states of Markov clain is equal to the time of retransmit
2. Figure 3 show the Markov chain for U2D cycle of BGP and Eq (4) show the expected flap time based on that:
    - the variable n is based on RTO and RTT in TCP protocol, which is overestimating in the paper
3. The D2U model in BGP need to concerned the direction from client to service or opposite:
    - It need to build the TCP first and then the BGP
    - Building TCP need 2 message sending from client to service and 1 in opposite direction
    - The Markov chain is completely different in different direction, Figure 5 and 6 show the different chains
4. The U2D situation:
    - send KEEPALIVE message
        - every 60s
    - have keepalive timer if it expires declare link down
        - timeout in 180s

## Experimental Results

1. Packet size is unrelated
2. Buffer size is unrelated to 2 nodes model
3. RTT is overestimate so the flap time is overestimate
4. BGP takes longer to fail and OSPF takes faster to recover:
    - BGP retry faster due to RTT which is << 10s
    - however TCP need to setup

##Note For Gao01b

# Overview

1. overview of interdomain routing
2. different AS relationship:
3. BGP Exporting potocol for each kind of relationship
4. Routing table entry patterns
5. An algorithm for inferring AS relationship
6. Result of the algorithm and verification

# Interdomain Routing Overview:

1. Connectivity does one imply reachability
    - Physical connectivity at the IXP does not necessarily imply that every pair of ASs exchanges traffic with each other
2. Several Tiers of ISP, the relation between ISP is different due to its tier
3. The degree of an AS can be a good heuristic in determining the size(tier) of the AS
4. BGP policies:
    - import policies
    - export policies
    - loop-avoidance rule
    - AS prepending: appends its AS number twice before export

# AS Relationships, export policies and entry patterns

1. 3 Relationship:
    - customer-provider(directed)
    - peering(undirected)
    - sibling(undirected)
2. Export Policies:
    - To Provider and Peer: its routes and its customer routes
    - To Customer and Sibling: its routes and its customer routes, its provider or peer routes
    - Peer and Sibling is symmetric but Provider-Customer isn't.
3. Entry patterns:
    - All Entry should be Vally-Free(Using a customer to transit datagram)
    - Downhill Path: A path only include provider-to-customer and sibling-to-sibling edge
    - Uphill Path: A path only include customer-to-provider and sibling-to-sibling edge
    - An Entry should be:
        - A Maximal uphill path and the maximal downhill path
        - A Maximal uphill path, a peer-to-peer path and the maximal downhill path
    - The different between sibling path and peer-to-peer path;
        - The peer to peer path only exist in top level transit

# Algorithm to infer relationship

1. Find the top level AS: the AS have largest degree
2. Distinct provider-customer and sibling-sibling relationship:
    - For Connected Pairs(u,v), if u provide transit to v and v provide transit to u, then they are sibling-sibling relationship otherwise they are provider-customer relationship
    - A better edition consider misconfigured of router:
        - count the number of entry that conclude transit relationship
        - only use the relation that appear more than L time, L is a constant number need to be fine-tuned
3. Distinct peer-to-peer relationship
    - According peer-to-peer only happen in top level AS, delete all relationship that isn't top level
    - A better edition: Delete the relationship which one AS has far less degree than the other

## Verification and Conclusion

1. Many kinds of misconfiguration
    - Router Configuration Typo: Loop existed in Entry
    - Misconfiguration of small ISPs: provide its provider to another provider, so it may be the transit between them
    - Unusual AS relationship
    - Inaccuracy of the heuristic: some top level AS have relatively small degree

# Note For Caesar05a

## Overview

1. Important process in BGP
    1. Exchanging Routing State Process
    2. Decision process in BGP
    3. Advertisement process in BGP
2. Common policies using BGP:
    - local policy
    - bussiness
    - traffic balance
    - scalability
    - security
3. Some more policy in developping
4. key ideas:
    - evaluation of routing between ISPs:
        - between ASes(AS: routers that share the same BGP routing policy)
    - BGP protocol
        - the EGP we used
    - Whay can BGP do for us?
        - cost effective
        - shortest path

        - flexibility for traffic engineering and business relationship
    - We care about the BGP mechanism(route attributes)

# Important process

## Exchanging Routing State Process(Overview)

1. overview:
    1. Border routers exchange routes between neighboring ASes
    2. Border routers distribute routes to internal routers
2. BGP salient feature:
    - incremental protocol
    - path vector protocol
    - advertised at the prefix level
    - messages contain several fields

## Decision process in BGP

1. Absence of policy: minimum path length
2. Policy: an order list of attributes
    - Attributes classification by source:
        - Locally: LocalPerf, IGP(Internal GP) cost
        - Neighbor: AS path length, MED
        - Neither(some is fixed): origin type, eBGP-learned over iBGP-learned, router ID
    - The order of attributes allows the operator to influence the decision process

## Advertisement process in BGP

1. 3 steps to deal advertisements:
    - import policy
    - decision process
    - export policy

# Common policy for BGP

## Some Terms

1. AS
2. IXP
3. RIB

## What is include in BGP UPDATE message:

1. prefix beging routed
2. with attributees:
    - AS-path
    - MED,LocolPref, etc.

## Local policy for BGP:

1. Preference(with attributes list)
2. Filtering: both before(inbound) and after(outbound) preference
3. Tagging: using community attribute(defind and use only by ISP itself)

### How business influence BGP policy:

1. 3 types of relationship:
    - peer-to-peer (usually without money)
    - provider-to-customer (with money)
    - backup
2. In decision process: Set different LocalPref for different relationship so it can make more money
3. In export policy:
    - Use community attribute to mark different relation and export different tables to different relationships
    - Usually need to delete the community attribute when export

### Traffic Engineering(Load balance):

1. Outbound control:
    - early exit routing
    - reduce congestion on outbound links to neighbors(By adjusting LocalPref, alway need manually operate)
2. Inbound control:
    - Goal: Local ISP to influence route selection in remote ISPs(By changing MED)
    - Prepend multiple copies of AS number to artificially increase the path length
3. Remote control:
    - Sharing community attributes(high risk if these community attributes have different meaning in different AS)

### Scalability

1. Limiting Routing Table Size(For limited memory)
2. Limit the Number of Routing Change(By invoking flap damping, add penalty when routing is changing)

### Security

1. Discarding Invalid Routes(By sanity checks in import policy)
2. Protecting Integrity of Routing Policy(Delete some attributes in import policy)
3. Securing the Network Infrastructure(Filtering some important IP in export policy)
4. Blocking DOS Attack(Using black hole router)

## Developping policy

1. Configuration Checking
2. Language Design(RPSL)
3. New architectures

## Internal a ISP

1. Why OSPF?
    - OSPF react quickly(internal network changes)
    - OSPF reroutes around changes quickly
2. Why iBGP(Interal BGP between routers in AS)
    - propagate policies across out network

# Note For Goldberg14a

## Overview

1. Common BGP policy
2. Several Attacks types on BGP
    - prefix hijacking
    - route leaks
3. Defense Methods For Attacks:
    - prefix filtering
    - RPKI
    - BGPSEC
4. Sometimes fixes(Defense method) need to be deployed widely
    - by people who don't directly benefit

## Common BGP policy

1. IP prefixes
2. Longest-prefix-match Routing
3. Autonomous System
4. Business relationships and Routing policies
    - customer-provider relationship
    - settlement-free peering relationship
    - AS always avoid forwarding if it cannot generate revenue
    - rule of thumb: AS a will announce a route to AS b only if
        1. b is a customer of a
        2. the route is for a prefix originated by a
        3. the route is through a customer of a

## Attacking method on BGP

### Hijacks(2 types of hijack)

- prefix hijacks
    - hijack the entire prefix
    - split the network traffic, people choose one route or other
- subprefix hijacks:
    - based on Longest-prefix-match Routing policy

    - hijack all traffic to subprefix
- detecting hijacks attack

### Route Leaks

- providing announce violating the rule of thumb
    - maybe from customer to provider
    - maybe to its peers
- based on AS can generate more revenue by forwarding traffic through its customer
- the announcer get a lot more traffic than they expect

### Impact of routing incidents

- blackholes: drop all traffic enter the router
- interception:
    - interception is invisible to end users
    - the bogus AS will transit the traffic back to the legitimate origin AS

## Defenses method

### Prefix filtering

1. using whitelisting technique to filter the announces from each of its customers
2. benefit:
    - simple and effective
    - make sure wrong people don't announce stuff
3. challenge:
    - works only on customer links, hard for peers
    - Lopsided incentives: the AS other than the provider itself can do nothing using this method

### RPKI

1. RPKI: provide a trusted mapping table from IP prefixes to original AS
    - central authority gives each AS either prefixes and public key
    - ASes sign their prefixes and announce that(a ROA: Route Origin Authorization)
2. benefit:
    - Offline cryptography
    - Protection from hijacks
    - Effective incentives
    - prevent other ASes from announcing a given prefix
3. challenge:
    - RPKI takedowns and misconfigurations: attacker may attack RPKI itself
    - RPKI can be circumvented(Including route leak and path-shortening attack)
    - Need to trust central authority(RIRs)

### BGPSEC(Path validation):

1. path validation based on BGPSEC signature:

    - the prefix and AS-level path
    - the AS number of the AS **receiving** the BGPSEC message (the message provider need to signed its receiver)
    - all the signed messages received from the previous ASes on the path
2. benefit:
    - No path-shortening attacks
    - all BGP steps are verified
3. challenge:
    - not clear this protects against route leaks
    - online cryptography: high computational overhead
    - The transition to BGPSEC: The method is useful only if **all** ASes on the path have applied BGPSEC
    - ASes have deployed BGPSEC can still suffer from **protocol downgrade attacks** if some ASes don't use BGPSEC

# Week 4

1. P11. [Casado09a] Martin Casado, Michael J. Freedman, Justin Pettit, Jianying Luo, Natasha Gude, Nick McKeown, and Scott Shenker. Rethinking enterprise network control. ACM/IEEE Transactions on Networking, 17(4):1270–1283, August 2009. (Casado09a.pdf)
2. P12. [Dhamdhere18a] Amogh Dhamdhere, David D. Clark, Alexander Gamero-Garrido, Matthew Luckie, Ricky K. P. Mok, Gautam Akiwate, Kabir Gogia, Vaibhav Bajpai, Alex C. Snoeren, and kc claffy. Inferring persistent interdomain congestion. In Proceedings of the ACM SIGCOMM Conference, pages 1–15, Budapest, Hungary, August 2018. ACM. (Dhamdhere18a.pdf)
3. P13. [Quan13c] Lin Quan, John Heidemann, and Yuri Pradkin. Trinocular: Understanding Internet reliability through adaptive probing. In Proceedings of the ACM SIGCOMM Conference, pages 255–266, Hong Kong, China, August 2013. ACM. (Quan13c.pdf)
4. P14. [Oliveira08a] Ricardo V. Oliveira, Dan Pei, Walter Willinger, Beichuan Zhang, and Lixia Zhang. In search of the elusive ground truth: the Internet's AS-level connectivity structure. In Proceedings of the ACM SIGMETRICS, pages 217–228. ACM, June 2008. (Oliveira08a.pdf)

# Note For Oliveira08a

## NABC Overview

- Need: Reveal the different between real AS-level Internet topology(ground truth) and the public view inferred by system log, BGP data.
- Approach: Using all kinds of data to inffer the Internet topology and compare with the ground truth in different tiers of ASes.
- Benefit: Finding that the inferred maps have a huge percentages error compare with the ground trup and argue a new approach to generate Internet map.
- Challenge: Inferring Networking topology have fundamental limitation and the lack of monitor in low level AS(stub networks) will make it impossible to infer all connections in AS-level.

## Key Ideas

- Compare with other paper:
    - this paper talks about completeness of prior AS graphs(how much data is missing)

    - validation the prior AS-level graph
    - actual ground ttruth is usually propritary
- are there any patterns in what is missing?
    - peering links can be hard to find
- Data Sources:
    - BGP: has AS paths, public information, could see hijacking attempts
    - traceroute
    - IRR: official, might be out of date and incompletely
    - Router configuration files: operational, can have old, stale information

# Background Information(Section 2)

1. Relationship ship between inter-domain connectivits
2. No-valley-and-prefer-customer policy
3. Definition of **ground true**: Complete set of AS links
4. **Hidden Links** in network map: a connection has not yet been observed but will be revealed in long time period
5. **Invisible Links**: a connection will never be revealed due to no-valley policy
6. Several open source **data set** about Internet topology

# Case Study in different tier of AS

## Tier1 Network

1. Tier-1 AS's links are covered completely by public view
2. Tier-1 AS's links are covered completely by a single peer
3. Tier-1 AS's links are covered completely by a single customer
4. Other important thing:
    - Method to remove some outdated connection information

## Tier2 Network

1. Tier-2 AS's links are **not** covered completely by a single peer, all the missing links are peer-peer links
2. Tier-2 AS's links are covered completely by a single customer
3. Tier-2 AS's links are covered completely by public view, monitor by its customer or customer's customer, and the missing links are prefixes longer than /24

## Abilene and Geant

1. These two are academic used network in US and EURO
2. The public view is almost the same as the ground true
3. The different is from the passive monitoring session and IPV6 session(the paper only focus on IPV4)

## Content Provider

1. Content Provider's most links are public peer-peer link in IXPs
2. These links are almost **Invisible Links** so the public view is more than 85% miss, even include the IRR data

### Stub Network

1. The edge network of Internet, its degree is very small
2. The public view is almost the same as the ground true if the time period is long enough
3. The problem is almost no stub network have its monitor

## Conclusion

1. Tier1 network connection can be well inferred
2. Customer-Provider links can be well inferred
3. Peer-peer links can be high potentially misses in public view
4. The author argue a new heuristic approach to build the Internet maps by checking if two ASes both have routers in the same IXP(co-location) in another paper.

# Note For Casado09a

## NABC Overview

- Need: How could we make enterprise network more manageable.
- Approach: Ethane architecture which enforce a single network policy by a central computer(Controller)
    - Binding a packet and its origin(user, host, port, Switchs port, etc.)
    - All decision is made by a centralized Controller(PC)
- Benefits: The architecure is more manageable than other in these fields:
    - Indentify network problem based on journaling registrations and bindings
    - Straightforward to add new features to the network
    - very few Controllers can scale to support a large network
    - Switches in this architecture is very simple
    - Ethane can be incrementally deployed without any host modification
- Challenge: Some shortcome due to Ethane is base on end-to-end argument in application layer
    - Broadcast and Service Discovery(ARP, OSPF, etc.) still constituted most flow
    - The Ethane is in application layer, so it cannot prevent from lower layer(likes IP)
    - Still miss knowledge about user, so it is hard to detect which application is using the port
    - Spoofing Ethernet address in a no-Ethane-only network
- Other contribution in this paper:
    - FSL(Flow-based Security Language) to describe network policy

## Architecture Design:

1. Naming and Binding
    - switch port to machine(MAC)
    - machine(MAC) to IP
    - User to machine(MAC)
    - must know who sends packets and when they start a new route
2. Policy Language(FSL)
    - high-level policy
3. Controller:

    - Register and Track each names and binding
    - Journal all bindings and flow-entries in log, for diagnose network fault
    - Bootstrapping: Create and Maintaining a shortest path tree for switches
    - Authentication: allow or deny user's action in network
    - Setup Flow in the network: add new flow entry and policy to switch(only the first packet need to send from switch to Controller, and the subsequence can use the established entry)
4. Switch:
    - Much simple than origin switch
    - Switches setup: Send register message to Controller
    - Forwarding a packet:
        - If it has a entry, forward or deny as the entry
        - If it has not a entry, send to the Controller and add a new entry
        - The entry is per flow and per switch
        - The entry will be cancel when time is out or revoked by Controller
    - recover from failure:
        - contect the control for everything like new flows to rebuild the flow table
5. Replicating the Controller for Fault-Tolerance and Scalability
6. Recover the network from link failures
7. FSL language:
    - A lookup table of rules
    - Each entry consisting of a condition and a action
    - condition include the predicate function and arguments, also include boolean predicate
    - action include standard set and also C++ or Python function
    - 2 ways to deal with rules conflicts:
        - priorities ordering
        - apply least restrictive action
8. Concerns about a Central Controller
    - performance
    - what happens if it fails? backup-controller, revert back to a distributed netwrok, recover from cold-start all the flow
    - who has access to the controller: change policy, DOS attack, the metadata of network

## Compare to Clark88a

1. In 1988 the accountability was the bottom but in 2007 big focus on accountability
2. In Clark's paper show decentralized control, in this paper provide centralized control inside enterprises
3. In 1988 each user trust other, but now it loss these trust

# Note For Quan13c

## NABC Overview

- Need: Find a method to monitor the network outage in global Internet.

- Approach: A method, called Trinocular, which uses active probes to detect the disconnected and inferred the outage by Bayesian inference.
- Benefit: This method have higher coverage rate and can affordable traffic.
- Challenge: The parameter in the model is fixed by historical data which can be adaptive by real time measurement

# Some Detail about Trinocular Method

## Responses for Ping

1. positive reply: some computer is live at the IP address you're pinging
2. err(negative reply)
    1. if you have an error in your ECHO REQUEST
    2. you get a HOST UNREACHABLE from the router
3. no reply
    1. no one's there
    2. firewall and they don't want to reply
    3. query or reply was dropped
    4. when there's a network failure (**We really care about this**)

## Simple Outage-Centric Bayesian Model of Internet

1. b denote each /24 block
2. E(b) denote the ever active addresses in block b
3. A(E(b)) denote the expected response rate of the address in E(b)
4. Note that:
    - E(b) and A(E(b)) are independence
    - the method only send probes to address in E(b)
5. For a block which is up(reachable):
    - the rate to receive a positive probe: A(E(b))
    - the rate to receive a negative probe: 1-A(E(b))
6. For a block which is down(unreachable):
    - the rate to receive a positive probe: (1-l)/|b|
    - the rate to receive a negative probe: 1-(1-l)/|b|
    - l, denote the combination of probe and reply loss
    - |b|, denote the block size
    - Receive a positive reply while a block is down, only happen if a single router is up but all addresses behind the router is down
7. Update the Inferred State based on the new probe reply by Bayesian inference:
    - B(U), denote the current belief of the block is up
    - B'(U), denote the next belief of the block is up
    - B(U'),denote the current belief of the block is down
    - When receive a positive reply:
        - B'(U)=B'(U|Possitive)=P(possive&U)/P(possitive)=(P(possitive|U)B(U))/((P(possitive|U)B(U))+(P(possitive|U')B(U')))

8. Multiple Observers:
    - distinguish between outages near the VP vs. near the target

### The Probe rate and Traffic Analyze

1. Periodic probing: every 11 minutes
2. Adaptive probing:
    - If belief is more than 90% or less than 10%, we believe the state is determinded
    - If the state is uncertain, we need to probe more adaptive probe
    - Send new probe as soon as the last probe is resolved, until the we reach a conclusive belief
    - Maximum of 15 probes will be send, if it still uncertain then we mark this block uncertain
3. Recovery probing:
    - If the A(E(b)) is low in some block, the down-to-up recovery may be false negative due to some unoccupied address
    - We send 15 probe to decrease the false negative rate
4. total traffic: about 0.7% of total background package in the analyzable blocks
5. Coverage rate:
    - We delete the block with very low E(b) and very low A(E(b))
    - 24% of routed space is analyzable in this method
6. probe from multiple location:
    - parallelize the work
    - different locations can access different address

### Validation

1. The correctness: for block A(E(b))>0.3 and outage duration longer than 1 round will always be detect
2. The timing of duration is lated with a uniform distribution between 0 and 1 round
3. In reanalyzing historical Data:
    - Revealing several outages cause by human or nature

## Compare with other method

1. Control-plane is indirect and provide incomplete coverage of all outage types
2. Client-support Analysis allows better fault diagnosis but fewer coverage
3. Passive Data Analysis which is needed to mind "low-quality" data source.
4. Data-plane:
    - which address to choose:
        - .1 address and hitlist address, recall rate is relatively lower because some .1 address is unoccupied thus other address in the block is reachable
        - all address, the information per probe is too low and the traffic is unaffordable
    - Choosing blocks or prefiex:
        - using prefix have large coverage, but some prefixs are too sparse so prefix will overstates outages

# Note For Dhamdhere18a

## NABC Overview

- Need: Discussions of persistent interdomain congestion at IXP without direct access to them
- Approach: A third-party monitor system for interconnectiong congestion
- Benefit: This system is well inferring the congest in interdomain links and validation with several data
- Challenge: The router configuration, asmmetric routes will lead to wrong inference and the cause of congestion still unreveal in this paper

## Key Ideas:

1. probing remote inter-domain links
    - want to prove if there is persistent congestion
    - so we can fix the problem
2. goal: discover persistent congestion
3. mechanisms
4. validating the results

## Detail about inference and validation method

1. TSLP(Time-Series Latency Probing) Method:
    - sending ICMP probes to both the near and far ends of an identified interdomain link
    - measuring the latency between probe and reply
    - a possible cause of the increased latency is congestion
2. bdrmap Method:
    - bdrmap Method is used to infer the interdomain links between the host network and the neighbor networks
3. Level-shift Method:
    - the algorithm detects level-shifts of duration at least l/2.
    - l denotes cut-off length of time bin(5 minutes in paper)
4. Autocorrelation Method:
    - Find multi-day repetition of elevated delays at the same times of day
    - The time window in the paper is 50 days
5. Validation Method:
    - **Packet Loss** is used to validate the congest inferred by the system, used when at least one episode of congestion has been inferred by one of the two method
    - **Throughput** is provide by NDT, and used to validate the congest
    - **YouTube streaming** is using YouTube-Test Tool to measure ON-period throughput, Start-up delay and Streaming failure rate in congest and uncongest period.
    - Also validate by operator in ISP

## Conclusion

1. Several validation datas show the system can well inferred the congest in interdomain with monitor in edge

2. Congestion was not widespread on the peer.provider interdomain links
3. Some show significant congestion from AP to Content Provider

# Week 5

1. P15. [Jacobson88a] Van Jacobson. Congestion avoidance and control. In Proceedings of the ACM SIGCOMM Conference, pages 314–329, Stanford, California, USA, August 1988. ACM. (Jacobson88a.pdf)
2. P16. [Ramakrishnan90a] K. K. Ramakrishnan and Raj Jain. A binary feedback scheme for congestion avoidance in computer networks. ACM Transactions on Computer Systems, 8(2):158–181, May 1990. (Ramakrishnan90a.pdf)
3. P17. [Padhye98a] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modelling TCP throughput: A simple model and its empirical validation. In Proceedings of the ACM SIGCOMM Conference, pages 303–314, Vancouver, Canada, September 1998. ACM. (Padhye98a.pdf)
4. P18. [Cardwell17a] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. BBR: Congestion-based congestion control. Communications of the ACM, 60(2):58–66, February 2017. (Cardwell17a.pdf)

# Note For Jacobson88a

## NABC Overview

- Need: Need to avoid and control congestion in Internet
- Approach: Implement several congestion control protocols in TCP protocol
    - slow-start protocol(soft start)
    - RTT mean estimator
    - RTT variation estimator
    - congestion avoidance protocol
- Benefit: Increasing the effective bandwidth in Internet
- Challenge: These approaches were implemented in hosts, other protocols need to implement in Gateway to make the Internet **fairer**

## Key ideas

1. principles: packet conservation, avoid congestion collapse
2. proposes algorithms to avoid and recover from congestion
3. better RTT computation

## TCP protocol detail in Paper

1. 3 ways for package conservation to fail:
    - The connection doesn't get to equilibrium
    - A sender injects a new packet before the old one has exited
    - The equilibrium can't be reached because of resource limits along the path

### Slow Start Protocal

1. Deal with the 1st Failure

2. Add 1 paramater and 3 lines of code to TCP protocol:
    - Add a congestion windows,** cwnd**, to the per-connection state
    - When TCP starting or restarting after a loss, set **cwnd=1**
    - When receive an ACK, increase cwnd by 1
    - When sending send the minimum of receiver's advertised window and cwnd
3. Which seem to be increased slowly(by 1 each time), but acctually it double every round(since you will receive an ACK for every package in the windows), want to get to steady state quickly
4. When to start: at connection start or when we restart after congestion
5. When to end: When we detect congestion OR when we reach steady state(cwnd>ssthesh) (OR we reach the maximum **flow control window size**)
    - ssthread should approximate the bandwidth-delay products
    - ssthread initial value ... based on guess 64K(different constant values in different implementations)

## A cheap RTT estimator(in Appendix A)

1. Deal with the 2nd and 3rd Failure
2. A cheap method to estimate the RTT mean time and the variation we used these parameters:
    - g for mean: a filter gain constant for RTT mean with suggested value of 0.1~0.2
    - g for variation: a filter gain constant for RTT variation with relatively bigger value than g for mean
    - m: the measure time for transmit
    - a: the estimator for RTT mean time
    - Err: Err=m-a, the error between the real time and estimate time
    - v: the estimator for RTT variation time, Here **we use the mean deviation rather than standar deviation**
    - Err-v: the gain part for mean deviation
3. We use the **mean deviation** rather than standar deviation:
    - avoid multiple calculation
    - mdev = 1.25 sdev, in normally distributed
4. So we got the algorithm without scaling:
    - Err = m - a
    - a = a + g*Err
    - v = v + g'*(Err-v)
5. To avoid the float calculation, we need to keep everything integer:
    - g for mean = 0.125 = 1/8
    - g for variation = 0.25 = 1/4
    - Multiply the whole parameters in algorithm by 1/g(a=8*real_a and v=4*real_v)
6. Algorithm detail after scaling:
    - Err = m - (a >> 3)
    - a = a + Err
    - Err for v = Err - (v>>2)
    - v = v + Err
    - rto = (a>>3) + v

## Congestion Avoidance Protocol

1. Deal with 3rd Failure

2. Key Idea:
    - increase the window sizes slowly:
        - If it increase in multiplicative speed will drive the net into saturation and hard to recover
3. Algorithm:
    - On any timeout set swnd to half and current window size
    - On each ACK increase cwnd by 1/cwnd
        - probing for unused capacity
        - The rate increase linearly
    - When sending send the minimum of receiver's advertised window and cwnd
4. The slow-start and congestion avoidance protocol seem to be confused, and they actually independent

### Slow-start and congestion avoidance protocol together

1. Key Idea:
    - Add a new paramater, ssthresh, to record the threshold size
    - Use slow-start to deal with the situation when the loss happen and need to recover
    - Use congestion avoidance to deal with the threshold and window increased after hitting the threshold
2. Recover:
    - ssthread = cwnd/2 , binary search is a good way to converge quickly on the true best rate
    - cwnd =1, give the network some time to clear its full buffers
3. If we want to avoid the pipe empty due to recovering:
    - Fast retransmit, duplicate ACK
    - duplicate ACk imply you missed packet n+1, but get some packet out of order

# Note for Ramakrishnan90a

## NABC Overview

- Need: Network congestion need to be control and the user should be fair
- Approach: Policy implemented in both router and user
    - Router: Set the congestion bit in packet header
    - User: Copy the congestion bit to Ack packet and adjust the window based on congestion bit and policy
- Benefit: The network is more power(higher utilization) and fairness and this algorithm doesn't need to send additional packet
- Challenge: Each user in the network should obey the policy, otherwise this policy will lead to truly unfair
- Key ideas:
    - for connectionless network(like IP)
    - using a binary feedback scheme(and packet loss) and hopefully *avoid* congestion and not just go to packet loss
    - this paper carefully lines up each design option

## Policy Detail

1. Different between congestion control and congestion avoidance:
    - Congestion Control: Let network never go to cliff
    - Congestion Avoidance: Let network on the knee point of throughput-response time(delay)
2. Former Research: Need to send additional packet when the network is already congest

## Overview

1. For Router
    - Congestion Detection: Based on average queue length
    - Feedback filter: deal with network jittera(avoid oscillation)
2. For User
    - Decision frequency
    - Use of receive information: Whether maintain the information after decision
    - Signal filtering: deal with network jitter
    - Decision function: Additive increase and multiplicative decrease

## Optimization Criteria

1. Power:
    - define: Throughput ^ a / Response time
    - if a=1, then power is maximized is the knee of the delay curve
2. Efficient:
    - define: resource power / resource power at knee
3. Fairness: which is fairness index still used today
    - define: square of sum / n * sum of square

## Router Part

1. Using simple thresholding policy or hysteresis policy:
    - The paramater for thresholding
    - The paramater for hysteresis
    - set constant value by experiement
2. Using instantaneous queue size or average queue size:
    - 2 disadvantage for instantaneous queue size:
        - delay for feedback
        - result in unfair since different user receive different signals
    - The paramater for average interval: using adaptive algorithm
        - From last busy regeneration point to now

## User Part

1. instant decision when ACK or use a relatively long period data:
    - instant decision will lead to prematurely alters the window size and causing overcorrection
    - The time period set to Wp+Wc:
        - Wp: is the former windows size, for ensure all preview package is ACK
        - Wc: is now windows size, which is used to make decision

2. How many ACK packet used to decision: Wc
3. How to deal with jitter:
    - use single threshold for average rate of congestion bit
4. How to alter window size:
    - Why not additive increase and decrease:
        - unfairness, the gap between 2 user will preserved due to add or subtraction the same number
    - Choose additive increase and multiplicative decrease:
        - If only interage number used, still lead to unfairness in special case
        - Use float for windows

# Note For Padhye98a

## NABC Overview

- Need: Need to analyze the TCP throughput accurately.
- Approach: A computational model for TCP throughput based on loss rate, RTT, RTO and TCP protocols(Both fast retransmit mechanism and time-out mechanism).
- Benefits: The model provides a more accurate result for TCP throughput than former model over a wider range of loss rate.
- Challenge: Some part of TCP congestion control protocol is not considered, so the correctness of the network in the network with high congestion level needs to be confirmed.
- Key Ideas:
    - estimate steady-state throughput for bulk transfer TCP
    - easier to examine an equation to find behavior in different situation

## Other Information

1. Key Idea: Proposing and validating a model to predict TCP throughput based on TCP window-control protocol, likes triple-duplicate acks and time-outs.
2. Novelty: The former paper and model about throughput analyze only focused on TCP triple-duplicate acks protocol and this paper considered not only the former one but also the time-out protocol which happens more often and affected the total throughput.
3. The methodology the paper used is analysis and experiment. The author proposed the model by analyzing the protocol detail and validating the model by experiment in 37 TCP connection across the US and Europe.
4. The thing this paper can improve:
    - This paper only focuses on the part of the TCP protocols, however other protocols, likes slow-start, should be considered in the paper. The time-out event(TO) is relatively higher than triple-duplicate acks(TD) event. The increment of in experiments according to the paper. And the increment of TCP window size after recovering from time-out is much faster than the TD event. So consider slow-start in the model will improve the accuracy of the prediction.
5. Question 1: Last few weeks we talk about congestion in the network, what will happen in this model if this probability is high($p \approx 0.5$)?

6. Question 2: The [Jacobson88a] paper talks about the slow-start protocol in TCP, what will happen if we consider this in the model.
7. Answer 1:
    - When the probability of packet loss is high(p≈0.5), and setting b=1 in the protocol. I found something interested in this model:
    - The expected windows size before fail: E(W)=3
    - The probability that a loss indication ending a TDP is TO: Q≈1, which means almost all the failure is caused by time-out event if p is high
    - The average duration of a time-out sequence is: E[ZTO]=8T0
    - The prediction of throughput in this paper model is: B(p)=\frac{5}{2.5RTT\ +\ 8\ T0}
    - The prediction of throughput in only-TD model is: B(p)=\frac{4}{2.5RTT}, This result is much larger than the result of the article, because it misplaces the cause of the network error.
8. The paper is important or not:
    - This paper is an important paper. It expresses the TCP window management protocol in a computable model and derives the model for calculating TCP throughput. This perdiction model requires only three variables that can be easily obtained. Secondly, this model is easy to expand. The article shows the process of extending from a model considering the only TD to a model considering TD and TO.
9. Relate to [Jacobson88a] which introduce the detail of TCP congestion control and avoidance protocol, and this paper is building the model based on TCP protocol(some detail methods are different between two papers).

# Note for Cardwell17a

## Key Ideas:

1. another congestion control algorithm: defining optimal operating poing
    - In R&J paper, their knee as optimal throughput/delay
    - this paper: range of behavior near knee
        - left side: no queue but full bitrate
        - right side: large queue (before drop) and full bitrate
    - below the knee: application limited
2. estimate: RTT, bottleneck bandwidth
3. from observation in the network
    - avoid packet loss(Not use it as an observation)
    - estimate data delivery rate EstBtlBw
    - effective Round Trip Time( = physical RTT + queueing delay)
4. why model the network now?
    - we think we can do better with different parameters
    - want to do better than waiting for packet loss
    - changes to network: MUCH faster and MUCH bigger buffers
    - end hosts are faster(In Clark88a paper we cannot have a timer for each TCP connection, which is too expensive)

## NABC Overview

- Need: Since the internet change in last 30 years, the loss-based TCP congestion control is not well perfromed. We need to alter the protocol to ensure the Internet running on both maximum throughput and minimum delay.
- Approach: BBR TCP implement, which is a congestion control protocol based on **Round-trip propagation time(RTprop) and Bottleneck bandwidth(BtlBw)**.
    - Algorithm designed to estimate RTprop and BtlBW
    - Maximum Inflight = BtlBW * RTprop
    - Several algorithm to control packet send to ensure the inflight traffic never exceed the maximum
- Benefit: Almost full bandwidth utilization and least delay, compared with CUBIC.
    - BBR does better with large RTT
        - can avoid congestion by not probing until loss
        - without congestion, you don't have to recovery
    - BBR does better with large bitrate
        - multiplicative increase gets to "large" must faster than additive
- Challenge: Still need all sender in a network obey the BBR protocol

## Detail about BBR algorithm

### Ack-arrival half

1. Used to estimate and update the RTprop and BtlBW:
    - RTprop = minimum(RTT), assume there is no queueing in the bottleneck
    - BtlBW = delivered_packet_size / now-delivered_time

### Data Send half

1. Calculate Real-time bdp(Traffic in pipe)
2. Check if bdp exceed the inflight limitation
3. Check if the last packet already send and these sized of other packet already exited the network pipe
4. Send the packet
5. Set the next packet sending time, which is after this packet is completed sending

### Steady-state behavior

1. Realtime monitoring the network change
2. By sending probe in about 2% total time(200ms in every 10s)

### Start-up behavior

1. Just like other TCP implementation, it start from 1 packet and double every time
2. The different, after got the max size it need to drain the buffer cause by start-up stage

### Multiple BBR

1. Using probe and ProbeRTT(rather than RTProp) to realize fairness

# Week 6

1. P19. [Demers89a] Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and simulation of a fair queueing algorithm. In Proceedings of the ACM SIGCOMM Conference, pages 1–12, Austin, Texas, September 1989. ACM. (Demers89a.pdf)
2. P20. [Nichols12a] Kathleen Nichols and Van Jacobson. Controlling queue delay. Communications of the ACM, 55(7):42–50, July 2012. (Nichols12a.pdf)
3. P21. [Katabi02a] Dina Katabi, Mark Handley, and Charlie Rohrs. Congestion control for high bandwidth-delay product networks. In Proceedings of the ACM SIGCOMM Conference, pages 89–102, Pittsburgh, PA, USA, August 2002. ACM. (Katabi02a.pdf)

# Note for Demers89a

## NABC Overview

- Need: The gateway(router) need to provide fair allocation of throughput and delay to all of its users, even there are **some ill-behaved user**
    - **Fair Allocation** Based on** max-min fairness criterion**: (in Page 3)
        1. no user receives more than its request
        2. no other allocation scheme satisfying condition 1 has a higher minimum allocation
        3. codition 2 remains recursively true as we remoce the minimal user and reduce the total resource accordingly
    - **user** means **source-destination pairs** in network
- Approach: A fair queueing algorithm runing on gateway(router) and also simulate and compare with former queueing algorithm(FCFS)
    - The flaw in fromer papers[Nagle95&Nagle97] is lack of consideration of packet lengths
    - This paper's algorithm is implement a packet-by-packet algorithm to emulate the bit-by-bit algorithm(each users share **the same amount of throughput in bit-level**)
    - This algorithm also punished the ill-behavior in the network
    - In comparison part:
        - This paper concerned different kind of congest avoidance algorithm run on source mechine
        - It also concerned if the network existed ill-behavior
- Benefit: The Fair-Queueing(FQ) algorithm significantly increase the fairness in network and decrease the delay for small packet source, who use lower throughput than its fair share
    - This algorithm also punished the ill-behavior which can be saw in *Scenario 3*, which may lead to network fault in all FCFS queueing algorithm
    - In almost all test cases from the paper, FQ have better proformance in the delay of small-packet source
    - The network is fair even it a highly congestion network, but not totally fair due to the generic TCP congestion avoidance algorithm's flaws
- Challenge: The fair doesn't mean equal to every user and it should provide some user higher throughput and the technology to product such an router is difficult at that time

## Key Ideas

1. interaction between queueing at routers and congestion
    - maybe routers can indirectly prevent congestion

    - FQ can control how different flows mix
    - different from TCP fairness control:
        - TCP is in the end host and FQ in the router
        - **FQ enforce fairness**, which is very powerful
2. methodology: start with impossible but understandable model, then convert to realizable but harder model
3. Fair in conversation(aka flow: src-dest addr-port pairs) level

# Note For Katabi02a

## NABC Overview

- Need: This problem needs to be solved that with an increment of bandwidth and delay makes TCP protocol less efficient and unstable, no matter which queueing algorithm used.
- Approach: XCP protocol
    - The Sender part:
        - Add a congestion control header to packet with current cwnd and rtt
    - The Receiver part:
        - Receive the packet and copy the congestion control header to ACK packet
    - The Router part:
        - Decouple fairness control and utilization control(EC)
            - maximize link utilization while minimizing **drop rate** and persistent queues
            - EC deals only with the aggregate behavior, it doesn't care about which percents every user get
            - Basically uses Multiplicative-Increase Multiplicative-Decrease law
            - Unlike TCP increase slowly, XCP increase the aggregation throughput to the limited in one RTT
        - For fairness control(FC):
            - Apportion the feedback to individual packets to achieve fairness
            - Allocate increment for all flows the same amount(or based on their price)
            - Allocate decreasement for a flow proportional to its current throughput
            - The Additive-Increase Multiplicative-Decrease law convert the network to fairness(or priority based on their price)
        - Without Keeping per-flow state(Using the packet Header to store the value and control)
- Benefit: XCP outperforms TCP both in conventional and high bandwidth-delay environment
    - Compare XCP with TCP(with different queueing algorithms)
        - The XCP always keep utility near optimal
        - The XCP much less likely to drop any packet
        - The XCP do well in a dynamic environment(convert to stable faster)
        - (no per flow state)
    - XCP can deal with the ill-behavior source and deal with other normal TCP sources
- Challenge: Since XCP already change the header struct of the packet, this protocol needs to be implemented in all intermediate router. Otherwise the packet will not available in the network

## Some Other Idea

1. Challenge in 1990s:

- signalling congestion was hard:
    - everyone has to help
    - backwards compatibility is hard
- Why didn't they just enforce ECN?

2. The problem of TCP:

- 3-way handshake and expensive connection setup(Especially for small packet)
- retransmissions are hard
    - B(p) is a function of 1/RTT
    - high bandwidth-delay links => AIMD is strict and has big changes if you're going fast

3. Why AIMD need shuffle?

- Even if full channel, still need to even out allocation

4. Why not XCP in 1990s?

- router CPU were slow

5. Possitive feedback per flow and negative feedback per (send) rate:

- additive increase per flow, multiplicative decrease per rate
- large pkg => more thorghput => more feedback
- short rtt => more packet => less feedback per packet

# Note For Nichols12a

## NABC Overview

- Need: Since the buffer of the router is relatively large, the increment of queue length significantly increases the delay of the network. A simple, robust queue algorithm that can manage buffer delay without a negative effect on utilization is needed.
    - Some design criterias of the algorithm:
        - Parameterless
        - Treating good and bad queues differently
        - Controling delay
        - Adapting to dynamically changing link rates with no negative impact on utilization(throughput)
        - Simple and Efficient(can implement in different kinds of backend)
- Approach： Design the CoDel algorithm and compare with RED and tail-drop algorithms in the simulation.
    - CoDel Algorithm's Detail:
        - Focus on local **minimum queue**
        - Using **packet-sojourn time** to measure the queue size
        - Keep a single variable of how long the minimum queue has been above or below the **target size**

        - When the **queue size** exceeded **target** for at least **interval**, CoDel start to drop packet until the **queue size less than target or the queue is less than 1 MTU**
        - **Target** and **Interval** are constant in algorithm
- Benefit: From simulation:
    - CoDel provide lower delay than tail-drop algorithm and almost keep the utilization almost 100%
        - Provide higher utilization than RED algorithm in all kind of network.s
        - In dynamic network, CoDel is much better than RED algorithm since it respond to changes quickly.
        - Packet drop is more fair than RED, according to Jain fairness index.
        - CoDel algorithm can also implement in Consumer Edge
        - CoDel fit different implementation TCP in source mechine
- Challenge: CoDel cannot work very well in Cable Modem and edge network

## Other Important Information

1. good queue and bad queue(from Page 4)
    - good queue: acting as shock absorbers to convert bursty arrivals into smooth, steady departures
    - bad queue: doing nothing(with throughput) but creating excess delay
2. Jain fairness index(from Page 6)

# Week 7

1. P22. [Bharghavan94a] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang. MACAW: A media access protocol for wireless LANs. In Proceedings of the ACM SIGCOMM Conference, pages 212–225, London, UK, September 1994. ACM. (Bharghavan94a.pdf)
2. P23. [Intanagonwiwat00a] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In Proceedings of the ACM International Conference on Mobile Computing and Networking, pages 56–67, Boston, MA, USA, August 2000. ACM. (Intanagonwiwat00a.pdf)
3. (NOT INCLUDE) P24. [Huang13a] Junxian Huang, Feng Qian, Yihua Guo, Yuanyuan Zhou, Qiang Xu, and Z. Morley Mao. An in-depth study of LTE: Effect of network protocol and application behavior on performance. In Proceedings of the ACM SIGCOMM Conference, pages 363–374, Hong Kong, China, Aug 2013. ACM. (Huang13a.pdf)

# Note For Bharghavan94a

## NABC Overview

- Need: Even though MACA resolve some of the plights in CSMA protocol, there are still some key problems to be solved in the media access protocol for wireless LAN's.
    - The contention is at the receiver, rather than the sender.
    - The congestion is location dependent.
    - The allocation of media access should be fairly.
    - The media access protocol should propagate synchronization information about contention periods.
- Approach: Analy the problem in MACA and modify MACA to MACAW.

    - Blackoff Algorithm:
        - The basic idea of Blackoff: slow down to avoid collisions when you retry
        - The MACA algorithm makes the advantage link completely occupy the network, and the disadvantage link has extremely high Blackoff.
        - Firstly, MACAW let the sender include its Blackoff in RTS. Then receiver copy that number in CTS. Then every sender copy this number to its Blackoff counter. So every sender shares the same Blackoff number.
        - MACAW also modifies the rule for BO which multiplies by 1.5 when increase and minus by 1 when decrease.
        - To deal with no response machine in the network, the Blackoff number for different destination station in one sender should be separated and the same destination station should have the same Blackoff number in the different sender.
    - In multiple streams model, MACAW allocates the bandwidth equally to streams by using a random algorithm.
    - Link-Level ACK:
        - why?
            - could depend on TCP to timeout and retry
            - can be faster: over shorter distance, know loss is due to corruption
    - In Massage sending:
        - Using ACK to replace CTS, when sending TCP and the ACK packet is lost
        - Using DS for a kind of *hidden terminal state*
        - Using RRTS for a kind of *exposed terminal state*
        - These three packet increase the throughput and fairness in the network, however, MACAW cannot solve all kinds of *hidden terminal state* and *exposed terminal state*(an example is listed in Figure 7)
- Benefit: The fairness of the algorithm is much better than MACA. Even the ACK, DS, RRTS packet use part of the network throughput; the throughput is still higher than MACA consider the congestion and the noise.
- Challenge: The MACAW cannot solve all kind of hidden terminal and exposed terminal problem.

# Other Informaiton

## Plight of CSMA(carrier sense) protocol

1. hidden terminal: collisions occur at the receiver
2. exposed terminal: unnecessary collision detection

## MACA protocol

1. The sender sends RTS packet to the receiver
2. If the receiver is IDLE, it replays CTS to the sender
3. After receiving the CTS packet, the sender starts to send DATA
4. If it has not received CTS packet, retransmissions occur after several slot times. The number is calculated by Backoff-counter(BO).
5. The BO doubles for each retransmission and decreases to 1 when success.

## Contention-based vs. token-based(scheduled) based

1. Contention-based
    1. each source fights for a chance to send
    2. could be simpler but easy to be unfair
2. Scheduled-based
    1. Everyone has a designated time to send
    2. easier to be fair
    3. not ideal if sources are moving, have to re-compute schedule
3. MACAW : contention-base, 802.11: both, 4G LTE: scheduled-based
4. Sometime change from contention-based to scheduled-based because it need to guarantee the receiver get the message

### Base-station vs. ad hoc

1. base-station
    1. device with (big) antenna, usually connect to wired networks
    2. hopefully base-station more reliable
    3. no coverage where it was not planned
2. ad hoc
    1. on-the-fly and peer-to-peer
    2. can work without pre-planning
    3. maybe don't work over long distance

# Note For Intanagonwiwat00a

## NABC Overview

- Need: With the improvement of the sensor network, the need for an energy efficient, robust, scalable network protocol for these devices is emerging.
- Approach: This paper designs and simulates the Directed Diffusion protocol.
    - Naming Data: Task and reply are named by a list of attribute-value pair. The choice of the naming scheme can affect the expressivity of tasks.
    - For each sensor, it records an *entry list* for *interests*.
    - In each entry, including these data:
        - Cache of the former data
        - The duration of the entry
        - The timestamp of the entry
        - The **gradient** of the entry
    - About the **gradient**: It records the data rate and controls the sensor to send or receive the data.
    - Sensor generate event samples **at the highest data rate**(according to gradient).
    - Sensor propagate the data according to the gradient.
    - Shortest Delay(Path) Finding by reinforcement and negative reinforcement in the network:
        - All data collection section **started with low-data rate**.
        - After the first packet arrived, the sink(the human operator) increase the data rate(the gradient) to the nearest(first replier) sensor and decrease(or time out) the other sensor.

        - All propagate the reinforcement and negative reinforcement to its neighbor
        - After the process, the shortest path has a high data rate and other path have a lower data rate.
        - So data will send faster in the shortest path, and the other path will save energy due to IDLE.
        - Due to other path still sending the data(in lower rate), if the shortest path is out, the network still working and the second shortest path will soon be reinforce to high rate.
- Benefit: The Directed Diffusion protocol is high energy efficient, lower delay and high robustness.
- Challenge: These protocol is really based on the MAC layer protocol.

# Other Information

## Key Idea

1. diffusion can be more efficient than IP
    - less energy
    - all the nodes collaborate
2. In Internet we know the topology(with pre-compute routing table) and do routing
3. In diffusion:
    - we don't know the topology, just know neighbors(**Localized interaction only**)
4. data-centric(**named data**)

- naming the data with attributes rather than IP addresses in the Internet

5. uses reinforcement to discover "routes" for data
    - a user **floods** interests through the network to discover the sensors
6. How to deal with loop, duplicate path and flooding:
    - negative reinforcement and caches to suppress duplicate path
    - reinforcement and high-rate data avoids flooding every time
    - use location information: exposed to the network with attributes
7. Computation in the network(In-Net Processing):
    - can't do this in a general way in IP
    - some big companies put machines around the internet(Google) to do caching
    - In this paper: for duplicate suppression for energy saving