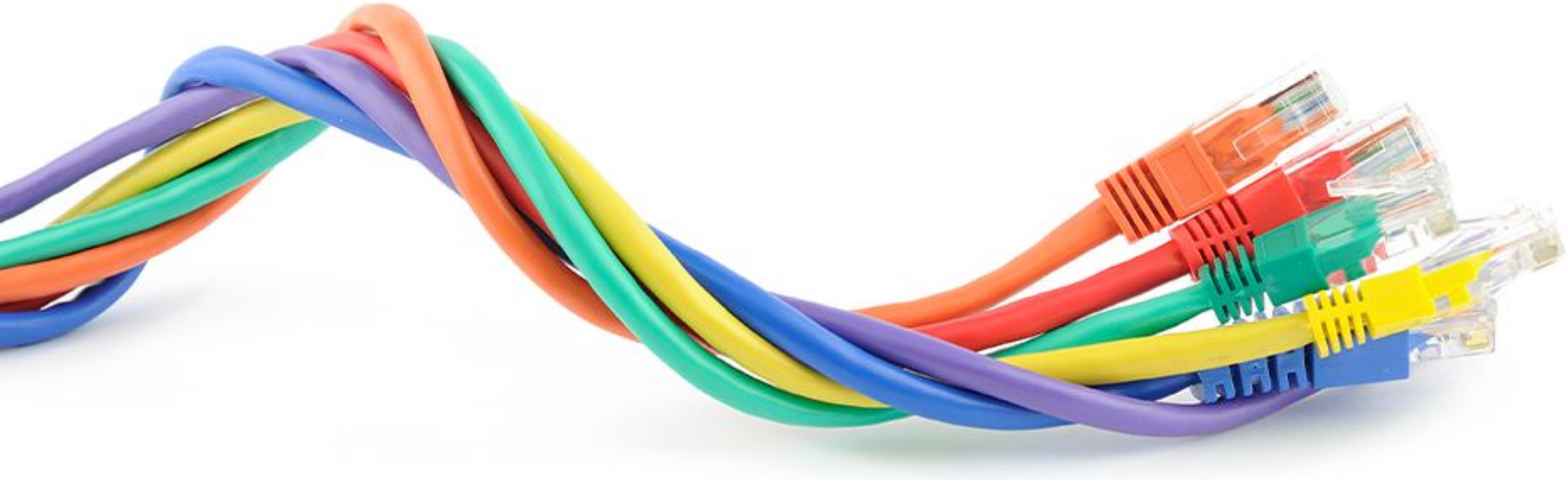
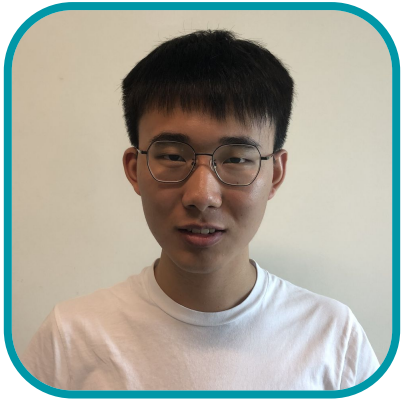
Does It Pay to be Selfish?

50.012 Networks Project Final Presentation



Group 2 – Han Xing Yi, Huang He, James RT, Qiao Yingjie, VS Ragul Balaji, Zhang Peiyuan

The Team



Recap

Problem Statement

What is the maximum threshold of selfishness* in a network that benefits* the adversary without causing major disruptions* to other users?

Our Definitions

selfishness → maximise benefits via active techniques
benefits ⟶ higher goodput / lower delay per task
disruptions → lower average goodput / higher average delay per task

Proposed Research Approach

1. We will create model networks and run simulations on those networks (assisted by Mininet) to:
   1.1. Iteratively find the maximum degree of selfishness through:
      1.1.1. Opening **multiple parallel TCP connections**
      1.1.2. Emulating **misbehaving receivers**
      1.1.3. **Aggressively sending packets**

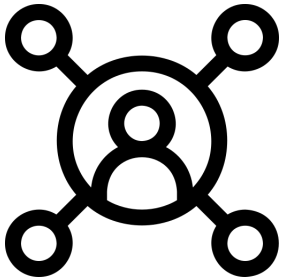
2. Repeat (1.) for:
   2.1. Different TCP versions:
      2.1.1. TCP Reno
      2.1.2. CUBIC TCP
      2.1.3. HighSpeed TCP
   2.2. UDP

3. Countermeasures (Stretch Goal!)

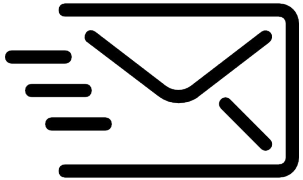
 TO

ATTACK!

Contents



Parallel TCP Connections

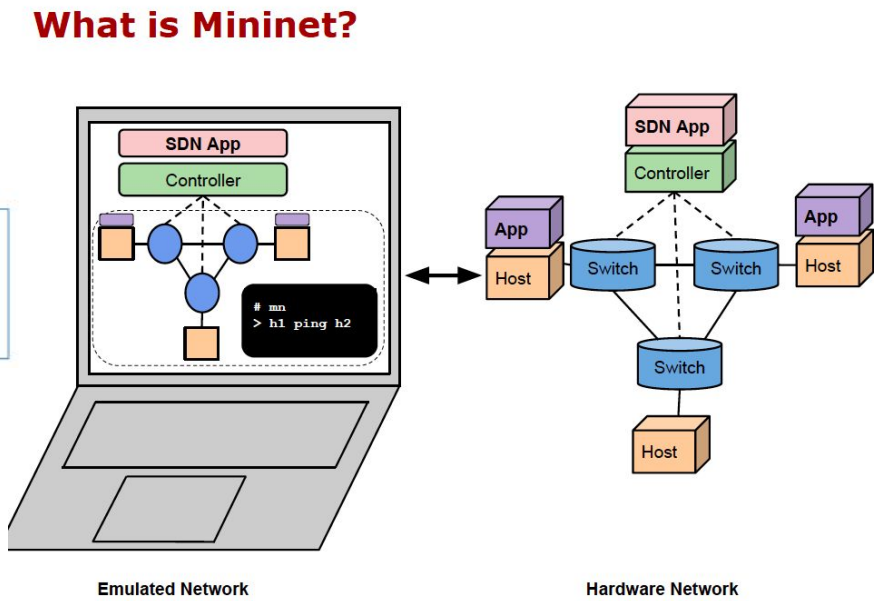


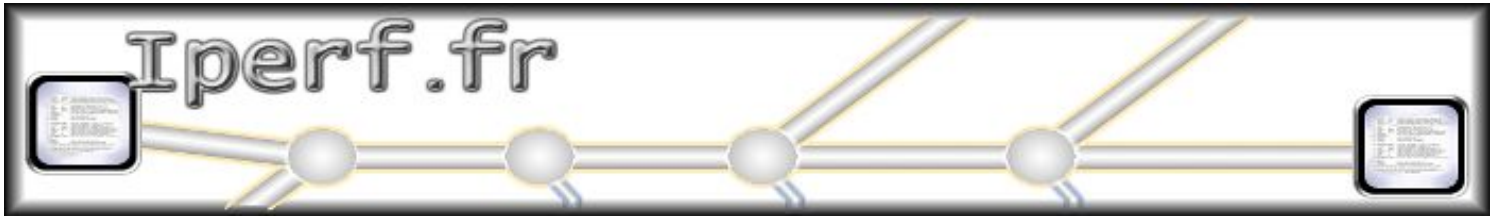
Aggressively Sending Packets

Strategic Abuse of ACKs

Toolset


> sudo mn
controllers
switches
hosts

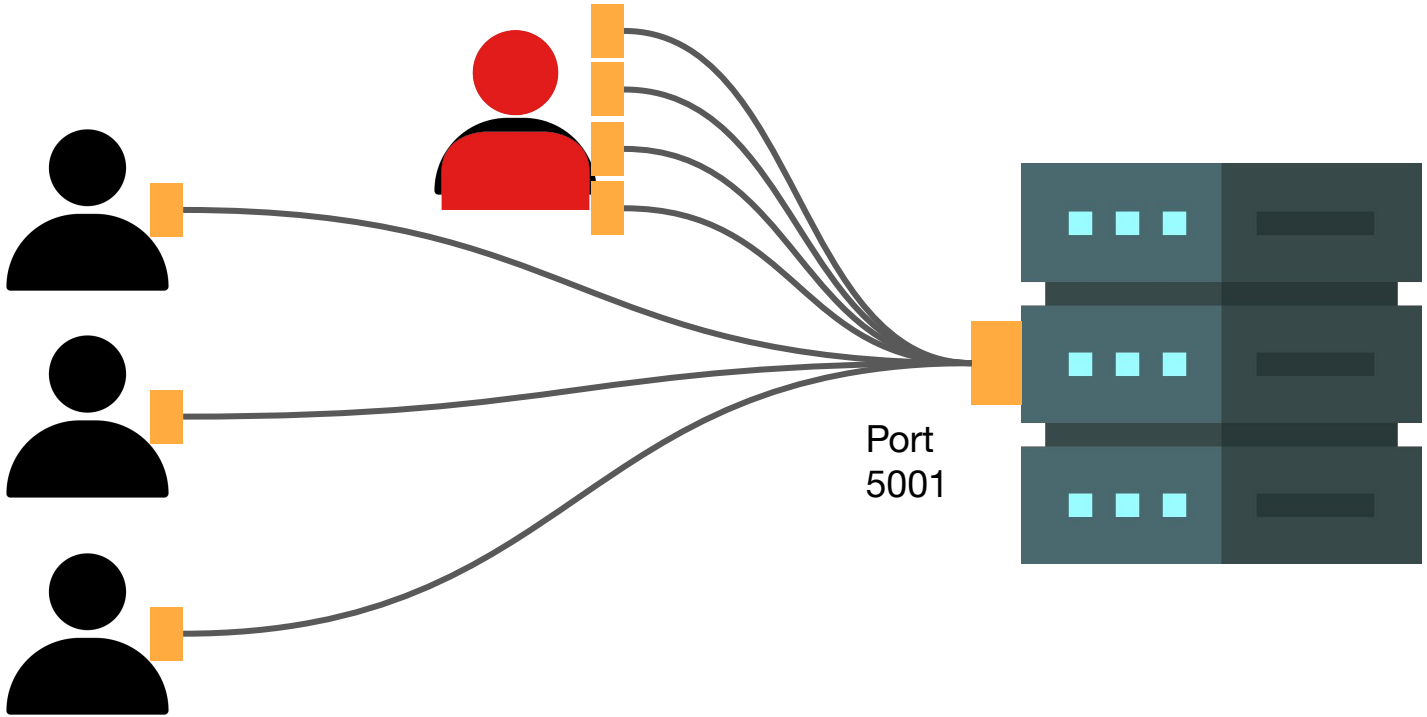

What is Mininet?
SDN App
Controller
mn
> h1 ping h2
SDN App
Controller
App
Host
Switch
Switch
App
Host
Switch
Host
Emulated Network
Hardware Network


Iperf.fr

1. Opening TCP Parallel Connections

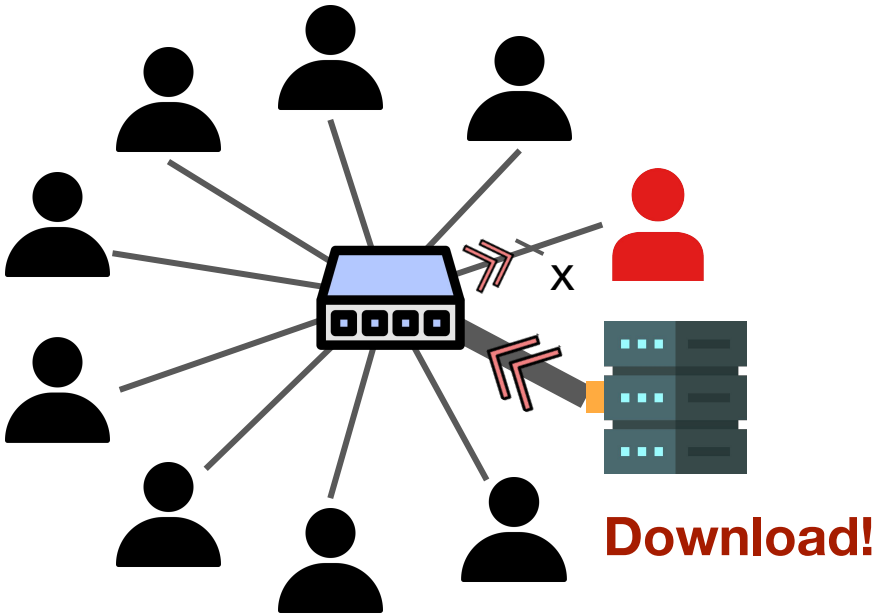
Opening TCP Parallel Connections

What is the attack all about? [Transport Layer Overview]



Opening TCP Parallel Connections

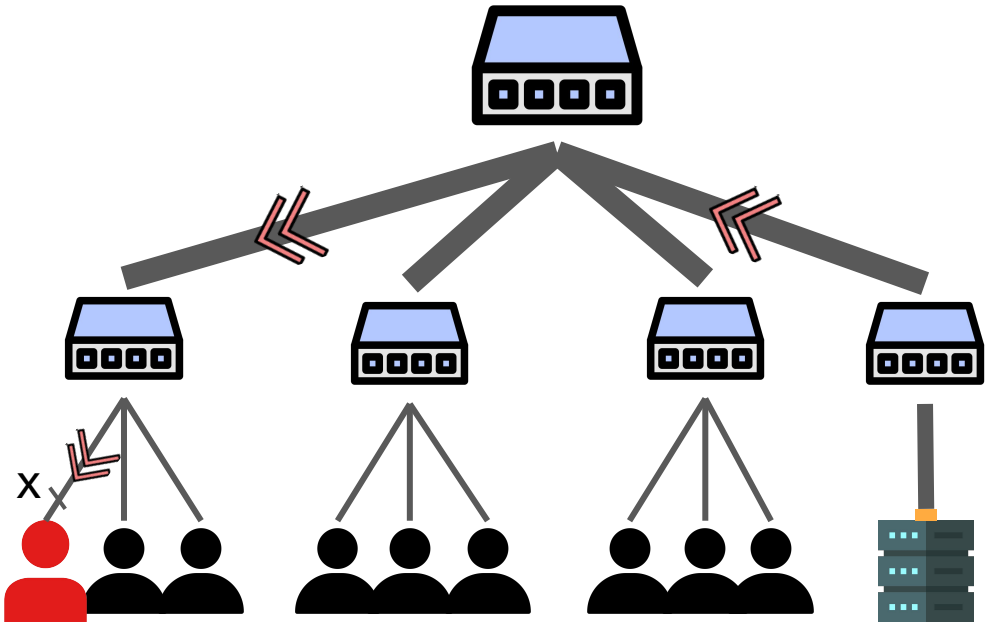
Set-up & Methodology



Parameter	Values
Consumer BW	[2, 4, 6] Mbps
Producer BW	[10, 16, 24] Mbps
// Connections	range(1, 250 + 1, 5)
TCP Algorithm	Reno, Cubic
CWND size (KB)	[16, 64, 256, 512, 1024]
Topology	[‘star’] # Future Work
Test Time	[5, 10] seconds

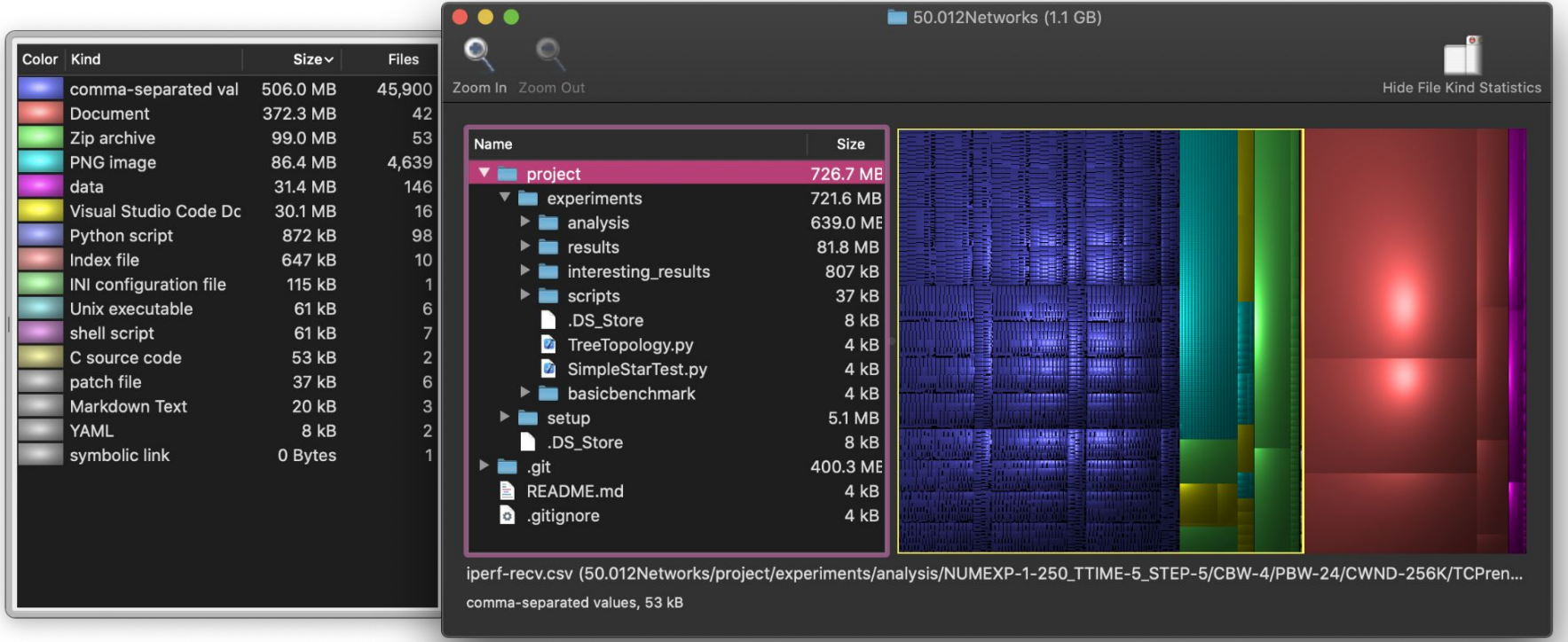
Opening TCP Parallel Connections

Set-up & Methodology

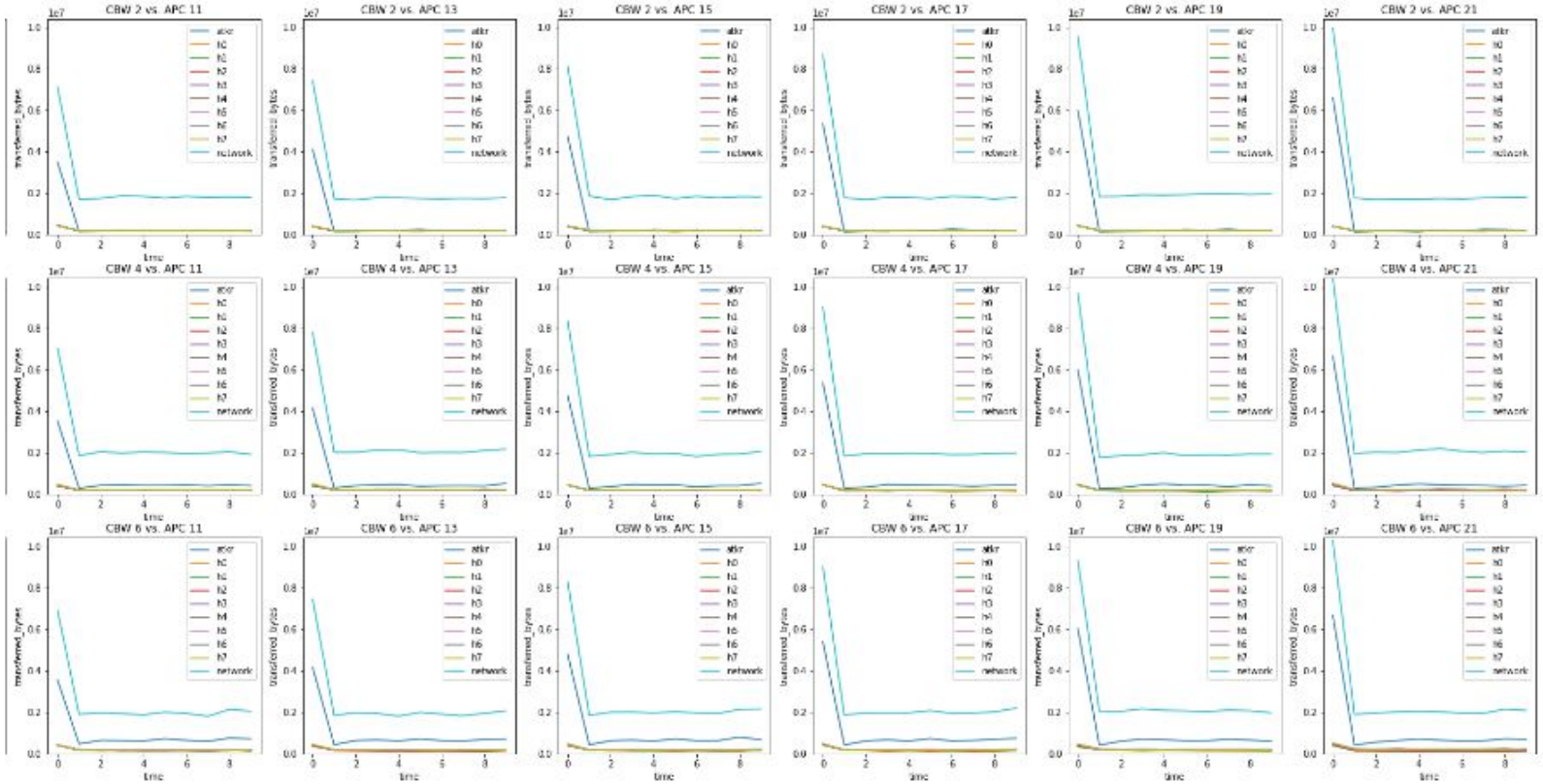


Parameter	Values
Consumer BW	[2] Mbps
Producer BW	[10, 16, 24] Mbps
// Connections	range(1, 250 + 1, 5)
TCP Algorithm	Reno, Cubic
CWND size (KB)	[16, 64, 256, 512, 1024]
Topology	[‘tree’]
Test Time	[5] seconds

Results

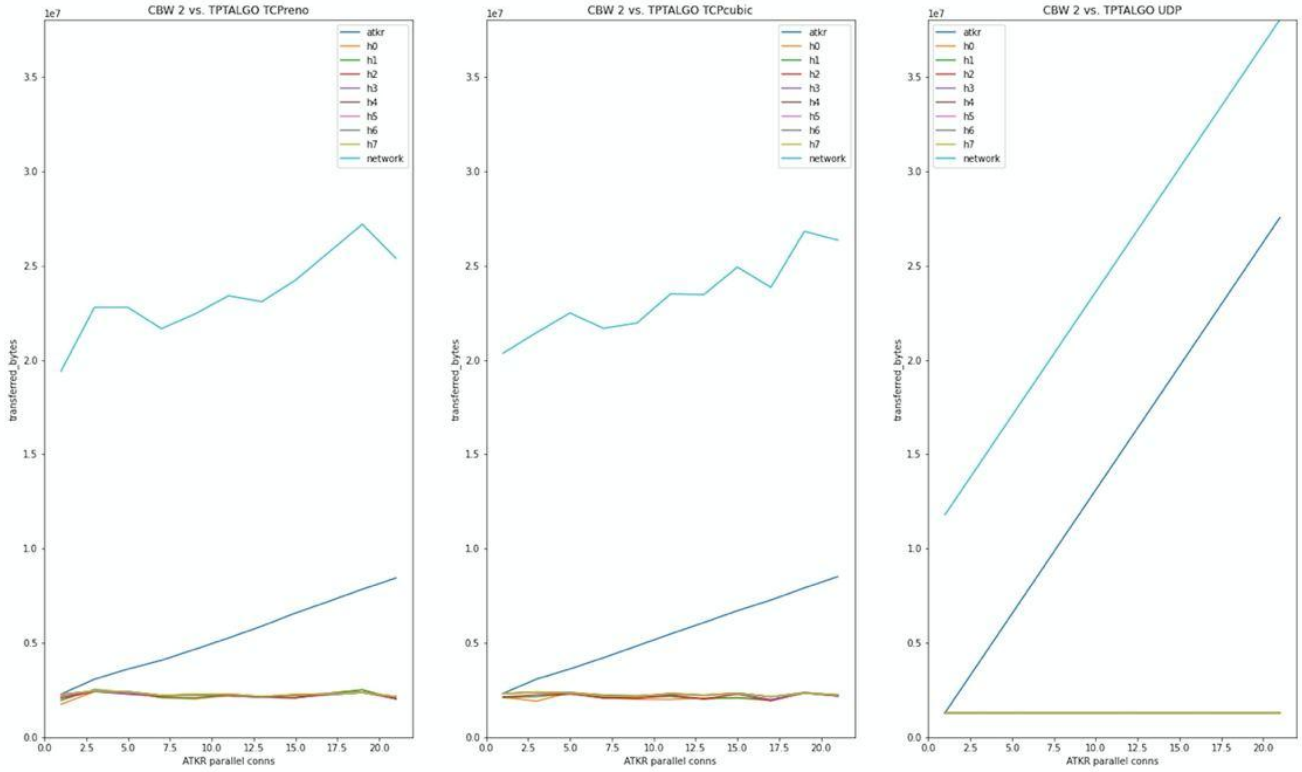

50.012Networks (1.1 GB)
Zoom In
Zoom Out
Hide File Kind Statistics
Color
Kind
Size
Files
comma-separated val 506.0 MB 45,900
Document 372.3 MB 42
Zip archive 99.0 MB 53
PNG image 86.4 MB 4,639
data 31.4 MB 146
Visual Studio Code Dc 30.1 MB 16
Python script 872 kB 98
Index file 647 kB 10
INI configuration file 115 kB 1
Unix executable 61 kB 6
shell script 61 kB 7
C source code 53 kB 2
patch file 37 kB 6
Markdown Text 20 kB 3
YAML 8 kB 2
symbolic link 0 Bytes 1
Name
Size
project 726.7 MB
experiments 721.6 MB
analysis 639.0 MB
results 81.8 MB
interesting_results 807 kB
scripts 37 kB
.DS_Store 8 kB
TreeTopology.py 4 kB
SimpleStarTest.py 4 kB
basicbenchmark 4 kB
setup 5.1 MB
.DS_Store 8 kB
.git 400.3 MB
README.md 4 kB
.gitignore 4 kB
iperf-recv.csv (50.012Networks/project/experiments/analysis/NUMEXP-1-250_TTIME-5_STEP-5/CBW-4/PBW-24/CWND-256K/TCPren...
comma-separated values, 53 kB

Trivial Temporal Analysis



Comparing Transports

$UDP to the moon 🚀🚀🚀🚀🚀🚀🚀🚀



crypto.com

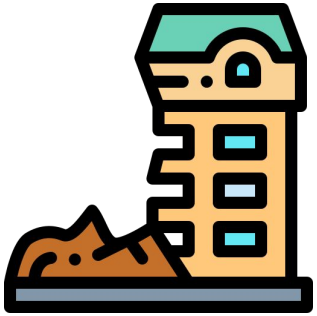
Varying Bandwidths

What About Changing CWND?

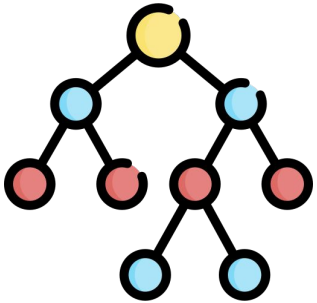
Opening TCP Parallel Connections

Conclusions & Limitations

Optimal Threshold is around **20-30 ports** for large cwnd size, around **60-70 ports** for small cwnd size.



Network Instability & Congestion occurs at **large** cwnd sizes.



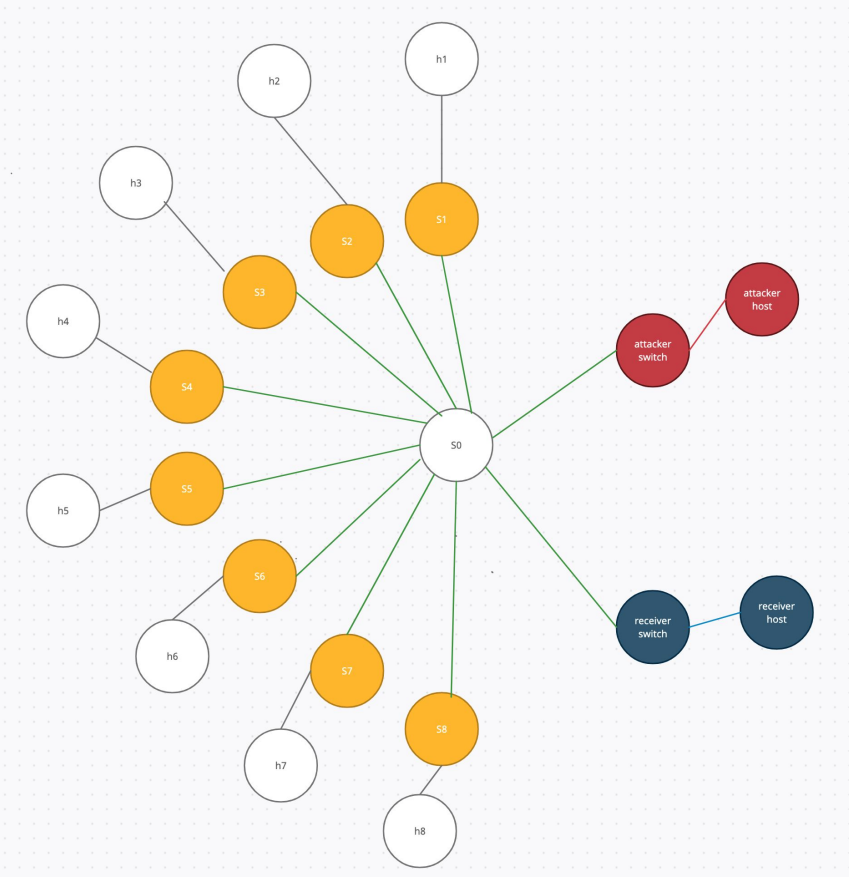
Future Work: Extended Topologies

2. Aggressive Sending of Packets

Aggressive Sending of Packets – Using Star Topology

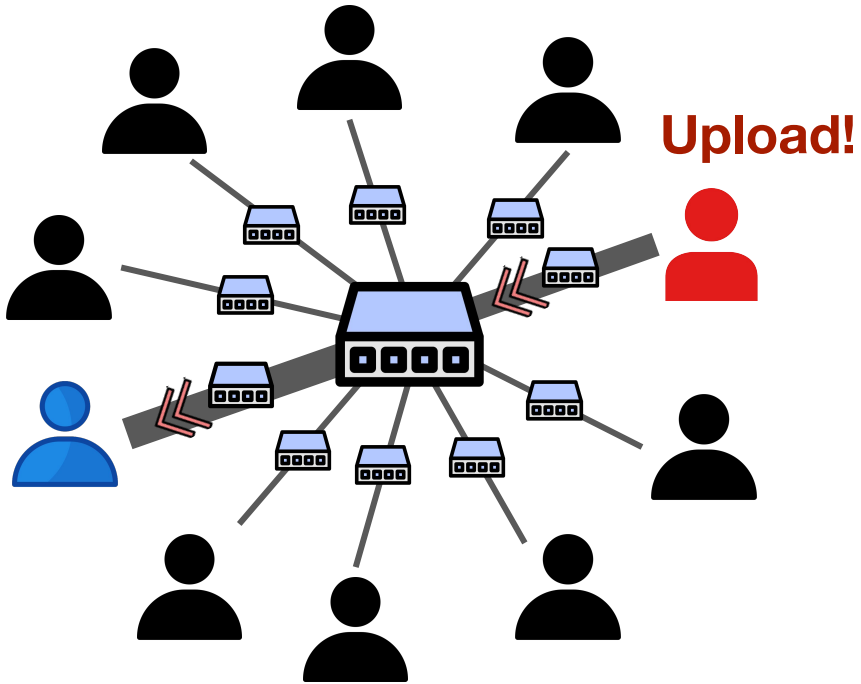
Experiment Setup

Attacker host - attacker link - attacker switch
Receiver host - receiver link - receiver switch
Normal host - normal link - normal switch



Aggressive Sending of Packets – Using Star Topology

What is this attack? + Experiment Setup



Difference:
A switch is added between each host and the central switch.

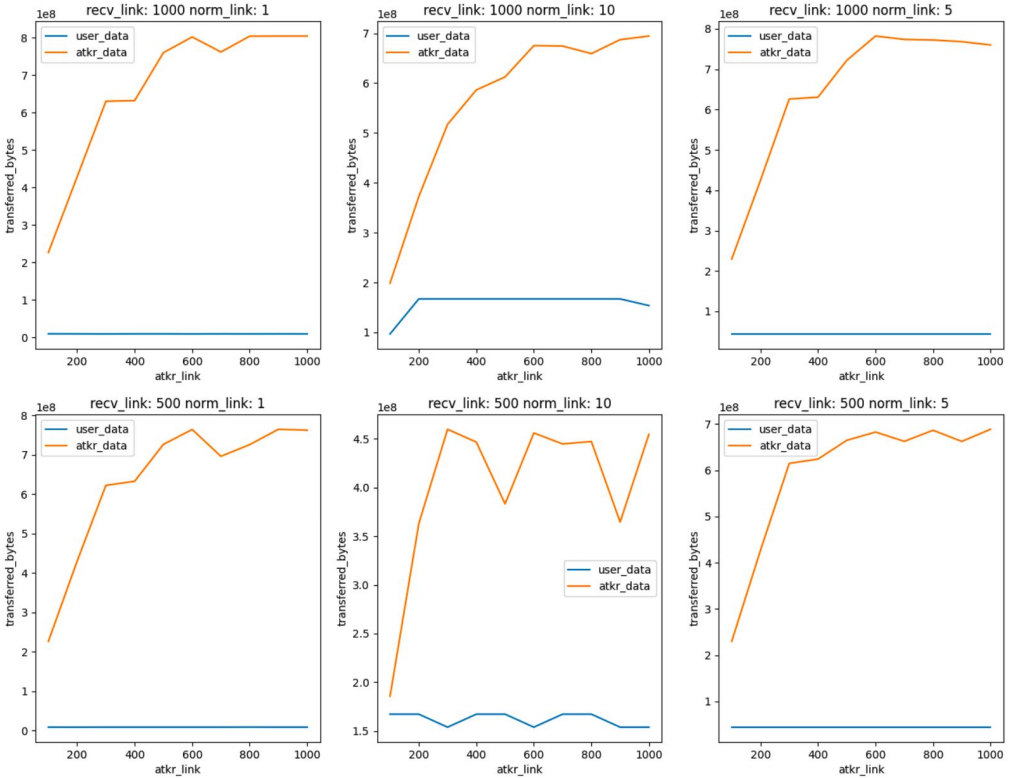
Attacker host - attacker link - attacker switch - central switch
Receiver host - receiver link - receiver switch - central switch
Normal host - normal link - normal switch - central switch

Aggressive Sending of Packets

Star Topology

TCP reno, uncongested

- Attacker: Sends more data with a higher sending rate
- Normal users throughput is not affected

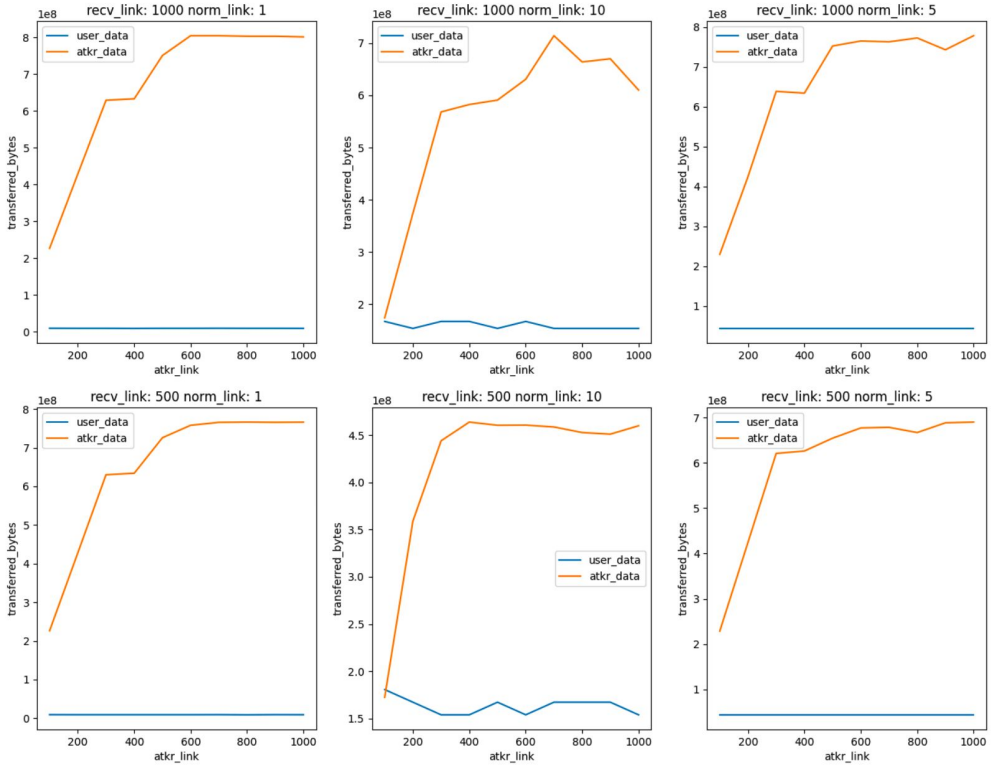


Aggressive Sending of Packets

Star Topology

TCP cubic, uncongested

- Attacker: Sends more data with a higher sending rate
- Normal users throughput is not affected



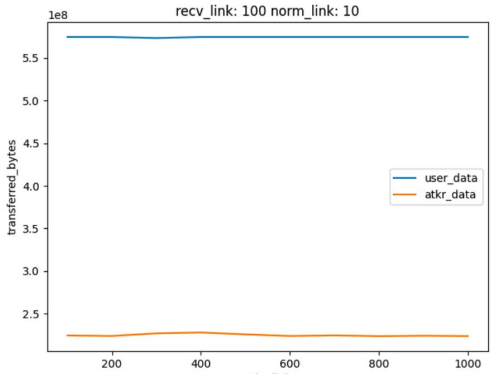
Aggressive Sending of Packets

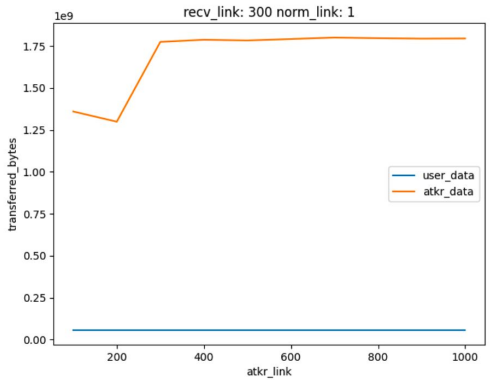
Star Topology

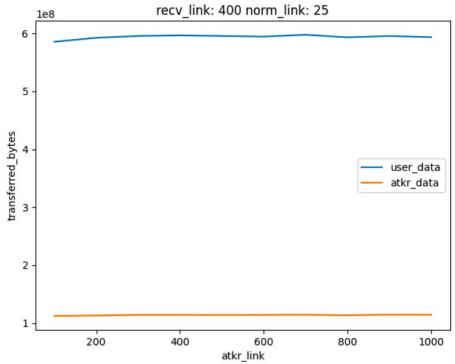
TCP reno, congested

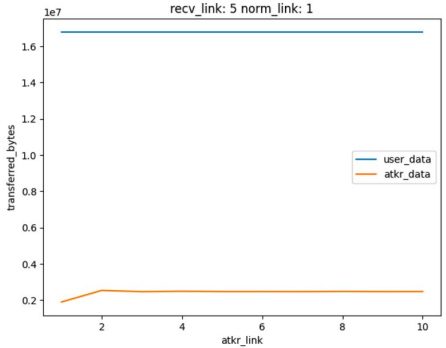
- Attacker: **<u>Cannot</u>** send more data with a higher sending rate

- Normal users throughput is not affected







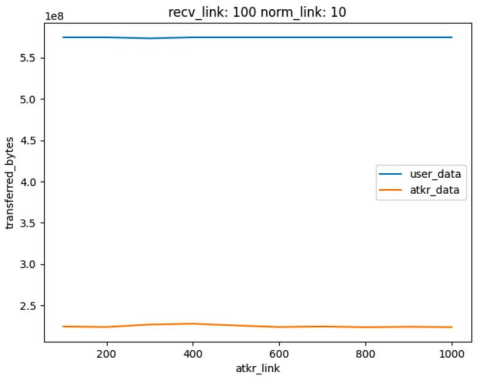


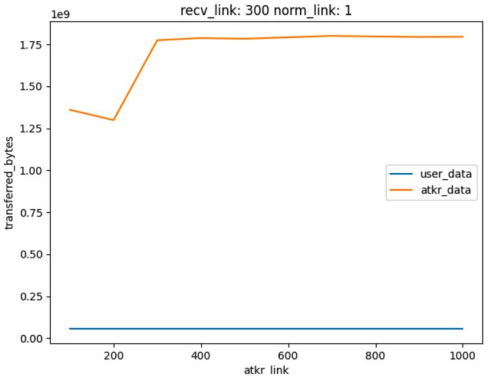
Aggressive Sending of Packets

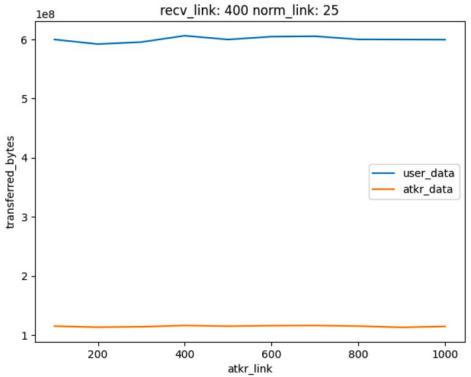
Star Topology

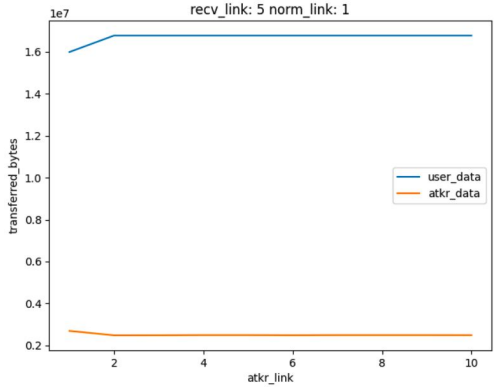
TCP cubic, congested

- Attacker: **Cannot** send more data with a higher sending rate
- Normal users throughput is not affected



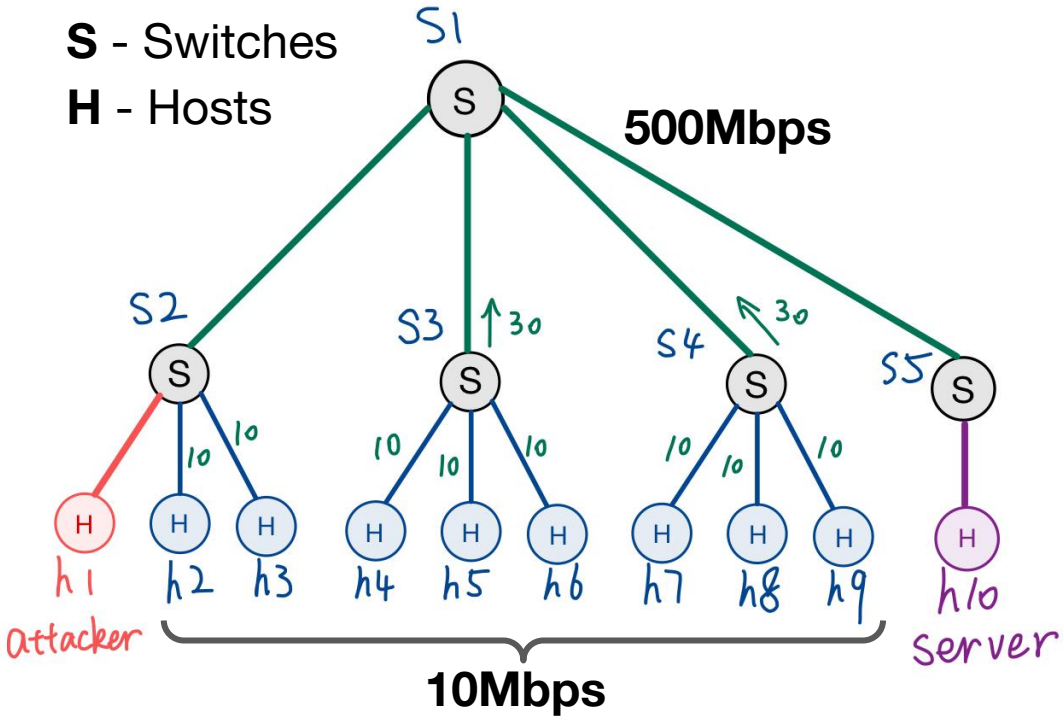






Aggressive Sending of Packets – Tree Topology

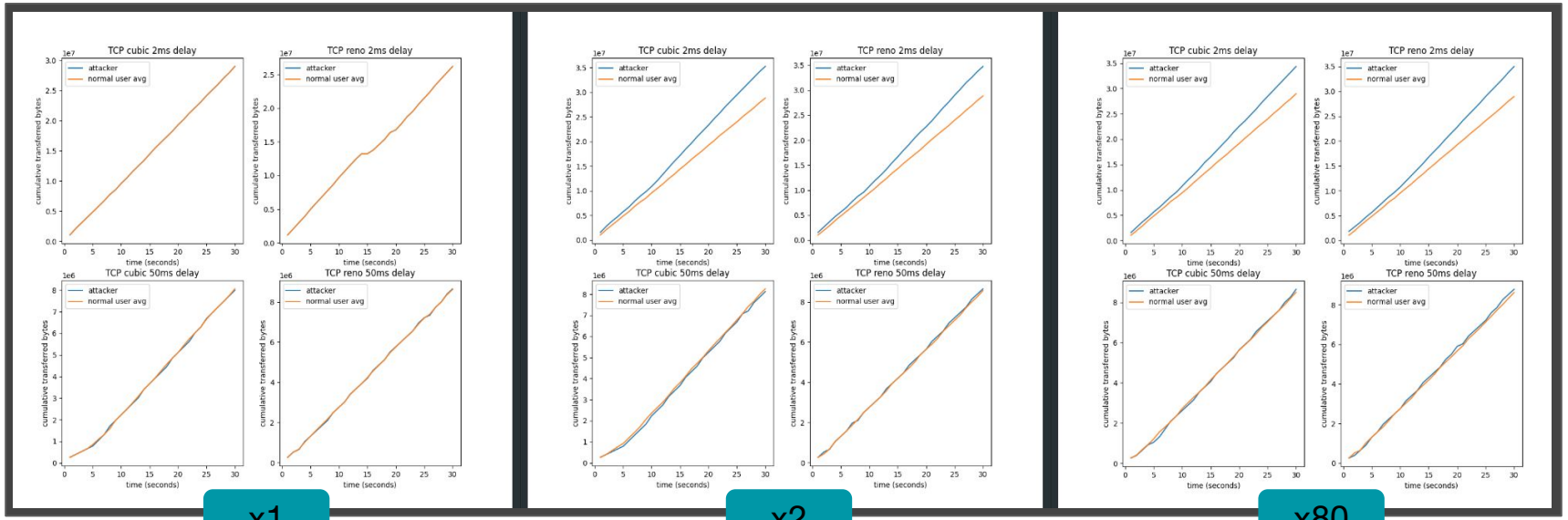
Experiment Setup



Parameter	Values
Server-Switch Bandwidth	[100, 500] Mbps
Attacker-Switch Bandwidth	[10, 20, 50, 200, 800] Mbps
Link Prop Delay	[2, 50] ms
TCP Algorithm	Reno, Cubic
CWND size	[16K, 1M]
Test Time	30 seconds

Aggressive Sending of Packets – Tree Topology

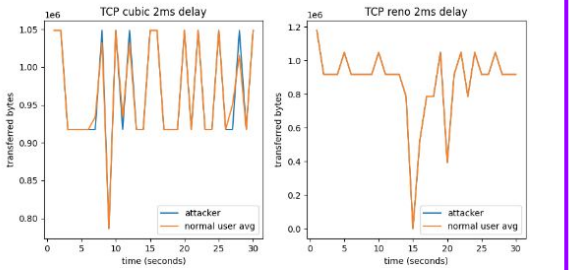
- Three ways of attack:
  - **Aggressively sending (by varying attacker link in Mininet topology)**
  - Aggressively starting (by setting initial window size much bigger for the attacker)
  - Purposely mix use of protocols (by using the alternative protocol for the attacker)

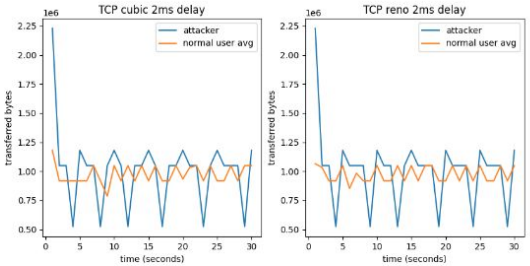


Aggressive Sending of Packets – Tree Topology

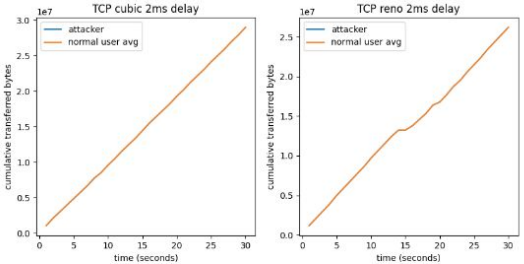
- Three ways of attack:
  - Aggressively sending (by varying attacker link in Mininet topology)
  - **Aggressively starting (by setting initial window size much bigger for the attacker)**
  - Purposely mix use of protocols (by using the alternative protocol for the attacker)

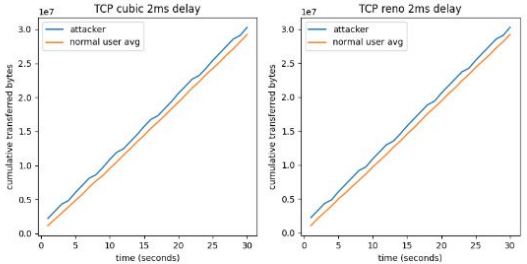
bytes transferred per interval





cumulative bytes transferred





same cwnd size

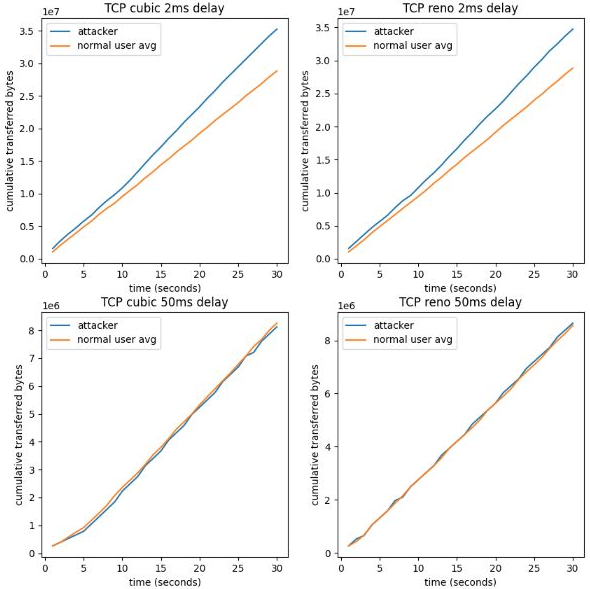
60x cwnd size

Aggressive Sending of Packets – Tree Topology

- Three ways of attack:
  - Aggressively sending (by varying attacker link in Mininet topology)
  - Aggressively starting (by setting initial window size much bigger for the attacker)
  - **Purposely mix use of protocols (by using the alternative protocol for the attacker)**

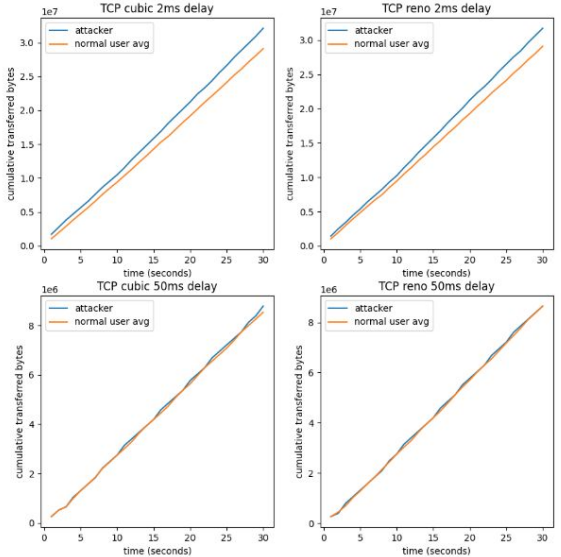
x2

NoMix



x2

Mix

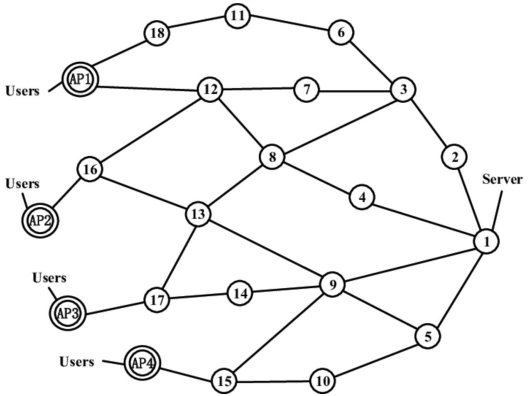


Aggressive Sending of Packets

Conclusion & Limitations

Attacker only gets limited gain by aggressive sending.

TCP Fairness.

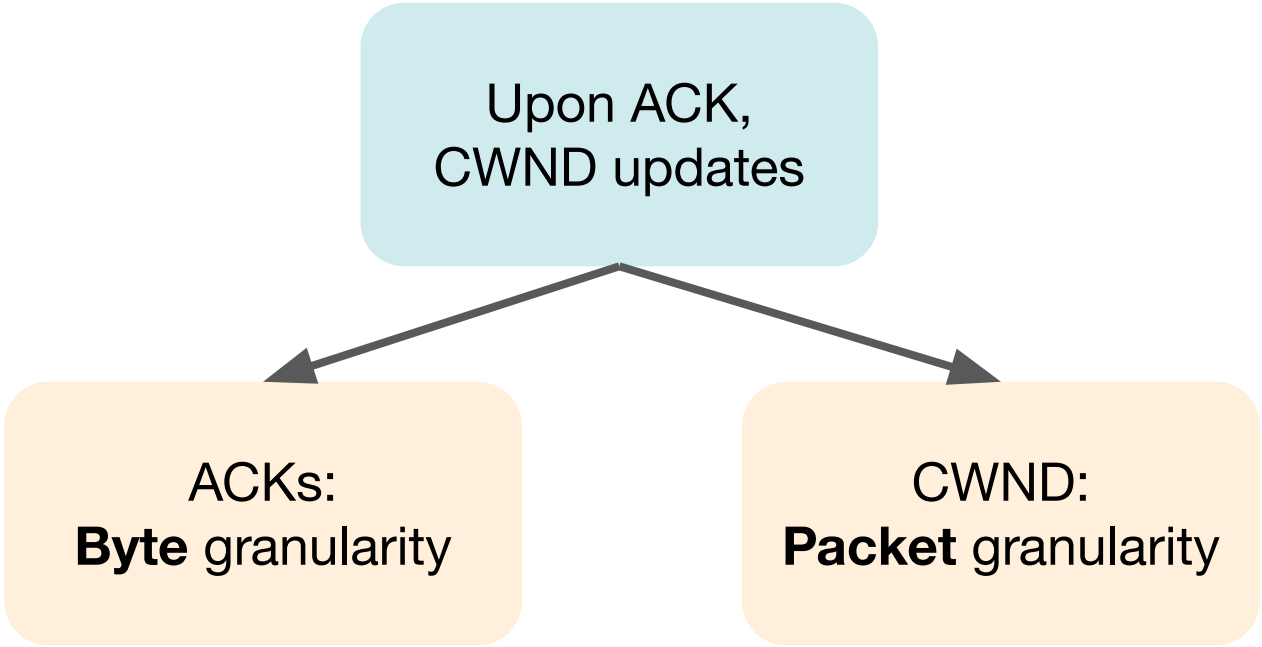


Future Work: Complex topology & custom tooling.

3. Strategic Abuse of ACKs

Strategic Abuse of ACKs

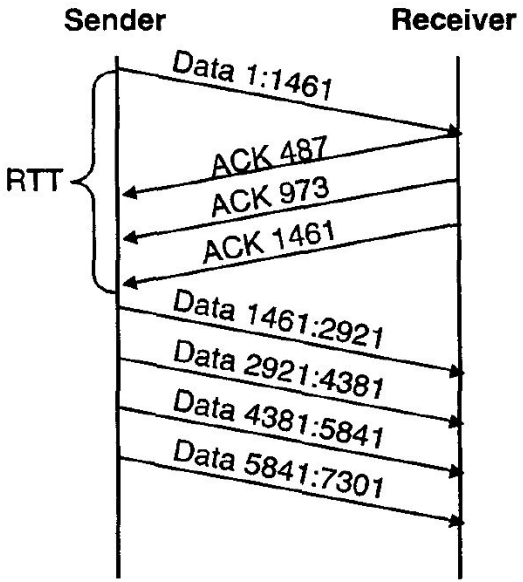
Key Strategies



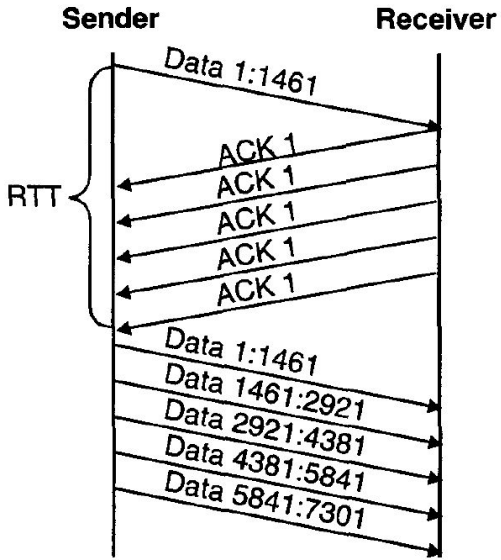
Strategic Abuse of ACKs

3 Attack Methods (Possible Question? 🤔)

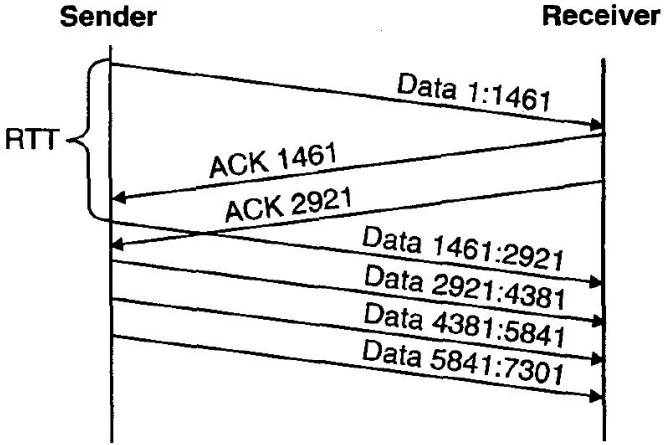
ACK Division



DupACK Spoofing

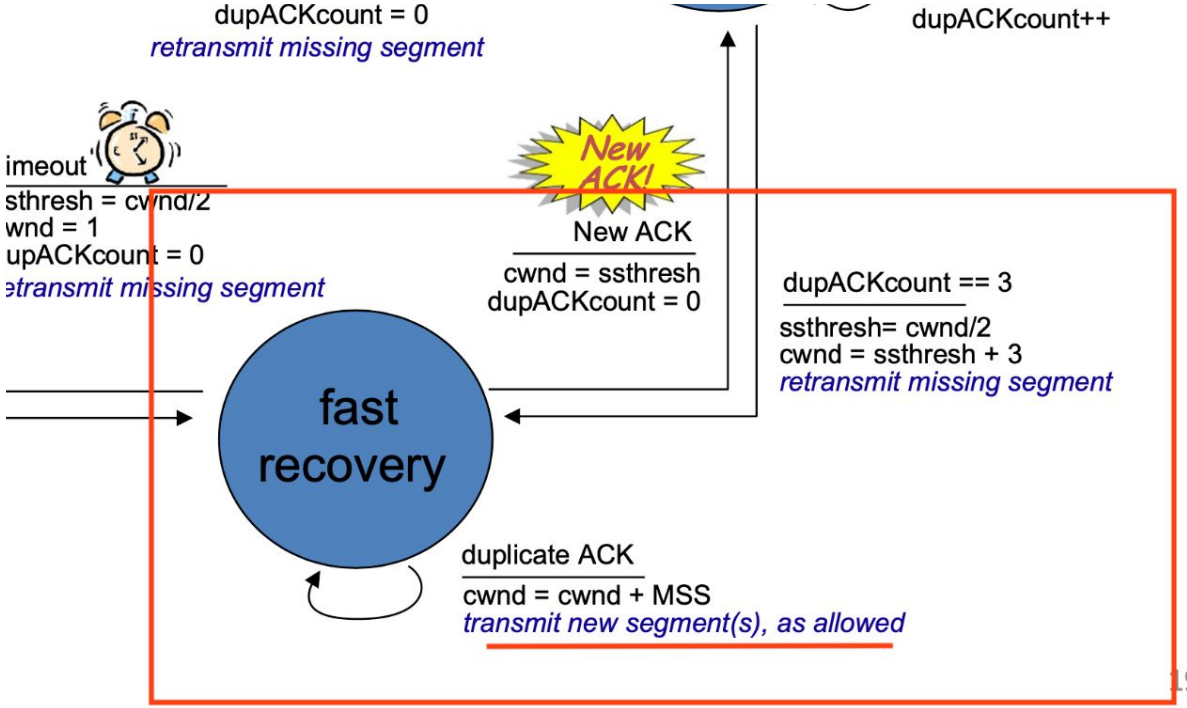


Optimistic ACKing



Strategic Abuse of ACKs

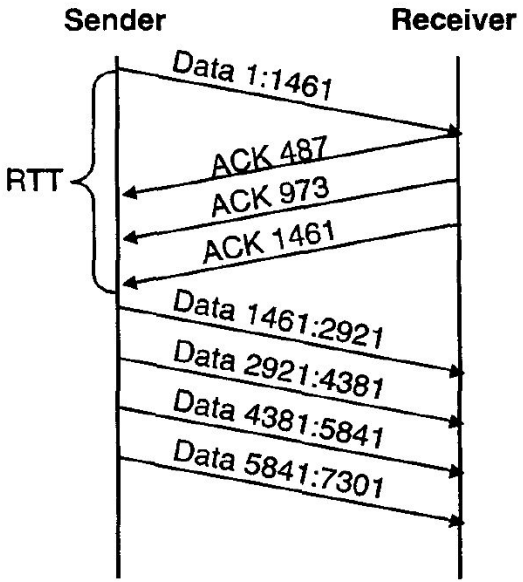
DupACK Spoofing (in Greater Detail)



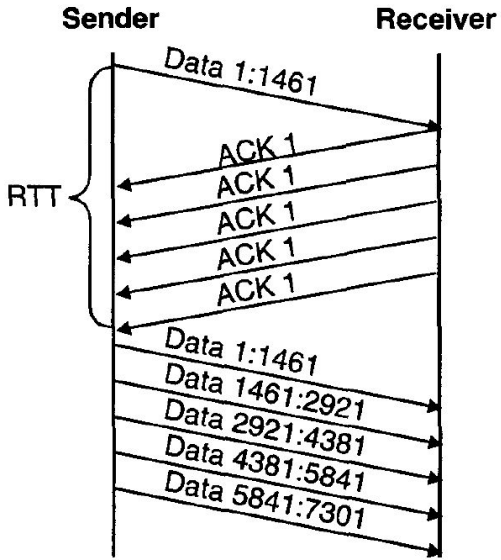
Strategic Abuse of ACKs

3 Attack Methods (Possible Question? 🤔)

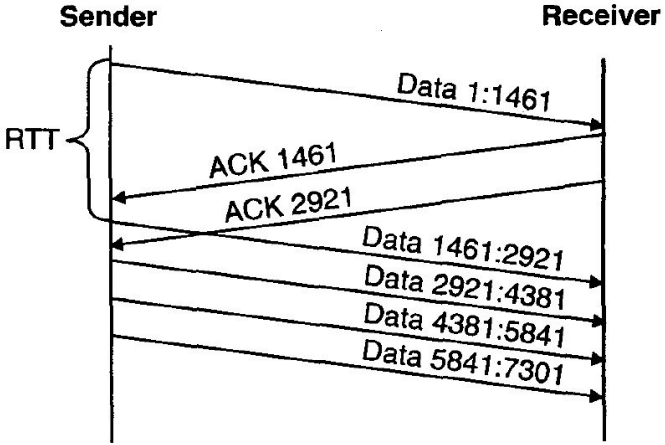
ACK Division



DupACK Spoofing



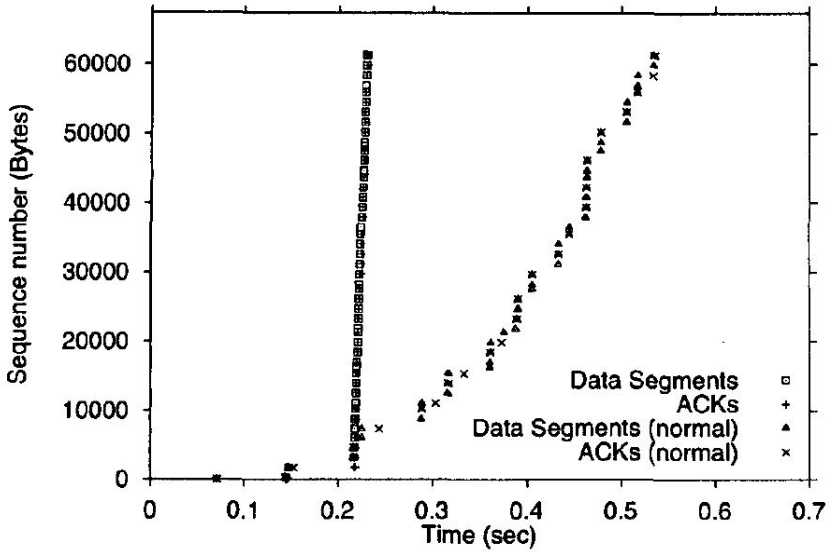
Optimistic ACKing



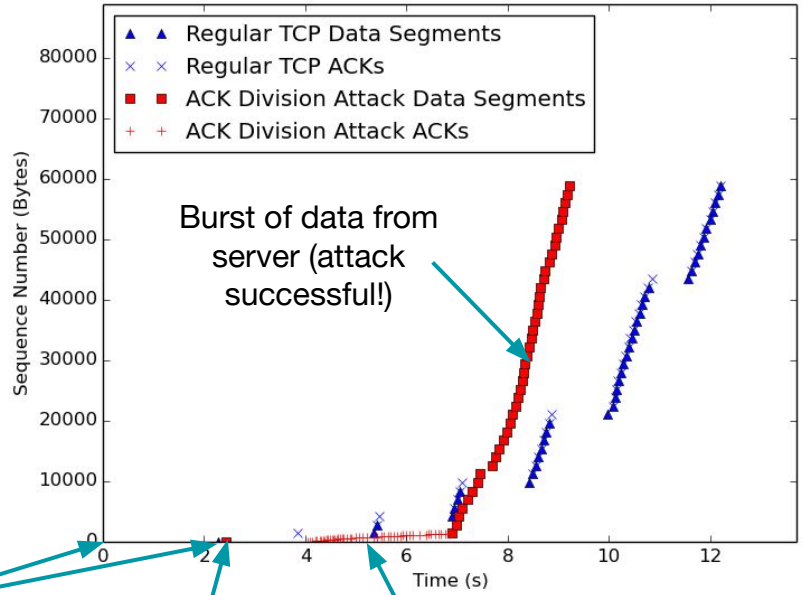
Strategic Abuse of ACKs - Results

ACK Division

Original Paper's Results



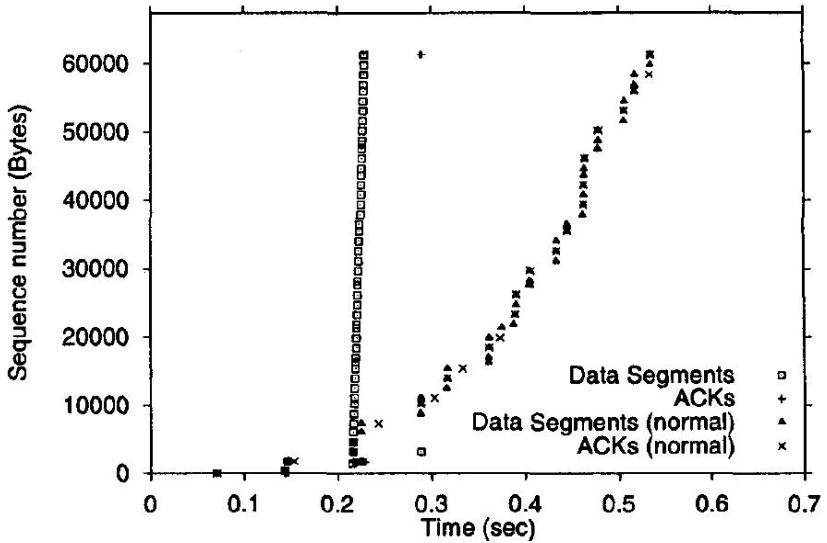
Our Results



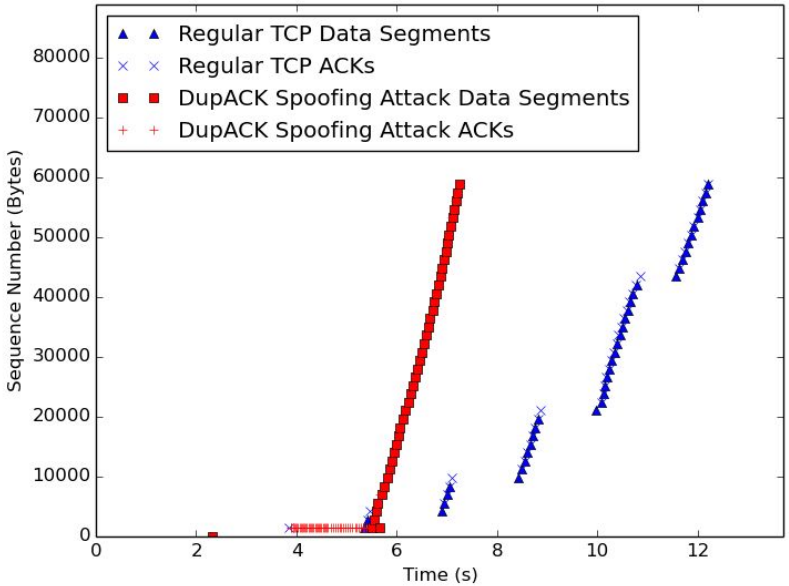
Strategic Abuse of ACKs - Results

DupACK Spoofing

Original Paper's Results



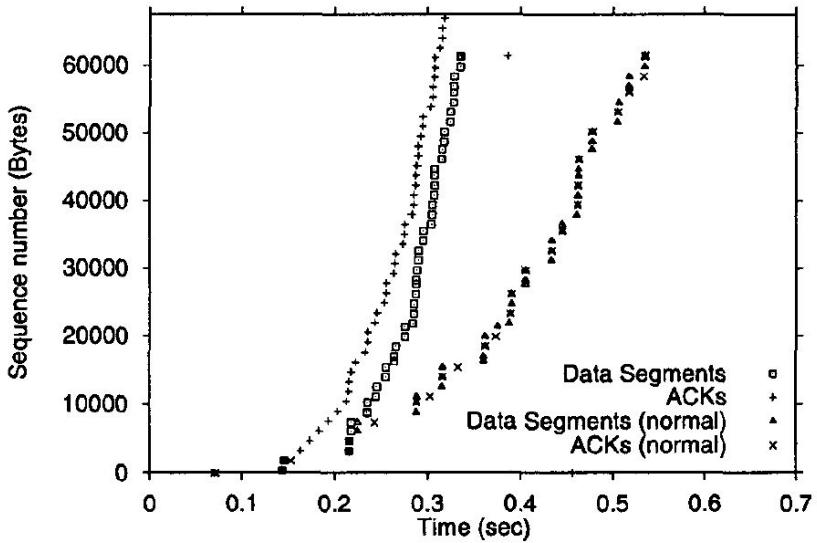
Our Results



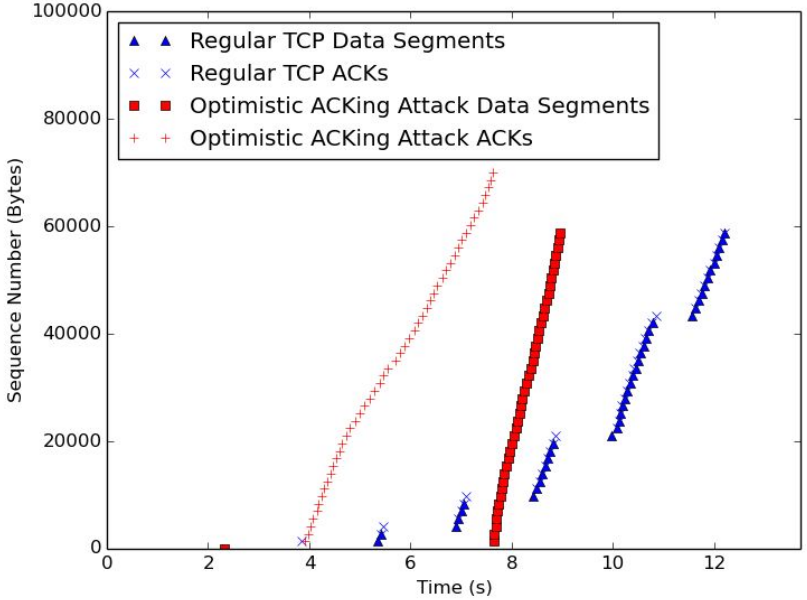
Strategic Abuse of ACKs - Results

Optimistic ACKing

Original Paper's Results

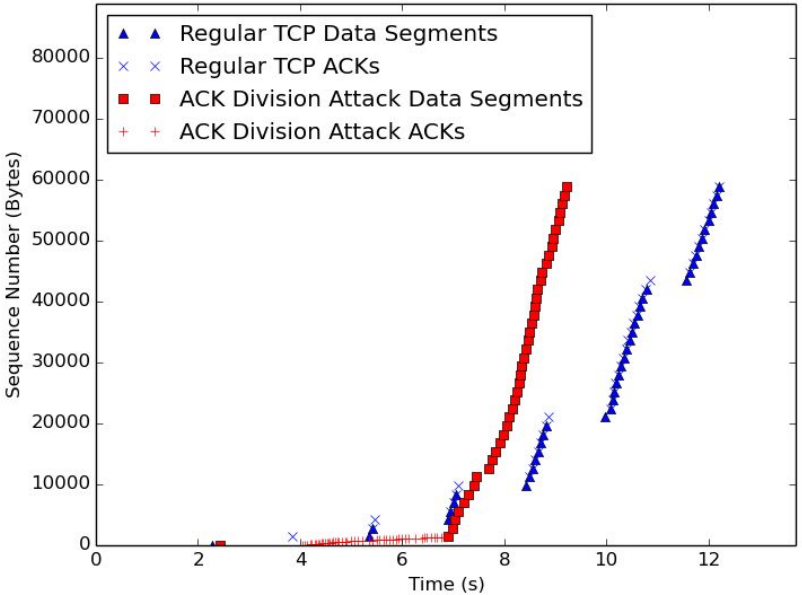


Our Results

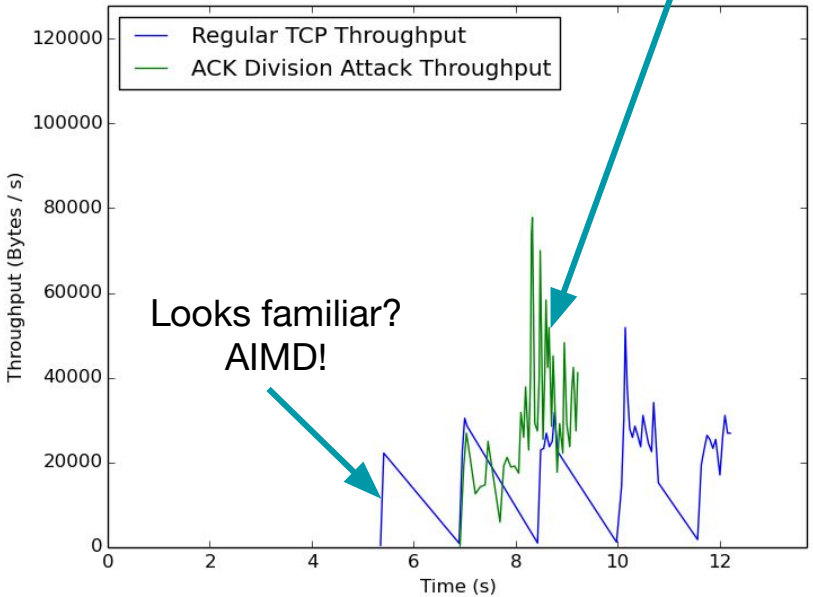


Strategic Abuse of ACKs - Results

ACK Division – Calculating Throughput from Seq Nums

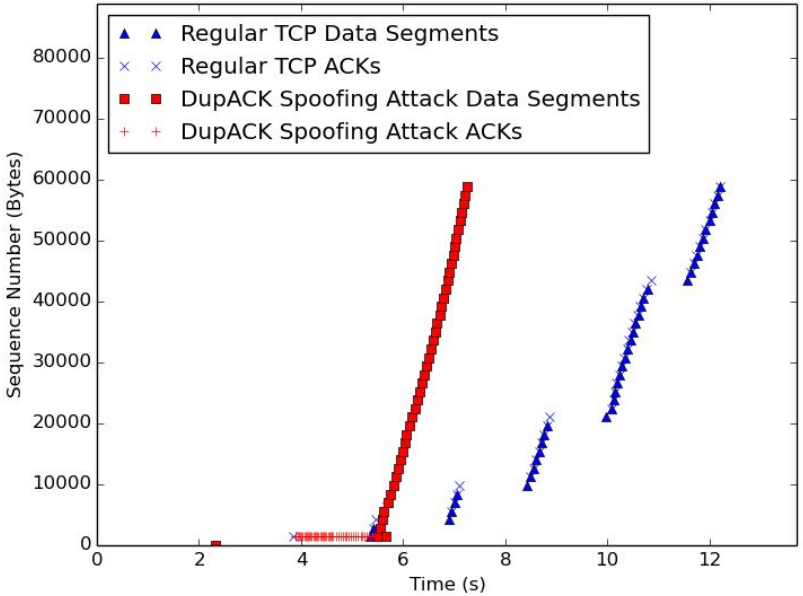


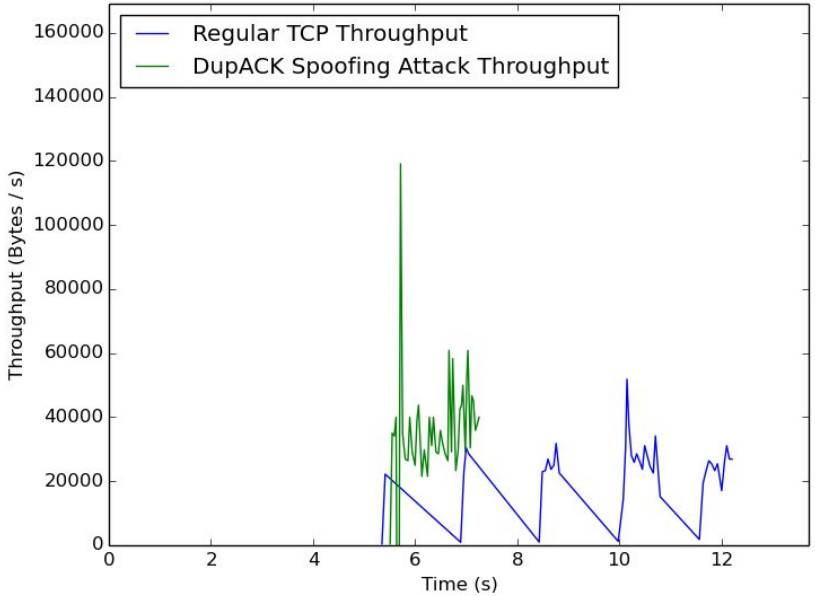
Such throughput, much fluctuations



Strategic Abuse of ACKs - Results

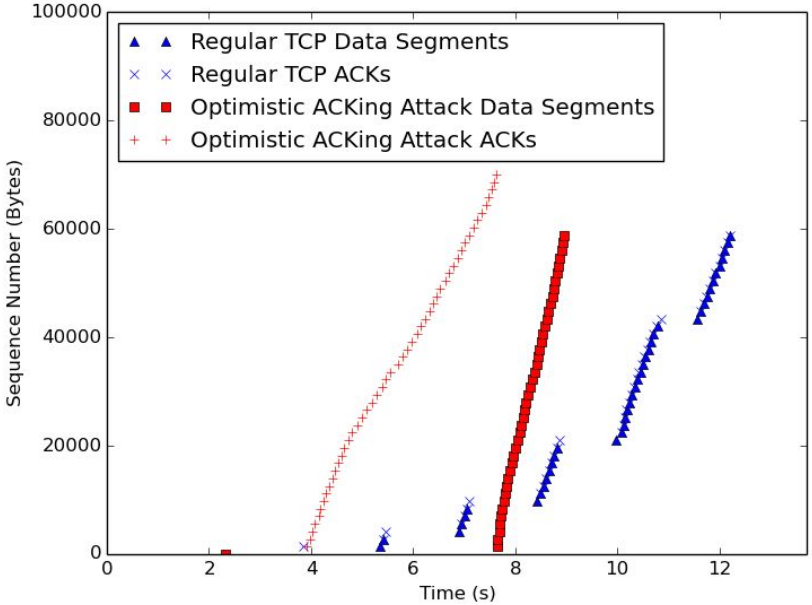
DupACK Spoofing – Calculating Throughput from Seq Nums

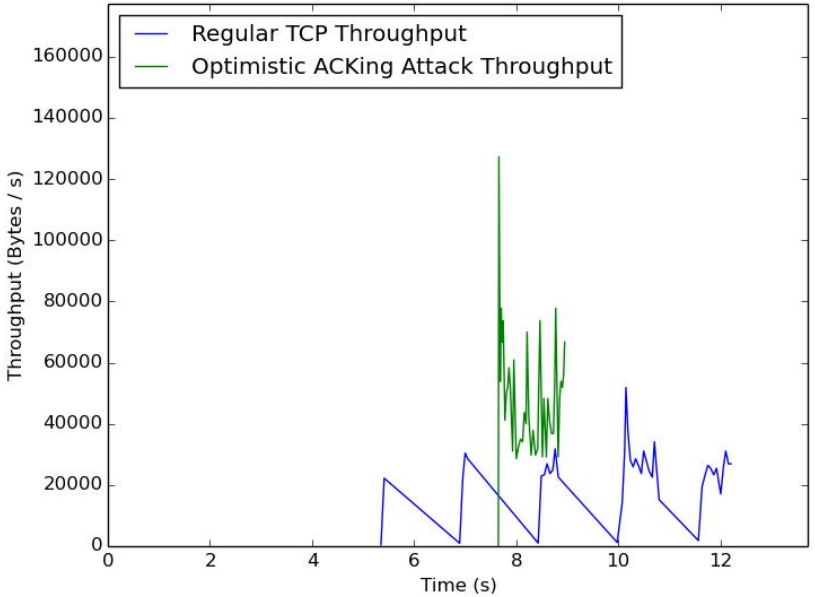




Strategic Abuse of ACKs - Results

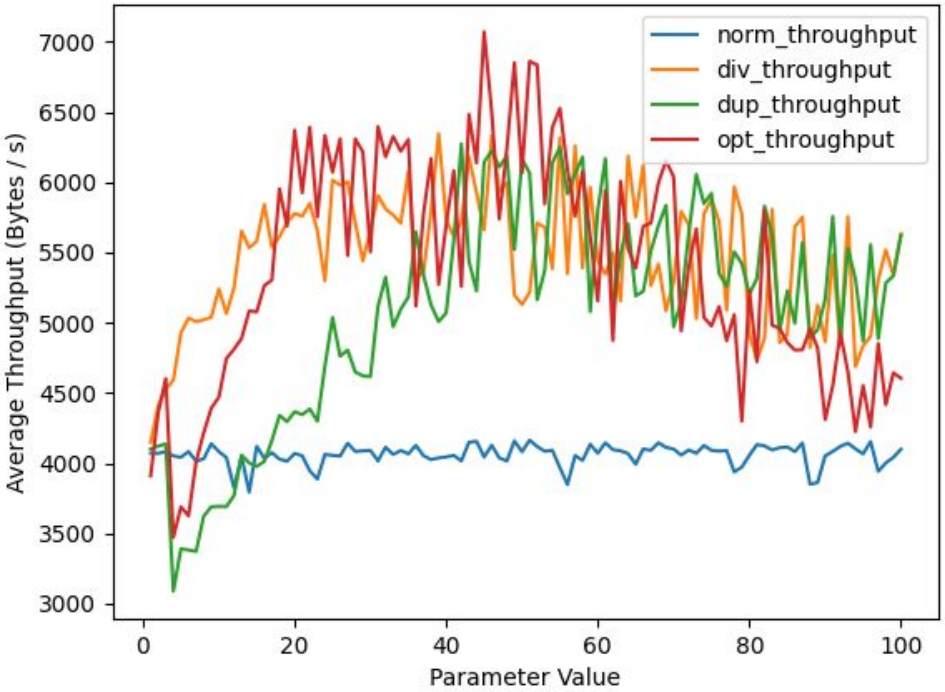
Optimistic ACKing – Calculating Throughput from Seq Nums





Strategic Abuse of ACKs - Results

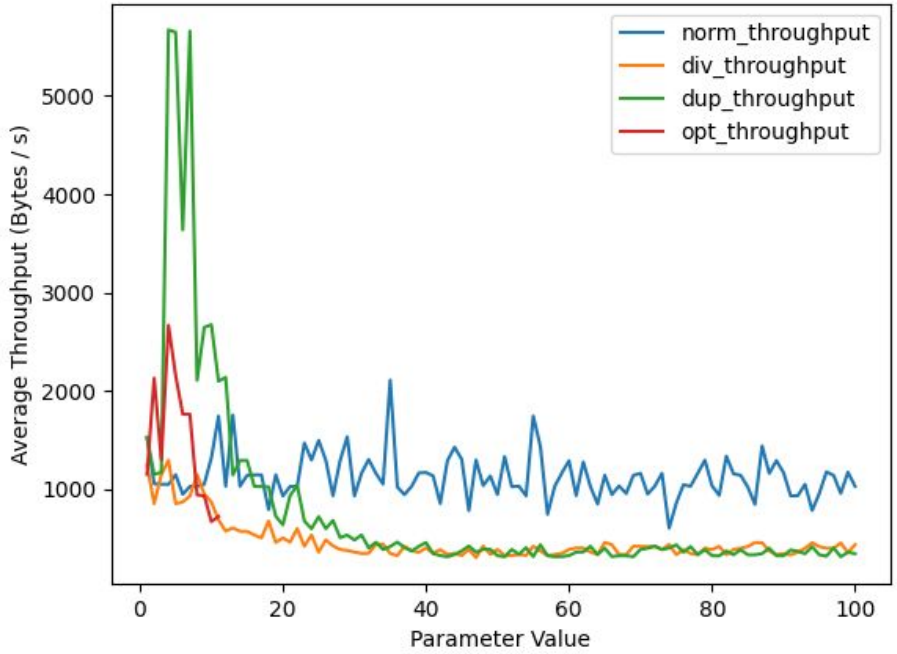
Ideal Standard Topology Average Throughput Comparison (TCP Reno)



Optimal Parameter Value:
Around **50.**

Strategic Abuse of ACKs - Results

Congested Topology Average Throughput Comparison (TCP Reno) - Limited Bandwidth & Queue Size



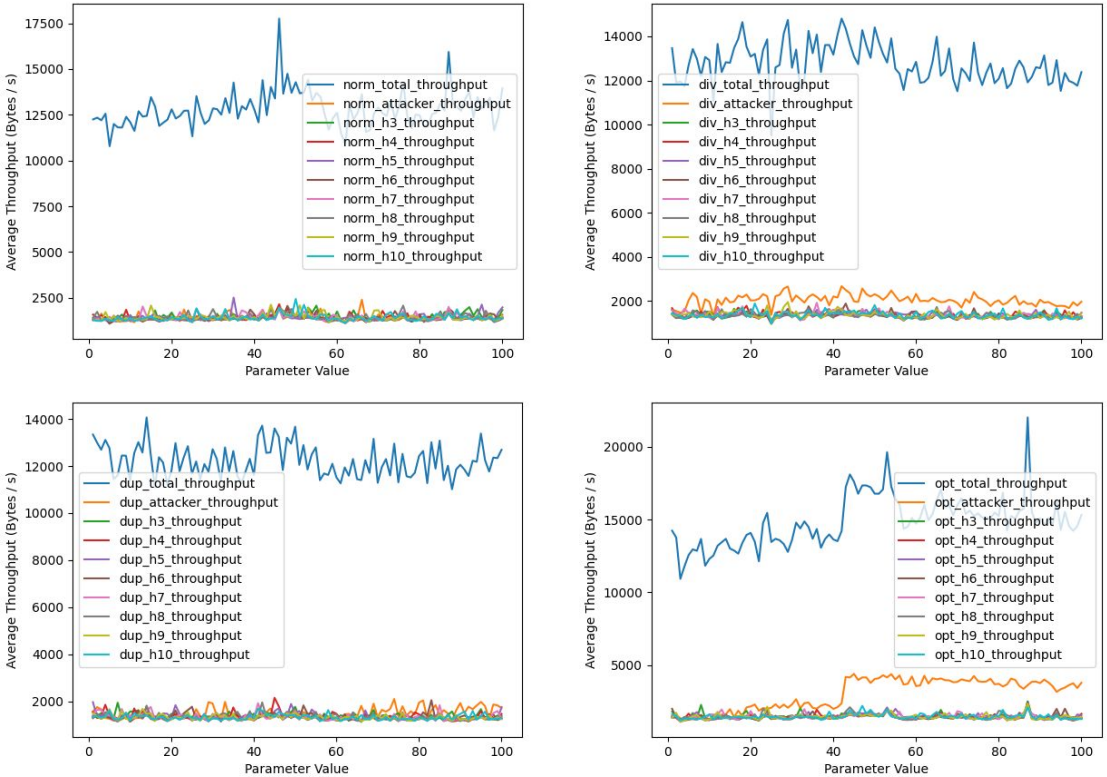
Optimal Parameter Value:
Around **3-5** (original paper)

Lesson:

- Network Parameters Matter!
- Congestion limits attacker, but does not prevent abuse.

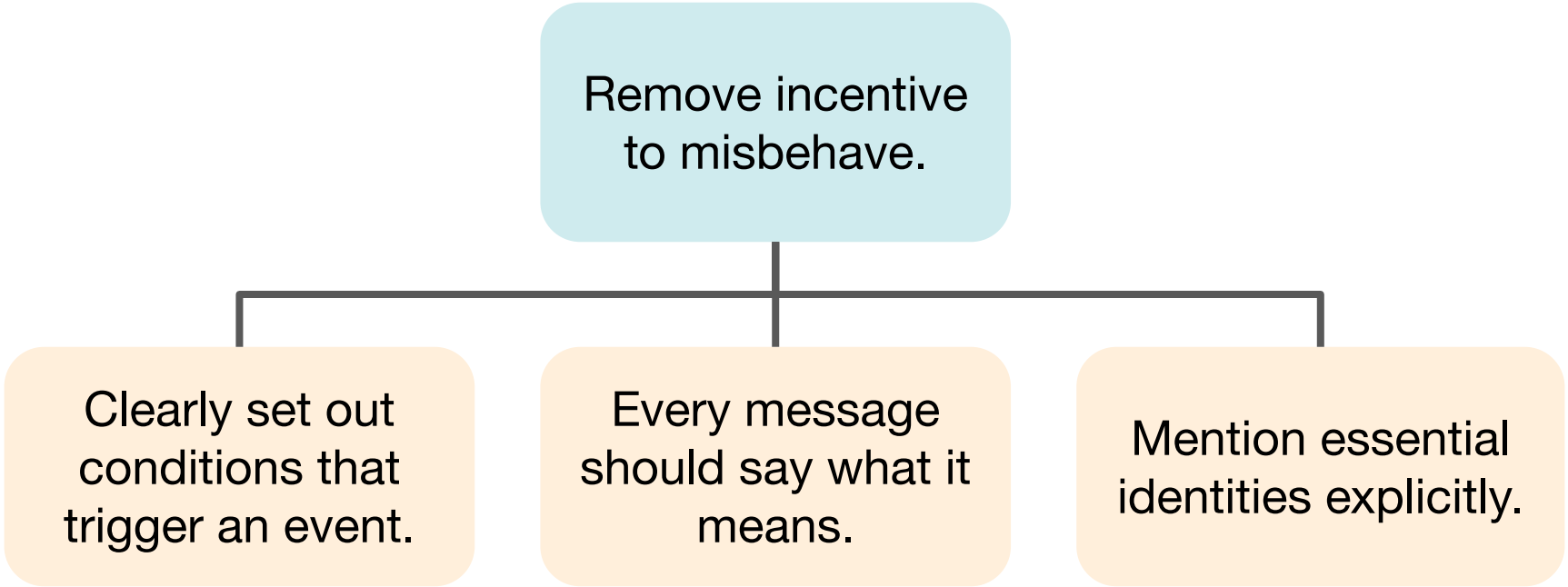
Strategic Abuse of ACKs - Results

Star Topology Average Throughput Comparison (TCP Reno)



Strategic Abuse of ACKs - Defense

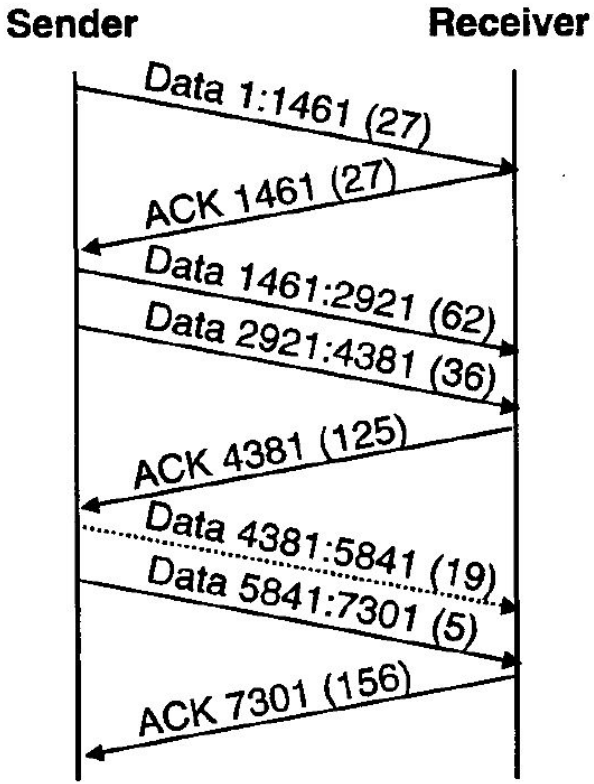
Defense mechanisms as proposed in original paper.



Strategic Abuse of ACKs - Defense

ACK Division	DupACK Spoofing	Optimistic ACKing
Byte granularity: Increase cwnd proportional to data ACKed	**Nonce:** Add 2 fields – 1. Nonce 2. Nonce Reply	**Sum of Nonces:** Cumulative sum of nonces in the ACKs

Strategic Abuse of ACKs - Defense



Strategic Abuse of ACKs - Defense

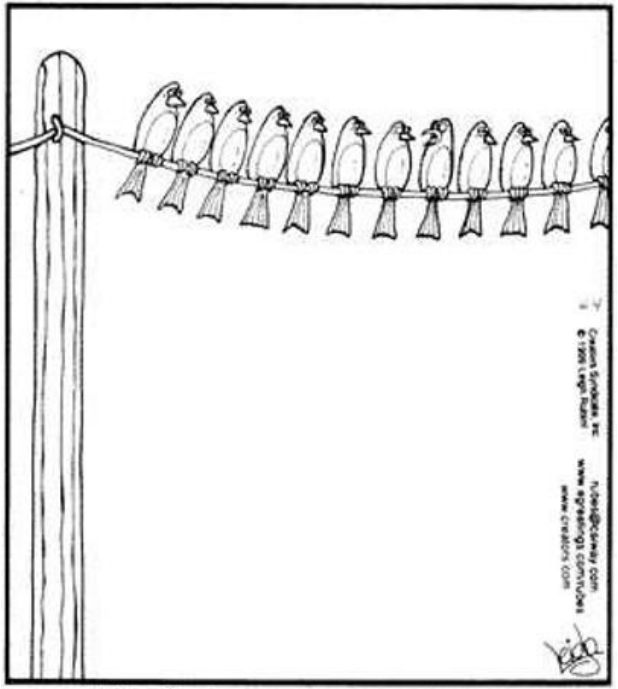
Implementing Countermeasures for TCP Reno

Strategic Abuse of ACKs - Discussion

Question: Are real-life operating systems vulnerable to these attacks?

	ACK Division	DupACK Spoofing	Optimistic Acks
Solaris 2.6	Y	Y	Y
Linux 2.0	Y	Y (N)	Y
Linux 2.2	N	Y	Y
Windows NT4/95	Y	N	Y
FreeBSD 3.0	Y	Y	Y
DIGITAL Unix 4.0	Y	Y	Y
IRIX 6.x	Y	Y	Y
HP-UX 10.20	Y	Y	Y
AIX 4.2	Y	Y	Y

Strategic Abuse of ACKs - Discussion



"Don't get me wrong, I'm all for the advancement of technology, but if the world goes completely wireless, just where in the heck are we supposed to sit?!"

Modern TCP implementations are **STILL vulnerable**

Solutions **hard** to implement

Results are **highly variable**

Hardships

Blood, Sweat & Tears 🥺

Incomprehensive Mininet documentation

Hard to debug Mininet code

Variable with Ubuntu version & hardware

Mininet is hard to parallelise -> Loss of sleep :(

iPerf is somewhat inflexible

Requires painful management for custom TCP protocols

Conclusion & Future Work

Yes, it does pay to be selfish! :D

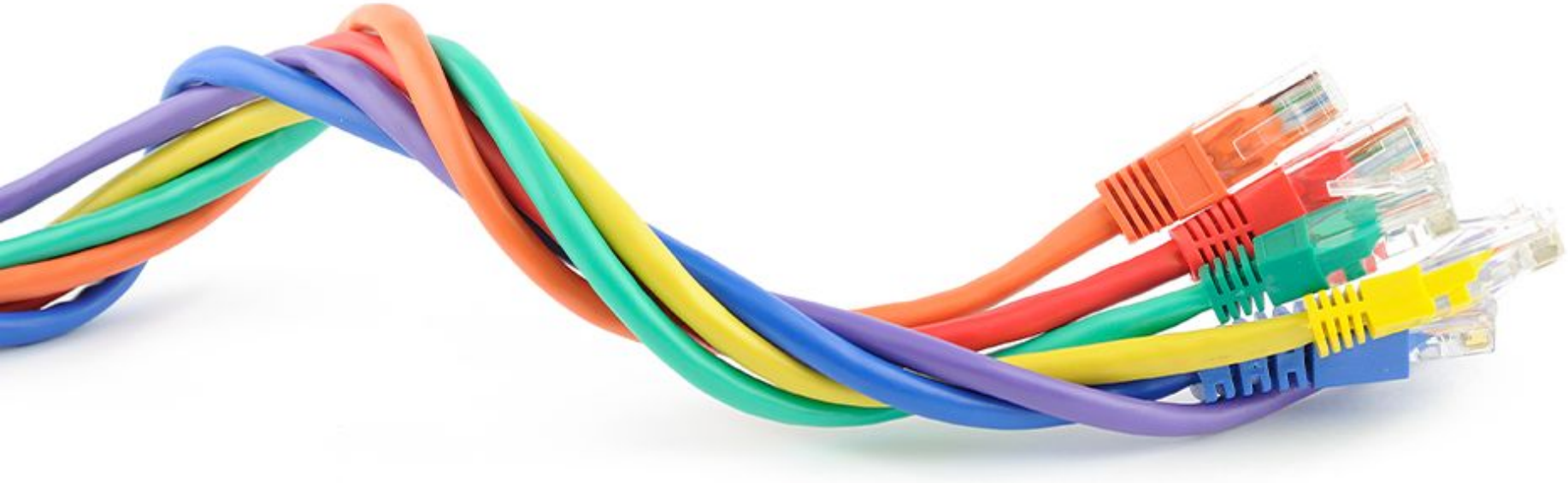
Only within a certain range, TCP Fairness is too strong :(

Vary with more (possibly random) topologies!

Experiments with finer granularity

Thank You!

Any Questions?



Group 2 – Han Xing Yi, Huang He, James RT, Qiao Yingjie, VS Ragul Balaji, Zhang Peiyuan

Appendix

Proposed Final Exam Question:

https://docs.google.com/document/d/1Sc6liyLozGKGvBUYS79vZEIRRMtrHClTwPpqgYdZrzM/edit?usp=sharing