

Approximating a Global Passive Adversary against Tor

Sambuddho Chakravarty , Columbia University , NY
Angelos Stavrou, George Mason University , VA
Angelos D. Keromytis, Columbia University , NY

Objective of Our Work

- Identifying Tor Clients and/or Hidden Services participating in an anonymous connection using a traffic analysis technique.
- Emulating a **Global Passive Adversary (GPA)** which performs this traffic analysis.
 - Induce bandwidth fluctuations in a Tor circuit
 - Detecting these at various routers and relays using a single-end controlled tool (**LinkWidth**).

Outline of the Presentation

- Overview of the Tor Architecture
- Previous Traffic Analysis Attacks Against Tor
- Our Approach : Traffic Analysis using LinkWidth
- Evaluation
 - Effectiveness of LinkWidth
 - "Linking" Tor Relays in a Circuit
 - "Tracing" the path of a Tor client communicating to a Tor Hidden Service
- Conclusion & Future Work

Overview of Tor Architecture

What is Tor ?

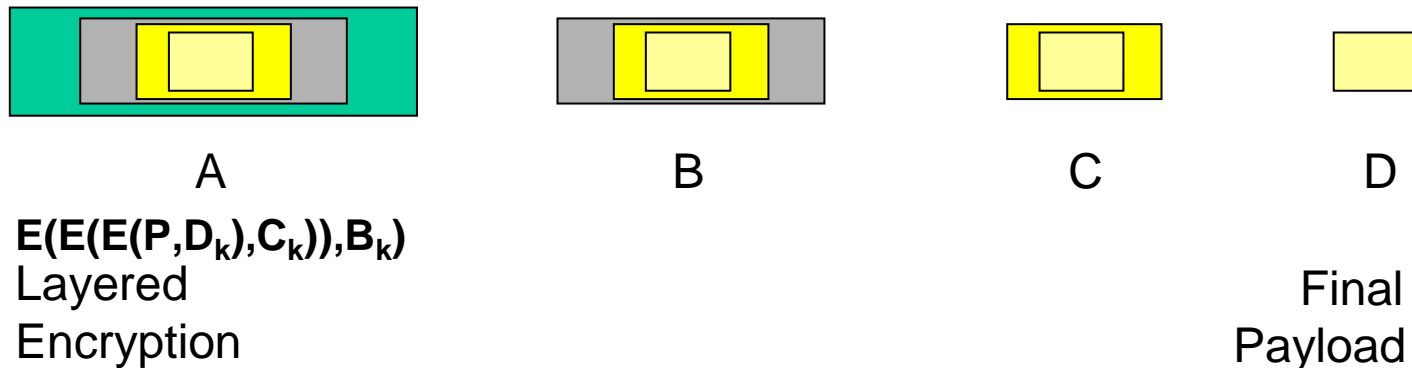
- The Onion Router
- Network Layer Anonymity System

Key Features of Tor

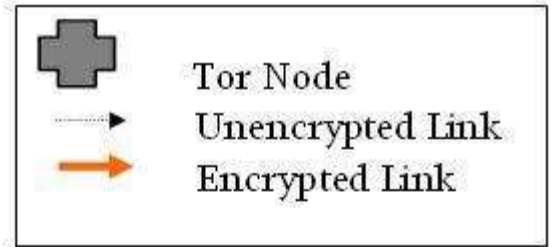
- Network of proxies
- Uses "3-hop" relays (ORs) by default - *Entry Node* , *Middleman* and *Exit Node* .
 - * A circuit can be extended to as many hops as possible.
- Many connections are multiplexed over the **same Tor Circuit**.

Key Features of Tor Contd...

- SSL encrypted connections to the the Entry Node , Middle Man and Exit Node
- No single Onion Router (OR) knows the entire circuit.
(Telescopic Layered Encryption (just like other anonymity systems))
 - Client encrypts and encapsulates the payload incrementally using the public keys (Onion Keys) of the ORs which it selects.



Anonymous Tor Circuit - Client-Server Circuit Set-up



Alice

Client (Alice) fetches the directory listing of ORs from the directory service (Dave) (hard-wired in the source)



Dave

<IP address> : 9002



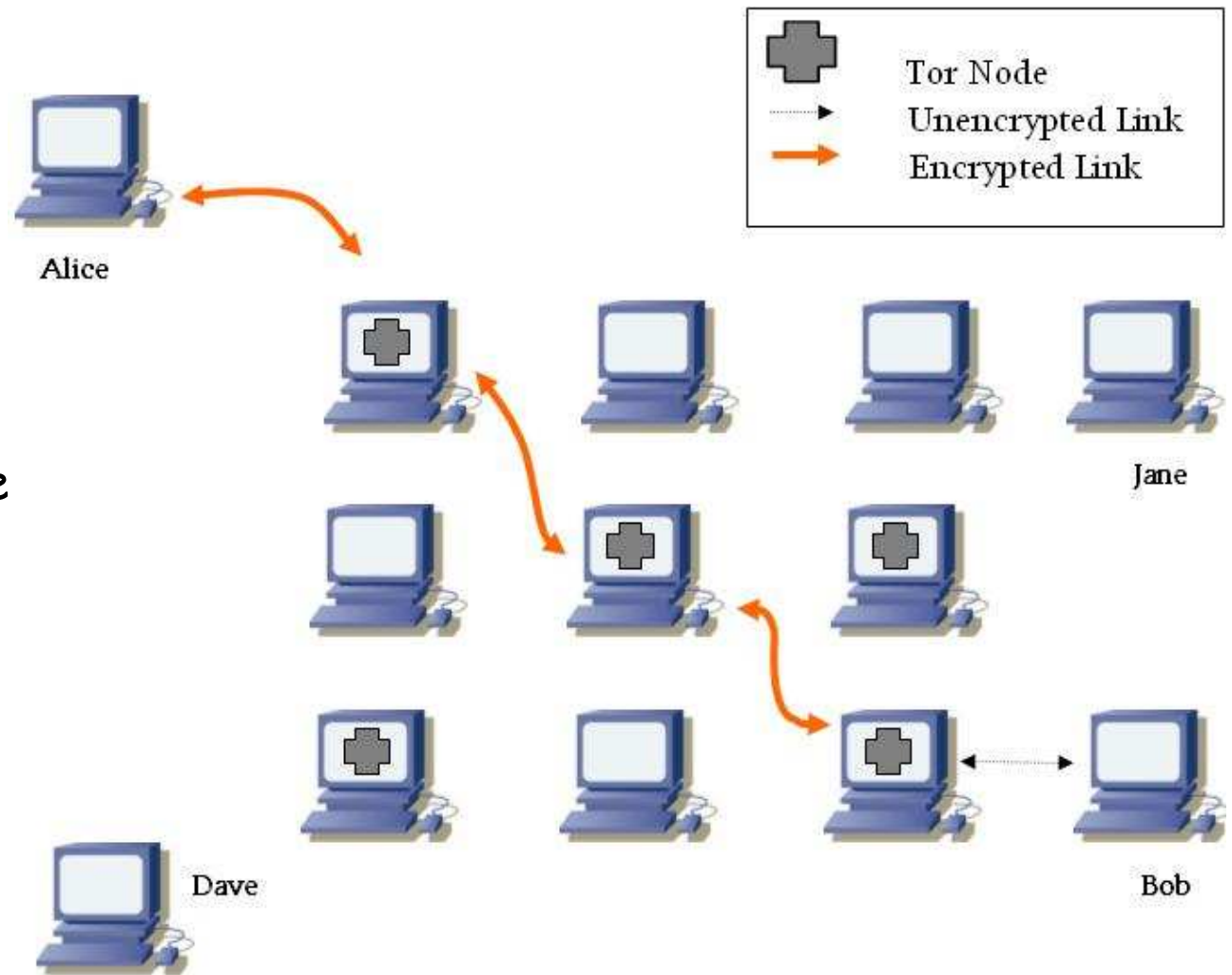
Jane



Bob

Tor Circuit Creation Continued...

Alice establishes a SSL connection to the Tor Entry Node and makes it extend the connection to the Middle Man and Exit Node.



Key Features of Tor Contd...

- Doesn't guarantee defence against a true GPA.

Focus : Performance of interactive TCP/IP applications, eg. www. (There are anyways not many GPAs).

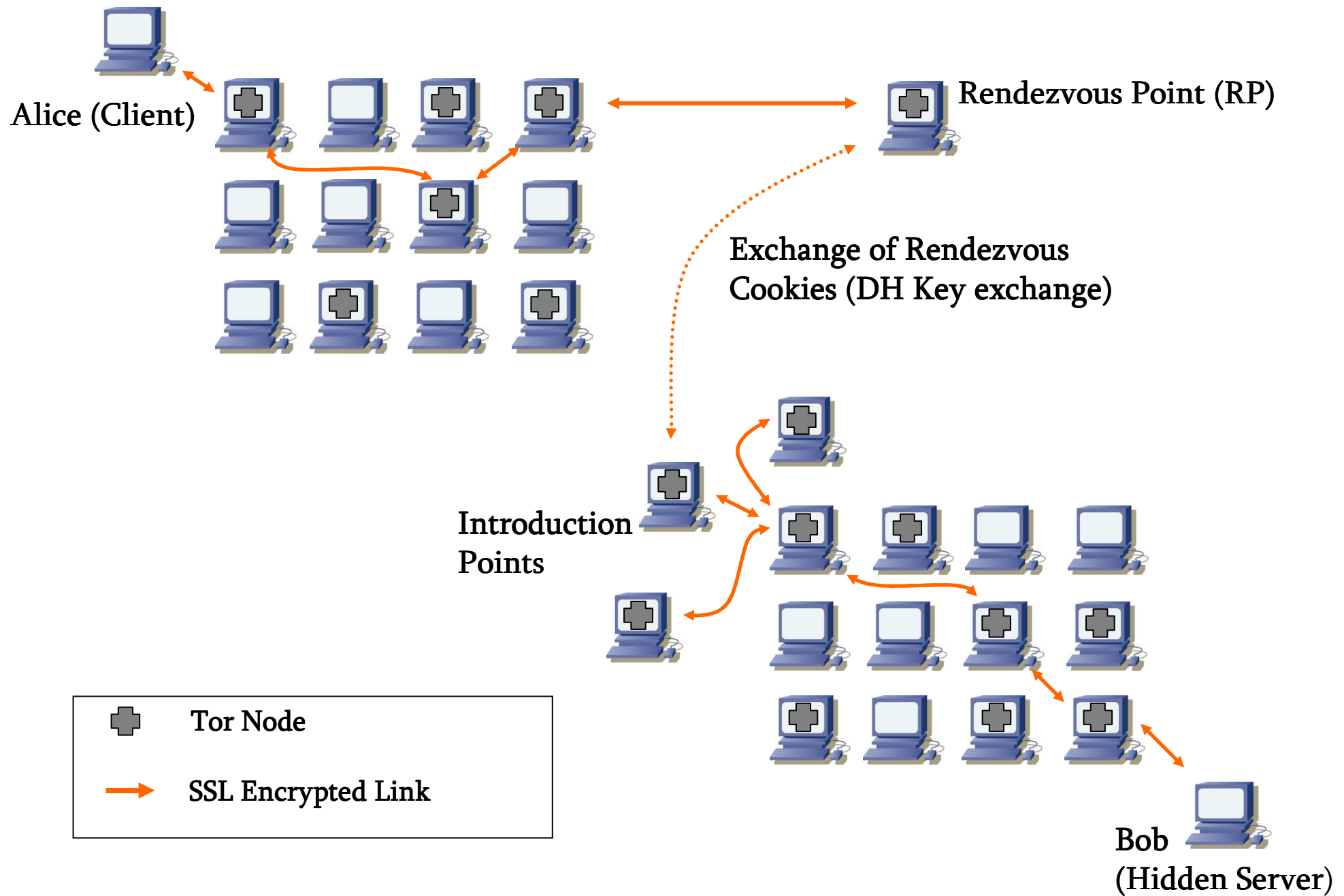
- Hidden Service (*responder anonymity*) :

- Service not accessible for an outsider

- Tor creates a new URI for the server (a string , NOT DNS NAME) within the .onion domain.

- (ex. : <http://qtlyzdk236qalfmq.onion/>)

Tor Client Establishing Circuit with a Hidden Service



Previous Attacks on Tor

- *CCS '07 : "How much anonymity does network latency leak ?" [2]*
 - Machine learning techniques to predict the host identity using One Way Delay (OWD). (uses MIT King data set [7] as training set . Assumption : *No background noise*)
 - Proposes using one-way latency / RTT variation in interactive user level applications such as IRC

Drawbacks:

- Possibly high false positives/negatives because One Way Delay only gives a "range" of probable hosts and not the exact host.
- Only suitable to determine the end-points of a communication; cannot be used to detect the network layer routers involved in circuit creation.

- IEEE S&P '05 : *"Low Cost Traffic Analysis of Tor"* [1]

- Victim client communicates to the server using a client - server

program (which measures the one way latency

(OWD)).

- Adversary uses a modified Tor client to establish a single hop Tor circuit to the victim relay.

- Adversary uses the same program to measure the OWD "through" the victim OR.

- Adversary's probes are distorted in the presence of client-server traffic flowing through the victim OR.

Drawbacks:

- Different network bottlenecks between the client and the server and the adversary OR and its victim OR may result in False Negatives.
- Only attempts to link Tor Relays participating in a client circuit .
- Fail to address network congestion which interfere with adversarial

Emulating a GPA

- Only a true GPA having knowledge of all the links can accurately say "who" communicates to "whom" via "whom".
- We are not a GPA. But what is the closest we can get ?
 - Induce traffic fluctuations through a Tor connection (through a client or server under the adversary's control) and detect it across IP routers and ORs when probed by an outsider.
 - How do we detect this bandwidth fluctuation : *LinkWidth*



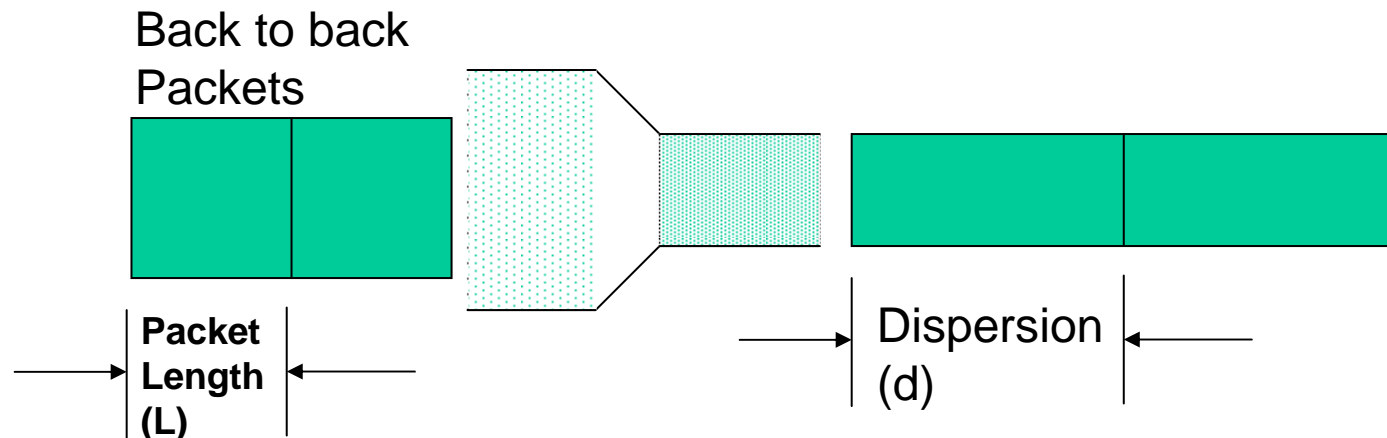
Using our technique, an adversary can probe both Tor Relays and routers along the underlying IP network, involved in an anonymous circuit.

Common Bandwidth Estimation Techniques

Packet pair method

- First proposed by S. Keshav [3]. as a part of his PhD dissertation
- Send a pair of "back-to-back" packets to the destination. The pair "spreads" in time. This spread is called *dispersion*.

$$BW = (\text{Packet Length} * 8) / (\text{dispersion}) \quad (\text{measured in bps})$$



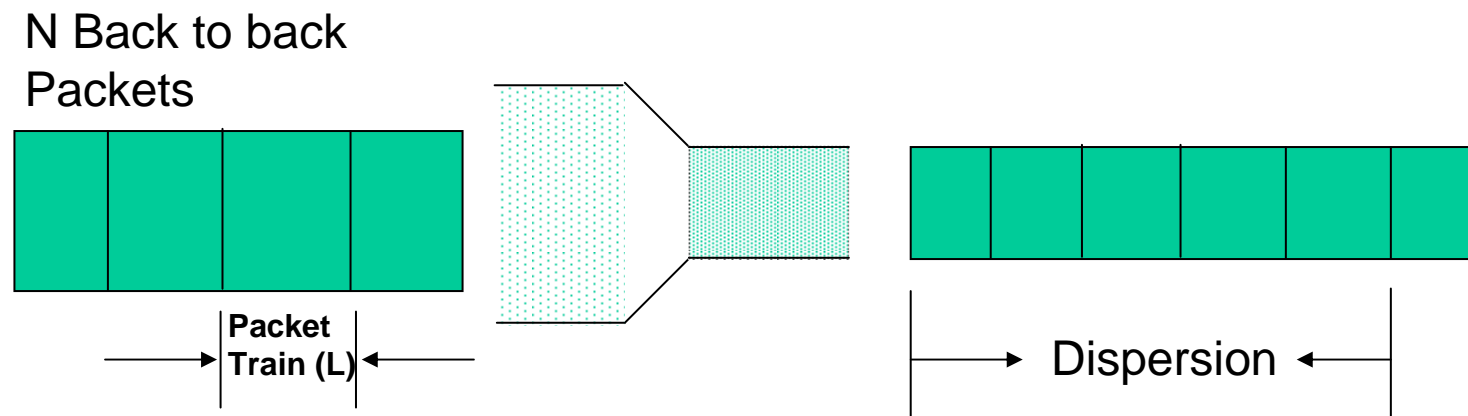
Packet pair goes from high-capacity link to a low-capacity link

Common Bandwidth Estimation Techniques

Packet train method:

- Augments packet pair method by sending 'n' back-to-back packets to the destination [5]
- Advantages: Less errors in the presence of noisy background traffic (equivalent to many independent packet-pair measurements)

$$BW = (\text{Packet Length} * n * 8) / (\text{dispersion}) \quad (\text{measured in bps})$$



Packet pair goes from high-capacity link to a low-capacity link

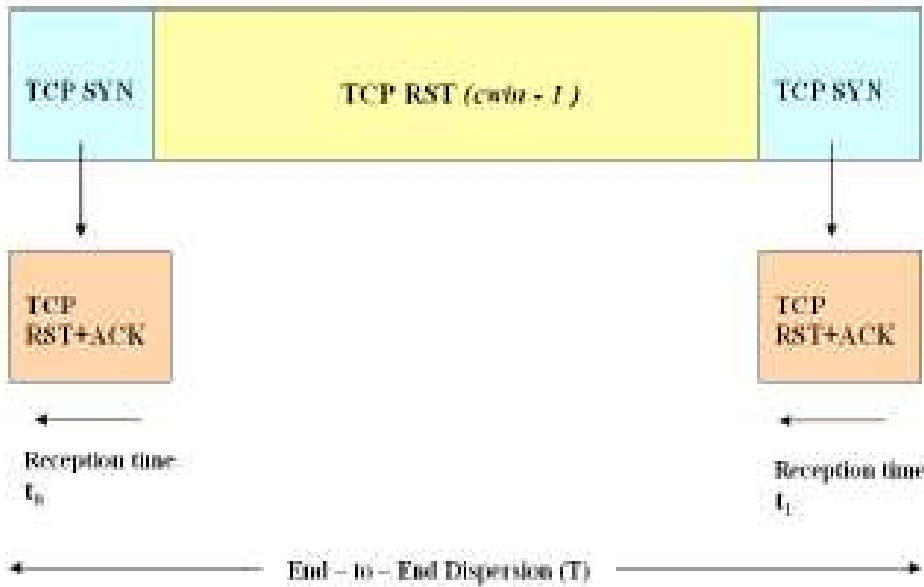
LinkWidth : Single End Controlled Available Bandwidth and Throughput estimation tool

- Emulates TCP Westwood sender in order measure Available Bandwidth and Throughput
 - TCP Westwood [4] uses bandwidth estimation every RTT seconds for adjusting the TCP *slow start threshold* (ssthresh) whenever congestion is detected
 - Congestion Detection : Coarse Timeout
 - Intuition: Available bandwidth / throughput measurement using a variable length packet train (train length = TCP *congestion window* (cwin)) .

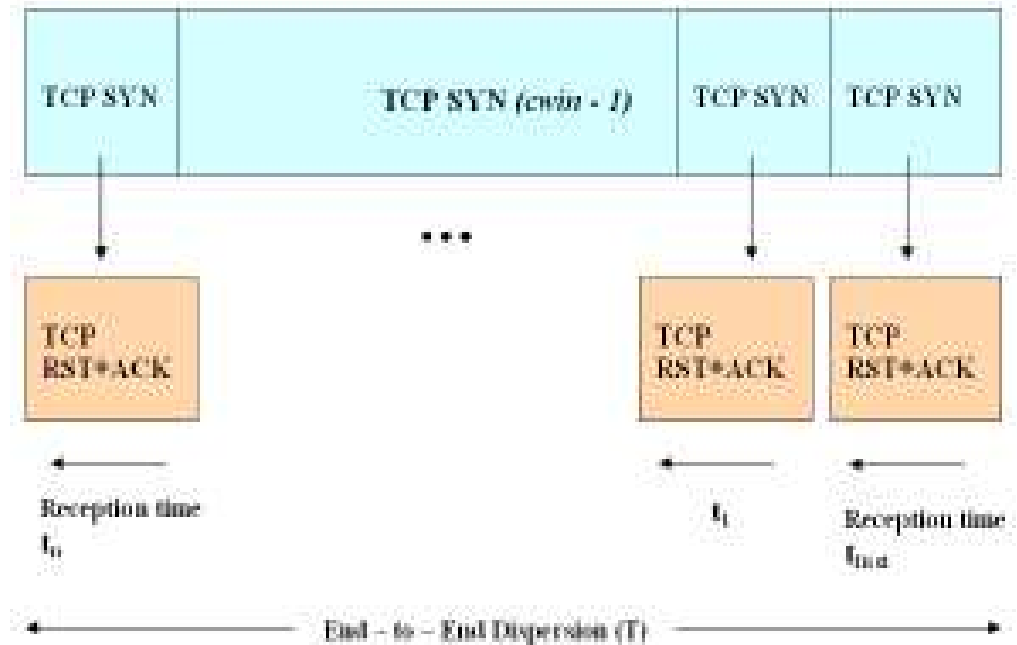
How LinkWidth Works

- LinkWidth emulates a TCP Westwood sender
- A train of TCP RST packets "sandwiched" between TCP SYN packets
- TCP SYN packets are destined to closed ports. These evoke TCP RST+ACK replies from the receiver (*Hence LinkWidth can be used from a single host*)
- Wherever TCP is filtered/rate-limited, we replace the TCP SYN packets with ICMP ECHO packets

Arrangement of Packets in LinkWidth (TCP)

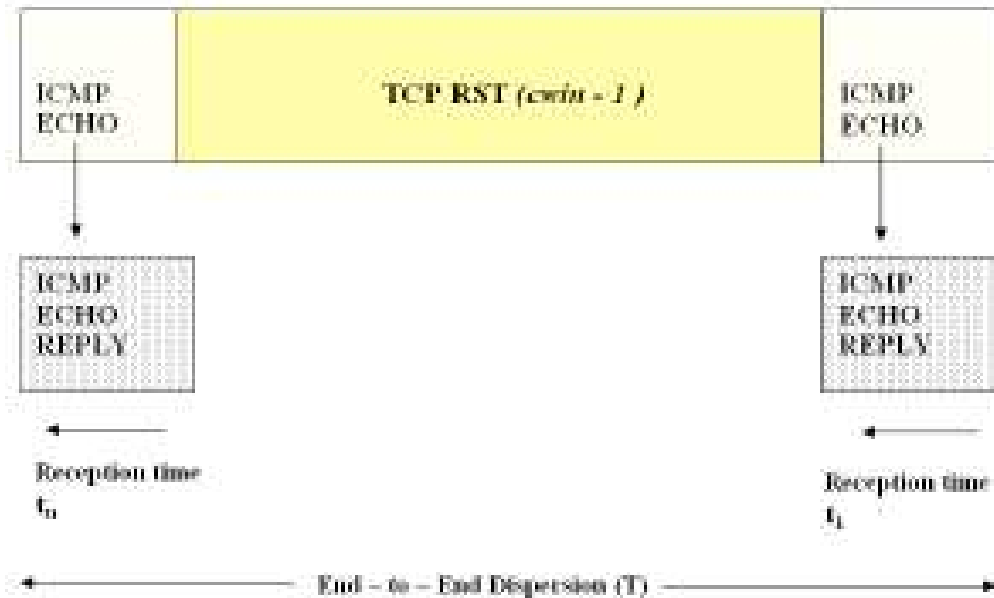


Arrangement of packets for measurement of available bandwidth (using TCP)

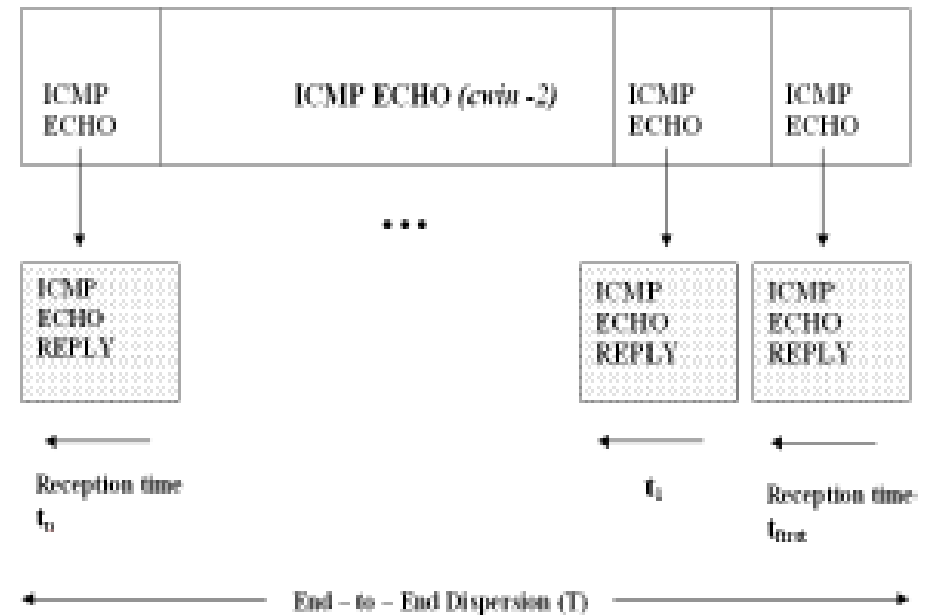


Arrangement of packets for measurement of throughput (using TCP)

Arrangement of Packets in LinkWidth (ICMP)

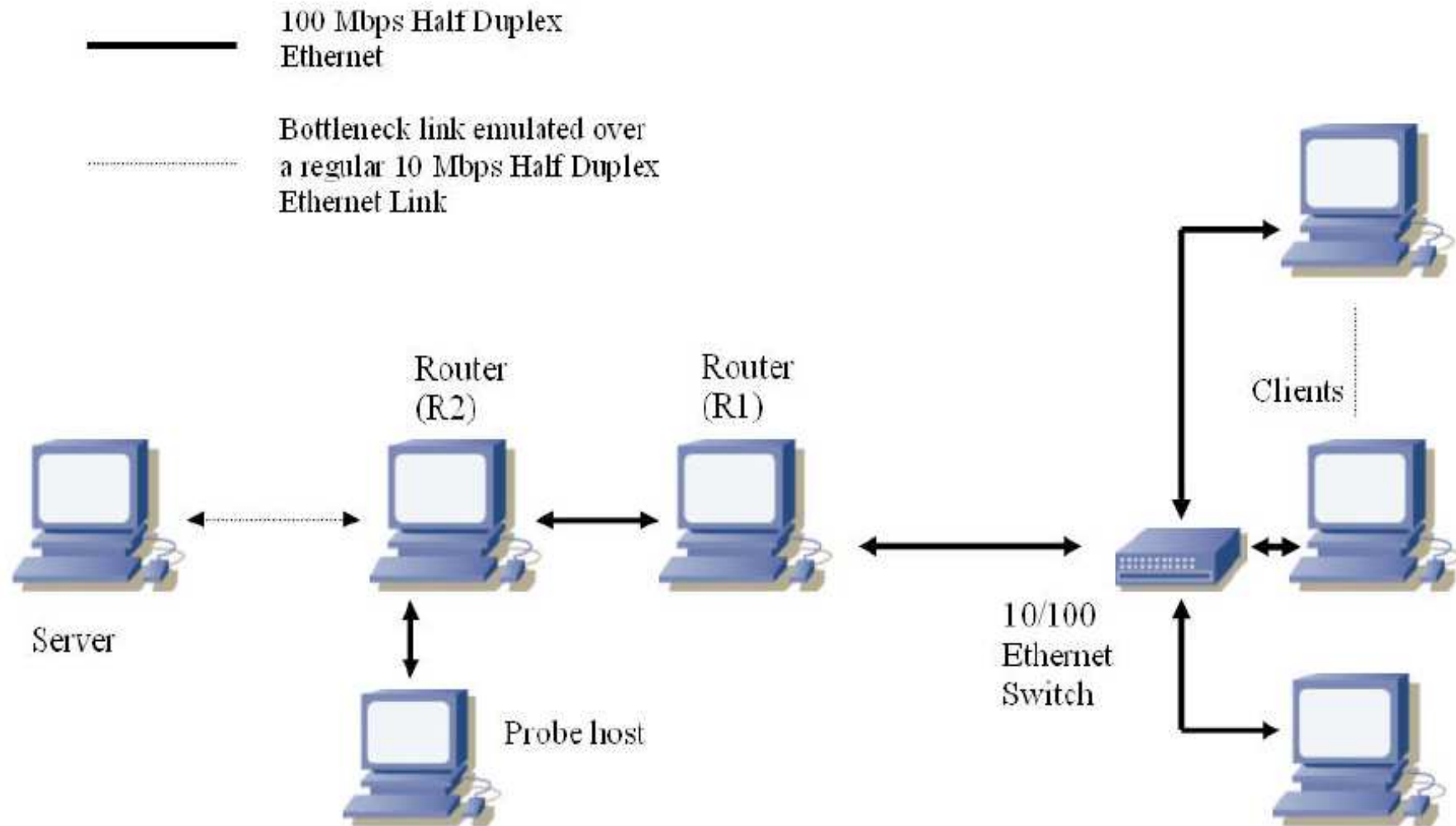


Arrangement of packets for measurement of available bandwidth (using ICMP)



Arrangement of packets for measurement of throughput (using ICMP)

Experiment to Test Effectiveness of LinkWidth



Experimental setup for testing of LinkWidth's effectiveness within a controlled environment

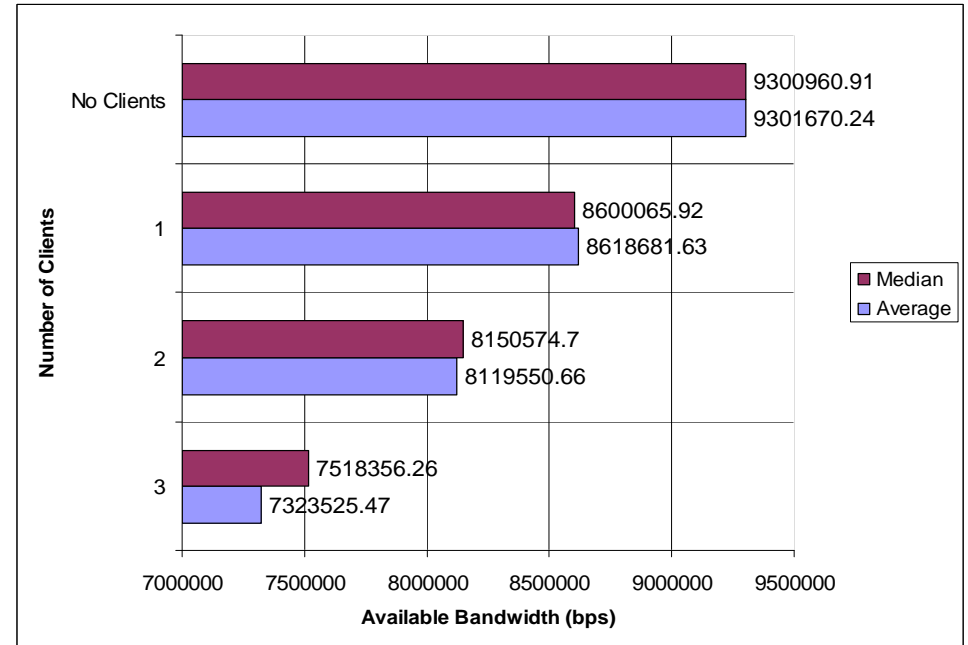
Experiment to Test Effectiveness of LinkWidth

Contd. ...

Experiment 1 (Test for accuracy)

Medium : 10 Mbps Ethernet Link being shared by 3 simultaneous http connections .

Max. bandwidth/throughput per connection : 500 Kbps.

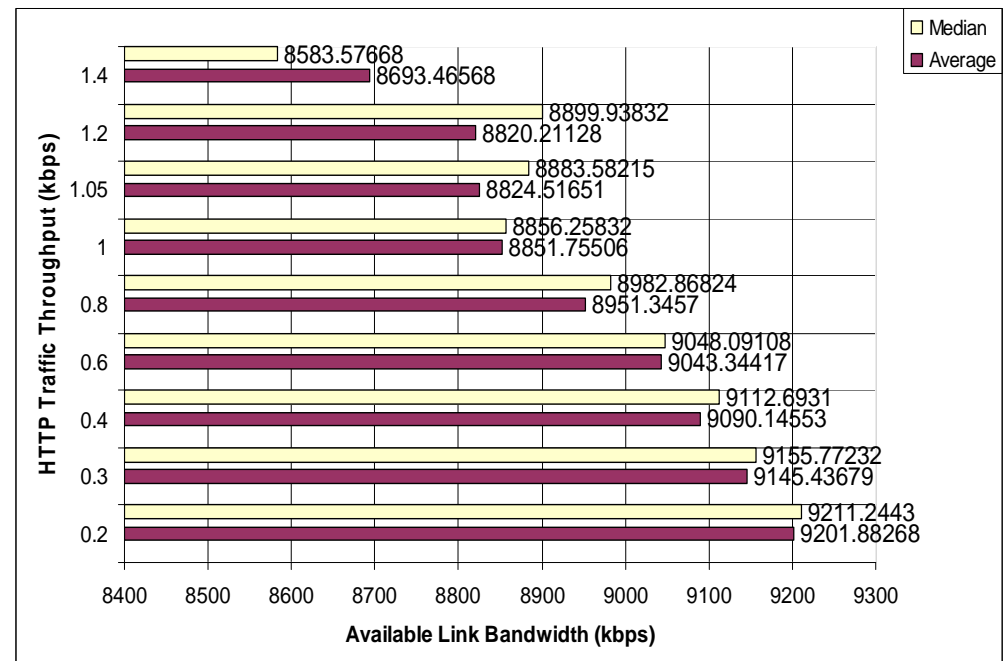


Experiment 2 (Test for speed of convergence)

Medium : 10 Mbps Ethernet

File Size : 2 Mbyte

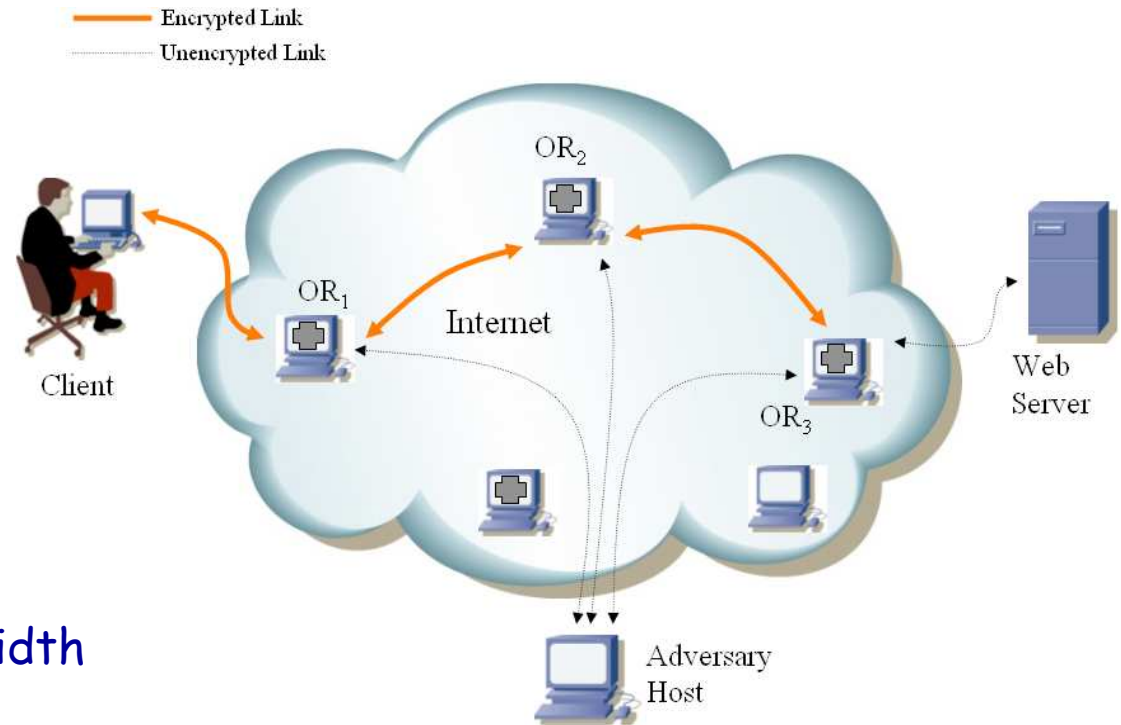
The bandwidth of the http connection is increased from **200 Kbps - 1.4 Mbps**, within a time window < 2 minutes.



Detecting Tor Relays Participating in a Circuit

Experiment :

- Probe Tor relays participating in a circuit from the client to the web server.
- The web sever fluctuates bandwidth of the http connection so as to introduce a "pattern" which may be detected by a GPA



Result from Probing the Tor Relays Participating a Circuit

Circuit	Adversary Host	OR_1	OR_2	OR_3	No. of ORs Detected
1	site1	Eureka(Y)	ieditconfiggg333(Y)	sipbtor(Y)	3
2	site1	CH1rrSkur7(Y)	nixnix(Y)	BostonUCompSci(Y)	3
3	site2	moria2(Y)	hawkitsupport(Y)	whistlersmother(Y)	3
4	site2	uberbabe(Y)	r2600(Y)	superbad(Y)	3
5	site2	desyn(Y)	croeso(Y)	BostonUCompSci(Y)	3
6	site2	foundry(Y)	ieditconfiggg333(Y)	tritlax2(Y)	3
7	site2	ieditconfiggg3(Y)	nixnix(Y)	plek(Y)	3
8	site1	spycat(N)	croeso(Y)	gentoo64(Y)	2
9	site1	redpineapple(Y)	slowturtle2(N)	whistlersmother(Y)	2
10	site1	foundry(Y)	evilwire(Y)	BostonUCompSci(N)	2
11	site1	nixnix(Y)	TorLuWakOrg(N)	superbad(Y)	2
12	site1	Pascal2(N)	Augustus(N)	clanspum(N)	0
13	site1	drunkenmonkeys(N)	lefkada(N)	charlesbabbage(N)	0

of circuits probed : 13

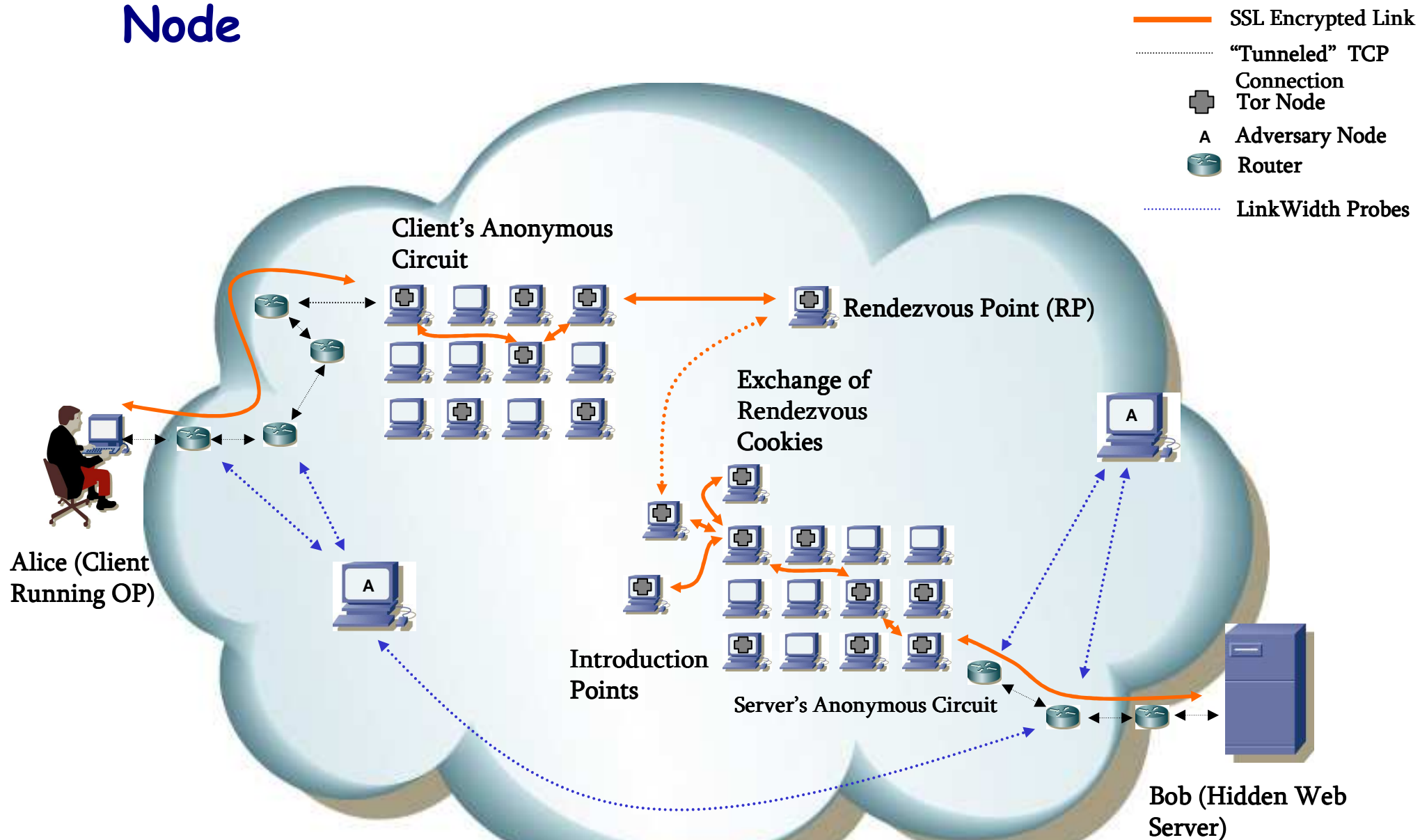
of connections where all the relays were detected : 7

of connections where 2 of the 3 relays were detected : 4

of connections where none of the relays were detected : 2

Client throughput should be at least 30 - 40 KBytes/sec to be detected

"Tracing" the Path of a Tor Client to its Entry Node and from the Hidden Server to its Entry Node



Overview of the Experimental Set-up / Process

1. The client fluctuates the traffic connection bandwidth (without support from the hidden server)
2. The Tor Client and the Hidden Server select Entry Nodes which they aren't change for the entire duration of an experiment (so that the adversary knows the routers to probe). The adversaries only probe the path from the Tor Client and the Hidden Server to their respective Entry Nodes.
3. Two separate adversary nodes are chosen (both having high bandwidth connection to the Internet).
4. Due to lack of network interconnectivity information we rely on `traceroute` to determine the path from client / server to their entry nodes.

Results from Probing the Path from the Tor Client to its Entry Node and from the Server to its Corresponding Entry Node

Circuit Number	# of hops from Client to its Entry Node	Correctly Detected Server-Entry Node Hops	Unresponsive Routers	Router Not Reporting Enough Fluctuation	Success Rate
1	14	9	1	4	64.30%
2	15	7	2	6	46.67%
3	14	7	2	5	50.00%
4	14	4	2	8	28.57%
5	15	6	4	5	40.00%

Circuit Number	# of hops from Server to its Entry Node	Correctly Detected Server-Entry Node Hops	Unresponsive Routers	Router Not Reporting Enough Fluctuation	Success Rate
1	13	4	2	7	30.70%
2	12	9	0	3	75.00%
3	11	7	1	3	63.64%
4	14	5	4	5	35.71%
5	12	9	0	3	75.00%

In all the experiments the Client and the Server gateways reported fluctuations . This can give the adversary clues about the sub-network to which the anonymous client/server belongs.

Conclusions / Observations from the Experiments

1. Fluctuation in bandwidth is detected by observing an increase in packet losses.
2. In most cases client achieved "very low" throughput (<30 K Bytes/sec ~ 250 kbps)
3. (2) makes detection difficult . Elastic / non-aggressive TCP client traffic (having low throughput) doesn't cause the probe traffic to drop "drastically". Instead, it adjusts its own throughput to adapt to "aggressive" probe traffic.
4. These measurements don't include possible errors due false positives . One may possibly result when an adversary tries to "search" for possible routers along the path from the Tor client to its Entry Node.

Discussion / Issues

Why is probing routers for bandwidth fluctuation using LinkWidth difficult ?

- Vantage point of the adversary :
(Client - Server bottleneck bandwidth may be different
Adversary - OR bottleneck.

Possible Solution : More geographically/topologically dispersed adversaries)

- Noisy background traffic / OS and hardware issues (ex. interrupt coalescence)/ traffic shaping makes correct estimation of bandwidth routers anyways a difficult problem.

Future Work

- Determine the scalability of LinkWidth using large scale experiments on testbeds such as DeterLab
- Perform more end-to-end measurements possibly on paths spanning across trans-continental links (which may pose as bottlenecks)
- Using network interconnectivity and probing “probable” routers along the path. Currently we only rely on known path between a client / server and its entry node.

References

- [1] S. J. Murdoch and G. Danezis. Low-Cost Traffic Traffic Analysis of Tor. In IEEE Symposium on Security and Privacy, pages 183-195, May 2005.
- [2] N. Hopper, E. Y. Vasserman, and E. Chan-Tin. How Much Anonymity does Network Latency Leak? In Proceedings of ACM CCS, October 2007.
- [3] S. Keshav. Congestion Control in Computer Networks. UC Berkely Technical Report TR-654, September 1991
- [4] M. Y. Sanadidi. Bandwidth Estimation Techniques , A Tutorial Presentation. In SBRC 2002:Brazilian Symposium on Computer Networks Date, Buzios, Brazil, May 2002.
- [5] C. Dovrolis and P. Ramanathan and D. Moore. Packet-Dispersion Techniques and a Capacity-Estimation Methodology. IEEE Transactions on Networking, December 2004
- [6] M. Gerla, M. Y. Sanadidi, R.Wang, and A. Zanella. TCP Westwood: Congestion Window Control Using Bandwidth Estimation. In Proceedings of IEEE Globecom, Volume 3, pages 1698-1702, November 2001.
- [7] Gil, T. M., Kaashoek, F., Li, J., Morris, R., and Stribling, J. The 'King' data set. <http://pdos.csail.mit.edu/p2psim/kingdata/>, 2005

Thank You