



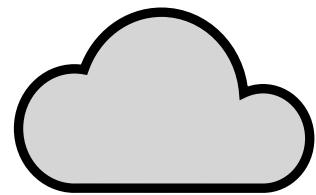
ISTD 50.012 Networks
Final Research Project
Term 6 – Fall 2021

Does it pay to be selfish?

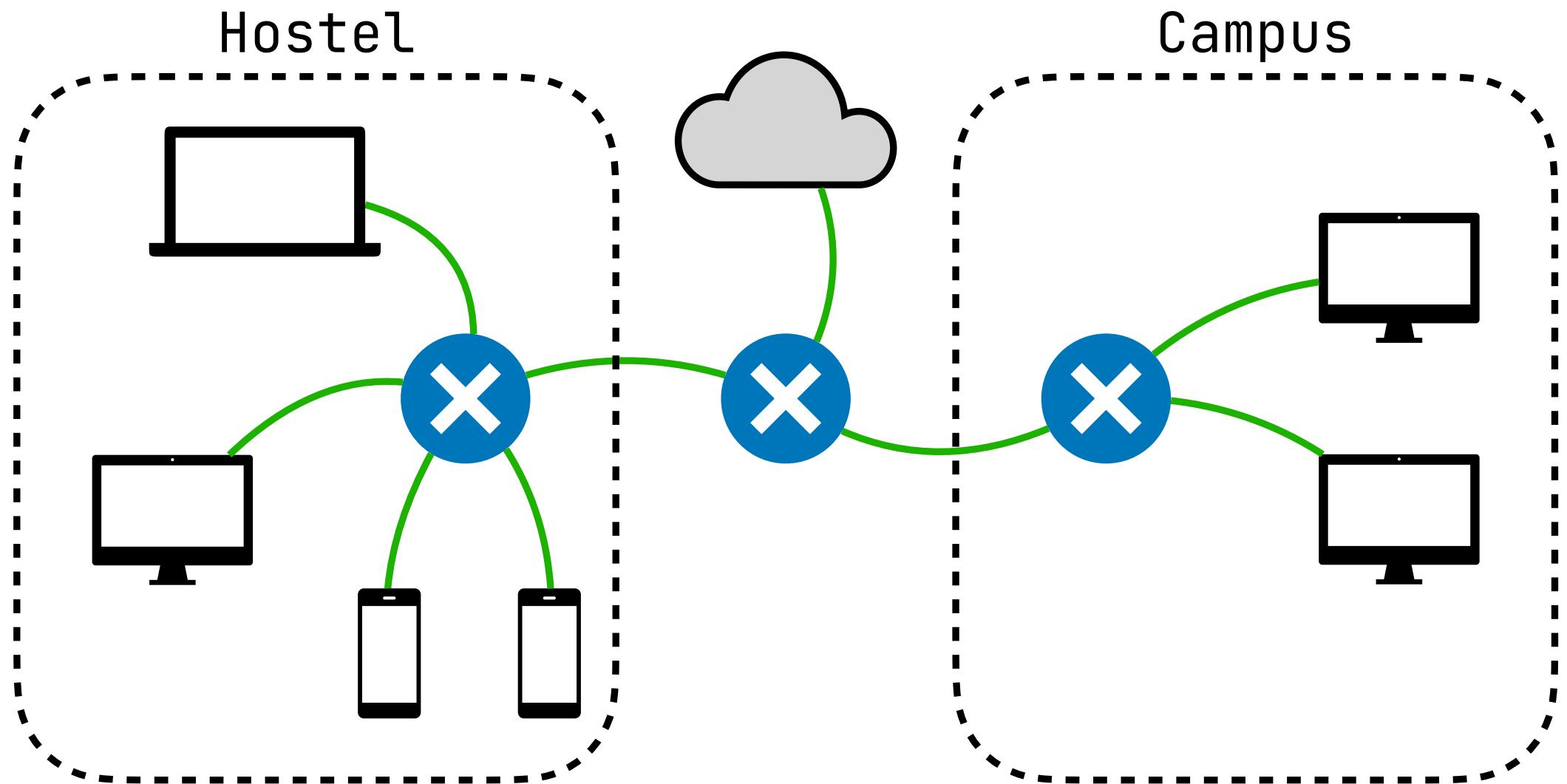
A study of congestion control abuse and countermeasures

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

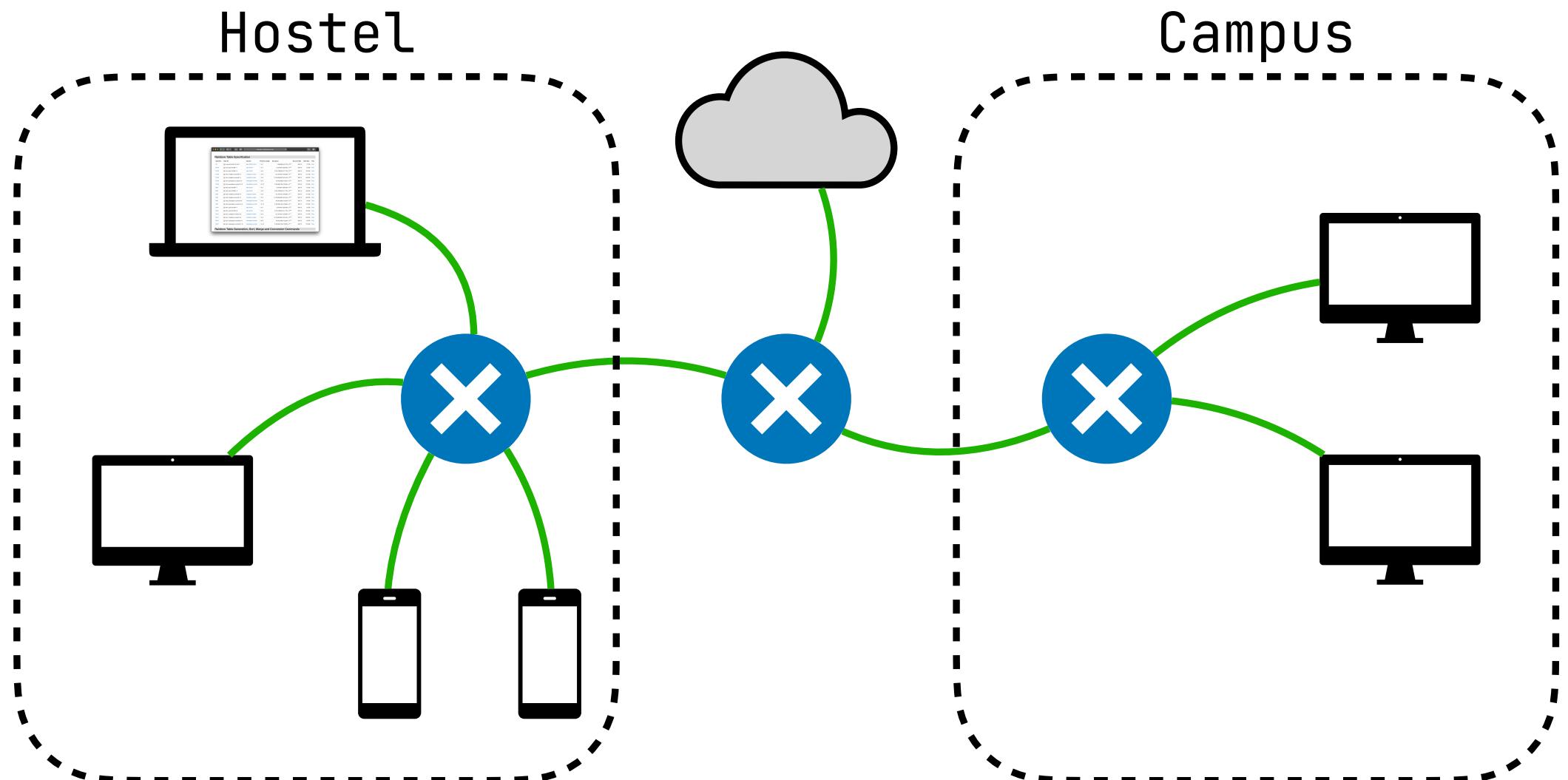
**** not the real SUTD network ****



**** not the real SUTD network ****



**** not the real SUTD network ****



**** not the real SUTD network ****

project-rainbowcrack.com

Rainbow Table Specification

Algorithm	Table ID	Charset	Plaintext Length	Key Space	Success Rate	Table Size	Files
LM	lm_ascii-32-65-123-4#1-7	ascii-32-65-123-4	1 to 7	$7,555,858,447,479 \approx 2^{42.8}$	99.9 %	27 GB	Files
NTLM	ntlm_ascii-32-95#1-7	ascii-32-95	1 to 7	$70,576,641,626,495 \approx 2^{46.0}$	99.9 %	52 GB	Files
NTLM	ntlm_ascii-32-95#1-8	ascii-32-95	1 to 8	$6,704,780,954,517,120 \approx 2^{52.6}$	96.8 %	460 GB	Files
NTLM	ntlm_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	$221,919,451,578,090 \approx 2^{47.7}$	99.9 %	127 GB	Files
NTLM	ntlm_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	$13,759,005,997,841,642 \approx 2^{53.6}$	96.8 %	690 GB	Files
NTLM	ntlm_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	$104,461,669,716,084 \approx 2^{46.6}$	99.9 %	65 GB	Files
NTLM	ntlm_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	$3,760,620,109,779,060 \approx 2^{51.7}$	96.8 %	316 GB	Files
MD5	md5_ascii-32-95#1-7	ascii-32-95	1 to 7	$70,576,641,626,495 \approx 2^{46.0}$	99.9 %	52 GB	Files
MD5	md5_ascii-32-95#1-8	ascii-32-95	1 to 8	$6,704,780,954,517,120 \approx 2^{52.6}$	96.8 %	460 GB	Files
MD5	md5_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	$221,919,451,578,090 \approx 2^{47.7}$	99.9 %	127 GB	Files
MD5	md5_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	$13,759,005,997,841,642 \approx 2^{53.6}$	96.8 %	690 GB	Files
MD5	md5_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	$104,461,669,716,084 \approx 2^{46.6}$	99.9 %	65 GB	Files
MD5	md5_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	$3,760,620,109,779,060 \approx 2^{51.7}$	96.8 %	316 GB	Files
SHA1	sha1_ascii-32-95#1-7	ascii-32-95	1 to 7	$70,576,641,626,495 \approx 2^{46.0}$	99.9 %	52 GB	Files
SHA1	sha1_ascii-32-95#1-8	ascii-32-95	1 to 8	$6,704,780,954,517,120 \approx 2^{52.6}$	96.8 %	460 GB	Files
SHA1	sha1_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	$221,919,451,578,090 \approx 2^{47.7}$	99.9 %	127 GB	Files
SHA1	sha1_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	$13,759,005,997,841,642 \approx 2^{53.6}$	96.8 %	690 GB	Files
SHA1	sha1_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	$104,461,669,716,084 \approx 2^{46.6}$	99.9 %	65 GB	Files
SHA1	sha1_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	$3,760,620,109,779,060 \approx 2^{51.7}$	96.8 %	316 GB	Files

Rainbow Table Generation, Sort, Merge and Conversion Commands

project-rainbowcrack.com

Rainbow Table Specification

Algorithm	Table ID	Charset	Plaintext Length	Key Space	Success Rate	Table Size	Files
LM	lm_ascii-32-65-123-4#1-7	ascii-32-65-123-4	1 to 7	$358,447,479 \approx 2^{12.8}$	99.9 %	27 GB	Files
NTLM	ntlm_ascii-32-95#1-7	ascii-32-95	1 to 7	$1,626,495 \approx 2^{46.0}$	99.9 %	52 GB	Files
NTLM	ntlm_ascii-32-95#1-8	ascii-32-95	1 to 8	$1,120 \approx 2^{52.6}$	96.8 %	460 GB	Files
NTLM	ntlm_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	$0 \approx 2^{47.7}$	99.9 %	127 GB	Files
NTLM	ntlm_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	$42 \approx 2^{51.5}$	96.8 %	690 GB	Files
NTLM	ntlm_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	$716,000 \approx 2^{53.6}$	99.9 %	65 GB	Files
NTLM	ntlm_loweralpha-numeric#1-11	loweralpha-numeric	3 to 11	$1,620,109,779,064 \approx 2^{51.7}$	96.8 %	316 GB	Files
MD5	md5_ascii-32-95#1-7	ascii-32-95	1 to 7	$1,626,495 \approx 2^{46.0}$	99.9 %	52 GB	Files
MD5	md5_ascii-32-95#1-8	ascii-32-95	1 to 8	$1,120 \approx 2^{52.6}$	96.8 %	460 GB	Files
MD5	md5_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	$0 \approx 2^{47.7}$	99.9 %	127 GB	Files
MD5	md5_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	$42 \approx 2^{53.6}$	96.8 %	690 GB	Files
MD5	md5_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	$42 \approx 2^{46.6}$	99.9 %	65 GB	Files
MD5	md5_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	$9,060 \approx 2^{51.7}$	96.8 %	316 GB	Files
SHA1	sha1_ascii-32-95#1-7	ascii-32-95	1 to 7	$1,626,495 \approx 2^{46.0}$	99.9 %	52 GB	Files
SHA1	sha1_ascii-32-95#1-8	ascii-32-95	1 to 8	$4,780,954,517,120 \approx 2^{52.6}$	96.8 %	460 GB	Files
SHA1	sha1_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	$221,919,451,578,090 \approx 2^{47.7}$	99.9 %	127 GB	Files
SHA1	sha1_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	$13,759,005,997,841,642 \approx 2^{53.6}$	96.8 %	690 GB	Files
SHA1	sha1_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	$104,461,669,716,084 \approx 2^{46.6}$	99.9 %	65 GB	Files
SHA1	sha1_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	$3,760,620,109,779,060 \approx 2^{51.7}$	96.8 %	316 GB	Files

Rainbow Table Generation, Sort, Merge and Conversion Commands

The screenshot shows a web browser window displaying the GitHub page for the `aria2` project. The URL in the address bar is `aria2.github.io`. The page has a dark header with the title "aria2" and the subtitle "The next generation download utility." Below the header, there is a navigation bar with links for "Top", "Documentation", "Документация", "Documentação", "Bugs", and "Source". A search bar and a RSS feed icon are also present in the header.

The main content area contains a paragraph about `aria2` being a **lightweight** multi-protocol & multi-source command-line **download utility**. It supports **HTTP/HTTPS**, **FTP**, **SFTP**, **BitTorrent** and **Metalink**. `aria2` can be manipulated via built-in **JSON-RPC** and **XML-RPC** interfaces.

Download

Download [version 1.36.0](#). There you can download source distribution and binaries for OS X, Windows and Android.

The legacy releases earlier than 1.19.1 are available [here](#).

Features

- **Multi-Connection Download.** `aria2` can download a file from multiple sources/protocols and tries to utilize your maximum download bandwidth. Really speeds up your download experience.
- **Lightweight.** `aria2` doesn't require much memory and CPU time. When disk cache is off, the physical memory usage is typically 4MiB (normal HTTP/FTP downloads) to

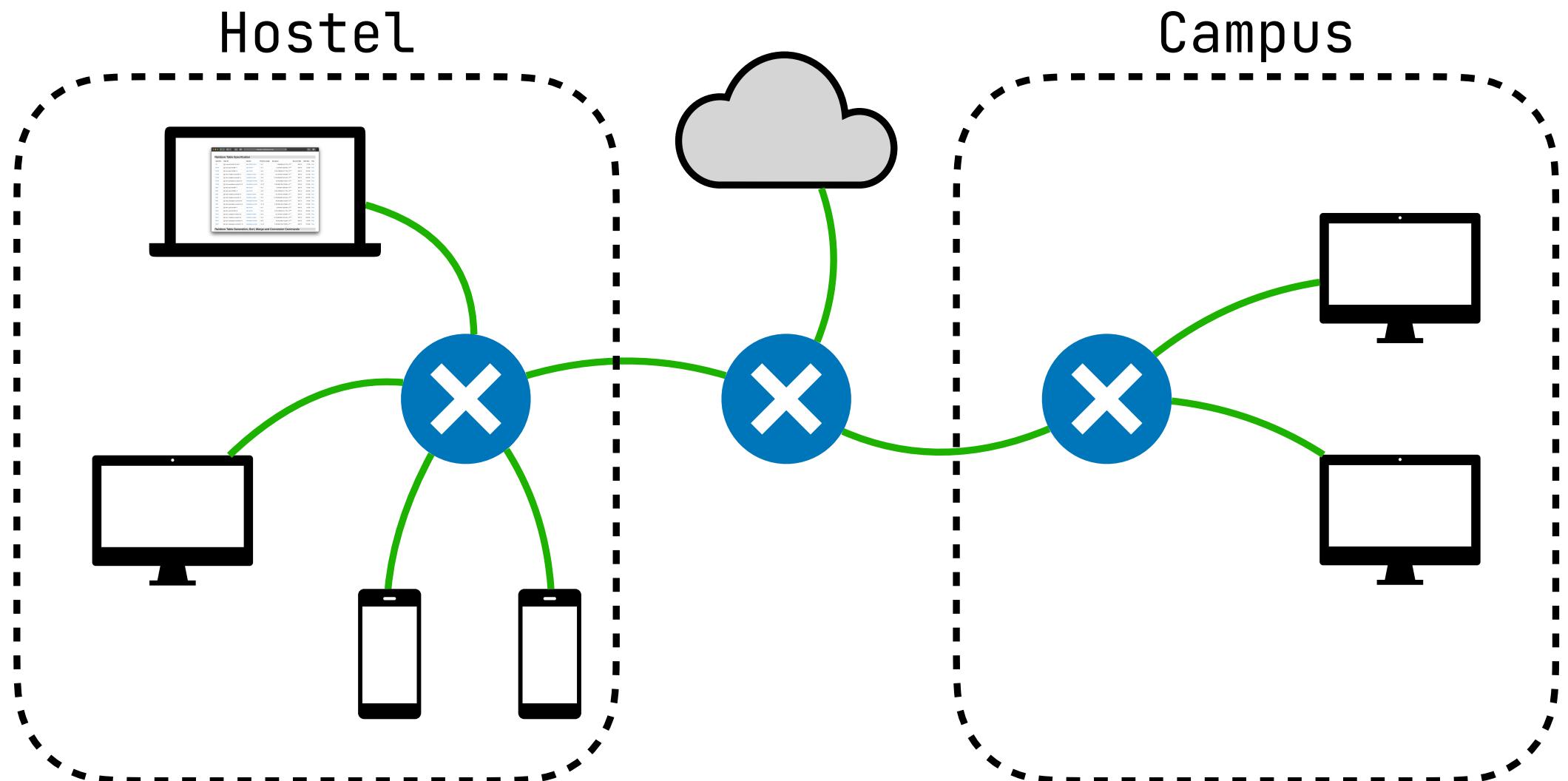
Stats

[Open Hub](#) **129K Lines**

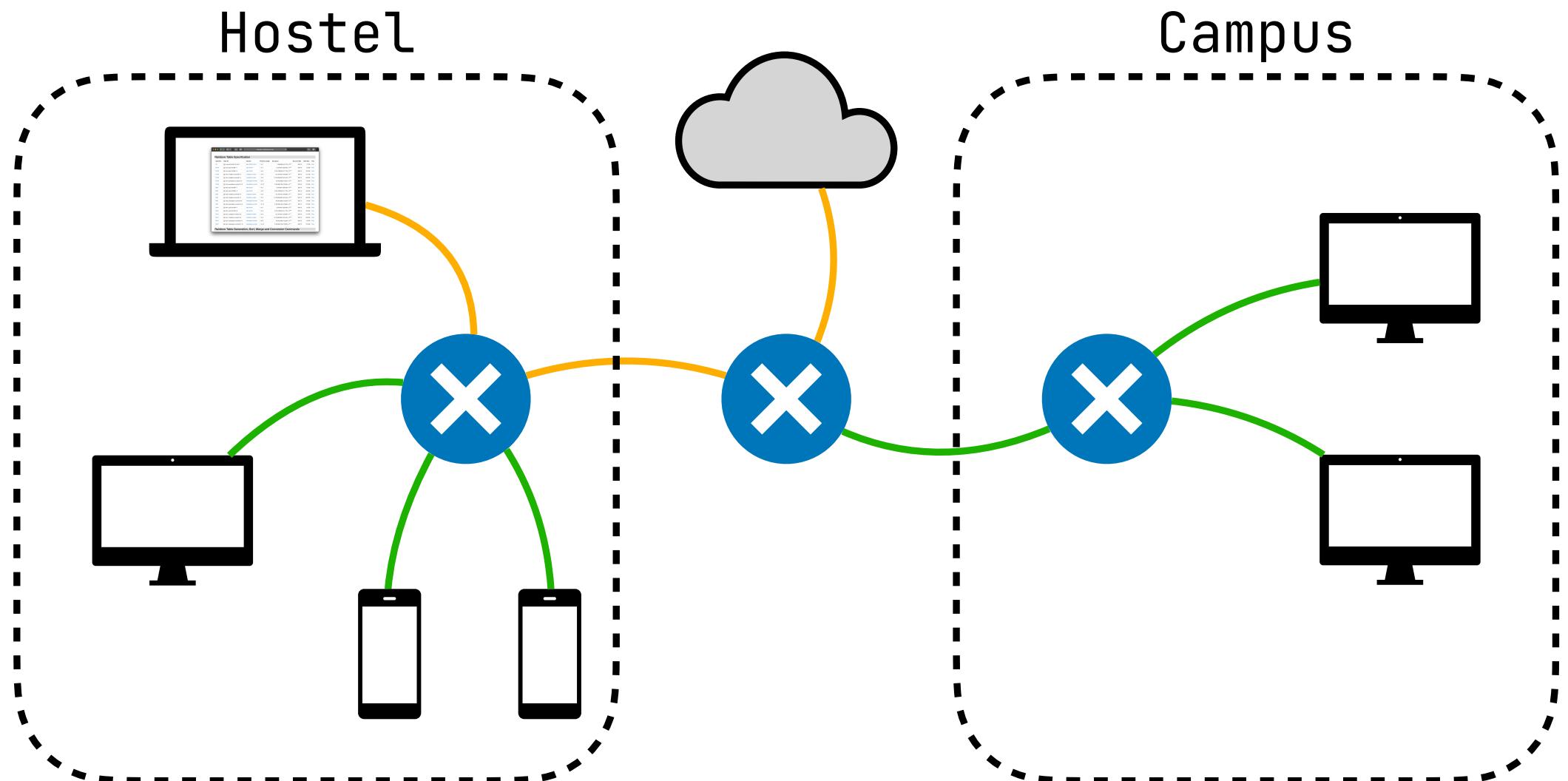
[Open Hub](#) **141 USERS**
I USE IT

Translation

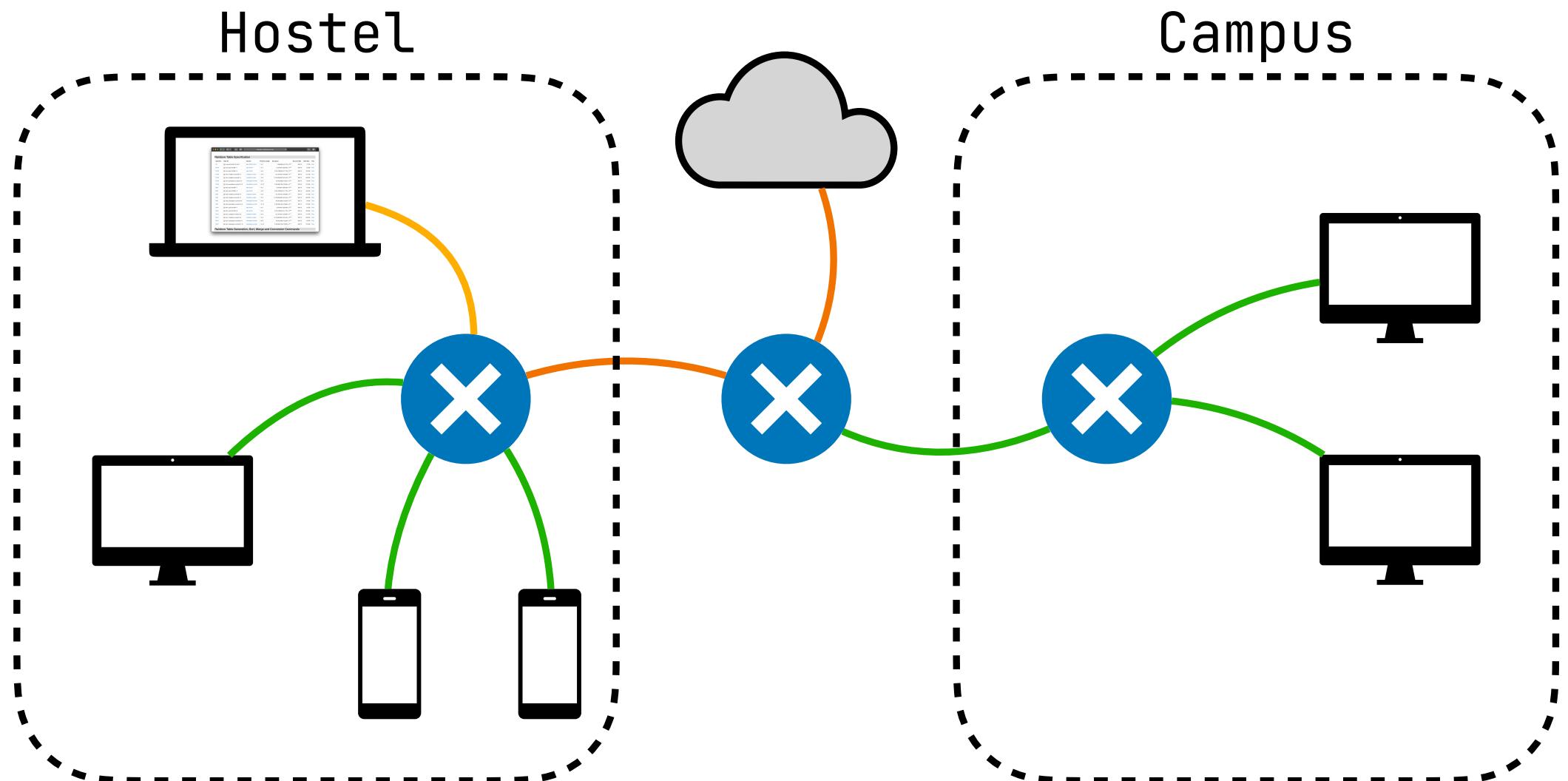
Localize `aria2` at [launchpad.net](#).



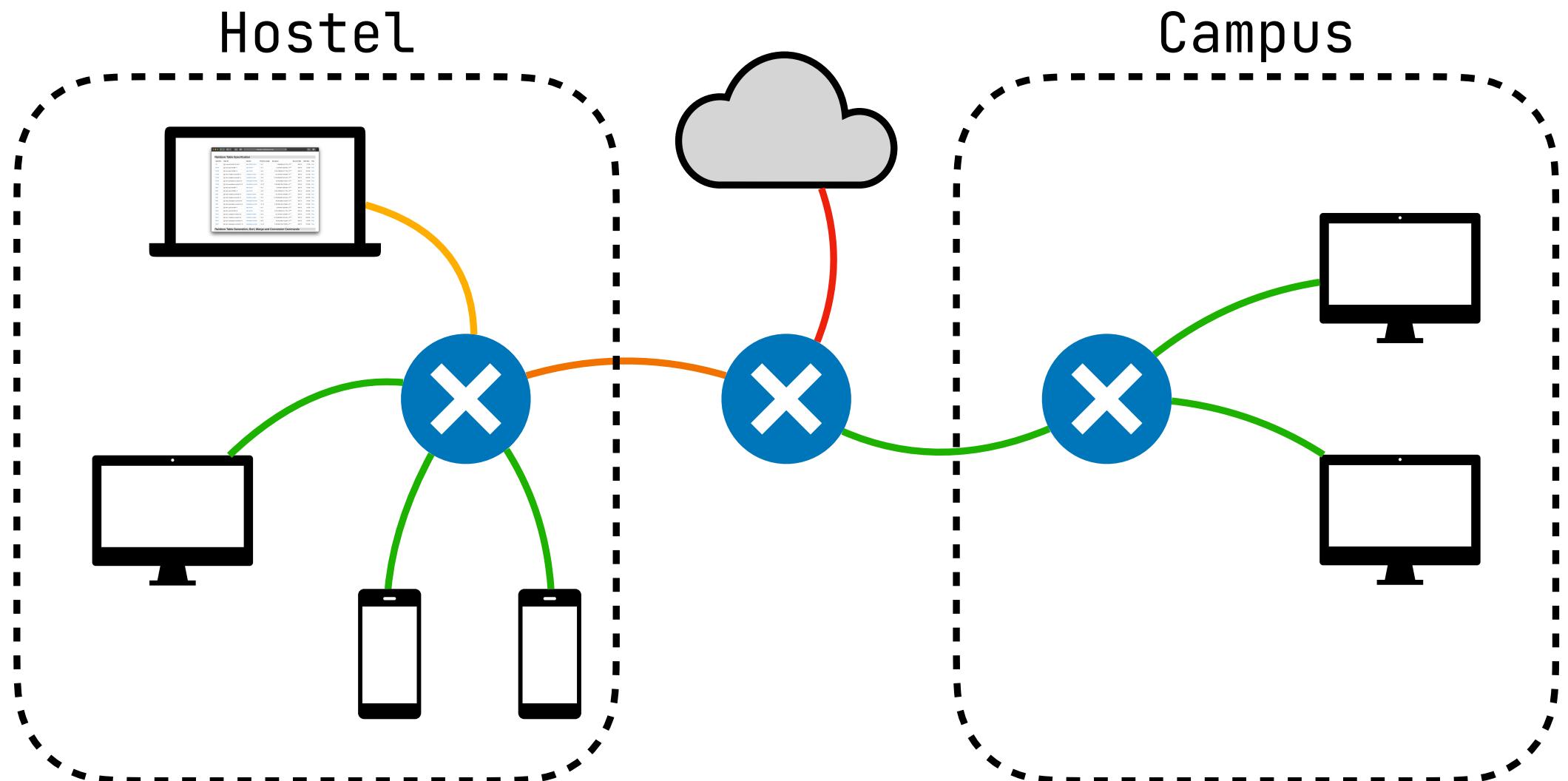
**** not the real SUTD network ****



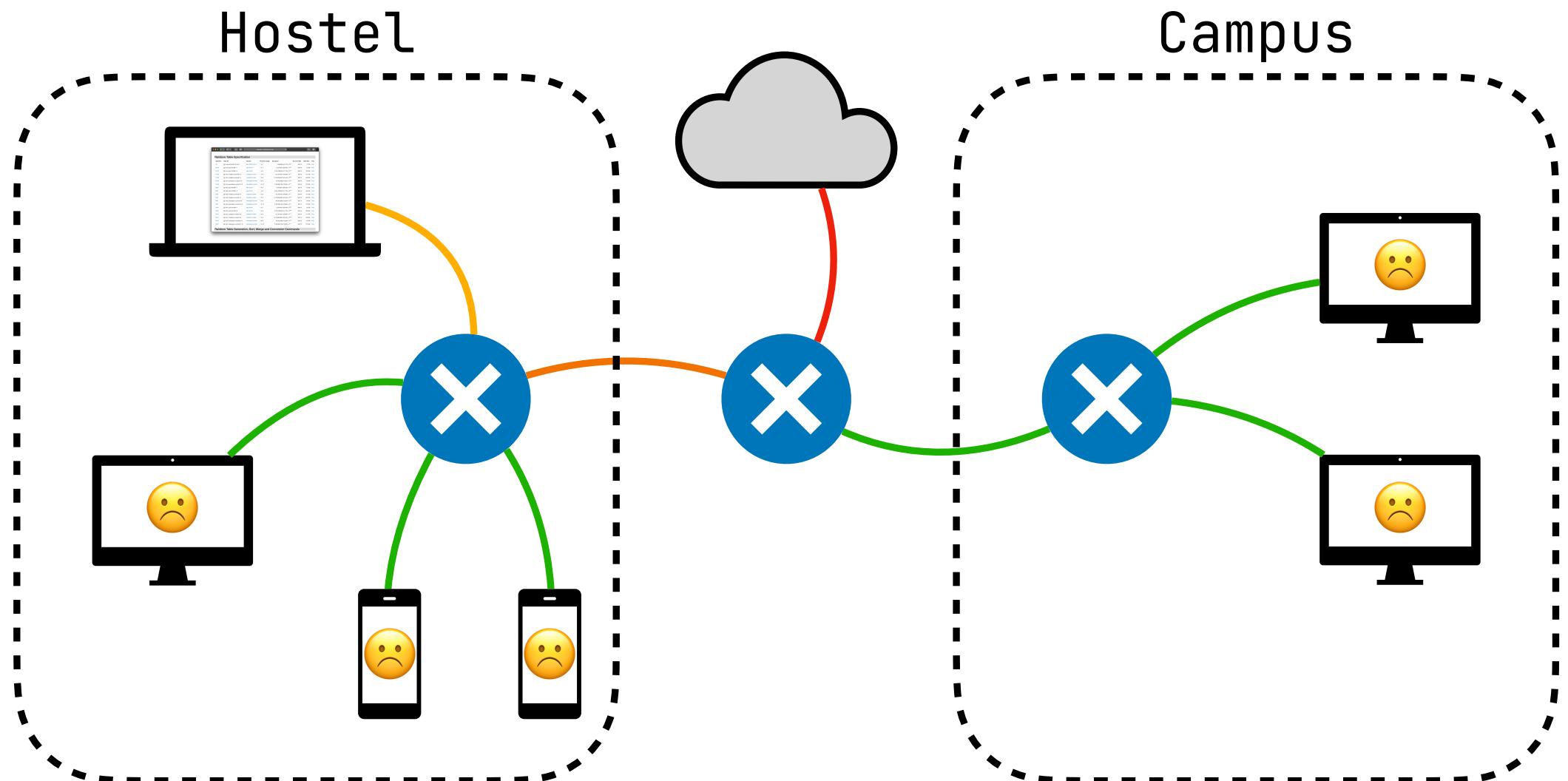
**** not the real SUTD network ****



**** not the real SUTD network ****



**** not the real SUTD network ****



**** not the real SUTD network ****

The screenshot shows a web browser window with the URL [en.wikipedia.org](https://en.wikipedia.org/wiki/Network_congestion) in the address bar. The page content is about network congestion avoidance, specifically focusing on TCP/IP avoidance. A pink highlight box is drawn around the first paragraph under the heading 'TCP/IP congestion avoidance'. The text in this box discusses the TCP congestion avoidance algorithm as the primary basis for congestion control on the Internet, mentioning references [8][9][10][11][12]. Below this, another paragraph explains problems with concurrent TCP flows experiencing tail-drops due to bufferbloat, leading to global synchronization. Further down, sections on Active queue management and Random early detection are listed.

Practical network congestion avoidance [edit]

Connection-oriented protocols, such as the widely used [TCP](#) protocol watch for [packet loss](#), or [queuing delay](#) to adjust their transmission rate. Various network congestion avoidance processes support different trade-offs.^[7]

TCP/IP congestion avoidance [edit]

The [TCP congestion avoidance algorithm](#) is the primary basis for congestion control on the Internet.^{[8][9][10][11][12]}

Problems occur when concurrent TCP flows experience [tail-drops](#), especially when [bufferbloat](#) is present. This delayed packet loss interferes with TCP's automatic congestion avoidance. All flows that experience this packet loss begin a TCP retransmit at the same moment – this is called [TCP global synchronization](#).

Active queue management [edit]

[Active queue management](#) (AQM) is the reordering or dropping of network packets inside a transmit buffer that is associated with a [network interface controller](#) (NIC). This task is performed by the [network scheduler](#).

Random early detection [edit]

One solution is to use [random early detection](#) (RED) on the network equipment's egress

https://en.wikipedia.org/wiki/Network_congestion#TCP/IP_congestion_avoidance

Problem Background

1. TCP congestion control is implemented at the edge of the network.
2. The responsibility lies in the hands of the client to carry out the algorithm in good faith.
3. As such, there is a possibility that it can be abused.

Problem Statement

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

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

Literature Review

"A mathematical model for the TCP Tragedy of the Commons." by López et al. proves that there exists a threshold whereby evil greediness will cause the game to take the form of "Tragedy of the Commons".

López, L., Almansa, G. d. R., Paquetelet, S., & Fernandéz, A. (2005, October 10). A mathematical model for the TCP Tragedy of the Commons. *Theoretical Computer Science*, 343(1-2), 4-26. <https://doi.org/10.1016/j.tcs.2005.05.005>

Literature Review

"TCP connection game: a study on the selfish behavior of TCP users" by Zhang et al. found that there is a Nash Equilibria and the overall network efficiency loss is bounded when given either a user's computation powers is limited or the user is socially responsible.

Zhang, H., Towsley, D., & Gong, W. (2005). TCP connection game: a study on the selfish behavior of TCP users. 13TH IEEE International Conference on Network Protocols (ICNP'05), 10 pp.-310. 10.1109/ICNP.2005.40}

Literature Review

"TCP congestion control with a misbehaving receiver" by Savage et al. explored 3 possible attacks that a receiver can exploit on the TCP Daytona congestion control behavior by abusing the frequency of ACKs the receiver sends back to the sender.

Savage, S., Cardwell, N., Wetherall, D. J., & Anderson, T. E. (1999, October 5). TCP congestion control with a misbehaving receiver. ACM SIGCOMM Computer Communication Review, 29(5), 71-78. 10.1145/505696.505704

Literature Review

"*Tolls for heterogeneous selfish users in multicommodity networks and generalized congestion games*" by Fleischer et al. proved the existence of tolls when heterogeneous network users independently choose routes minimizing their own linear function of tolls versus latency to collectively form the traffic pattern of a minimum average latency flow.

Fleischer, L., Jain, K., & Mahdian, M. (2004). *Tolls for heterogeneous selfish users in multicommodity networks and generalized congestion games*. Annual IEEE Symposium on Foundations of Computer Science, 45(1), 9. <https://ieeexplore.ieee.org/document/1366247>

Literature Review

"*Performance Analysis of TCP Congestion Control Algorithms*" by Truc et al. did a comparative study in which we present the results obtained by simulating multiple TCP variants: NewReno, Vegas, HighSpeed, Scalable, Westwood+, BIC, CUBIC, and YeAH using a fat tree architecture. Each protocol is evaluated in terms of queue length, number of dropped packets, average packet delay, and aggregate bandwidth as a percentage of the channel bandwidth.

Truc, Nguyen Nhat & Gangadhar, Siddharth & Sterbenz, James. (2016). *Performance Evaluation of TCP Congestion Control Algorithms in Data Center Networks.* 21-28.

Literature Review

1. We know a selfishness threshold exists. (López et al., 2005)
2. User's computation powers and socially responsibilities affect that threshold. (Zhang et al., 2005)
3. One method is a user can open many connections to obtain greater goodput at the Nash Equilibrium state. (Zhang et al., 2005)
4. Another method is strategically abusing the frequency of ACKs the receiver sends back to the sender. (Savage et al., 1999, 71-78)
5. Other network users will adjust to any disruptions caused by independently choosing routes minimizing their own delay / maximising their own goodput (Fleischer et al., 2004, 9)

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!)

Proposed Research Approach

The image shows two web browser windows side-by-side, comparing GNS3 and ns-3.

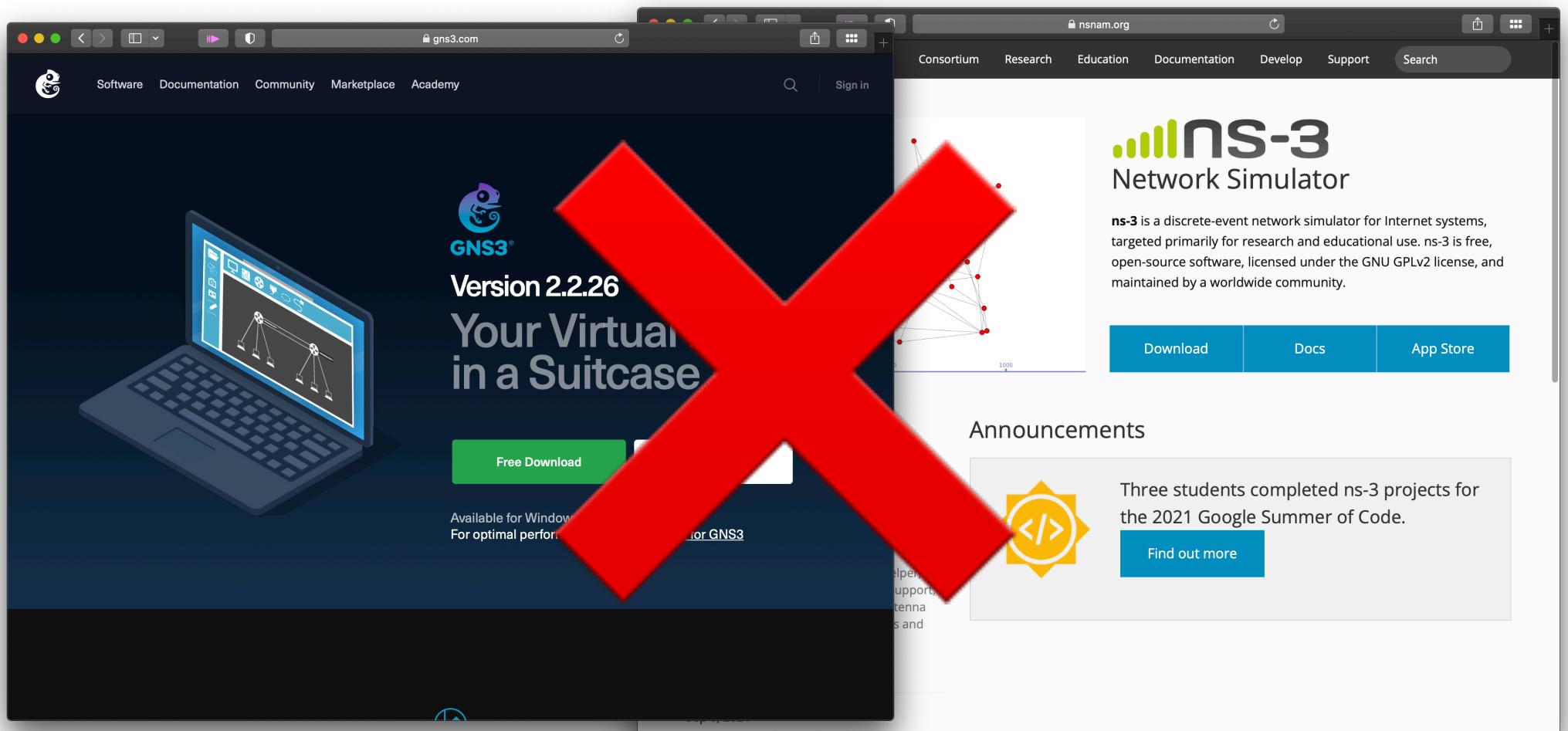
GNS3 (Left Window):

- Header:** Software, Documentation, Community, Marketplace, Academy, Sign in.
- Image:** A laptop displaying a virtual network interface with multiple nodes and connections.
- Text:** Version 2.2.26, Your Virtual Network in a Suitcase.
- Buttons:** Free Download, Watch Video.
- Text:** Available for Windows, Linux & Mac. For optimal performance, [Download VM for GNS3](#).

ns-3 (Right Window):

- Header:** Consortium, Research, Education, Documentation, Develop, Support, Search.
- Image:** A network graph visualization showing red nodes connected by white lines.
- Text:** **ns-3** Network Simulator.
- Text:** ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for research and educational use. ns-3 is free, open-source software, licensed under the GNU GPLv2 license, and maintained by a worldwide community.
- Buttons:** Download, Docs, App Store.
- Section:** Announcements.
- Image:** A yellow sun icon with a double slash symbol inside it.
- Text:** Three students completed ns-3 projects for the 2021 Google Summer of Code.
- Button:** Find out more.

Proposed Research Approach



Proposed Research Approach

Not Secure — mininet.org

Mininet

An Instant Virtual Network on your Laptop (or other PC)

Mininet creates a **realistic virtual network**, running **real kernel, switch and application code**, on a single machine (VM, cloud or native), in seconds, with a single command:

> sudo mn

Because you can easily [interact with](#) your network using the Mininet [CLI](#) (and [API](#)), [customize it](#), [share](#) it with others, or [deploy](#) it on real hardware, Mininet is useful for [development](#), [teaching](#), and [research](#).

Mininet is also a great way to develop, share, and experiment with Software-Defined Networking (SDN) systems using [OpenFlow](#) and [P4](#).

Mininet is actively developed and supported, and is released under a permissive BSD Open Source [license](#). We encourage you to [contribute](#) code, bug reports/fixes, documentation, and anything else that can improve the system!

Get Started
Download a Mininet VM, do the [walkthrough](#) and run the [OpenFlow tutorial](#).

Support
Read the [FAQ](#), read the [documentation](#), and join our mailing list, [mininet-discuss](#).

Contribute
File a [bug](#), download the [source](#), or submit a [pull request](#) - all on GitHub.

Mininet
[Get Started](#) [Sample Workflow](#) [Walkthrough](#) [Overview](#)

[Download](#) [Documentation](#) [Papers](#) [Videos](#) [Wiki](#) [Source Code](#) [CI Testing](#) [Apps and Tools](#) [Teaching](#) [FAQ](#)

[Support](#) [Contribute](#) [News Archive](#) [Credits](#)

News
[Announcing Mininet 2.3.0!](#) [Mininet Wins ACM SIGCOMM Test of Time Paper Award](#) [Congrats to Mininet SIGCOMM Hackathon Winners!](#) [Open Networking Foundation Promotes Mininet](#) [Mininet Wins ACM SIGCOMM SOSR Software Systems Award !](#)

Copyright © 2021 Mininet Project Contributors - powered by Octopress

1004101_Networks_Lab4 (page 9 of 10)

50.012 Networks Lab 4 V S Ragul Balaji (1004101)

Switch Queue Occupancy

TCP CWND for iperf

TCP CWND for wget

TCP CWND for wget

When the buffer is reduced from 100 packets to 20

Proposed Research Approach

The image shows a large green checkmark graphic overlaid on two screenshots. On the left is the official Mininet website (mininet.org) displaying its homepage with sections for 'Get Started', 'Support', and 'Contribute'. On the right is a PDF page titled '50.012 Networks Lab 4' by 'V S Ragul Balaji (1004101)' containing three line graphs under the heading 'FIGURES' showing 'Switch Queue Occupancy', 'TCP CWND for iperf', and 'TCP CWND for wget'.

Mininet
An Instant Virtual Network on your Laptop (or other PC)

Mininet creates a **realistic virtual network**, running **real kernel, switch and application code**, on a single machine (VM, cloud or native), in seconds, with a single command:

> sudo mn →

Because you can easily [interact with](#) your network using the Mininet [CLI](#) (and [API](#)), [customize](#) it, [others](#), or [deploy](#) it on real hardware, Mininet is useful for [development](#), [teaching](#), and [research](#).

Mininet is also a great way to develop, share, and experiment with Software-Defined Networking (SDN) using [OpenFlow](#) and [P4](#).

Mininet is actively developed and supported, and is released under a permissive BSD Open Source license. We encourage you to [contribute](#) code, bug reports/fixes, documentation, and anything else that can help the system!

Get Started
Download a Mininet VM, do the [walkthrough](#) and run the [OpenFlow tutorial](#).

Support
Read the [FAQ](#), read the [documentation](#), and join our mailing list, [mininet-discuss](#).

Contribute
File a [bug](#), download the [source](#), or submit a [pull request](#) - all on GitHub.

Open Networking Foundation Promotes Mininet
Mininet Wins ACM SIGCOMM SOSR Software Systems Award !

50.012 Networks Lab 4
V S Ragul Balaji (1004101)

FIGURES

Switch Queue Occupancy

TCP CWND for iperf

TCP CWND for wget

When the buffer is reduced from 100 packets to 20

Expected Deliverables

#	Attack Technique	Optimal Parameter Value	Optimal Parameter Type	TCP / UDP Variant (if any)
1	<i>Opening many parallel connections</i>	10	<i>Connections</i>	<i>TCP Reno</i>
2	<i>Receiver abuse frequency of ACKs sent</i>	42	<i>ACKs per second</i>	<i>TCP Tahoe</i>
3	<i>Opening many parallel connections</i>	69	<i>Connections</i>	<i>TCP Tahoe</i>
4	<i>Opening many parallel connections</i>	420	<i>Connections</i>	<i>UDP</i>
...

** these are obviously not real results **

Questions?