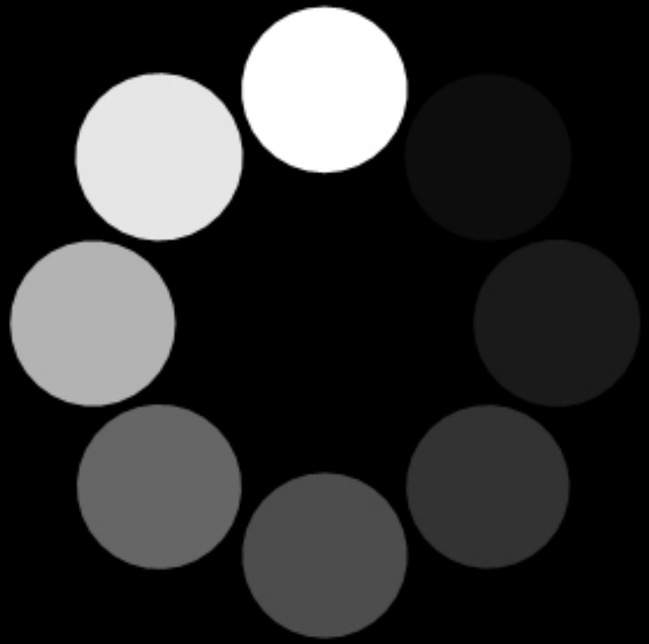
RouteScout

Performance-Driven Internet Path Selection



default Internet routing protocol

BGP

default Internet routing protocol

BGP statically routes traffic to each destination prefix via a single path despite Internet's path diversity, leading to sub-optimal performance.

Can we dynamically route traffic according to performance?

Can we dynamically route traffic according to performance?

Can we dynamically route traffic according to performance?

…why now?

Can we dynamically route traffic according to performance?

…why now?

- sharper focus on reliably high network performance

Can we dynamically route traffic according to performance?

…why now?

- sharper focus on reliably high network performance
- increasing diversity in available paths

Can we dynamically route traffic according to performance?

…why now?

- sharper focus on reliably high network performance
- increasing diversity in available paths
- programmable devices allows scalable monitoring & flexible forwarding

RouteScout: Performance-Driven Internet Path Selection

Maria Apostolaki

Ankit Singla

Laurent Vanbever

RouteScout: Performance-Driven Internet Path Selection

RouteScout Operations

RouteScout Architecture

Data-Plane Design

Evaluation

RouteScout: Performance-Driven Internet Path Selection

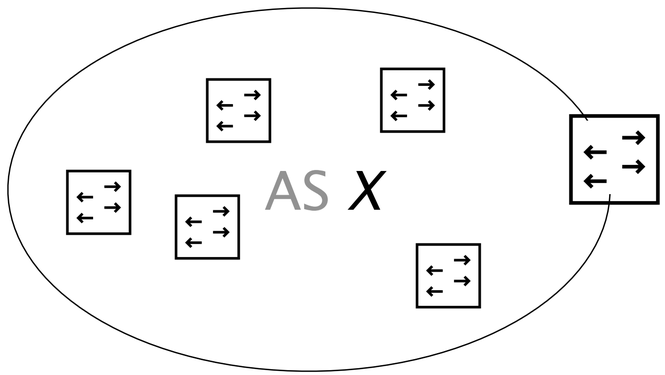
RouteScout Operations

RouteScout Architecture

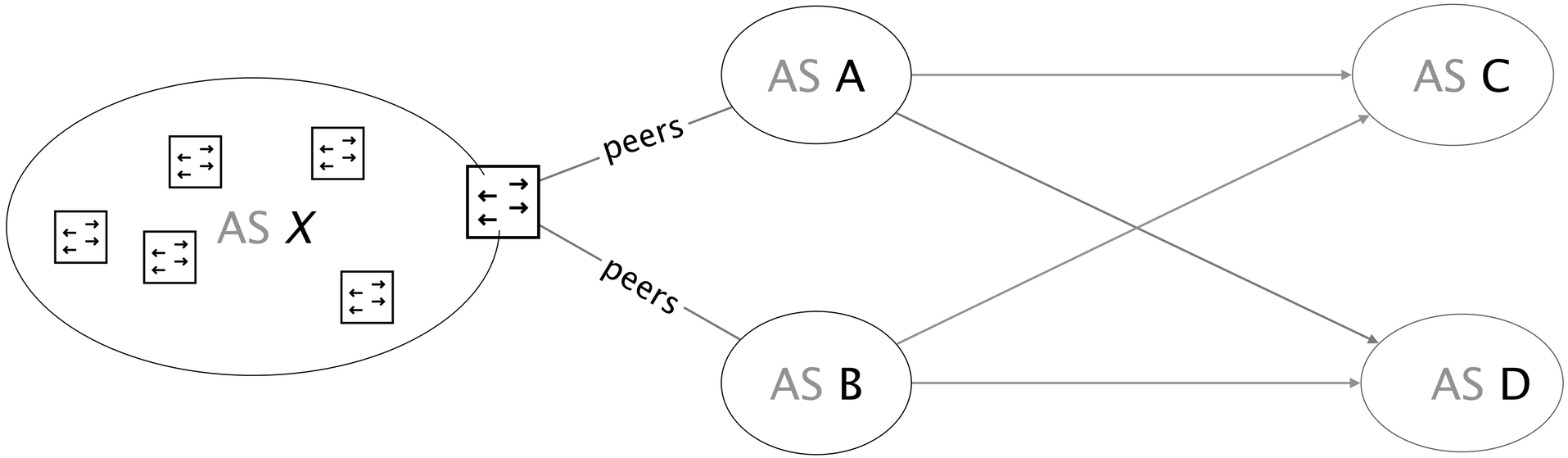
Data-Plane Design

Evaluation

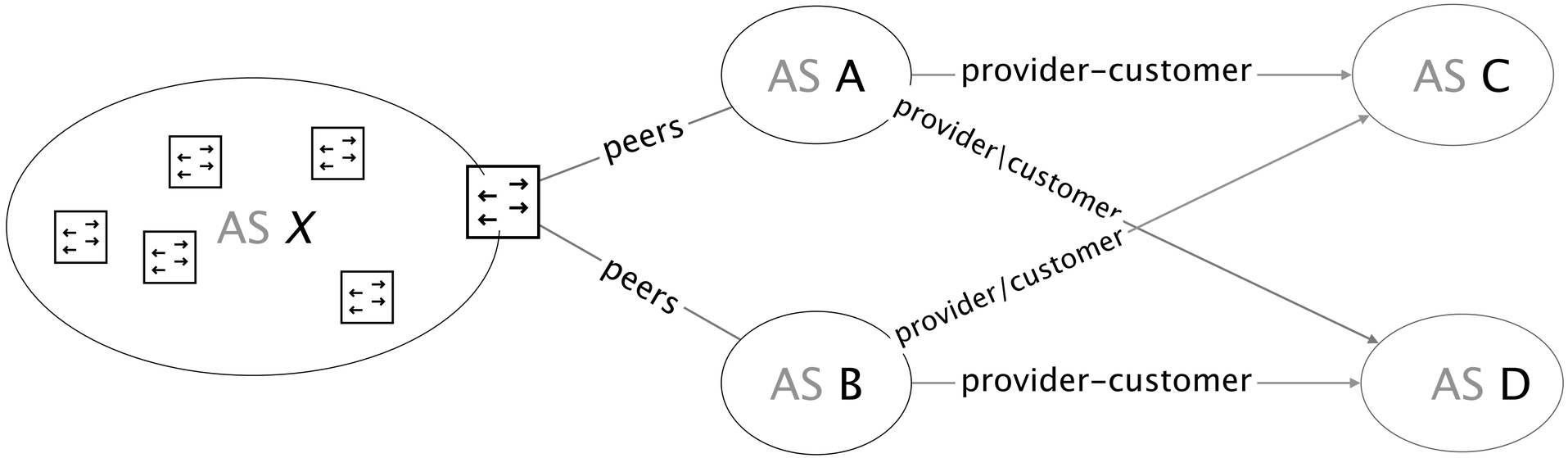
RouteScout selects the next hop per prefix



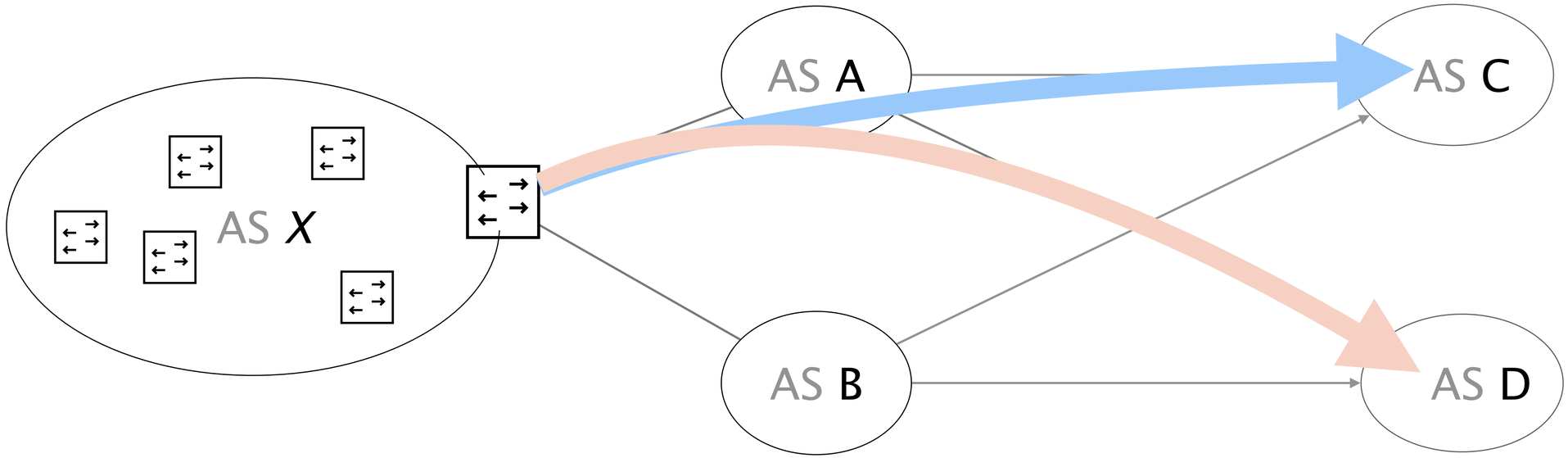
RouteScout selects the next hop per prefix



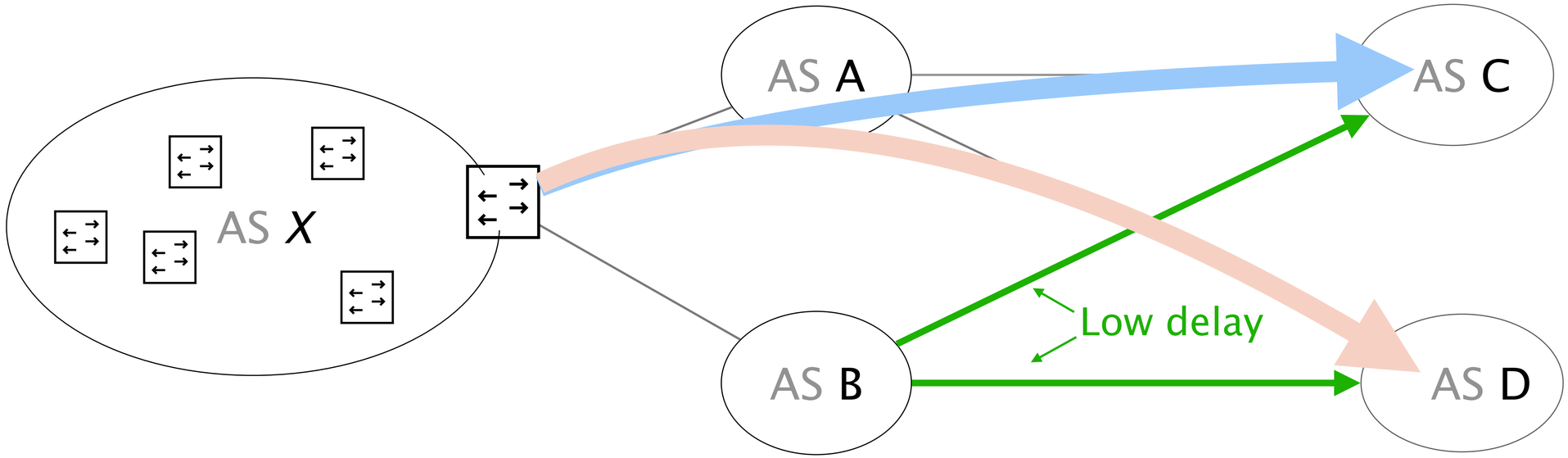
Routes via AS A and via AS B to AS C and AS D are equally preferred



BGP routes to both destinations via AS A

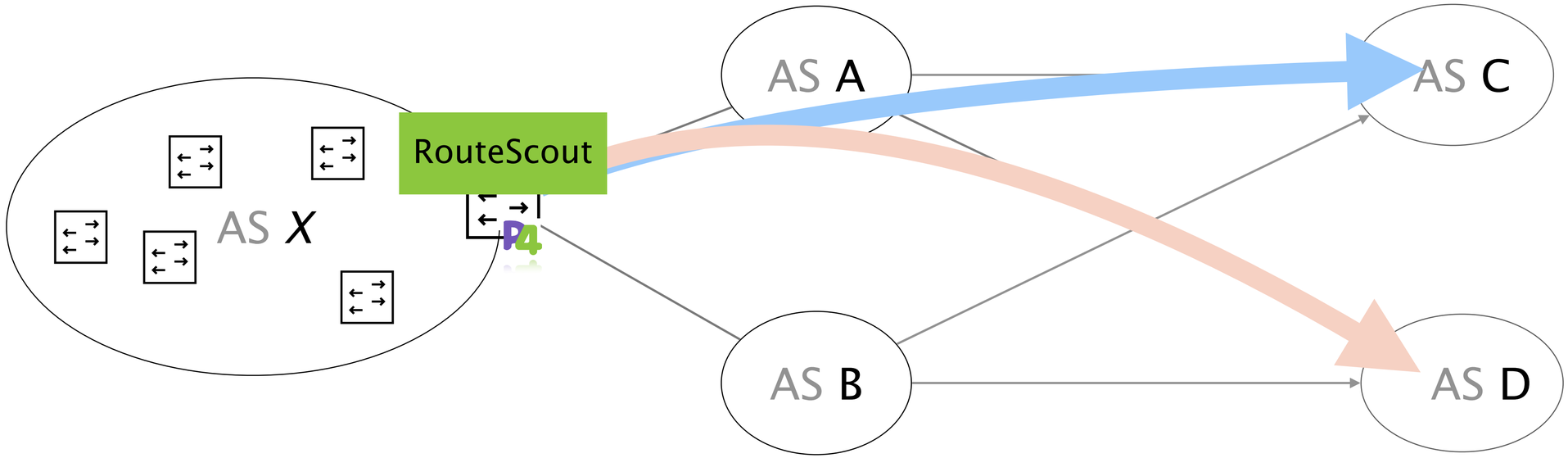


Unbeknownst to AS X Routes via AS B have much lower delay

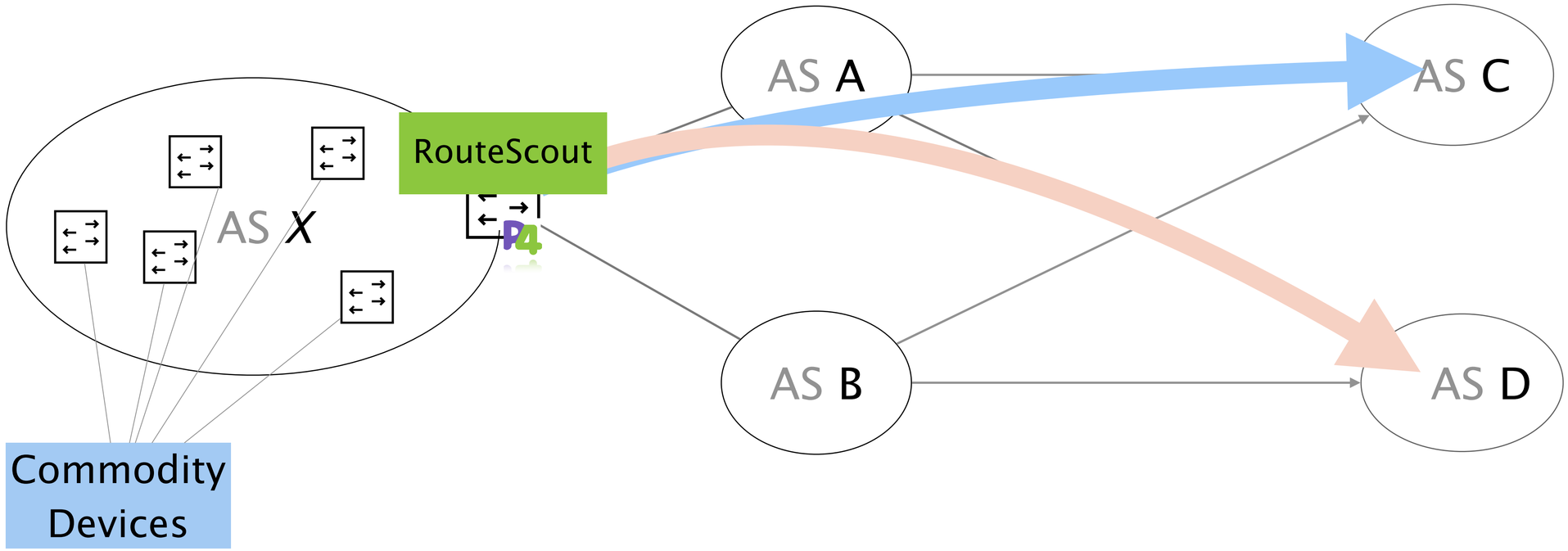


What does RouteScout need to explore and exploit this opportunity?

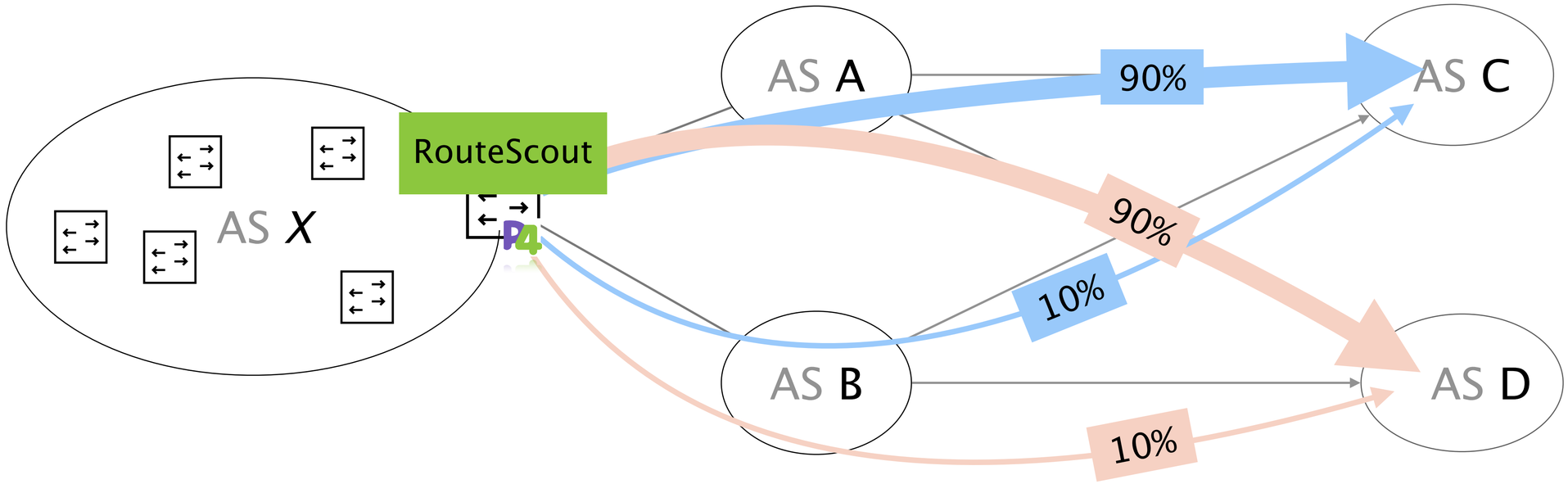
RouteScout runs at the network edge on a programmable switch and controls the paths of outgoing traffic



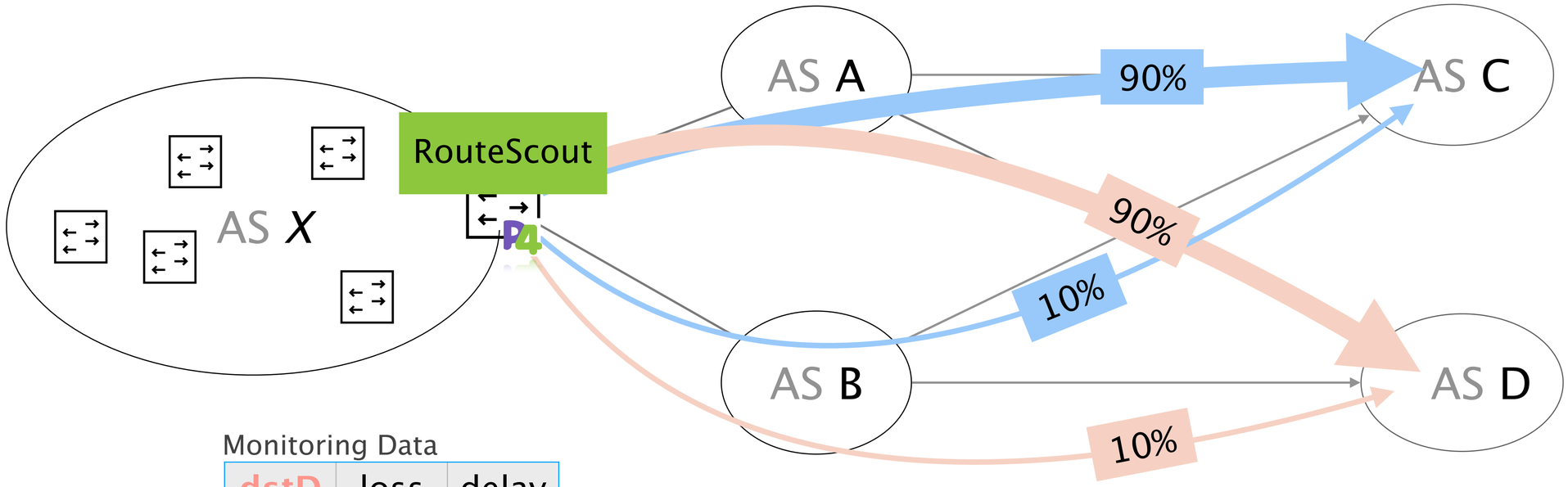
RouteScout runs at the network edge on a programmable switch and controls the paths of outgoing traffic



RouteScout directs traffic to alternative BGP-compliant next-hops



RouteScout monitors performance across prefix, next-hop pairs

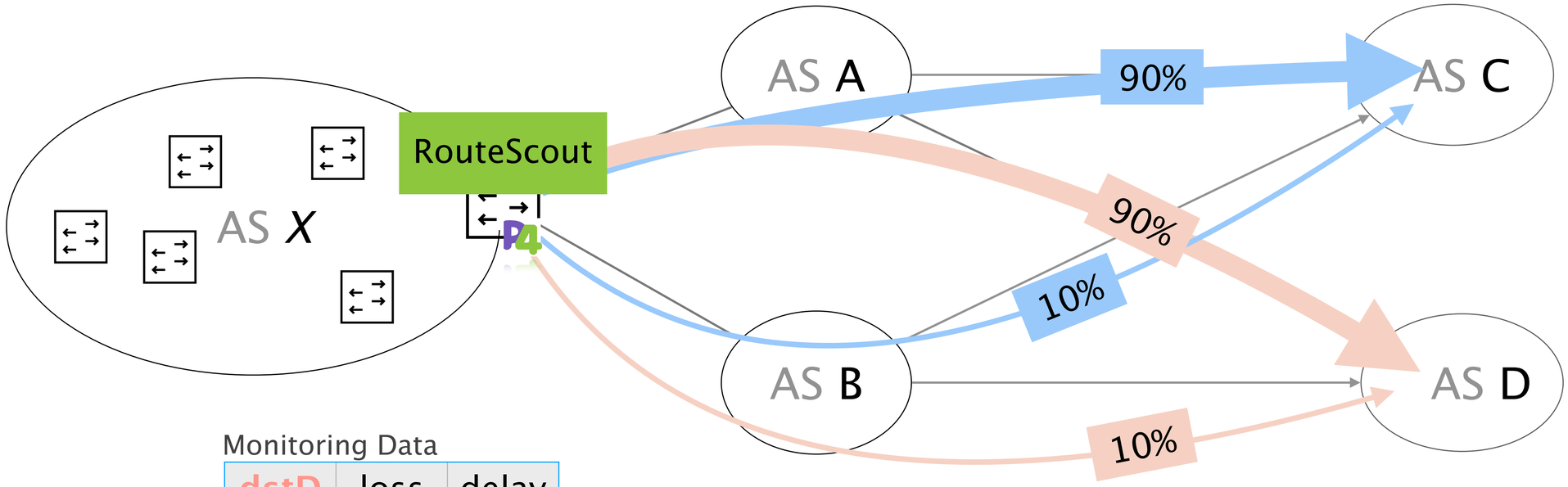


Monitoring Data

dstD	loss	delay
A	0	5
B	0	4

dstC	loss	delay
A	0	13
B	0	7

RouteScout monitors performance across prefix, next-hop pairs

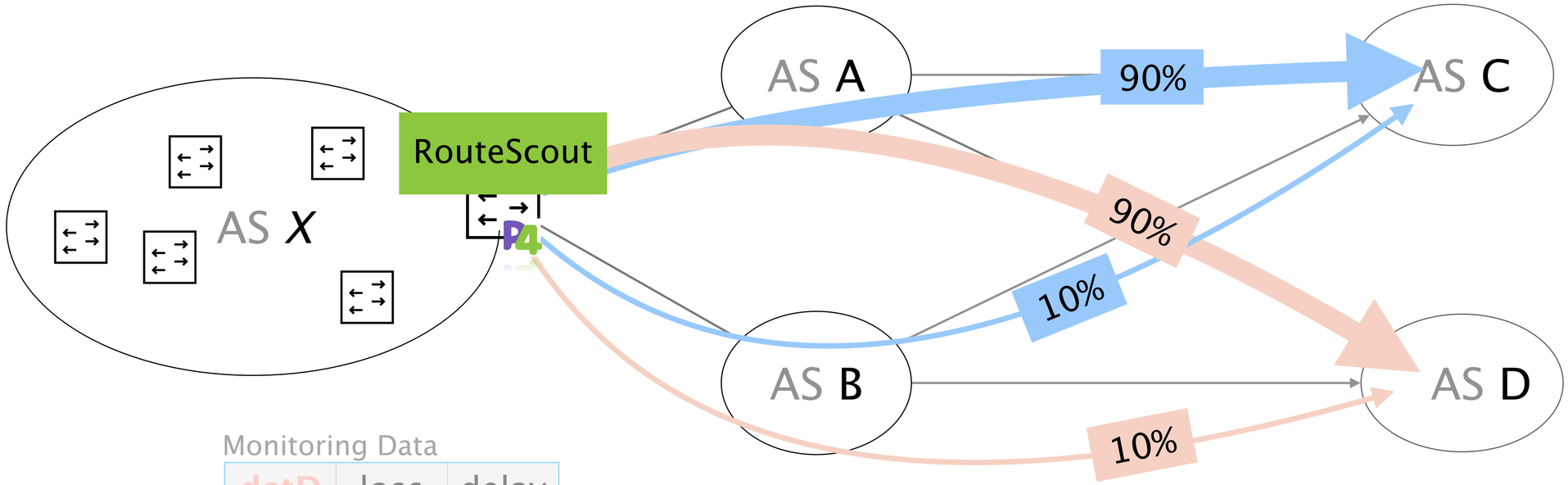


Monitoring Data

dstD	loss	delay
A	0	5
B	0	4

dstC	loss	delay
A	0	13
B	0	7

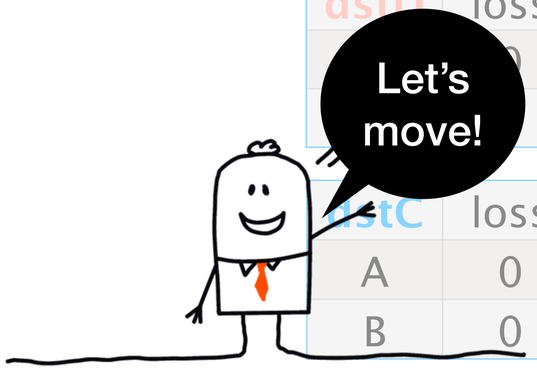
RouteScout monitors performance across prefix, next-hop pairs



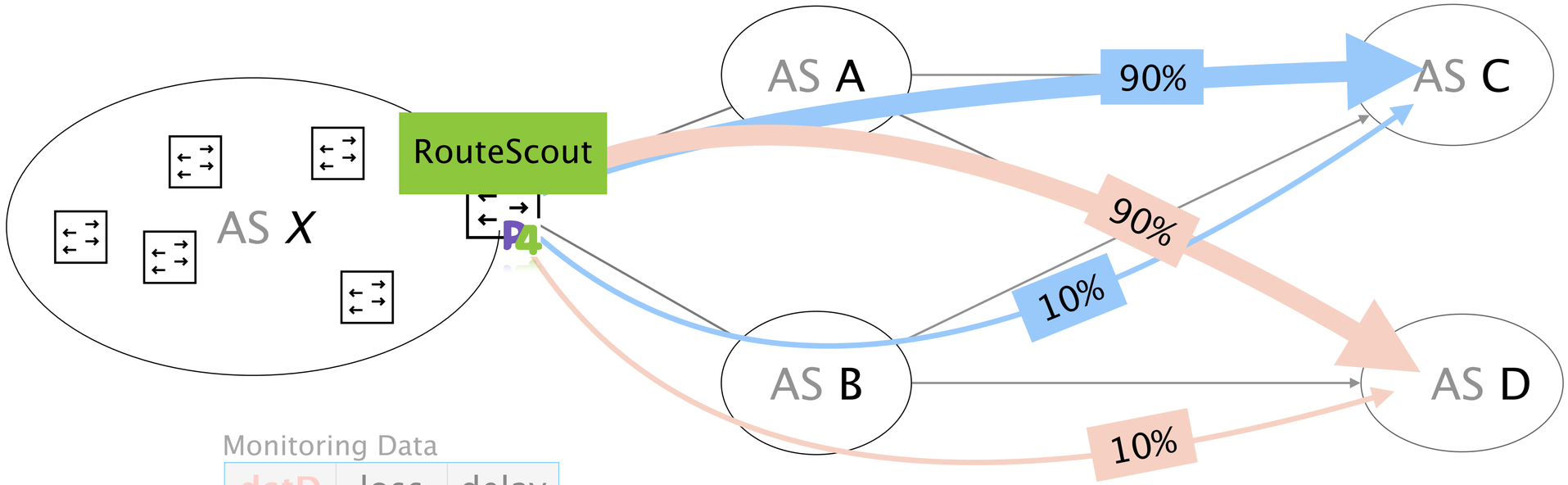
Monitoring Data

dstD	loss	delay
	0	5
		4

dstC	loss	delay
A	0	13
B	0	7



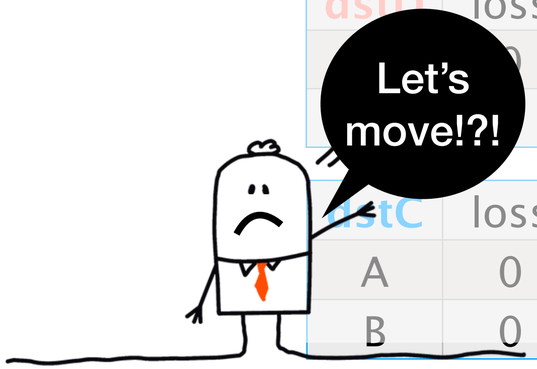
RouteScout monitors performance across prefix, next-hop pairs



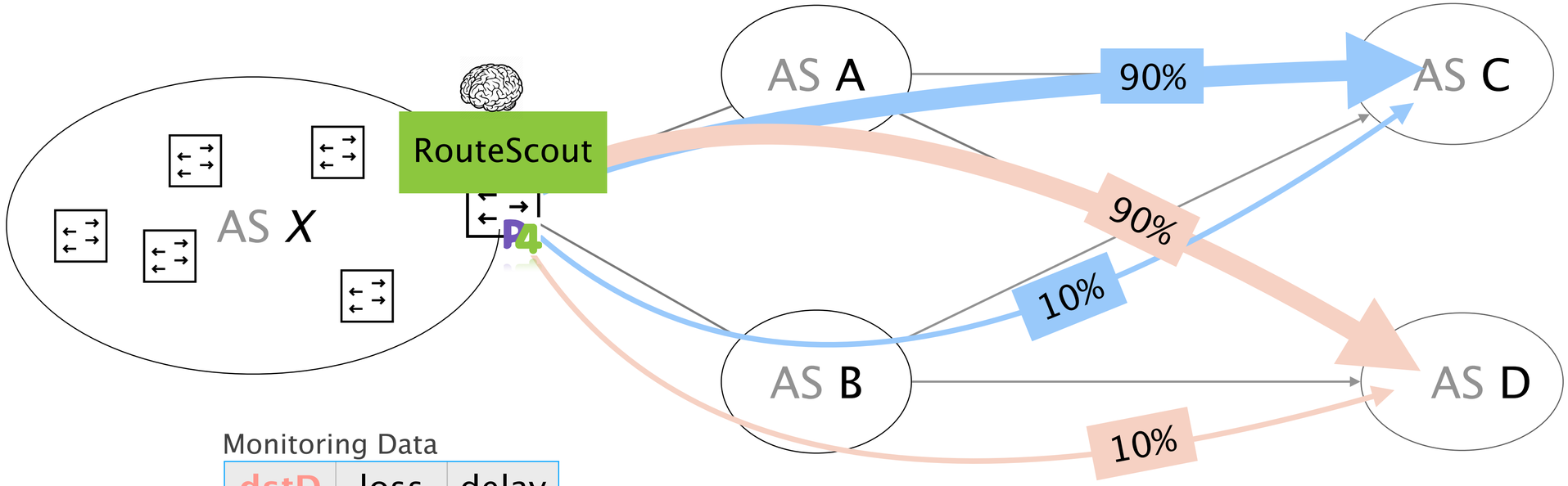
Monitoring Data

dstD	loss	delay
	0	5
		4

dstC	loss	delay
A	0	13
B	0	7



RouteScout using performance monitoring as input,
solves an optimization problem to find the most suitable forwarding plan



Monitoring Data

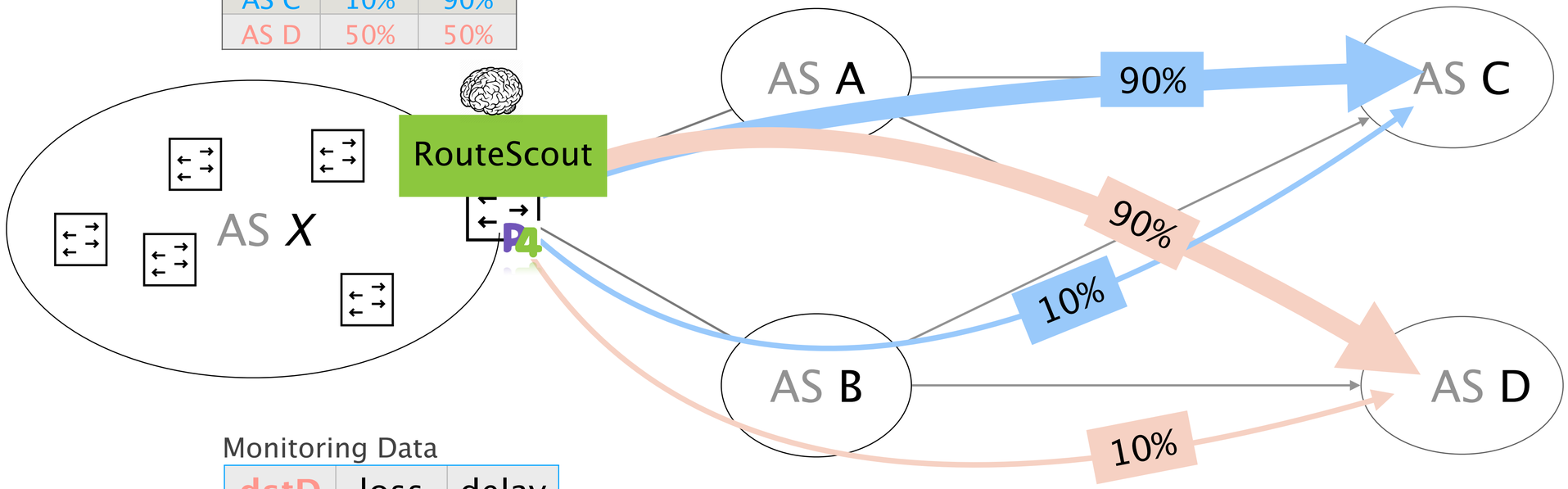
dstD	loss	delay
A	0	5
B	0	4

dstC	loss	delay
A	0	13
B	0	7

RouteScout using performance monitoring as input, solves an optimization problem to find the most suitable forwarding plan

Forwarding Plan

	AS A	AS B
AS C	10%	90%
AS D	50%	50%



Monitoring Data

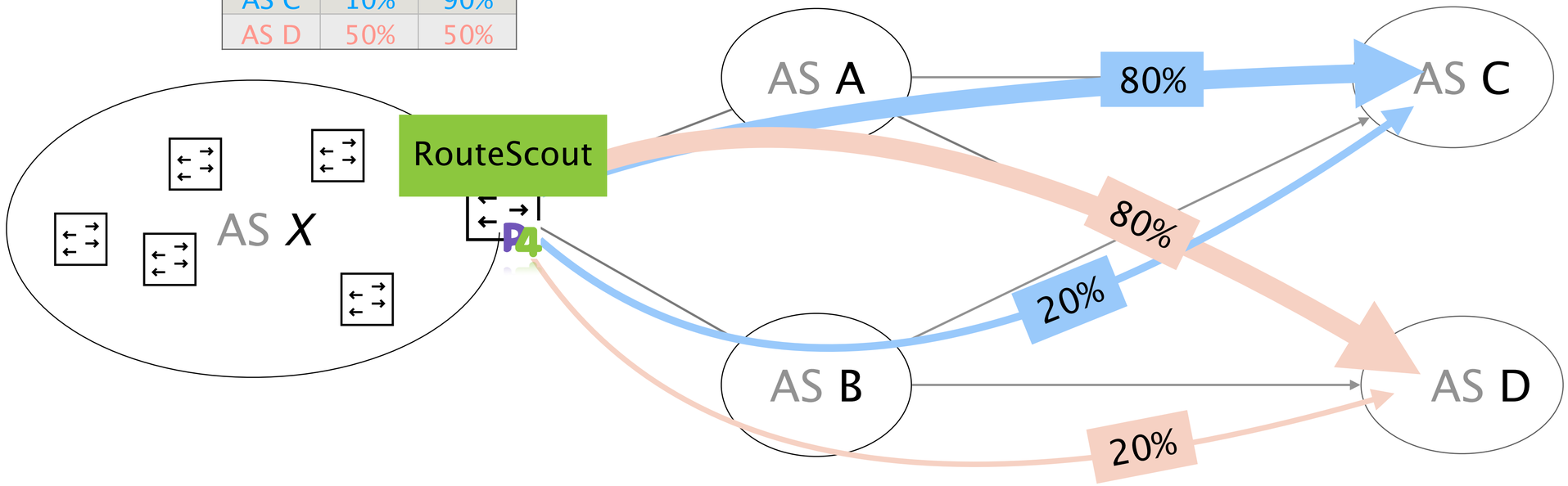
dstD	loss	delay
A	0	5
B	0	4

dstC	loss	delay
A	0	13
B	0	7

RouteScout actuates appropriate traffic shifts in the data plane

Forwarding Plan

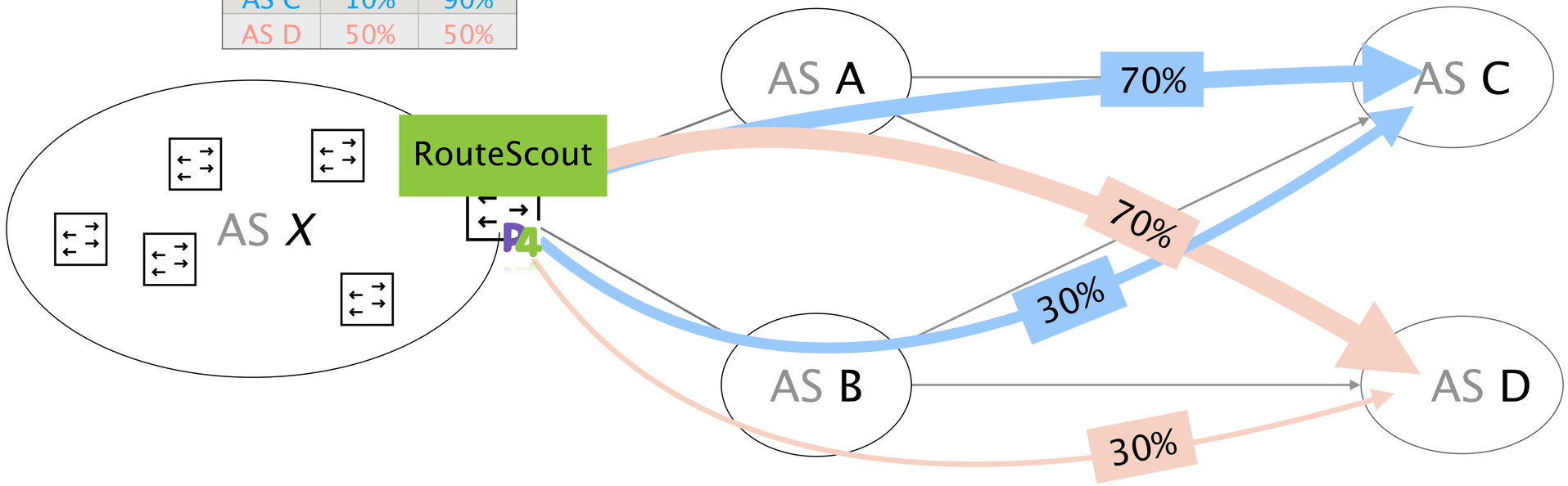
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout actuates appropriate traffic shifts in the data plane

Forwarding Plan

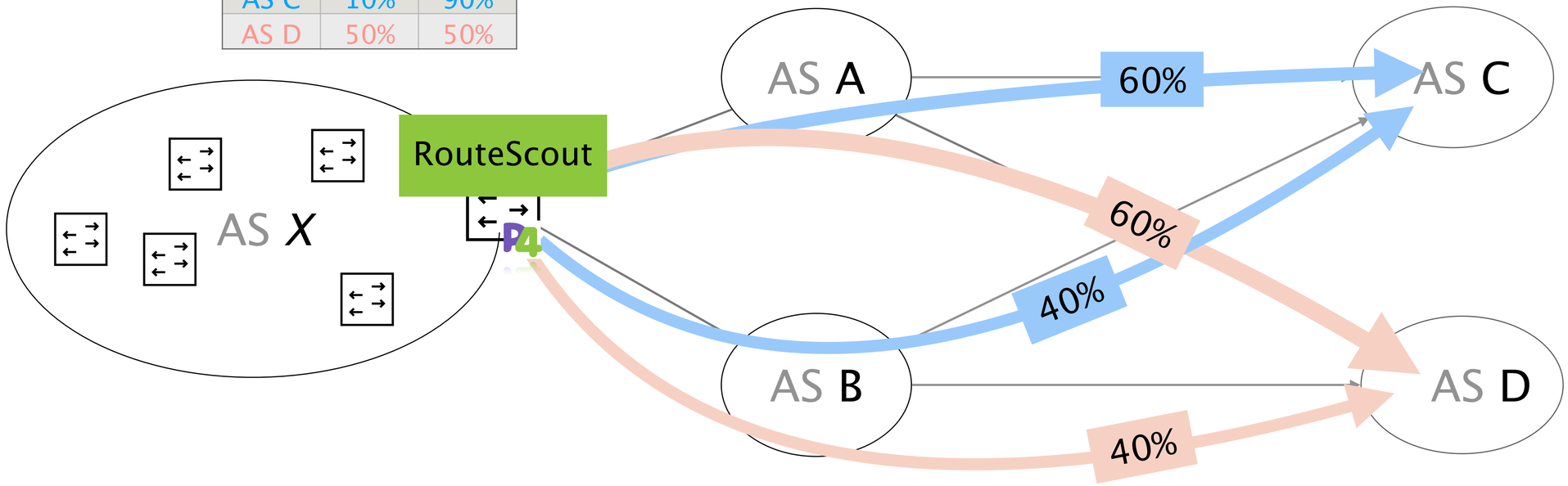
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout actuates appropriate traffic shifts in the data plane

Forwarding Plan

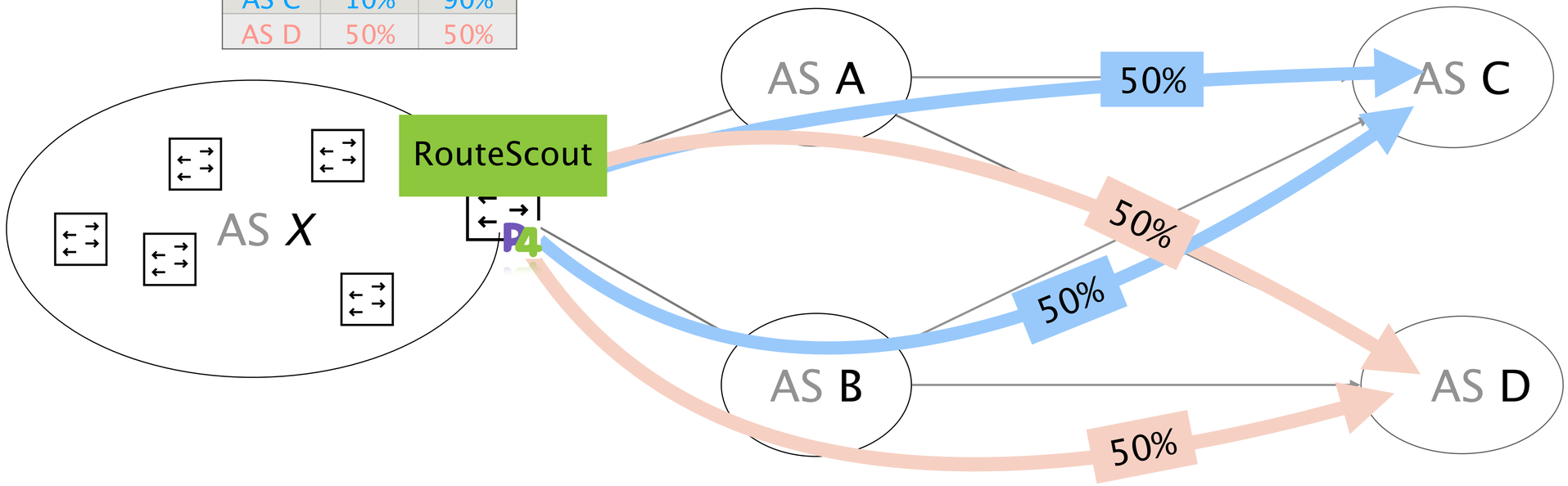
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout actuates appropriate traffic shifts in the data plane

Forwarding Plan

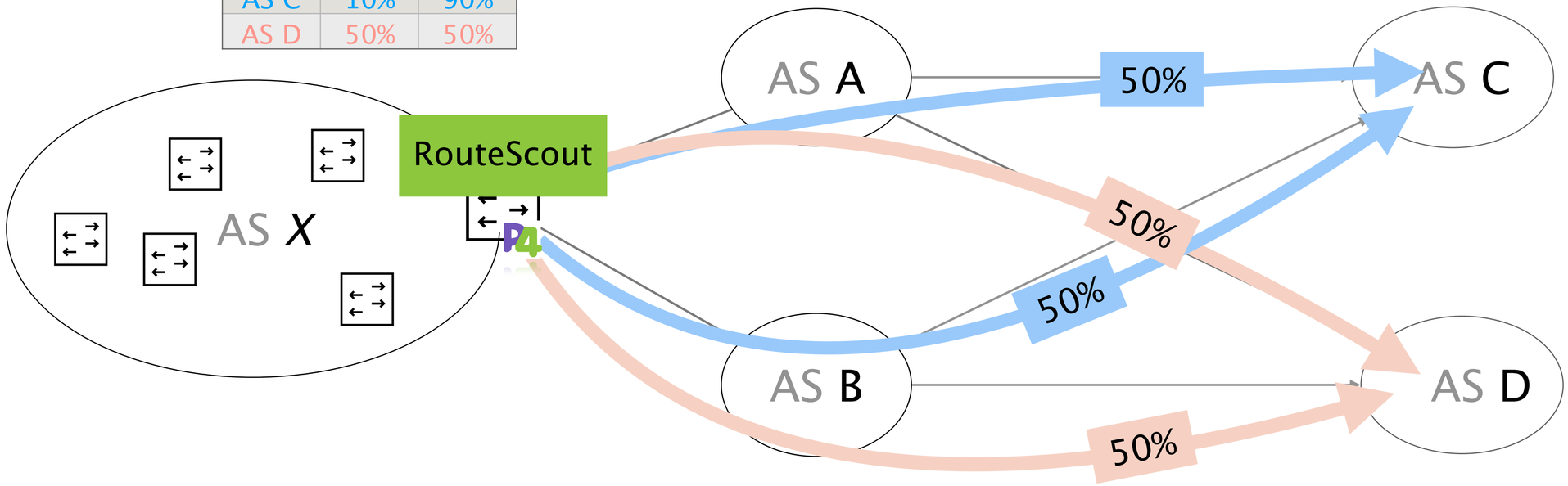
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout actuates appropriate traffic shifts in the data plane

Forwarding Plan

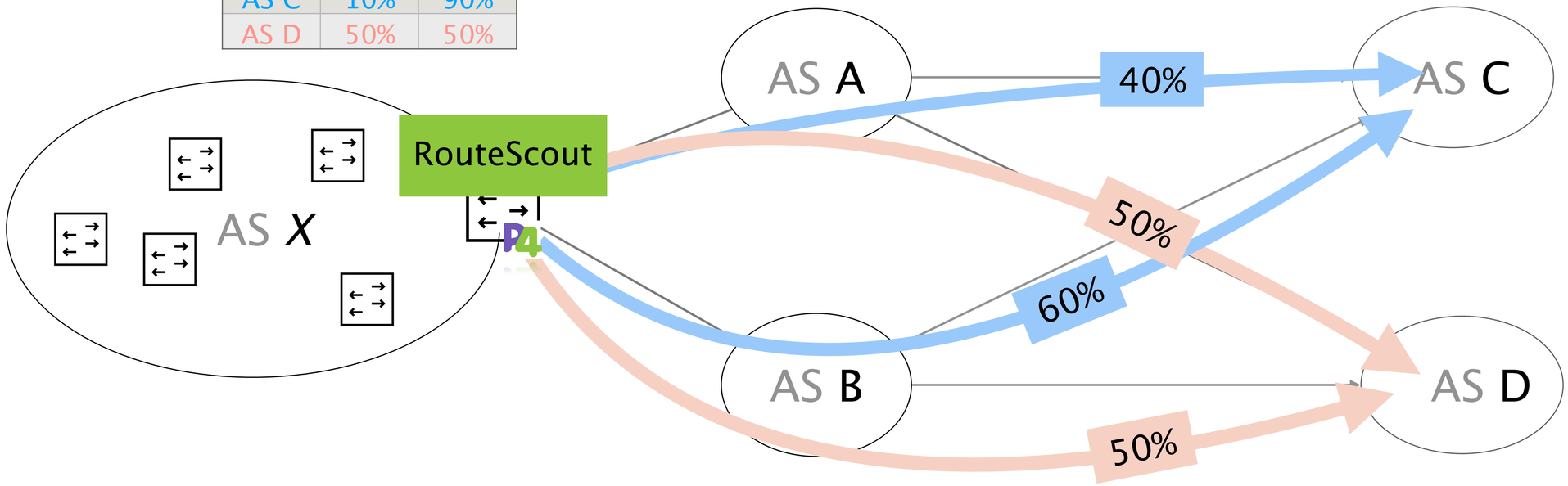
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout actuates appropriate traffic shifts in the data plane

Forwarding Plan

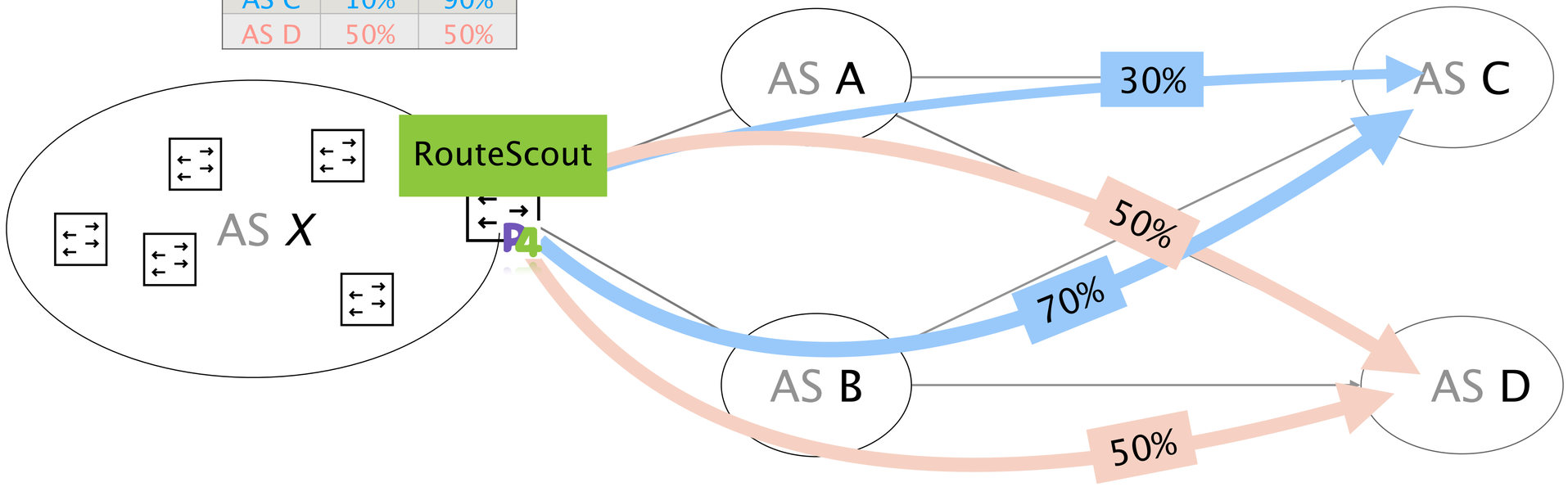
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout optimizes aggregate performance while respecting constraints

Forwarding Plan

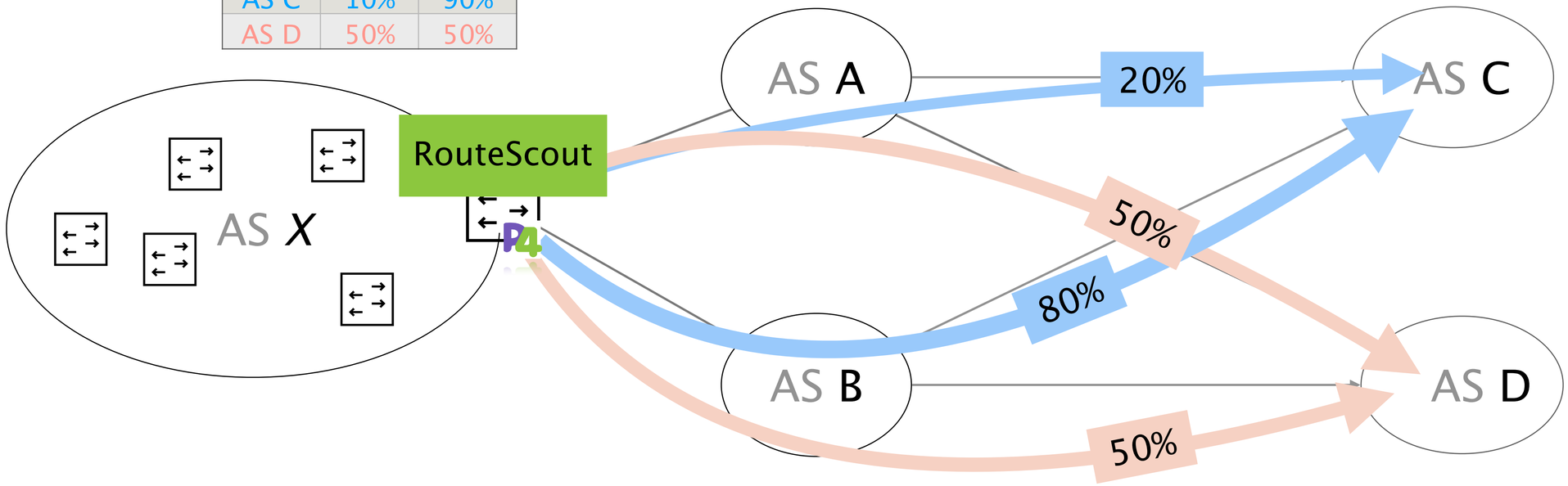
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout optimizes aggregate performance while respecting constraints

Forwarding Plan

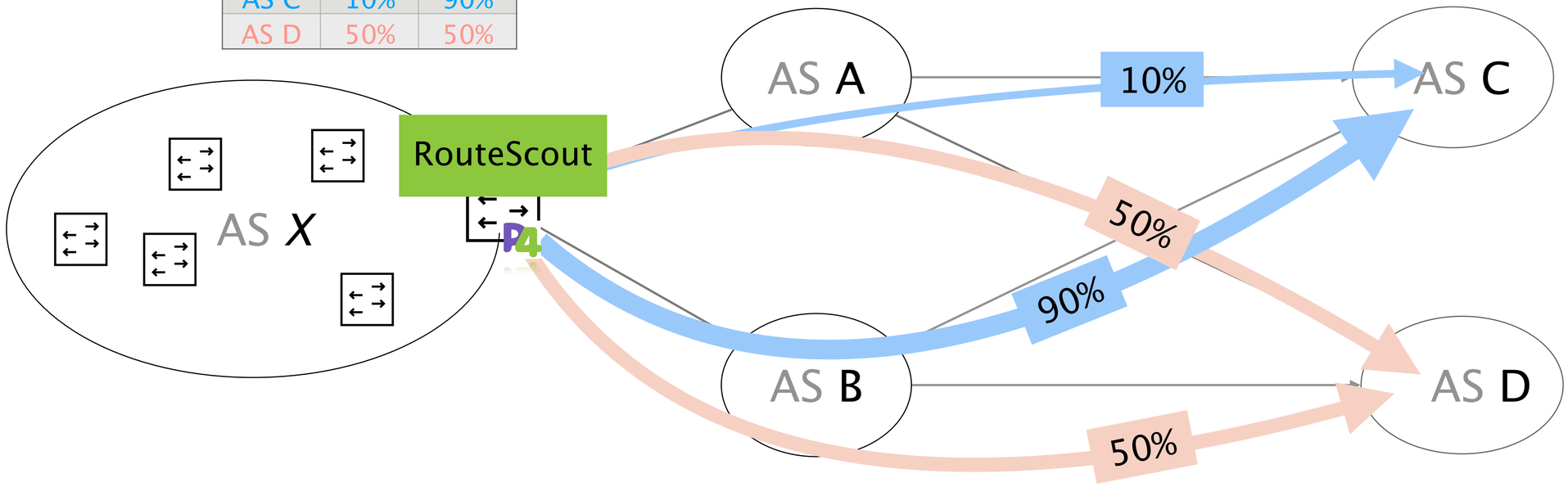
	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout optimizes aggregate performance while respecting constraints

Forwarding Plan

	AS A	AS B
AS C	10%	90%
AS D	50%	50%



RouteScout: Performance-Driven Internet Path Selection

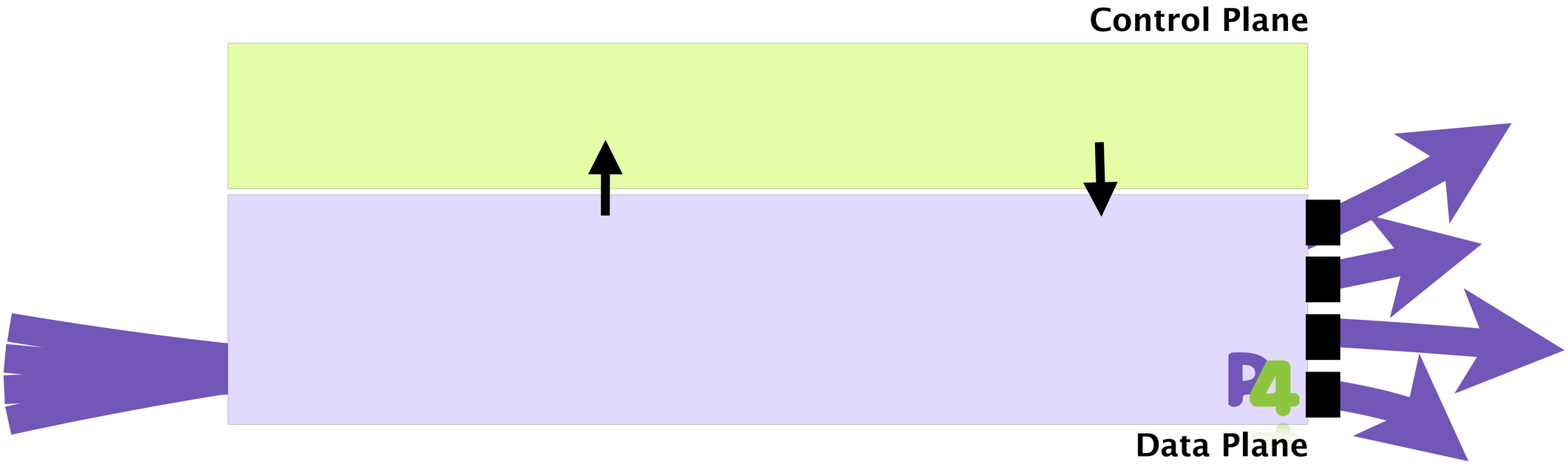
RouteScout Operations

RouteScout Architecture

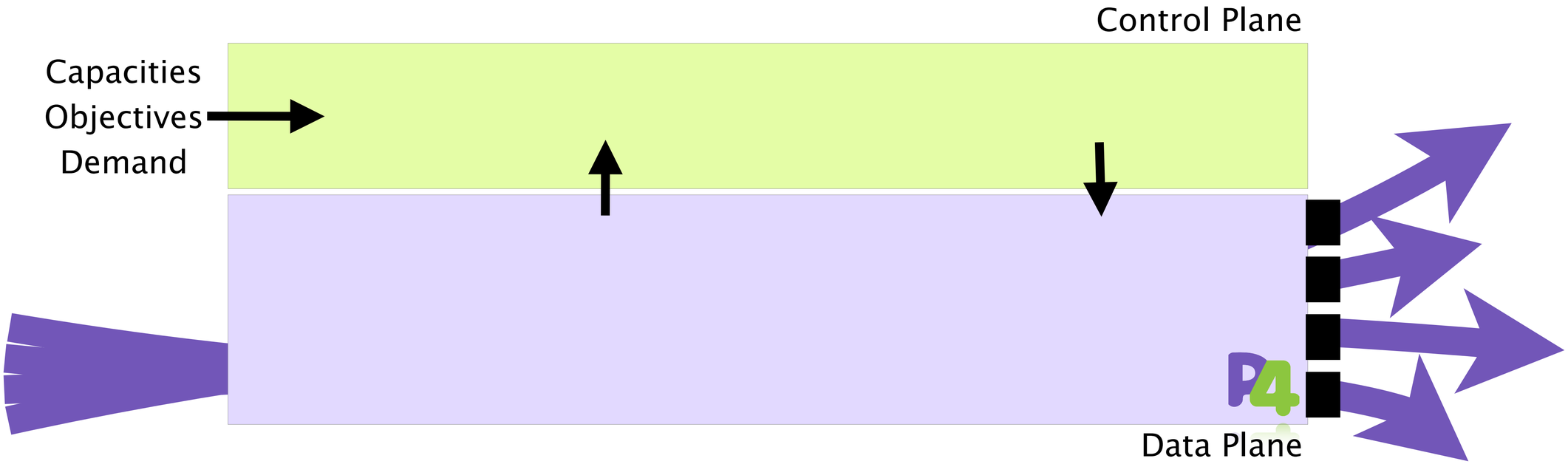
Data-Plane Design

Evaluation

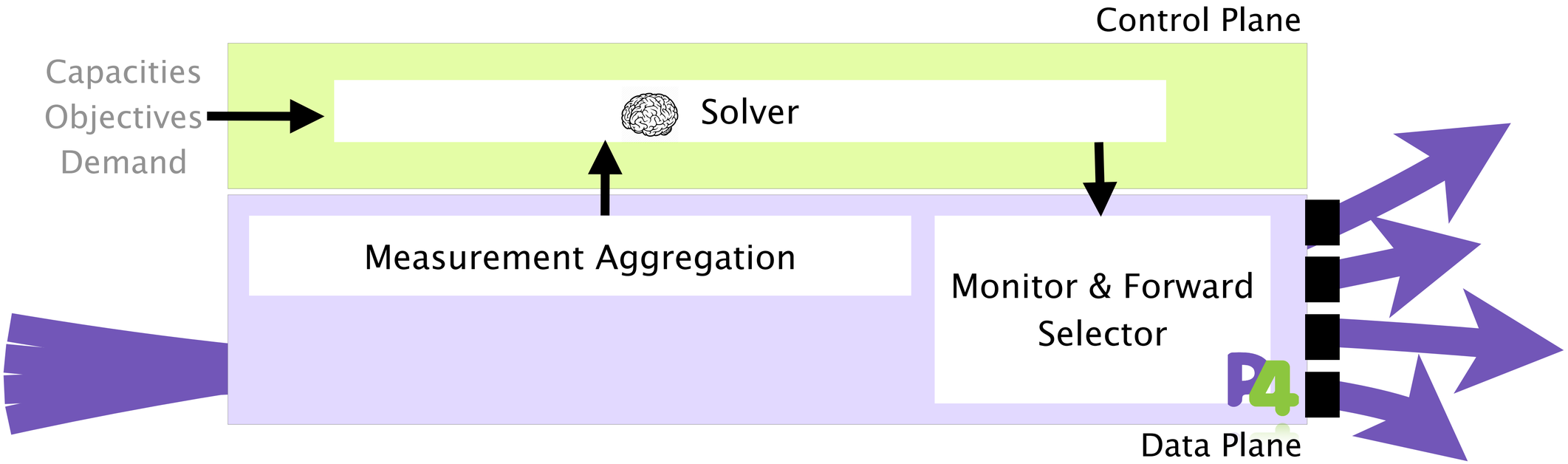
RouteScout is a closed-loop control system, implemented as a hardware-software design



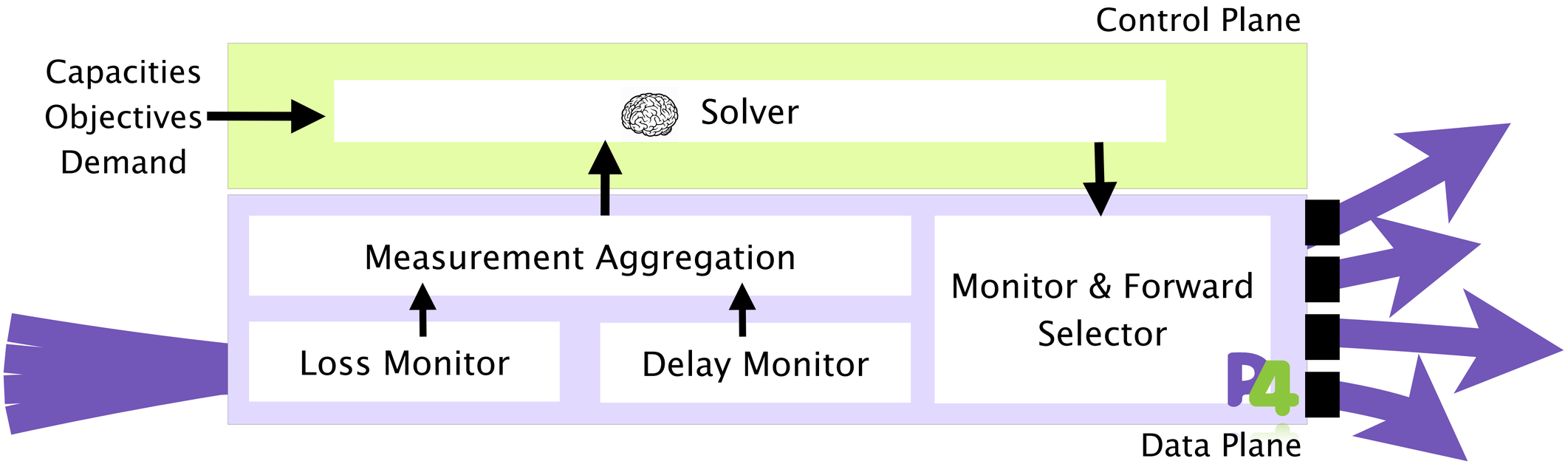
RouteScout is a closed-loop control system, implemented as a hardware-software design



RouteScout is a closed-loop control system, implemented as a hardware-software design



RouteScout is a closed-loop control system, implemented as a hardware-software design



RouteScout: Performance-Driven Internet Path Selection

- RouteScout Operations
- RouteScout Architecture
- Data-Plane Design
  - Loss Monitor
  - Delay Monitor
- Evaluation

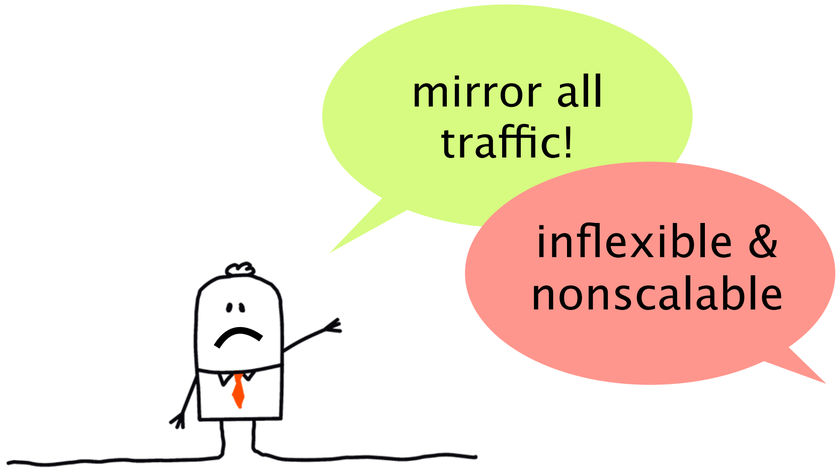
RouteScout: Performance-Driven Internet Path Selection

- RouteScout Operations
- RouteScout Architecture
- Data-Plane Design
  - Loss Monitor
  - Delay Monitor
- Evaluation

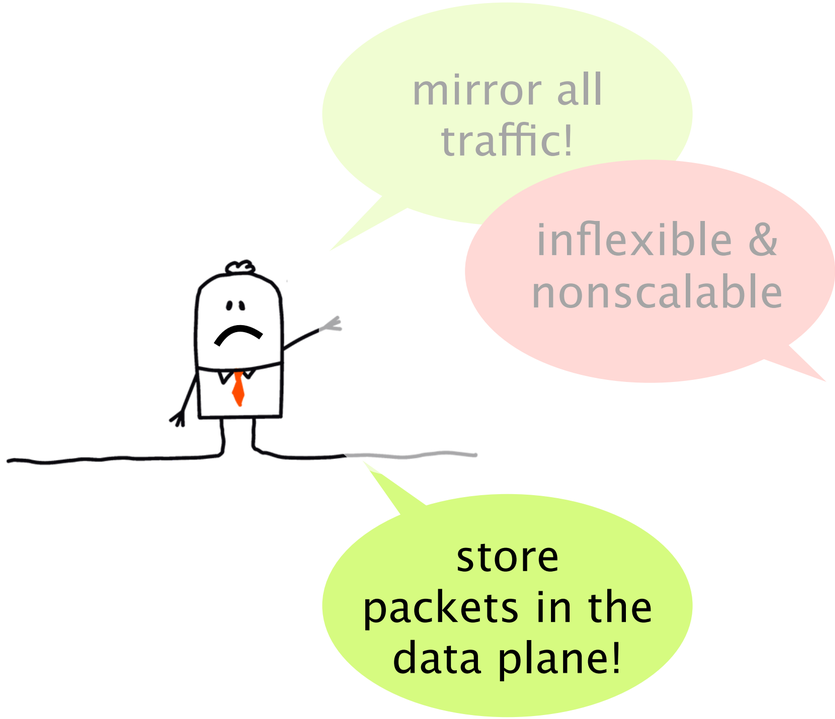
RouteScout measures loss rate via detecting retransmissions

RouteScout measures loss rate via detecting retransmissions

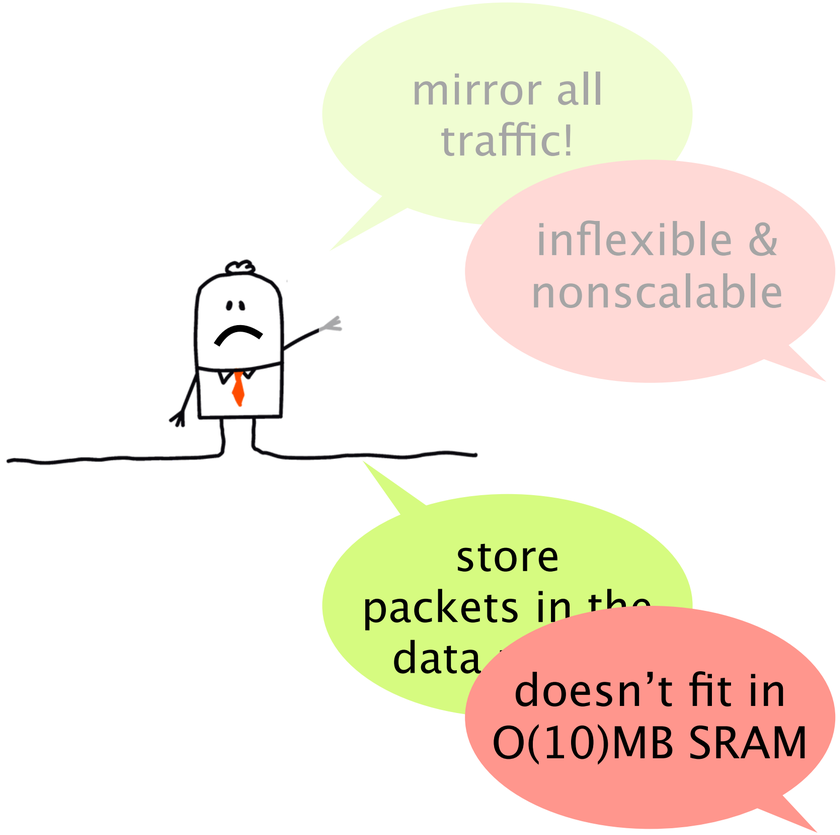
RouteScout measures loss rate via detecting retransmissions



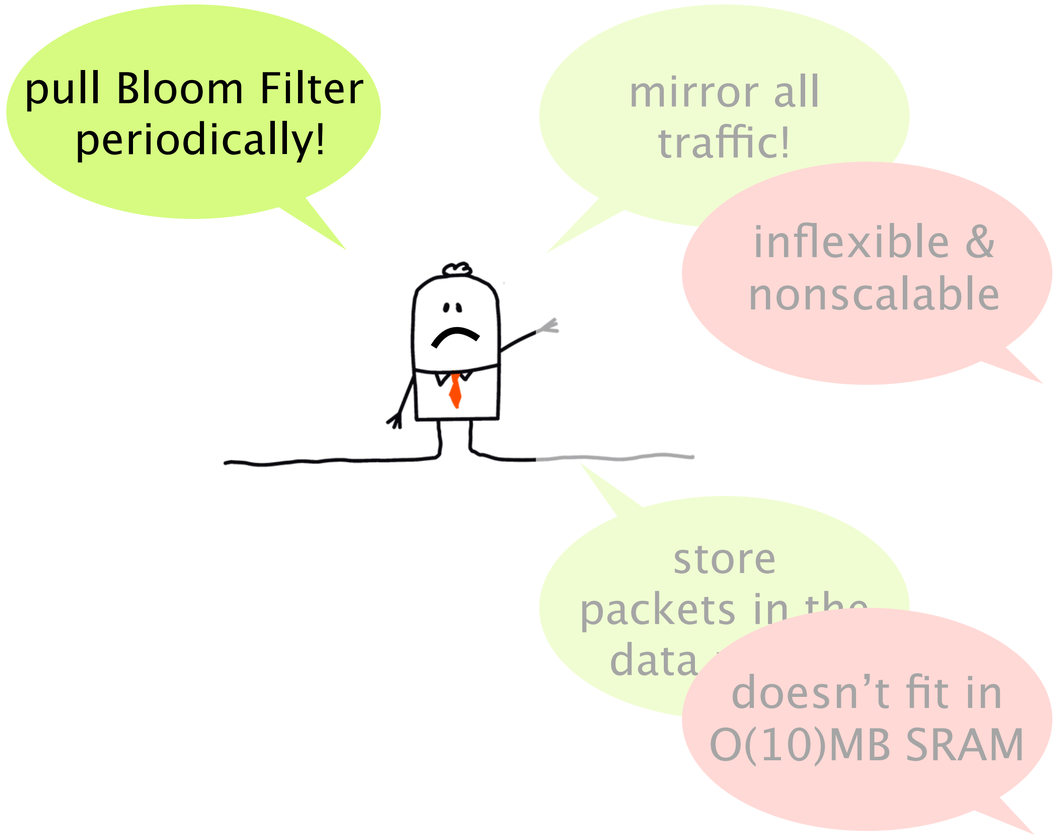
RouteScout measures loss rate via detecting retransmissions



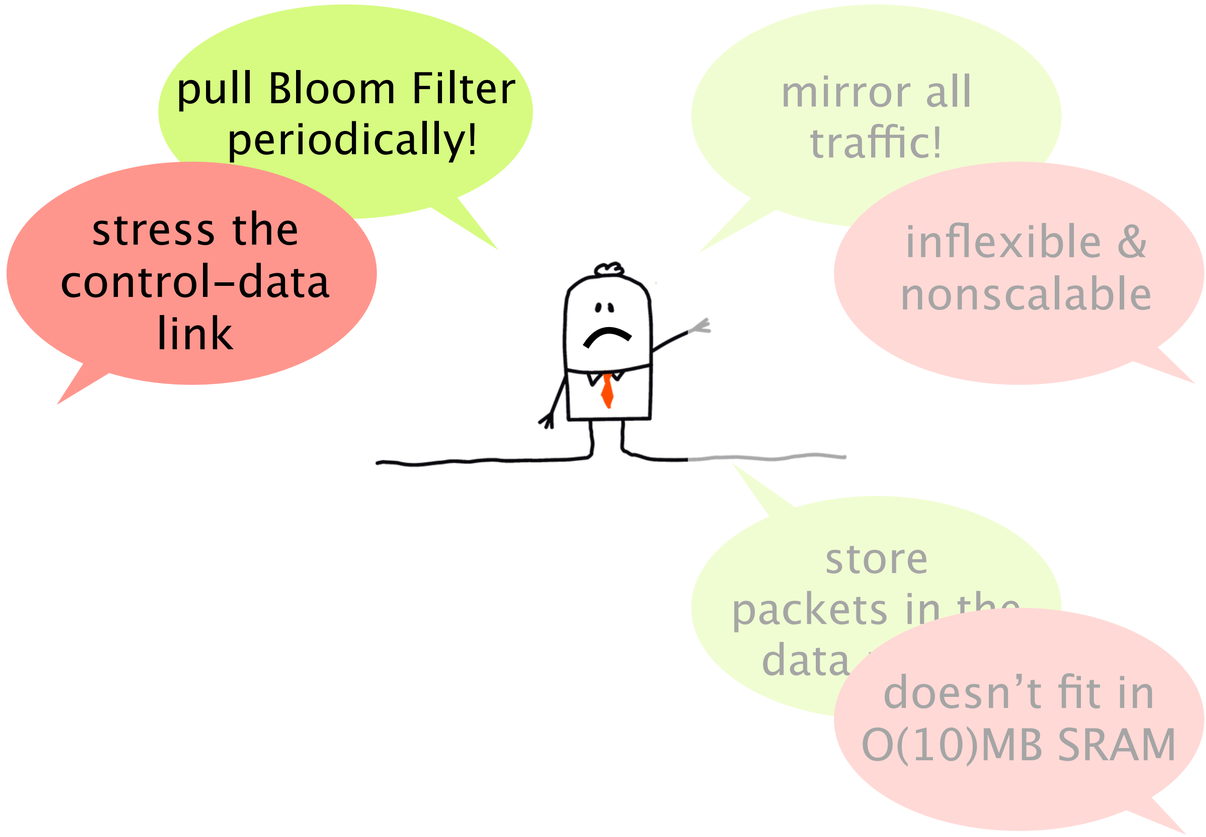
RouteScout measures loss rate via detecting retransmissions



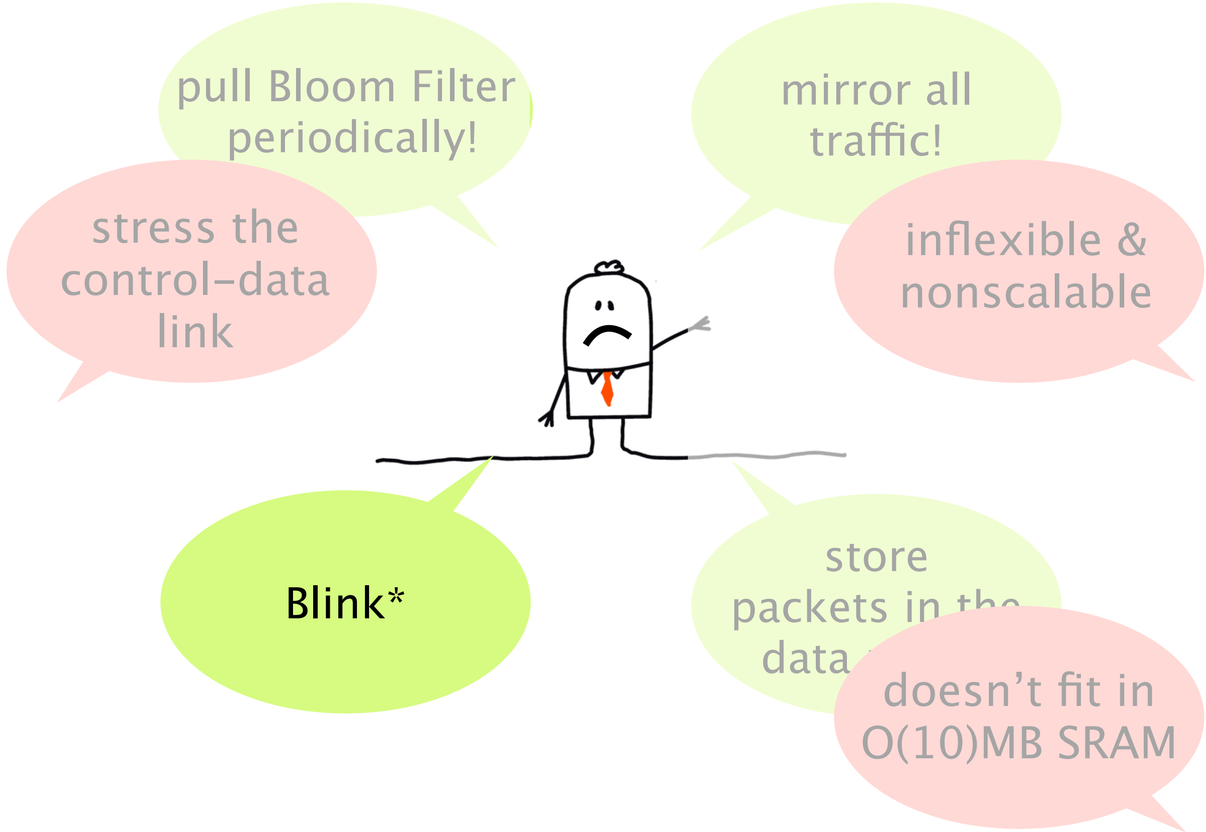
RouteScout measures loss rate via detecting retransmissions



RouteScout measures loss rate via detecting retransmissions



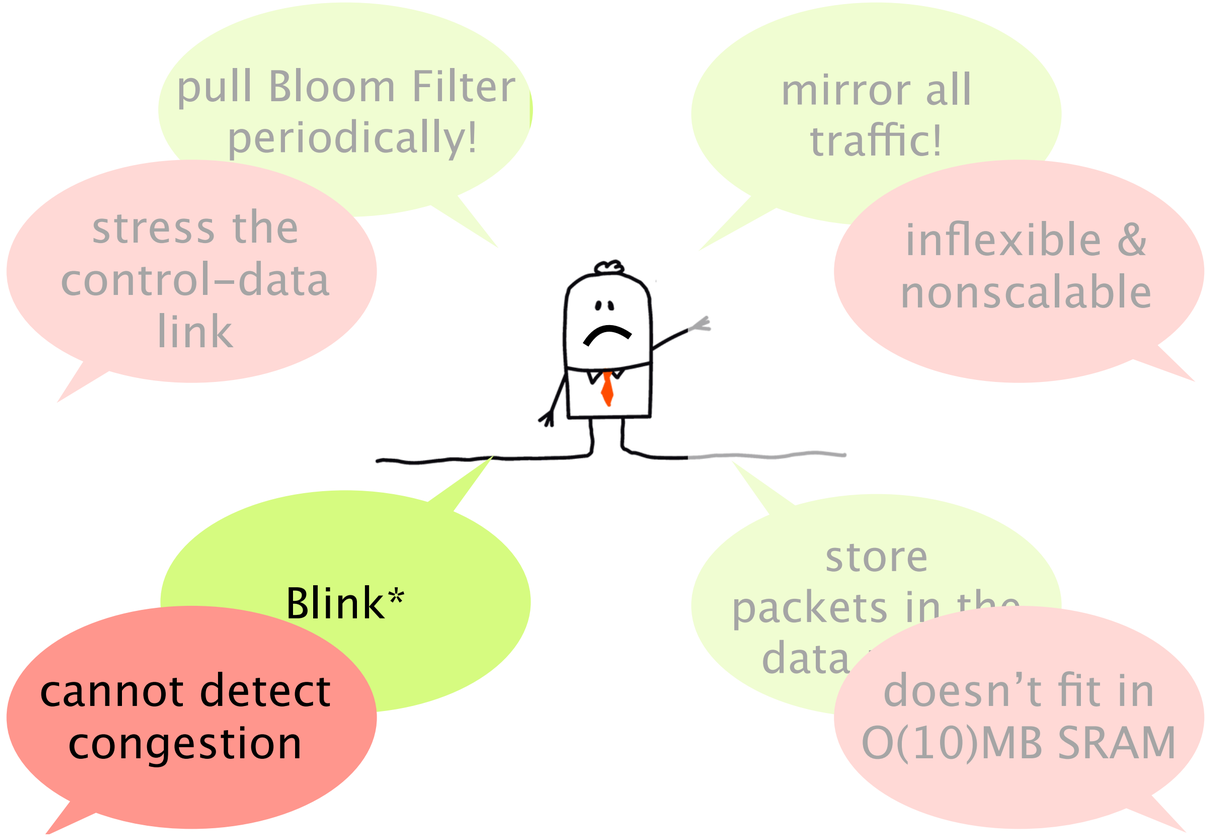
RouteScout measures loss rate via detecting retransmissions



*Holterbach et. al, NSDI '19

Blink: Fast Connectivity Recovery Entirely in the Data Plane

RouteScout measures loss rate via detecting retransmissions



*Holterbach et. al, NSDI '19

The Loss Monitor…

performs monitoring & aggregation in the data plane

The Loss Monitor…

- performs monitoring & aggregation in the data-plane
- minimizes communication between the control- and data-plane

The Loss Monitor…

- performs monitoring & aggregation in the data plane
- minimizes communication between the control and data plane
- scales with #flows (not #packets)

The Loss Monitor…

- performs monitoring & aggregation in the data plane
- minimizes communication between the control and data plane
- scales with #flows (not #packets)

To measure loss uses a Count Min Sketch and an Aggregator

Count Min Sketch	
1	
2	
3	
4	
5	
6	
7	
8	

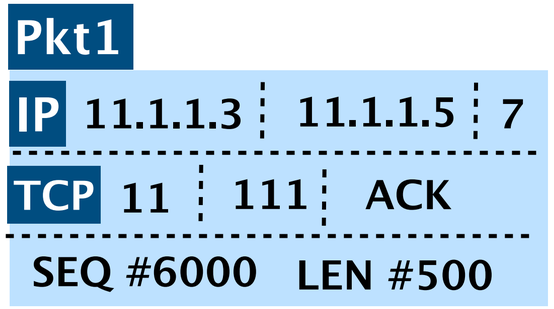
Loss Aggregator		
Prefix, NextHop	Expected	Lost
C,A		
C, B		

The Loss monitor uses a Count Min Sketch to detect retransmissions

	Count Min Sketch
1	0
2	0
3	1
4	2
5	0
6	0
7	0
8	0

Each non-empty packet, say Pkt1,

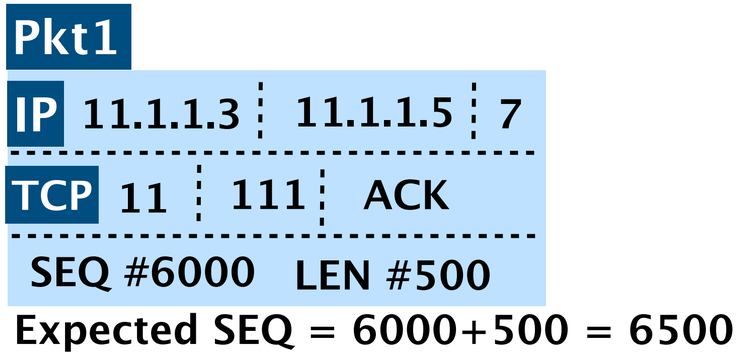
triggers the insertion of the next expected one to the CMS



Count Min Sketch	
1	0
2	0
3	1
4	2
5	0
6	0
7	0
8	0

Each non-empty packet, say Pkt1,

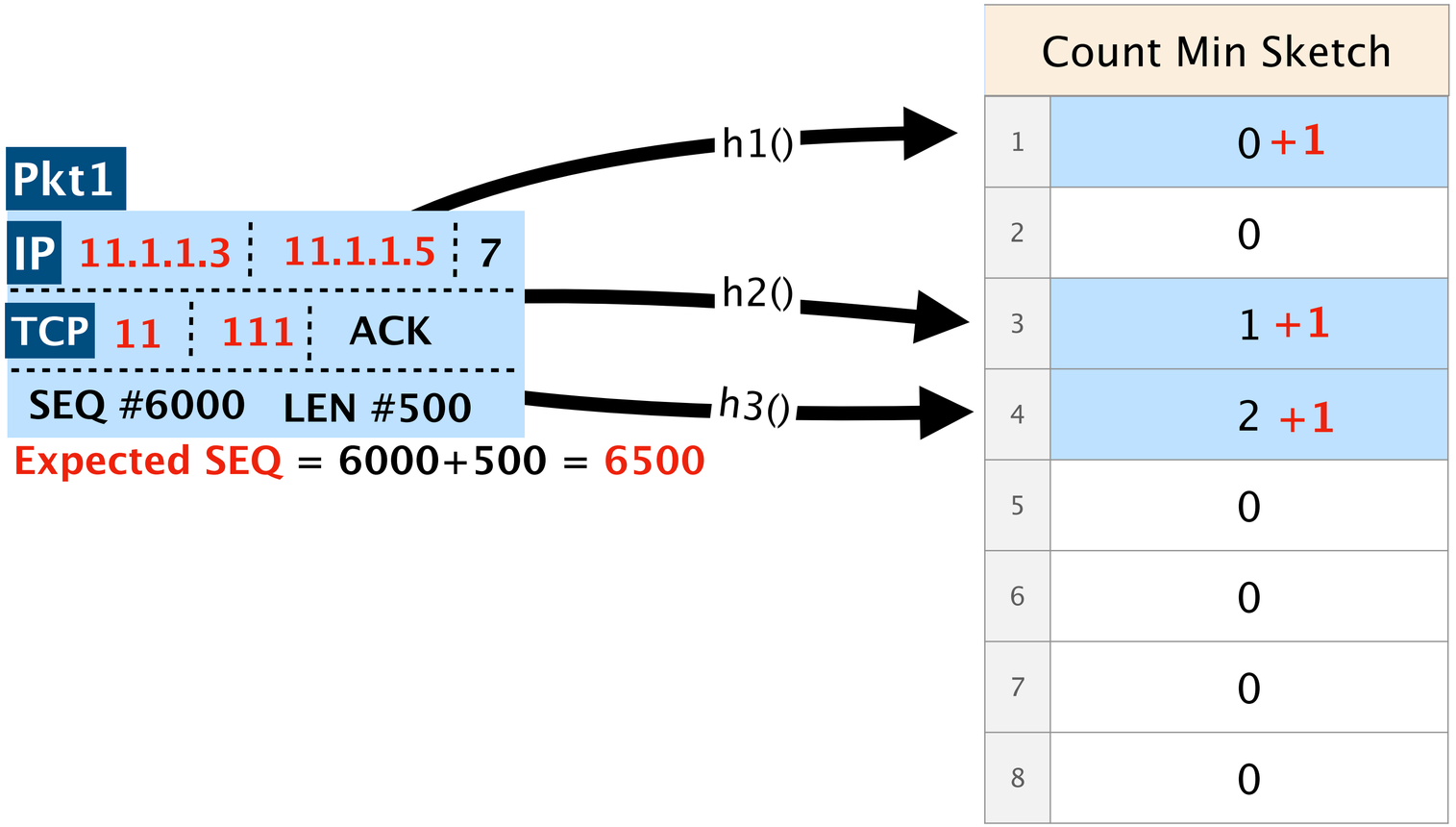
triggers the insertion of the next expected one to the CMS



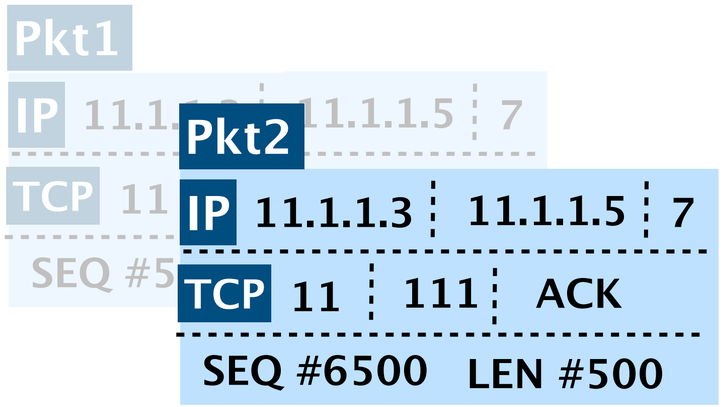
Count Min Sketch	
1	0
2	0
3	1
4	2
5	0
6	0
7	0
8	0

Each non-empty packet, say Pkt1,

triggers the insertion of the next expected one to the CMS

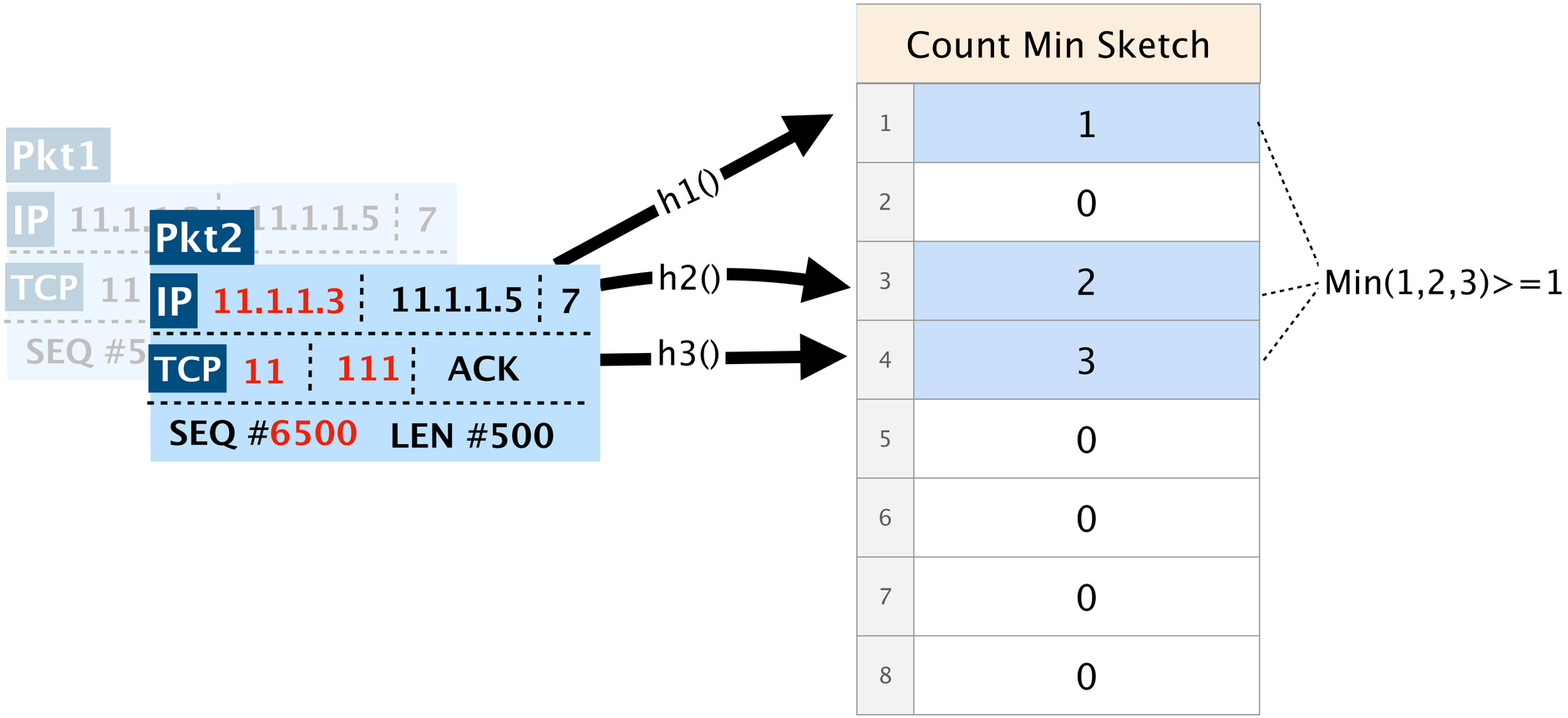


Step 1: Pkt2 arrives, it verifies that it is expected

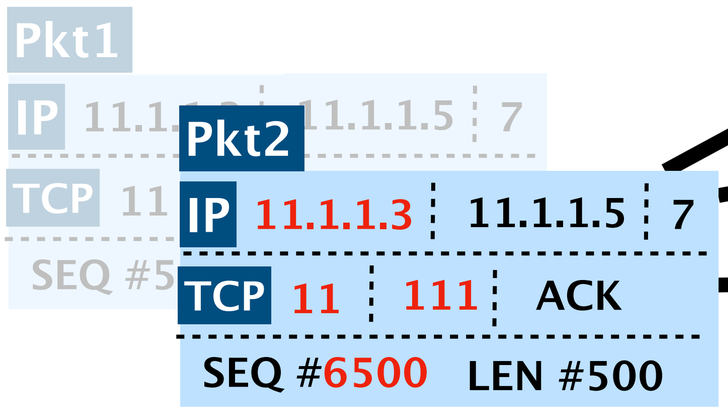


Count Min Sketch	
1	1
2	0
3	2
4	3
5	0
6	0
7	0
8	0

Step 1: Pkt2 arrives, it verifies that it is expected



Step 2: Pkt2 triggers an update in the Loss Aggregator



h1()

h2()

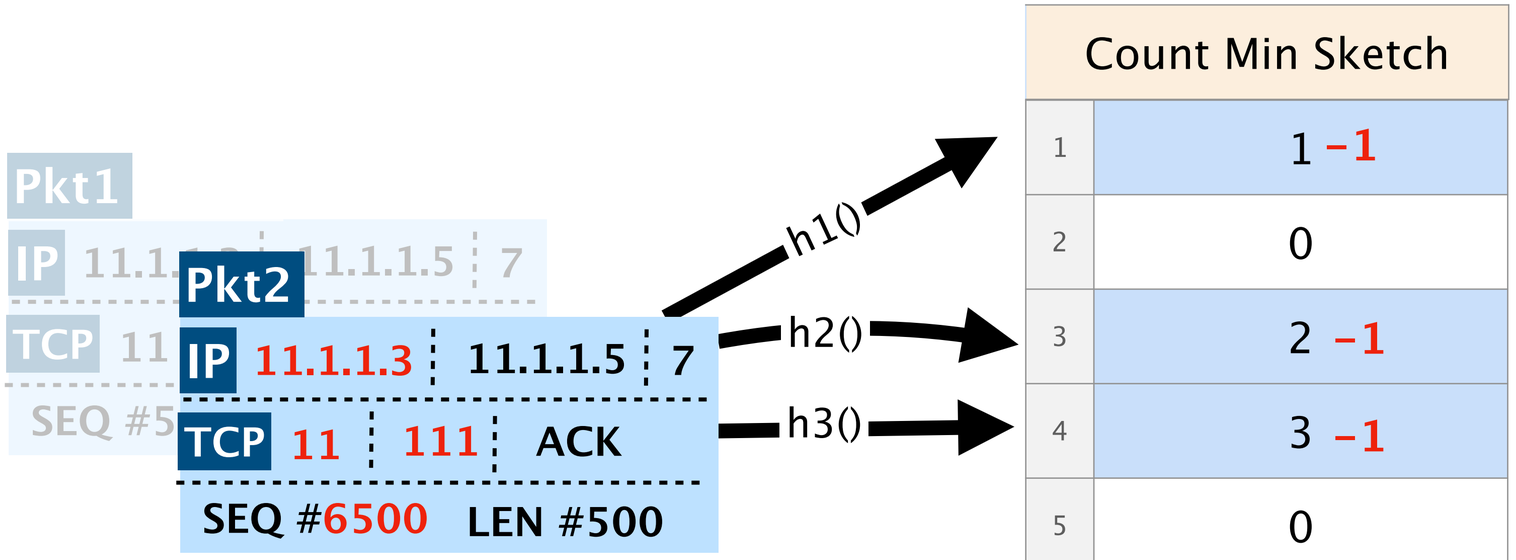
h3()

Count Min Sketch	
1	1
2	0
3	2
4	3
5	0
6	0
7	0
8	0

Min(1,2,3)>=1

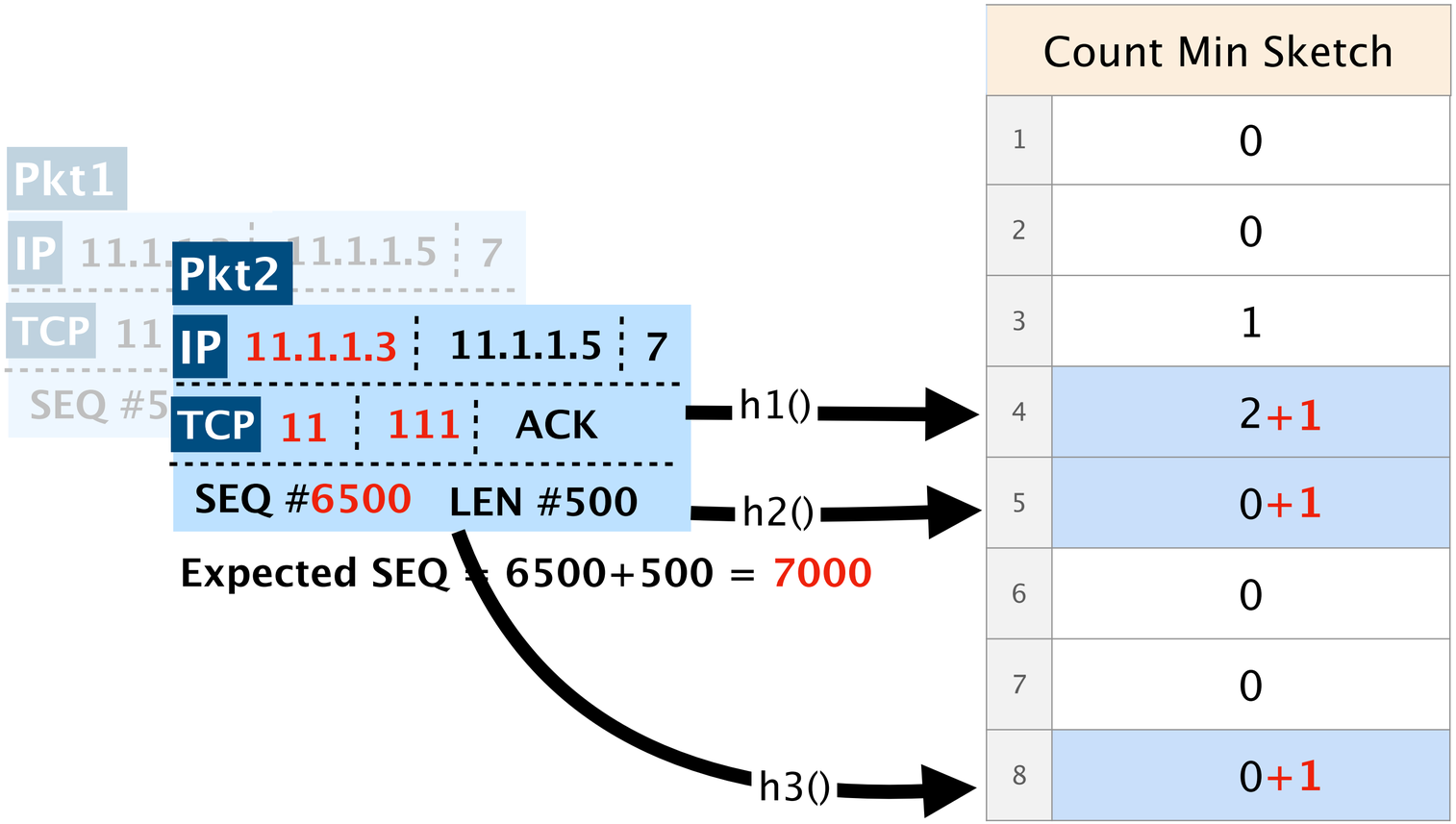
Loss Aggregator		
Prefix, NextHop	Expected	Lost
C,A	12+1	2
C, B	0	0

Step 3: Pkt2 cleans the filters



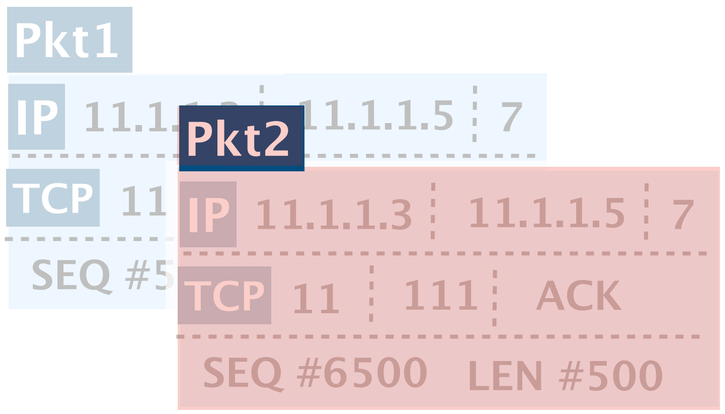
Count Min Sketch	
1	1 -1
2	0
3	2 -1
4	3 -1
5	0
6	0
7	0
8	0

Step 4: Pkt2 triggers the insertion the next expected packet, Pkt3



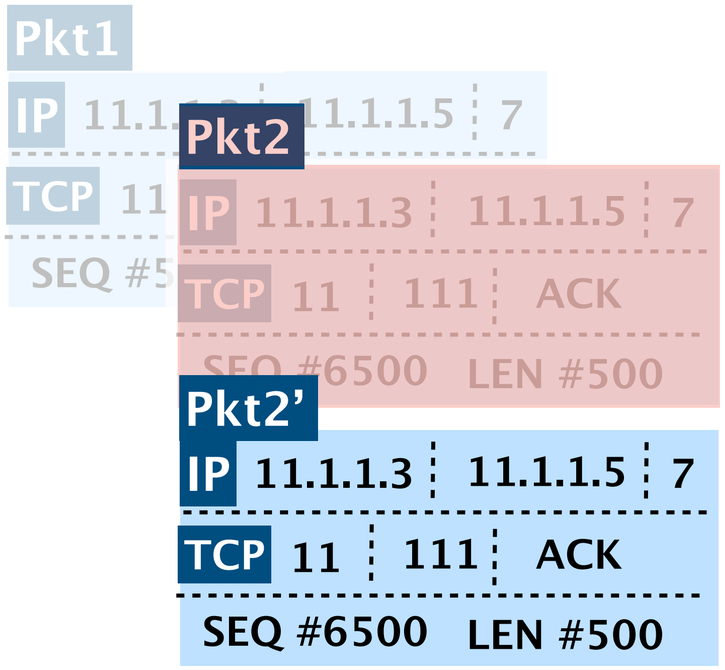
Count Min Sketch	
1	0
2	0
3	1
4	2+1
5	0+1
6	0
7	0
8	0+1

If Pkt2 is lost, it will be retransmitted



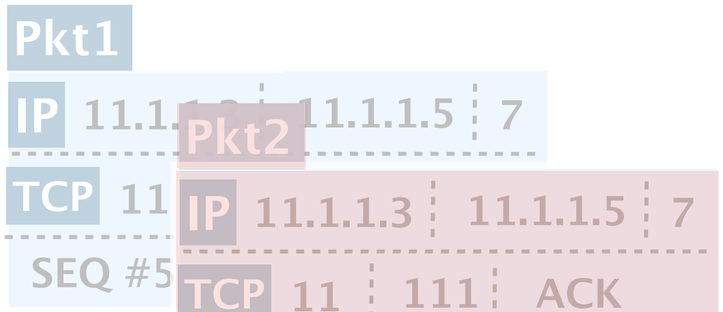
Count Min Sketch	
1	0
2	0
3	1
4	3
5	1
6	0
7	0
8	1

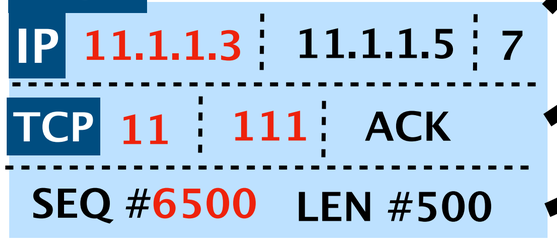
If Pkt2 is lost, it will be retransmitted

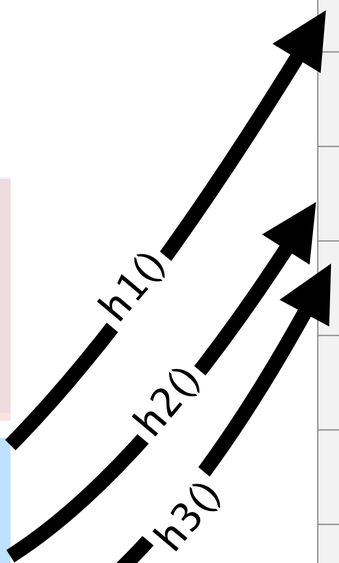


Count Min Sketch	
1	0
2	0
3	1
4	3
5	1
6	0
7	0
8	1

The retransmitted packet, Pkt2', will find that it is not expected





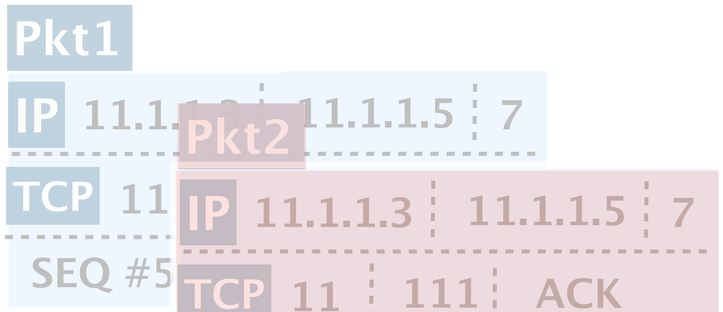


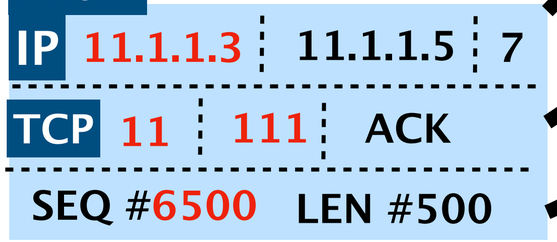
Count Min Sketch	
1	0
2	0
3	1
4	3
5	1
6	0
7	0
8	1

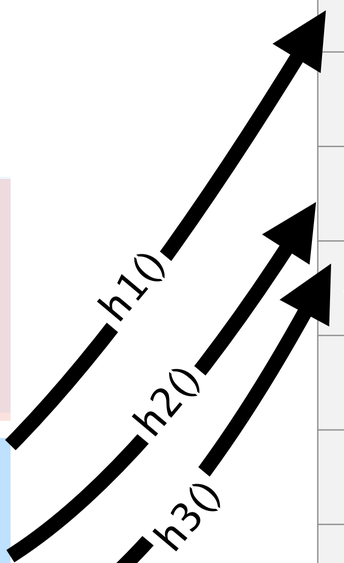
Min(3,1,1)>=1

Loss Aggregator		
Prefix, NextHop	Expected	Lost
C,A	13	2
C, B	0	0

The retransmitted packet, Pkt2', will find that it is not expected







Count Min Sketch	
1	0
2	0
3	1
4	3
5	1
6	0
7	0
8	1

Loss Aggregator		
Prefix, NextHop	Expected	Lost
C,A	13	2 +1
C, B	0	0

RouteScout: Performance-Driven Internet Path Selection

RouteScout Operations

RouteScout Architecture

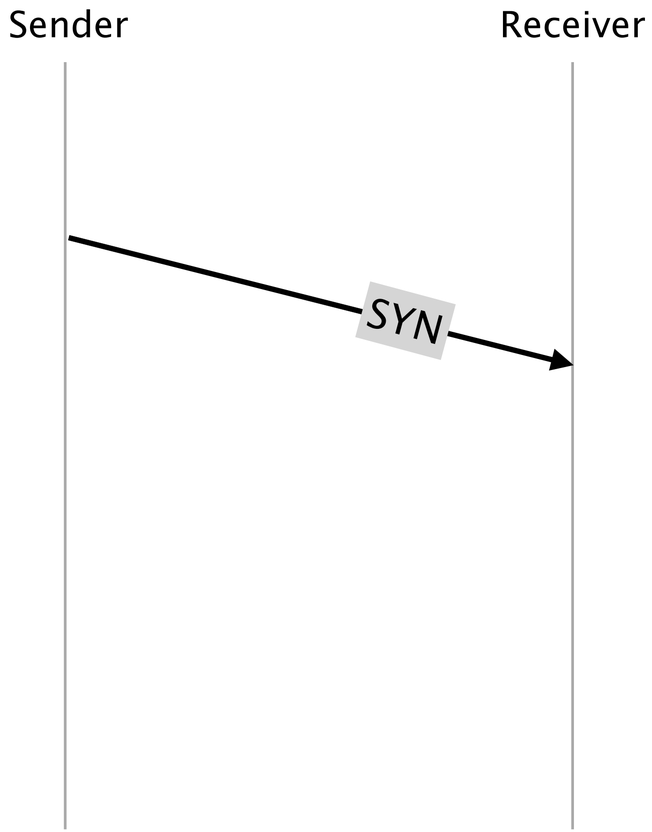
Data-Plane Design

- Loss Monitor
- Delay Monitor

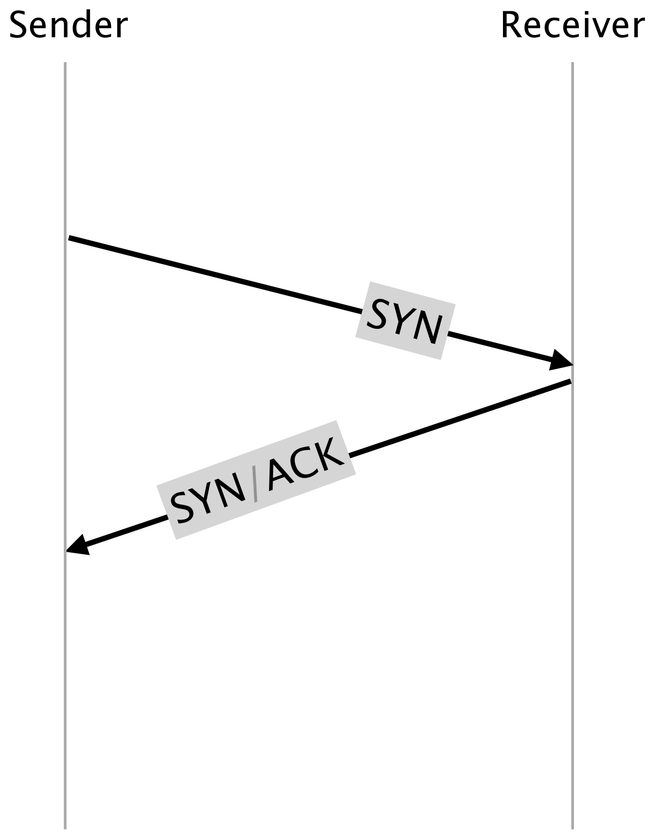
Evaluation

RouteScout measures delay by measuring the difference between SYN-ACK

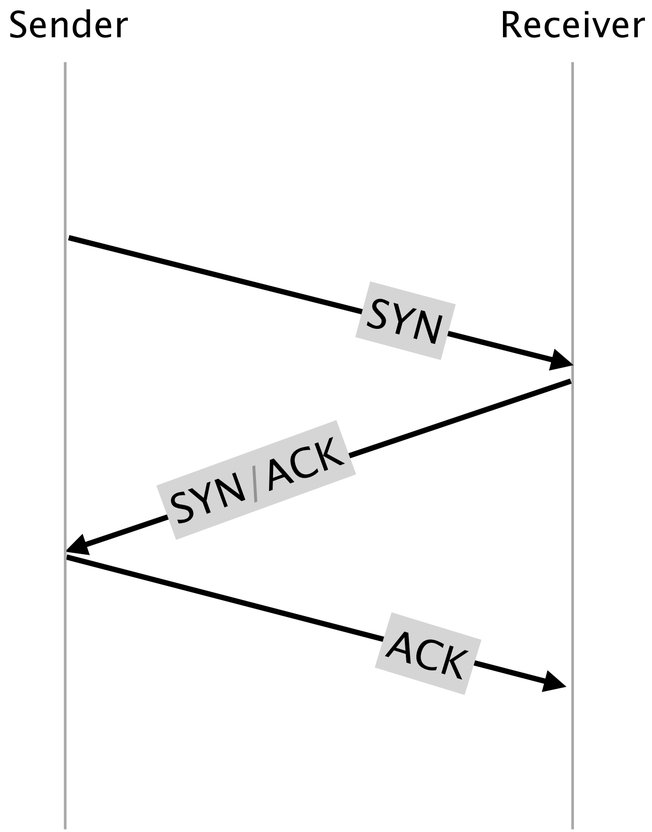
RouteScout measures delay by measuring the difference between SYN-ACK



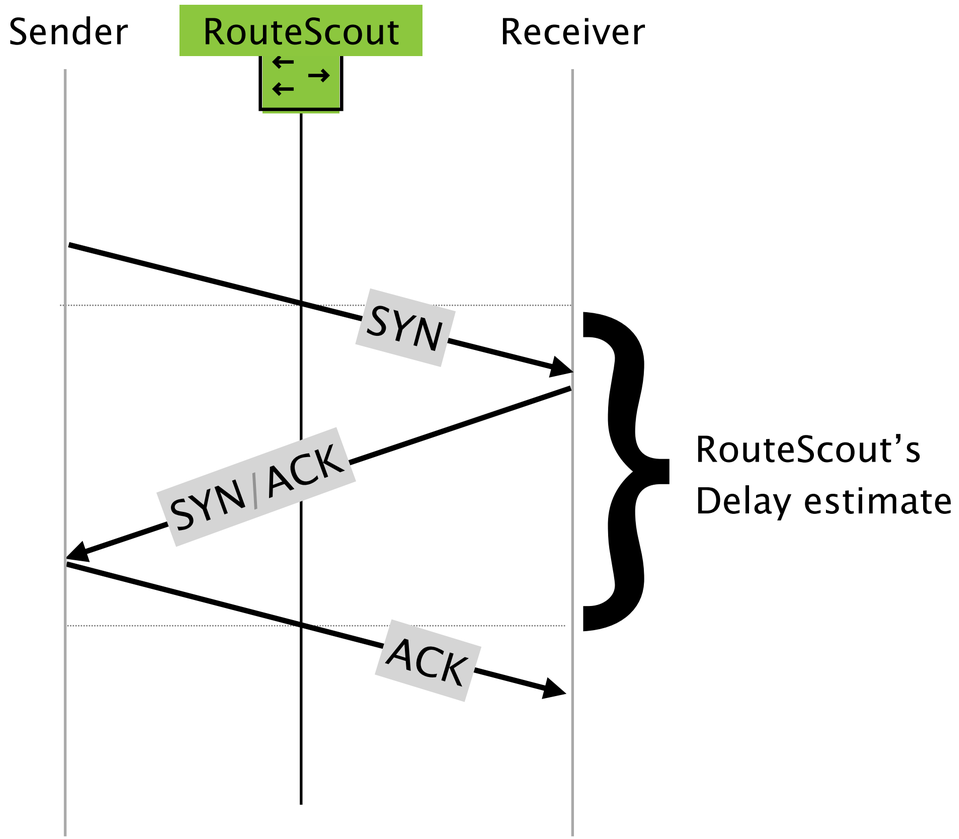
RouteScout measures delay by measuring the difference between SYN-ACK



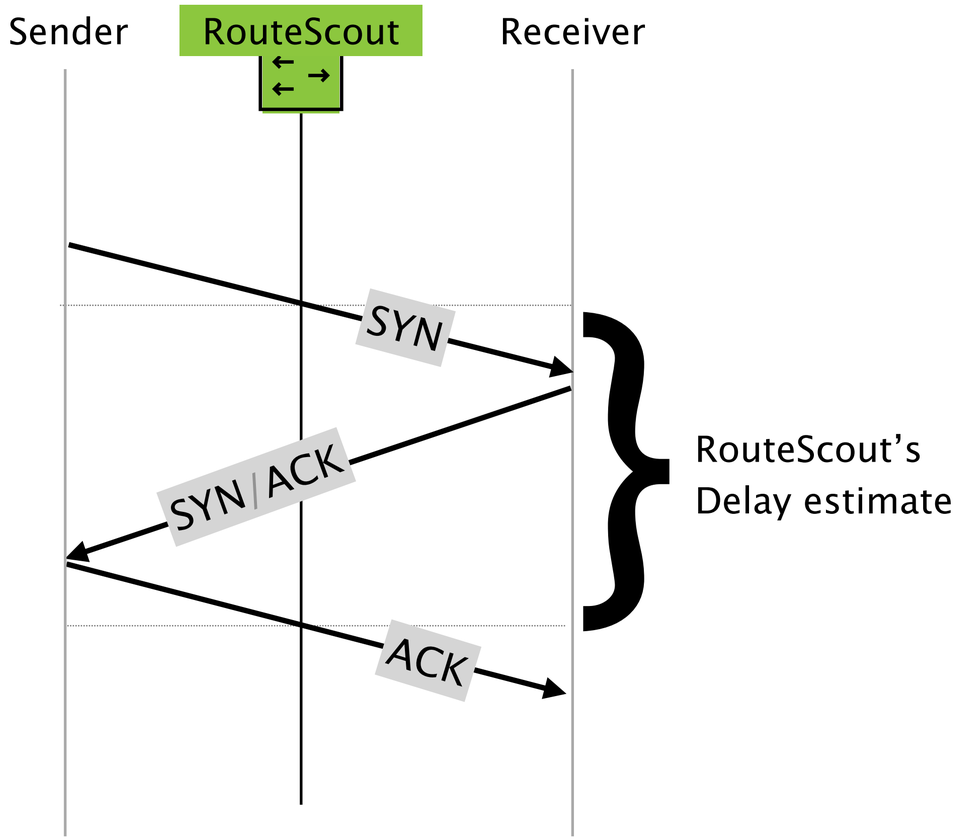
RouteScout measures delay by measuring the difference between SYN-ACK



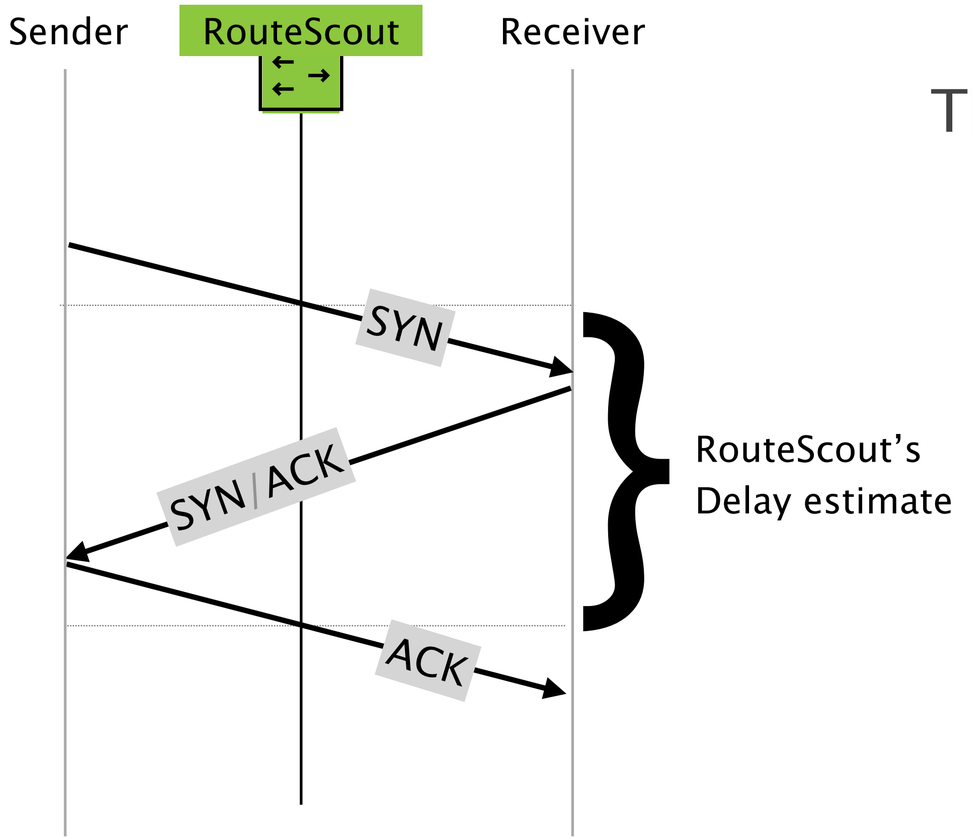
RouteScout measures delay by measuring the difference between SYN-ACK



RouteScout measures delay by measuring the difference between SYN-ACK



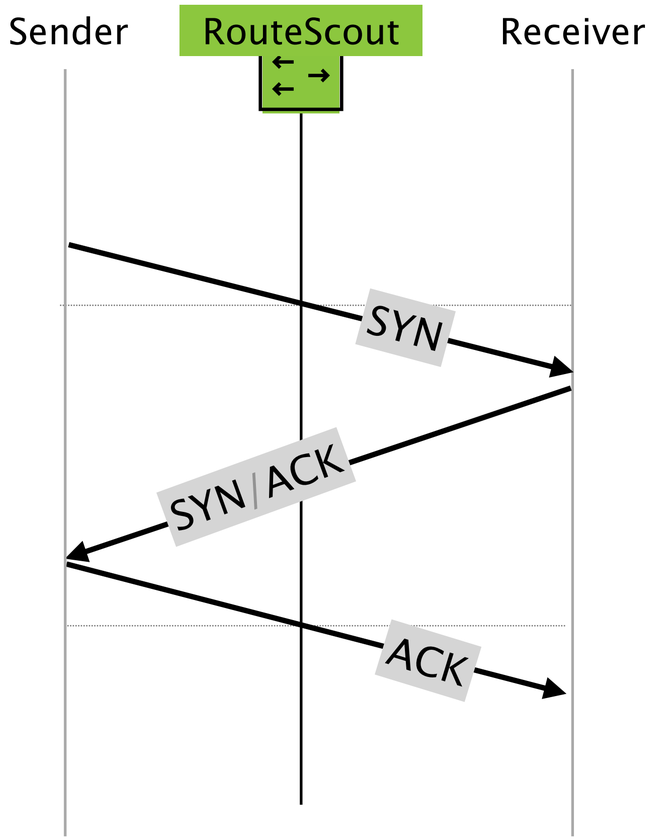
RouteScout measures delay by measuring the difference between SYN-ACK



The Delay Monitor…

does not require bidirectional traffic

RouteScout measures delay by measuring the difference between SYN-ACK



The Delay Monitor…

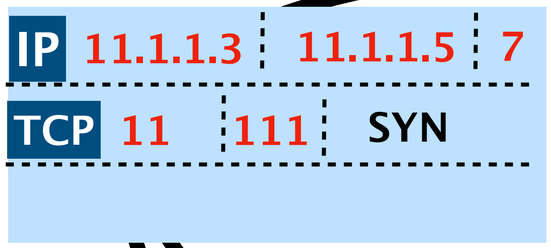
- does not require bidirectional traffic
- performs monitoring in the data plane
- performs aggregation in the data plane

To measure delay uses an Invertible Bloom Filter and an Aggregator

	Accumulator	CNT
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0	0

Delay Aggregator		
Prefix, NextHop	Sum Delays	# Delays

Upon arrival of a SYN packet its timestamp (T1) is XORed to multiple indexes



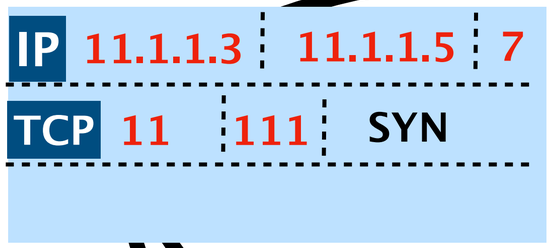
h1(5-tuple)

h2(5-tuple)

h3(5-tuple)

	Accumulator	CNT
1	$0 \oplus T1$	0
2	0	0
3	0	0
4	0	0
5	$0 \oplus T1$	0
6	$0 \oplus T1$	0
7	0	0
8	0	0

Upon arrival of a SYN packet the corresponding counters increase by one



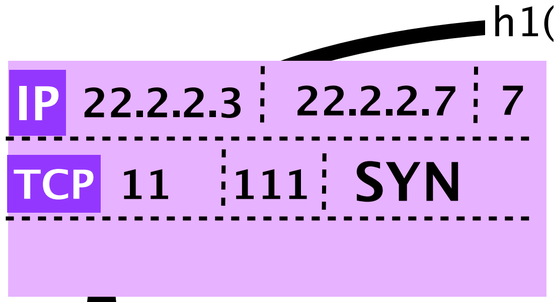
h1(5-tuple)

h2(5-tuple)

h3(5-tuple)

	Accumulator	CNT
1	T_1	0+1
2	0	0
3	0	0
4	0	0
5	T_1	0+1
6	T_1	0+1
7	0	0
8	0	0

Similar updates happen for each SYN packets



h1(5-tuple)

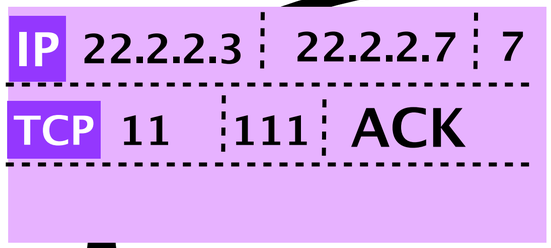
h2(5-tuple)

h3(5-tuple)

	Accumulator	CNT
1	$T1 \oplus T2$	2
2	0	0
3	0	0
4	0	0
5	T_1	1
6	$T_1 \oplus T_2$	2
7	0	0
8	$0 \oplus T2$	1

Upon arrival of an ACK Packet one of the indexes

Contains an unpolluted timestamp of the corresponding SYN



h1(5-tuple)

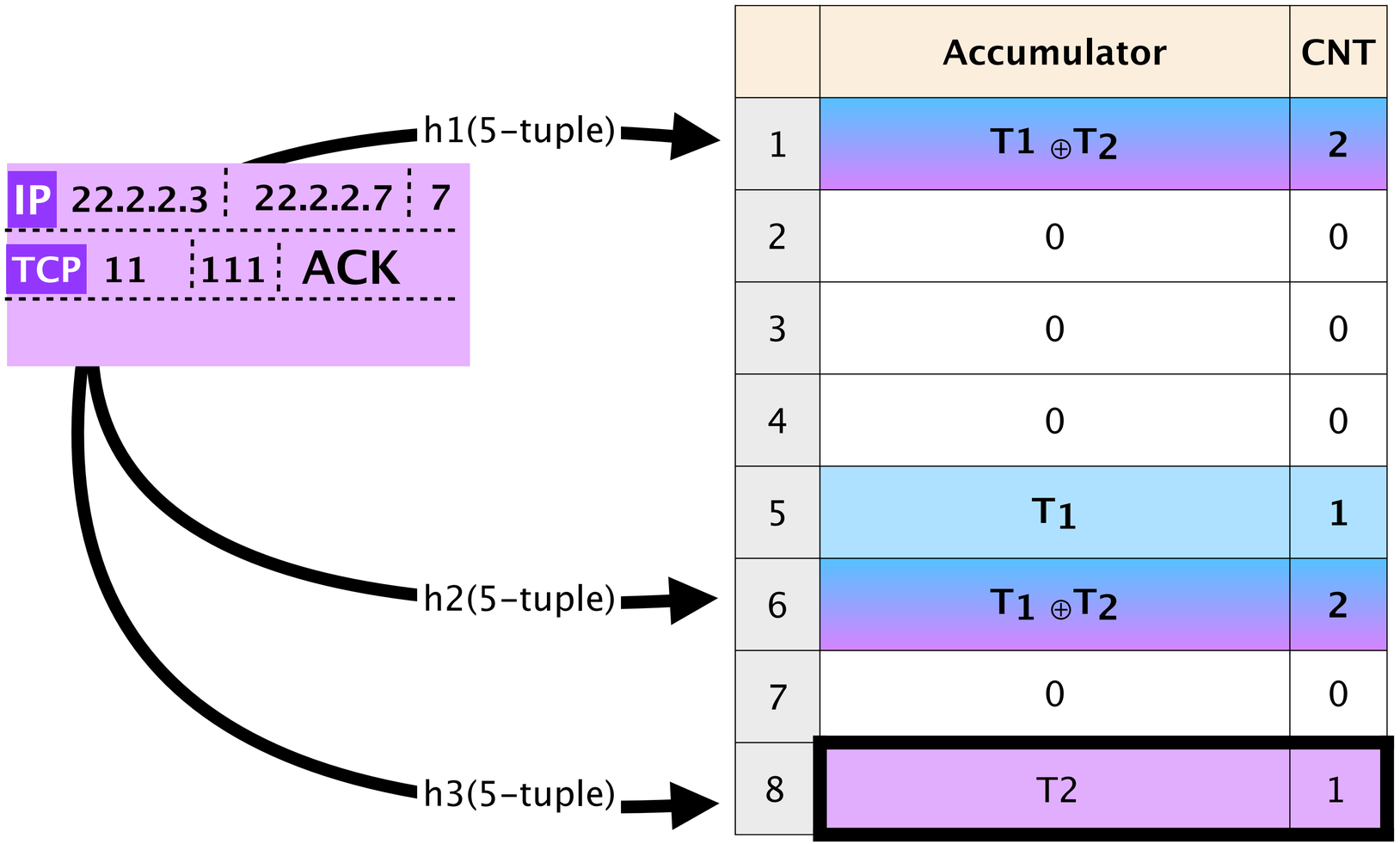
h2(5-tuple)

h3(5-tuple)

	Accumulator	CNT
1	$T1 \oplus T2$	2
2	0	0
3	0	0
4	0	0
5	T_1	1
6	$T_1 \oplus T_2$	2
7	0	0
8	T2	1

Upon arrival of an ACK Packet one of the indexes

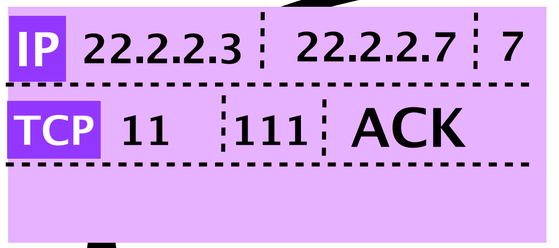
Contains an unpolluted timestamp of the corresponding SYN



	Accumulator	CNT
1	$T1 \oplus T2$	2
2	0	0
3	0	0
4	0	0
5	T_1	1
6	$T_1 \oplus T_2$	2
7	0	0
8	T2	1

Upon arrival of an ACK Packet one of the indexes

Contains an unpolluted timestamp of the corresponding SYN



h1(5-tuple)

h2(5-tuple)

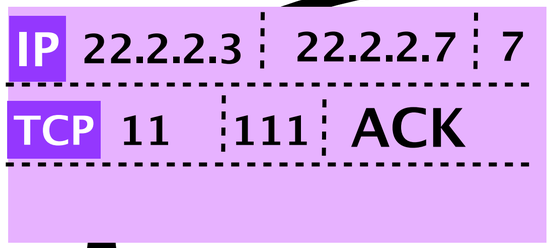
h3(5-tuple)

	Accumulator	CNT
1	$T1 \oplus T2$	2
2	0	0
3	0	0
4	0	0
5	T_1	1
6	$T_1 \oplus T_2$	2
7	0	0
8	T2	1

delay=Tack-Tsyn= T2-T3

Upon arrival of an ACK Packet one of the indexes

Contains an unpolluted timestamp of the corresponding SYN



h1(5-tuple)

h2(5-tuple)

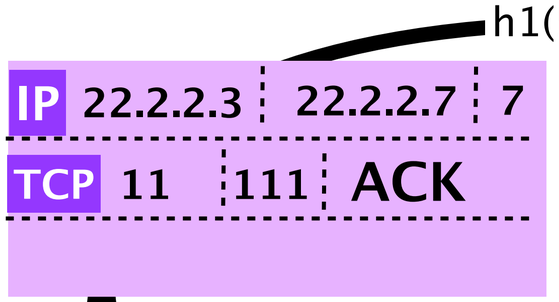
h3(5-tuple)

	Accumulator	CNT
1	$T1 \oplus T2$	2
2	0	0
3	0	0
4	0	0
5	T_1	1
6	$T1 \oplus T2$	2
7	0	0
8	T2	1

Delay Aggregator		
Prefix, NextHop	Sum Delays	# Delays
A	+=(T3-T2)	+=1
B		

delay=Tack-Tsyn= T3-T2

By XORing the timestamp to all indexes and the counters by one the IBF is cleared

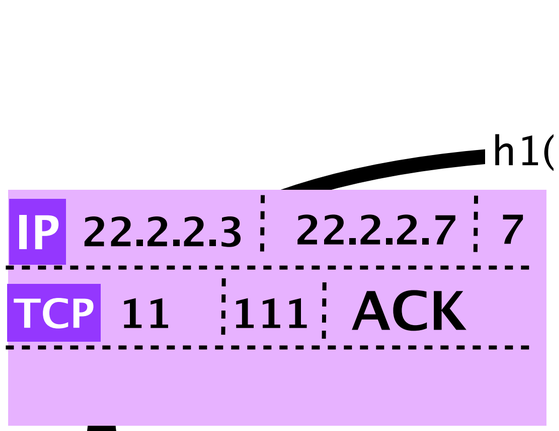


h1(5-tuple)

h2(5-tuple)

h3(5-tuple)

	Accumulator	CNT
1	$T1 \oplus T2 \oplus T2 = T1$	2-1
2	0	0
3	0	0
4	0	0
5	T_1	1
6	$T_1 \oplus T_2 \oplus T_2 = T_1$	2-1
7	0	0
8	$T2 \oplus T2 = 0$	1-1



h1(5-tuple)

h2(5-tuple)

h3(5-tuple)

	Accumulator	CNT
1	T_1	1
2	0	0
3	0	0
4	0	0
5	T_1	1
6	T_1	1
7	0	0
8	0	0

RouteScout: Performance-Driven Internet Path Selection

RouteScout Operations

RouteScout Architecture

Data-Plane Design

Evaluation

RouteScout: Performance-Driven Internet Path Selection

RouteScout Operations

RouteScout Architecture

Data-Plane Design

Evaluation

- Accuracy vs Memory Footprint
- Complexity vs Run Time
- Feasibility

RouteScout: Performance-Driven Internet Path Selection

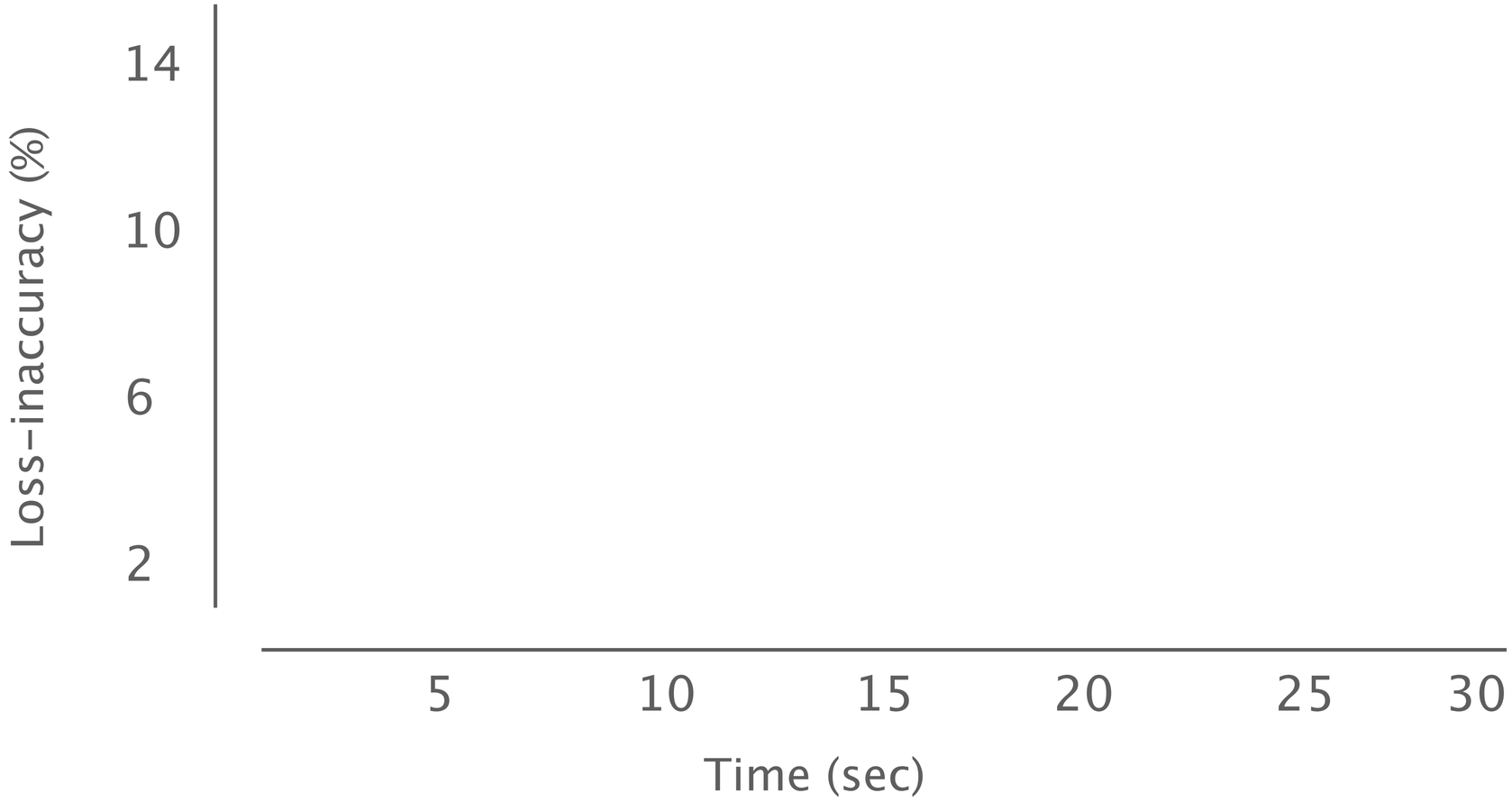
RouteScout Operations

RouteScout Architecture

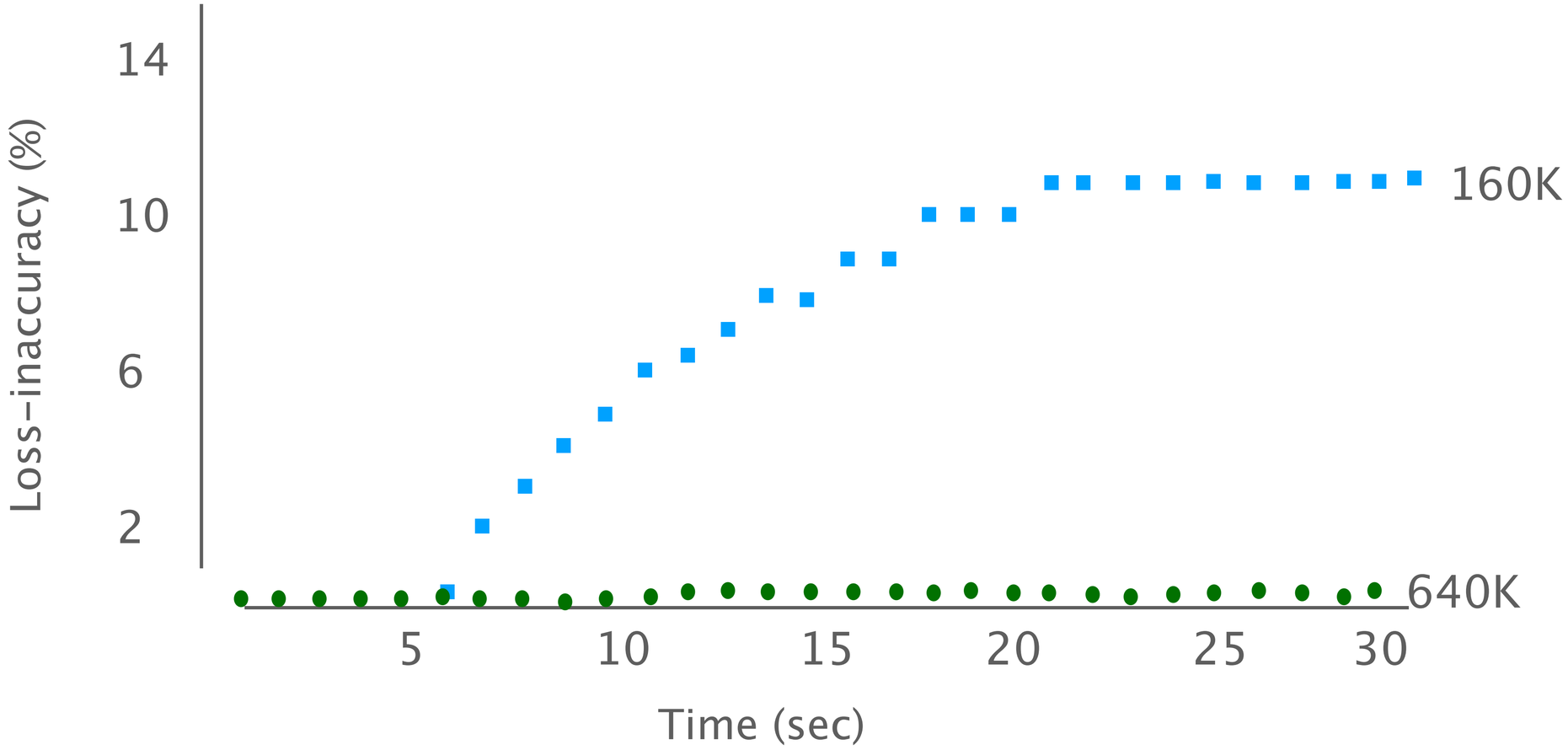
Data-Plane Design

Evaluation

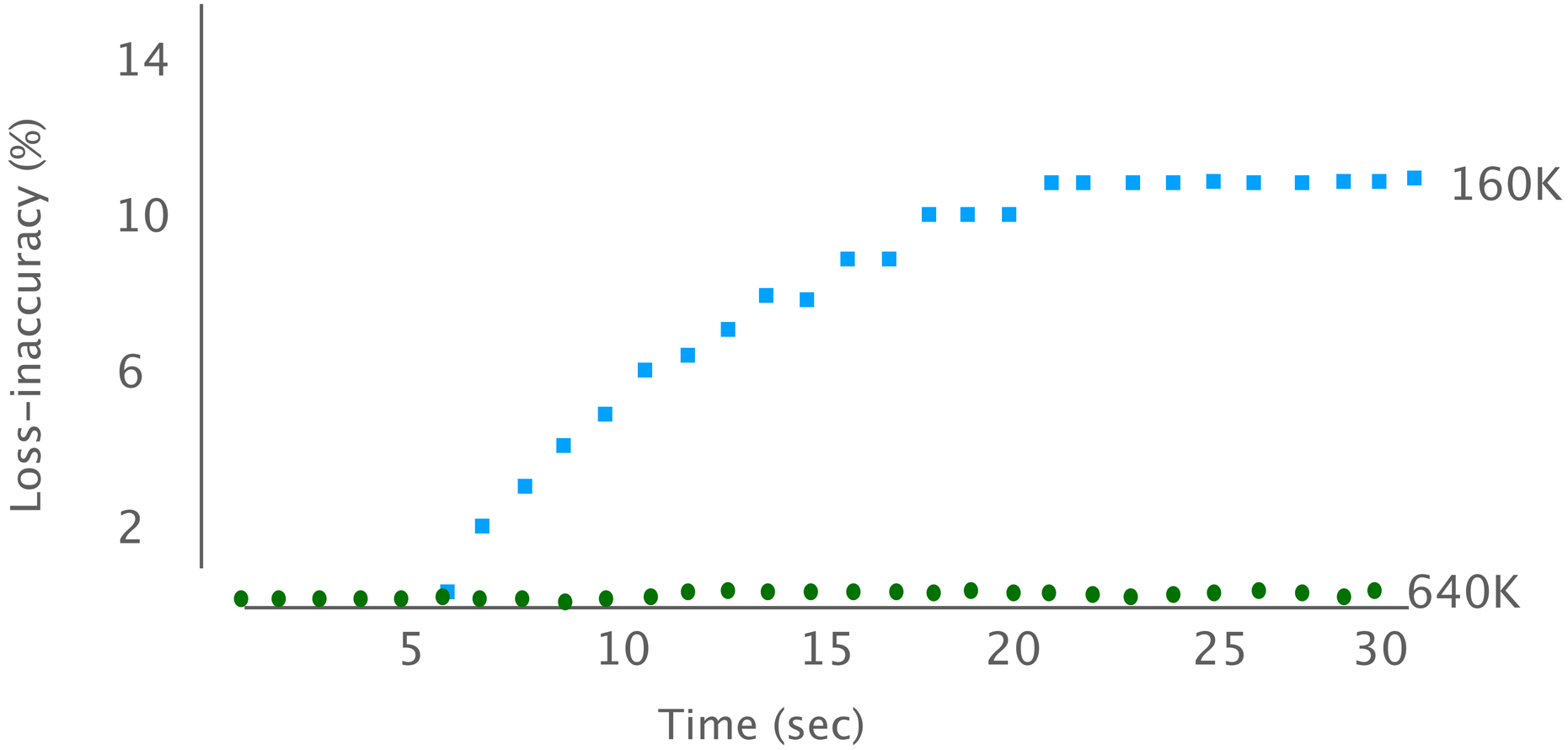
- Accuracy vs Memory Footprint
- Complexity vs Run Time
- Feasibility

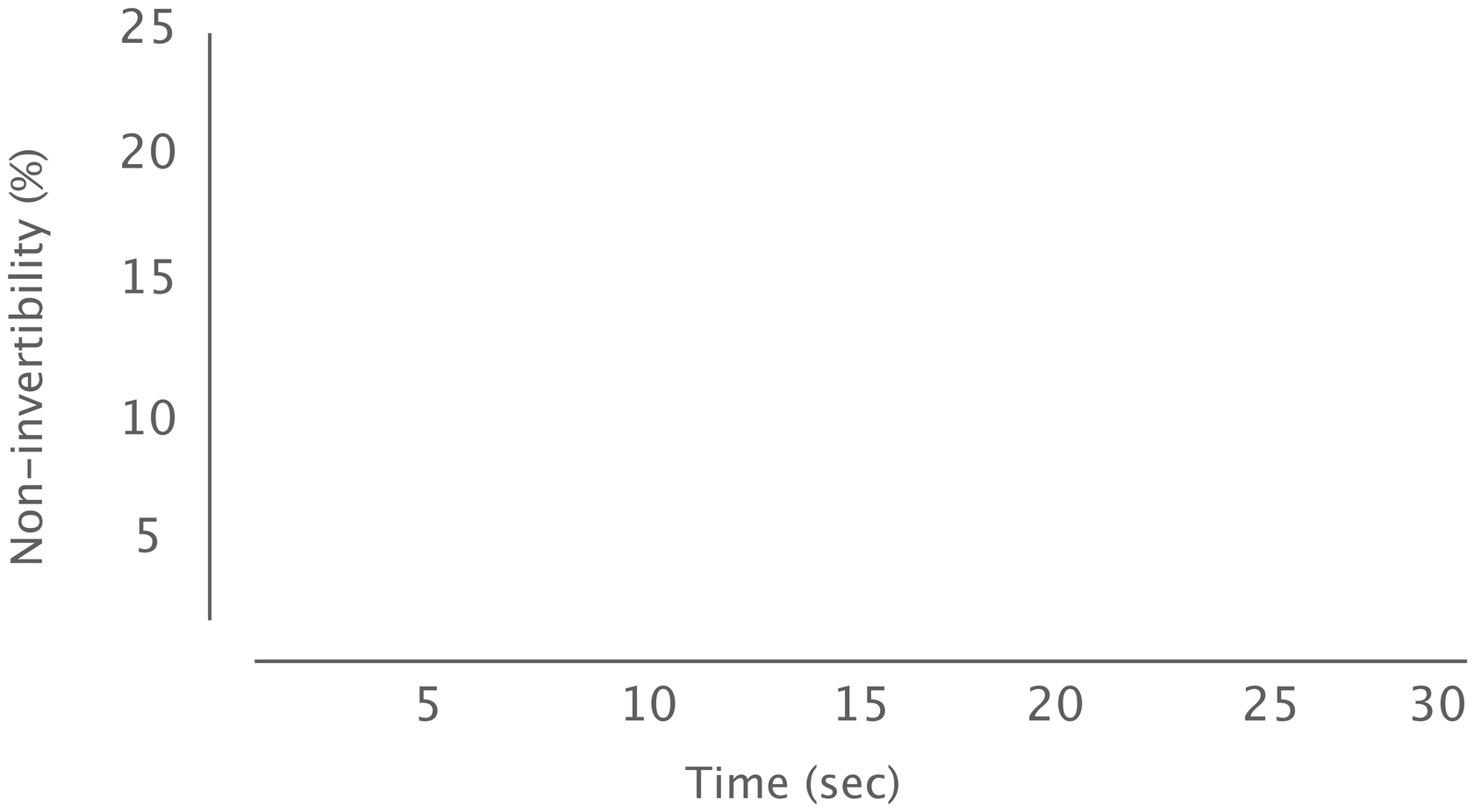

Loss-inaccuracy (%)
14
10
6
2
5
10
15
20
25
30
Time (sec)

Using 625KB (640K elements) and 2 hashes,
the Loss monitor calculates the loss rate with high accuracy

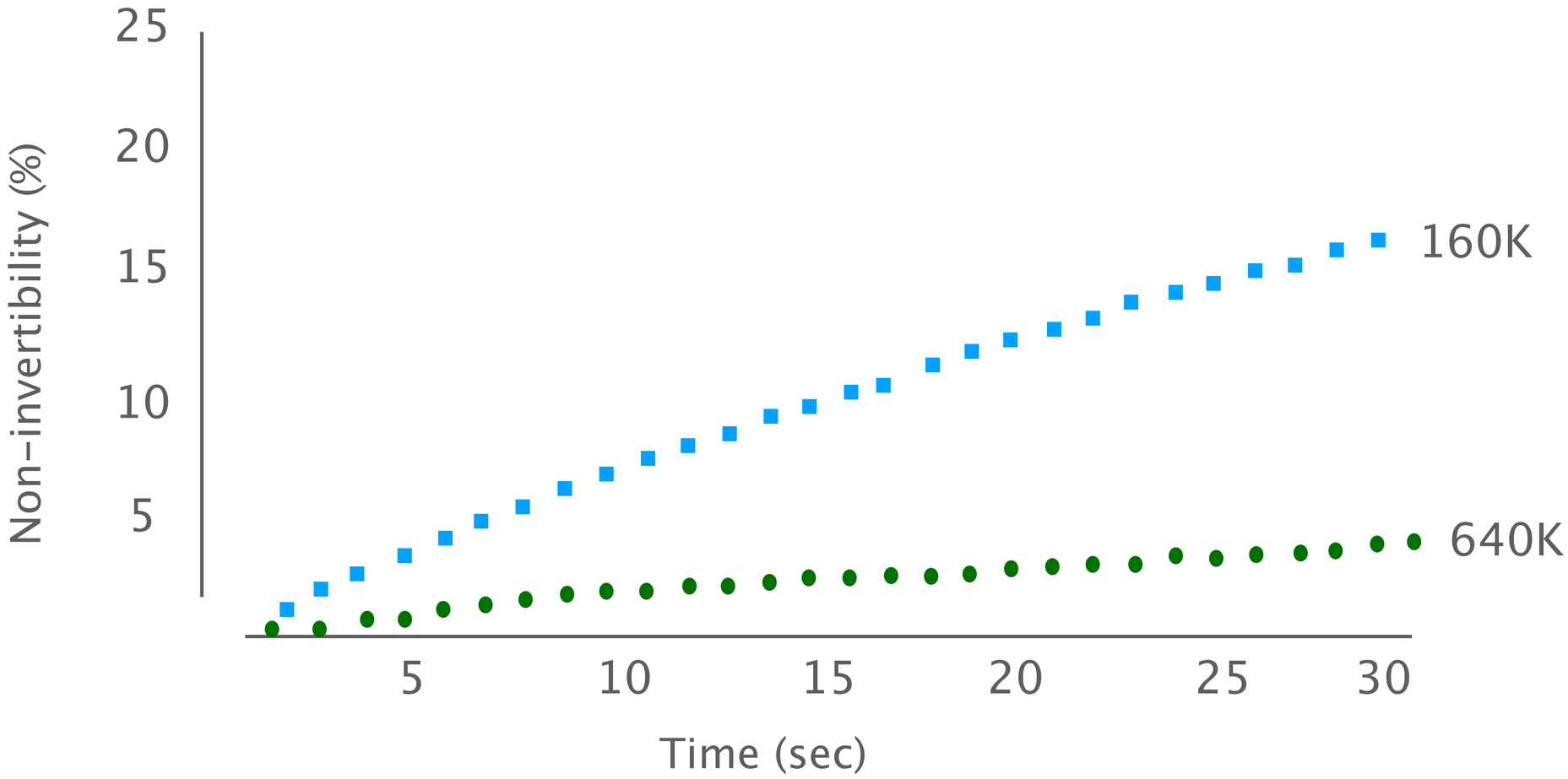


Using 625KB (640K elements) and 2 hashes,
the Loss monitor calculates the loss rate with high accuracy




25
20
15
10
5
Non-invertibility (%)
5
10
15
20
25
30
Time (sec)

The probability of an ACK to decode its SYN's timestamp is >95% with a 1MB (640K elements) delay monitor of 2 hashes



RouteScout: Performance-Driven Internet Path Selection

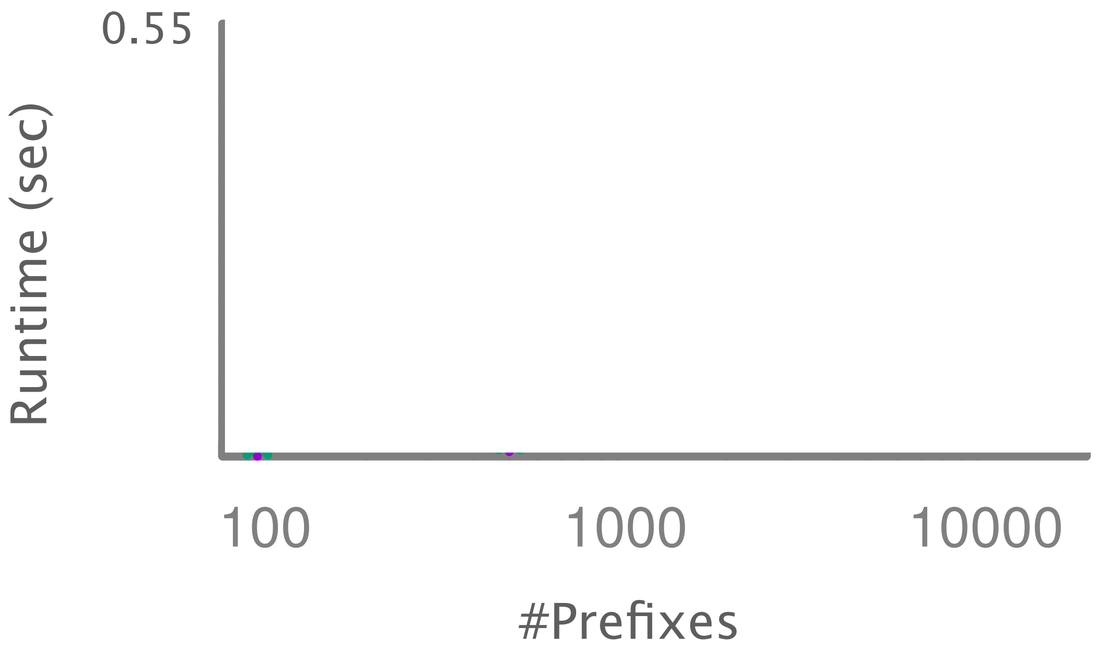
RouteScout Operations

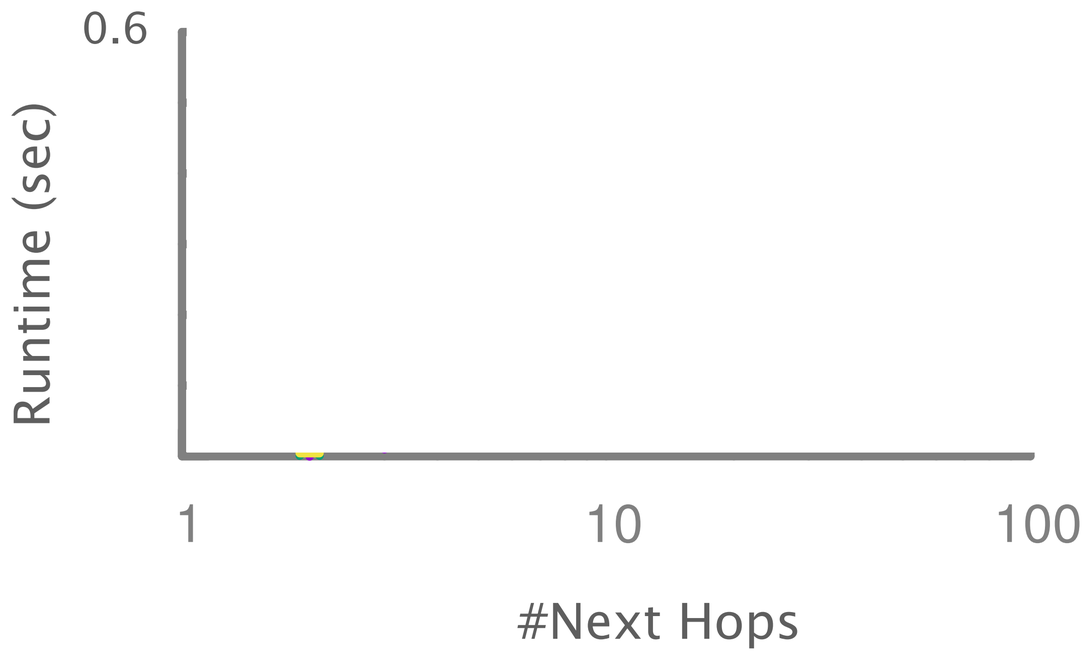
Routescout Architecture

Data-Plane Design

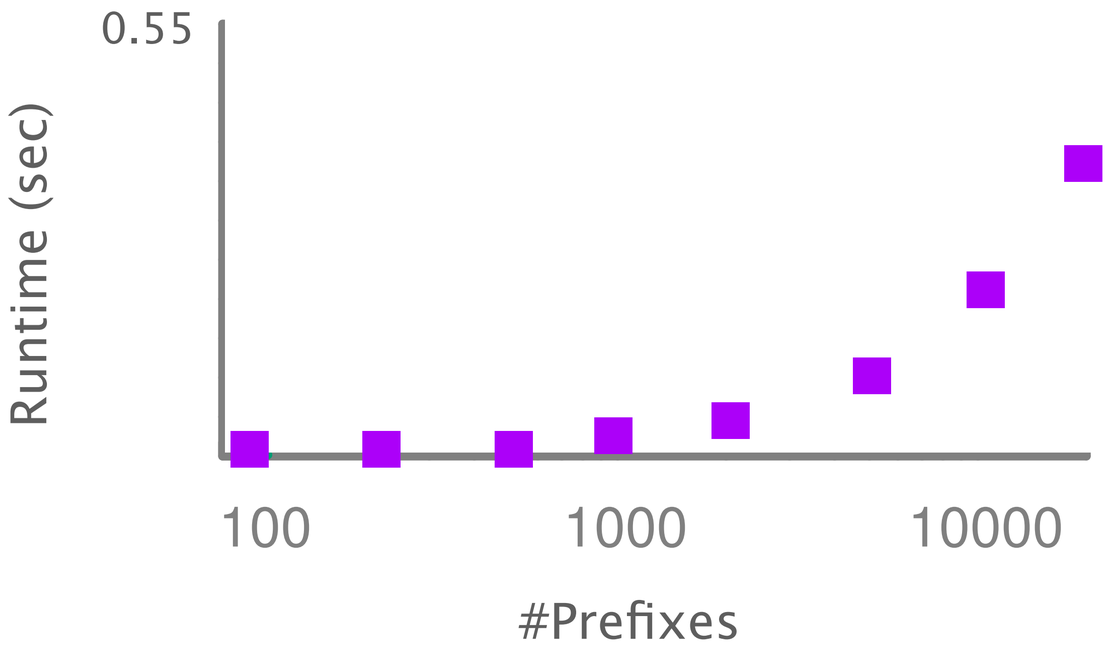
Evaluation

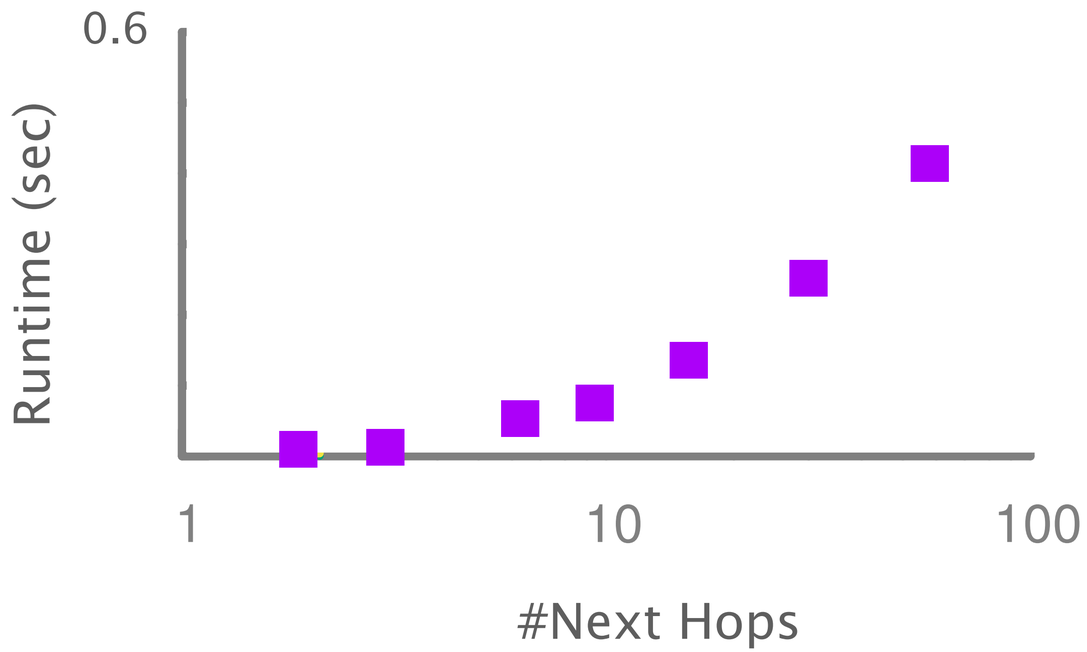
- Accuracy vs Memory footprint
- Complexity vs Run Time
- Feasibility


0.55
Runtime (sec)
100
1000
10000
#Prefixes


0.6
Runtime (sec)
1
10
100
#Next Hops

RouteScout is fast even when run with an increasing number of next hops and destinations





RouteScout: Performance-Driven Internet Path Selection

- RouteScout Operations
- RouteScout Architecture
- Data-Plane Design
- Evaluation
  - Accuracy vs Memory Footprint
  - Complexity vs Run Time
  - Feasibility

RouteScout: Performance-Driven Internet Path Selection

RouteScout Operations

RouteScout Architecture

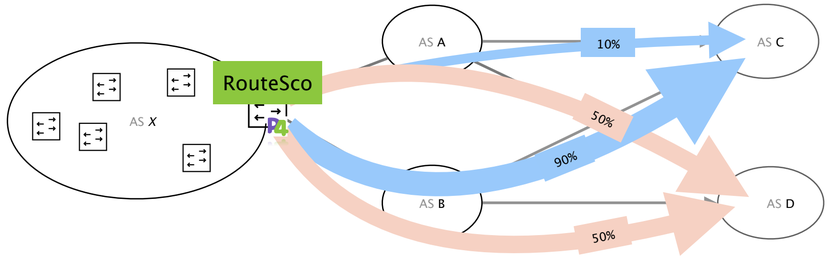
Data-Plane Design

Evaluation

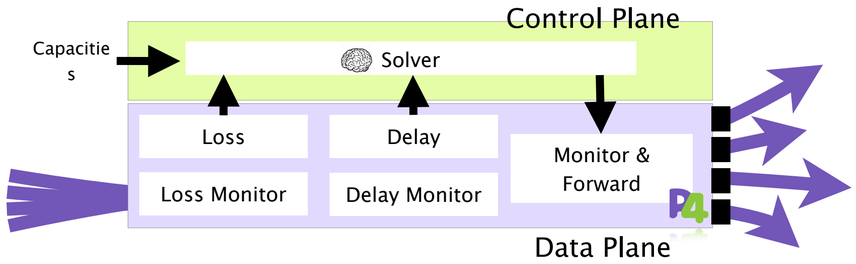
- Accuracy vs Memory Footprint
- Complexity vs Run Time
- Feasibility

RouteScout: Performance-Driven Internet Path Selection

RouteScout is a modern answer to the old problem of performance-aware Internet routing.



RouteScout is a closed-loop control system with hardware and software components that leverages programmable data planes.



RouteScout's data plane fits in today's devices while its control plane runs in sub-second operating times.

RouteScout

Performance-Driven Internet Path Selection