Source-domain DDoS Prevention

Bobby Bhattacharjee Dave Levin

Christopher Kommareddy Richard La

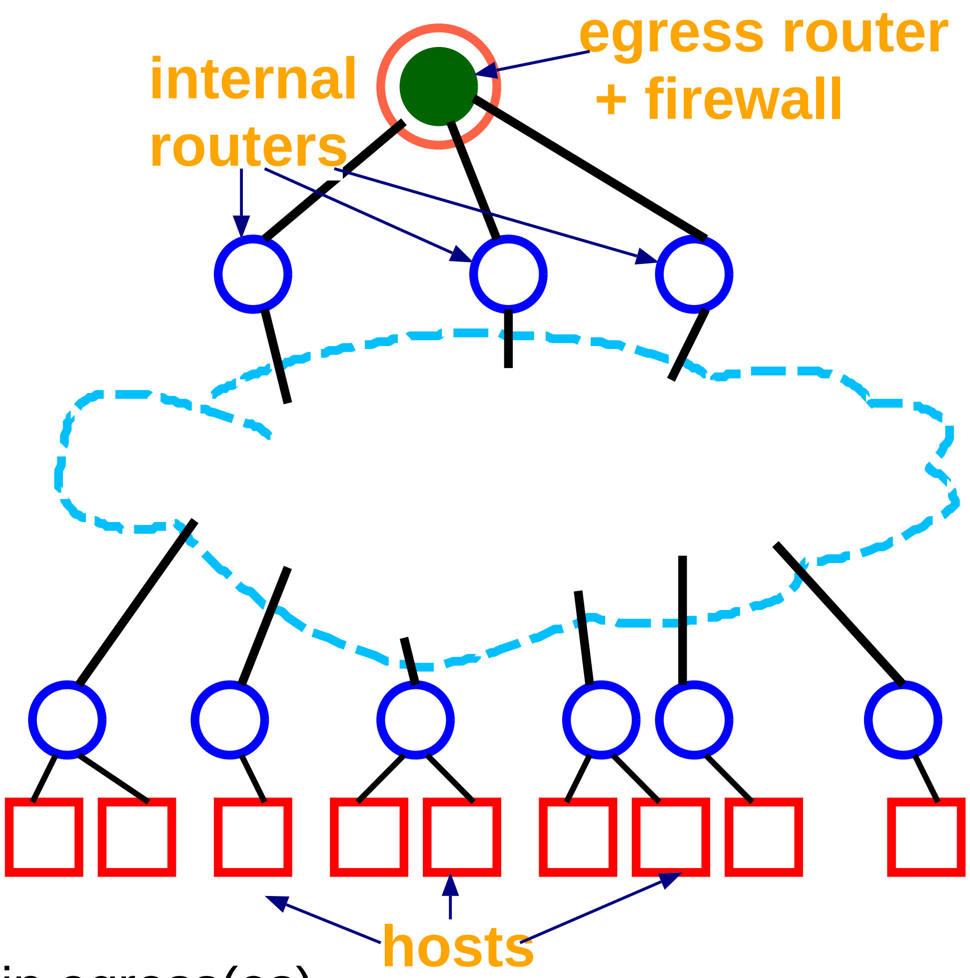
Mark Shayman Vahid Tabatabaee

University of Maryland

DDoS Prevention at the Source

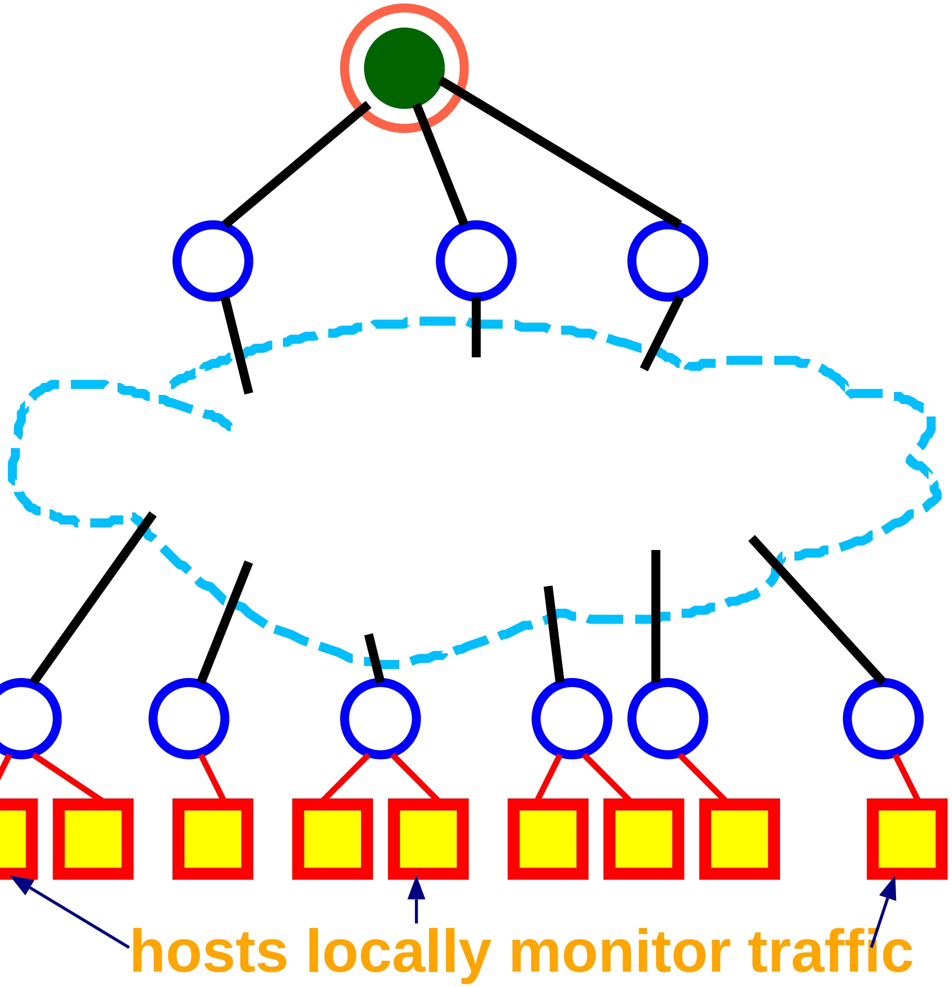
- Monitor and stop attacks at the *source* of the attack
- Does not require Internet-wide deployment
- Most efficient solution — attacks are stopped before they can do much damage
- Shares the cost of attack monitoring and prevention

Approaches



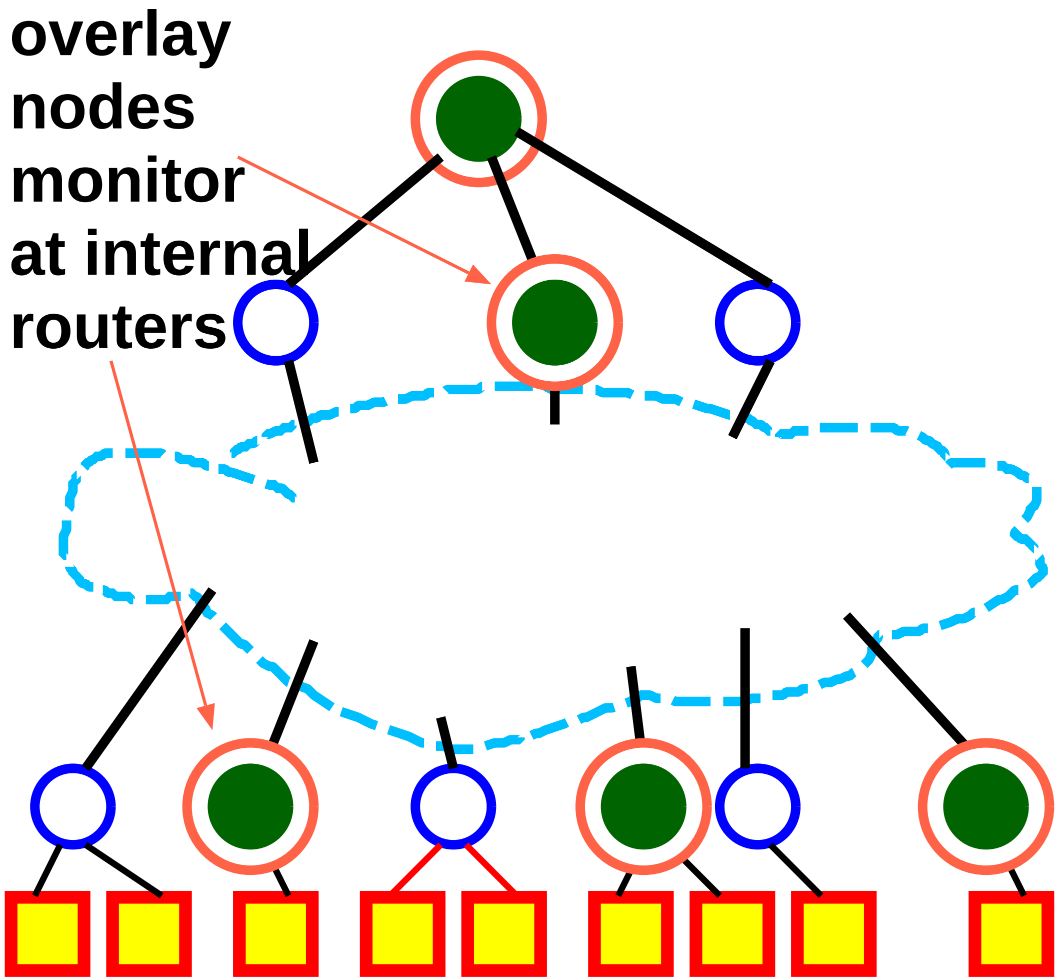
- Firewall at the domain egress(es)

Approaches



- Monitor at each host

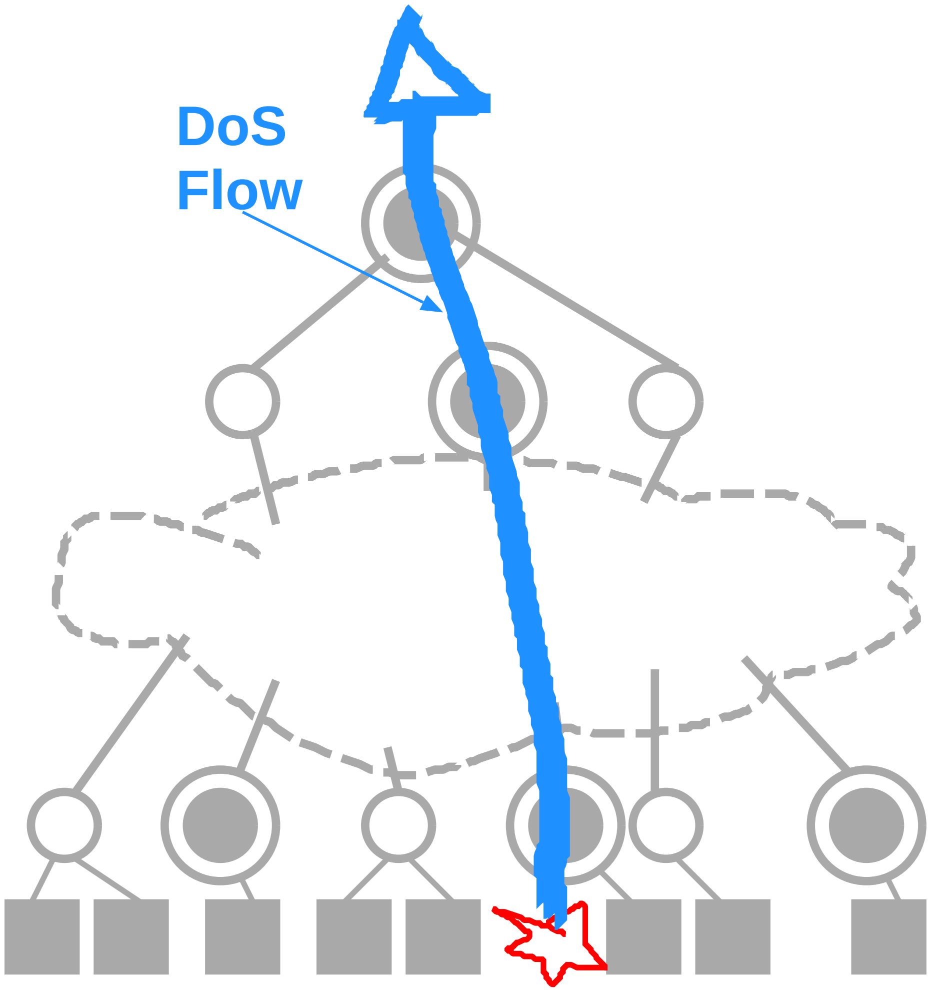
Approaches



- Overlay-based

Solution components — new ideas

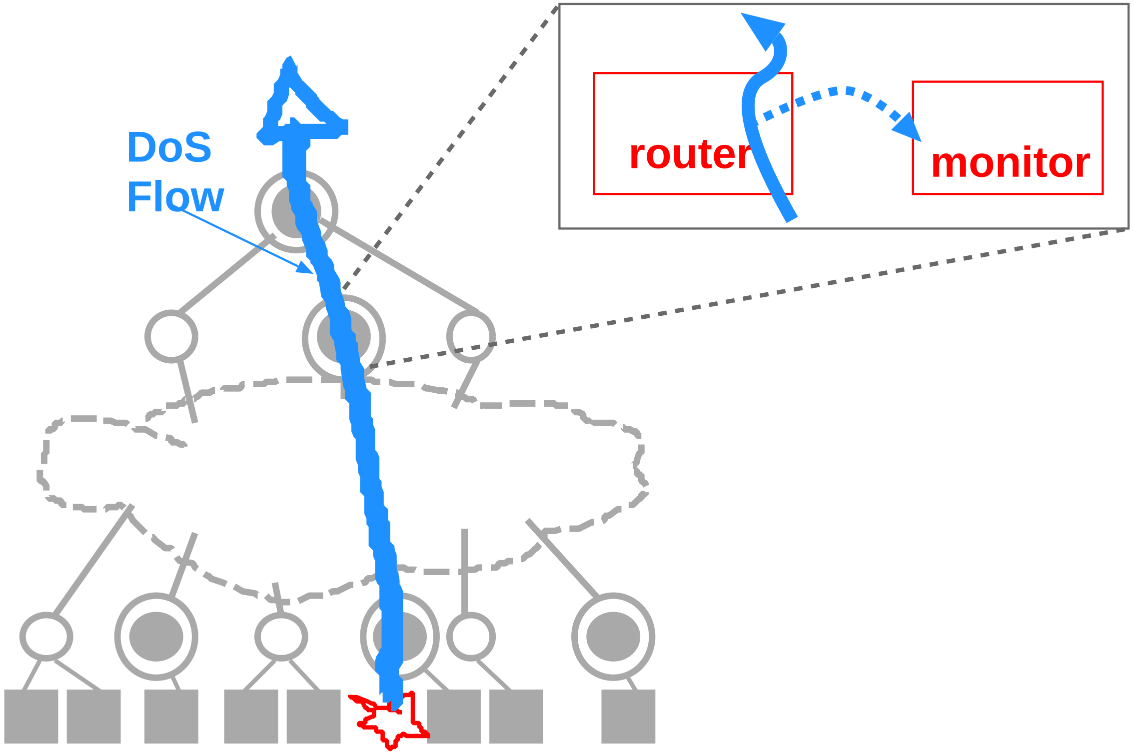
- Coordinate and Correlate information between nodes



- Local Oracle

Source-domain Monitoring

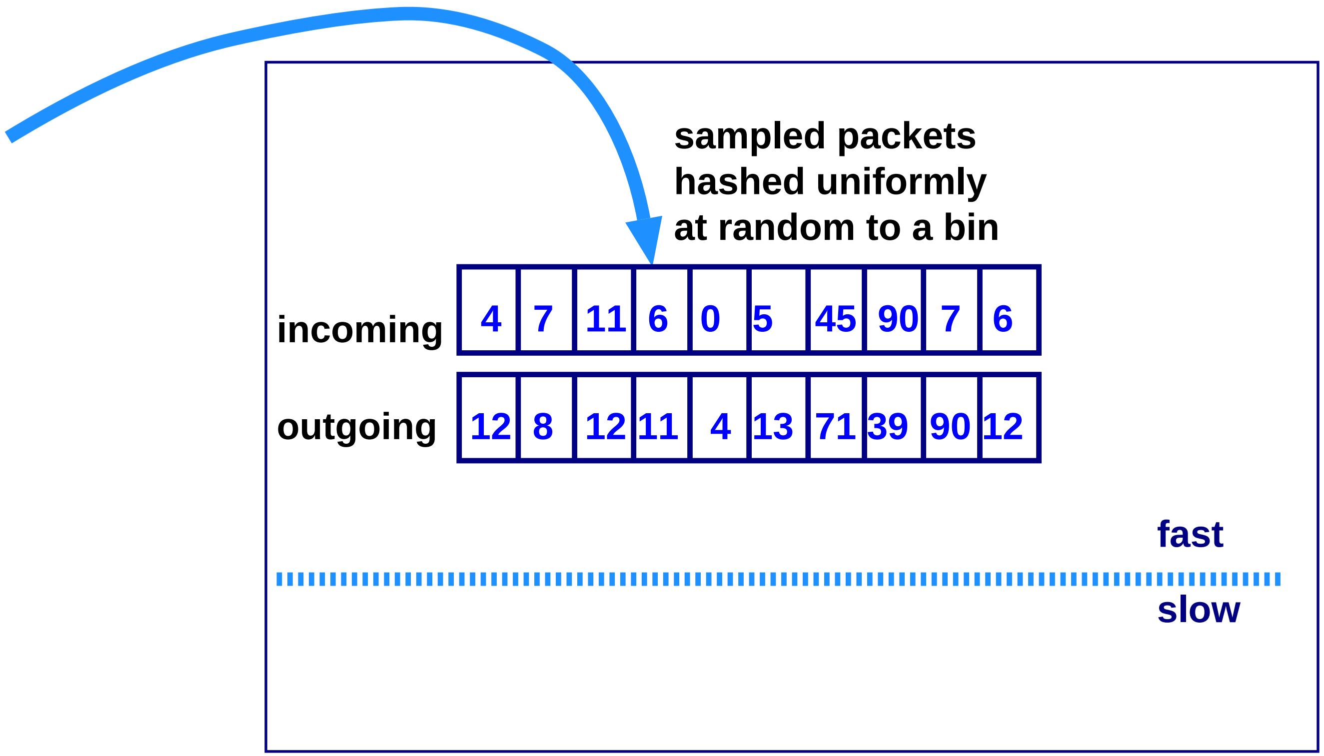
- Monitors are co-located with routers



- Packets are sampled at the router and sent to monitor

Detection Algorithm Schematic

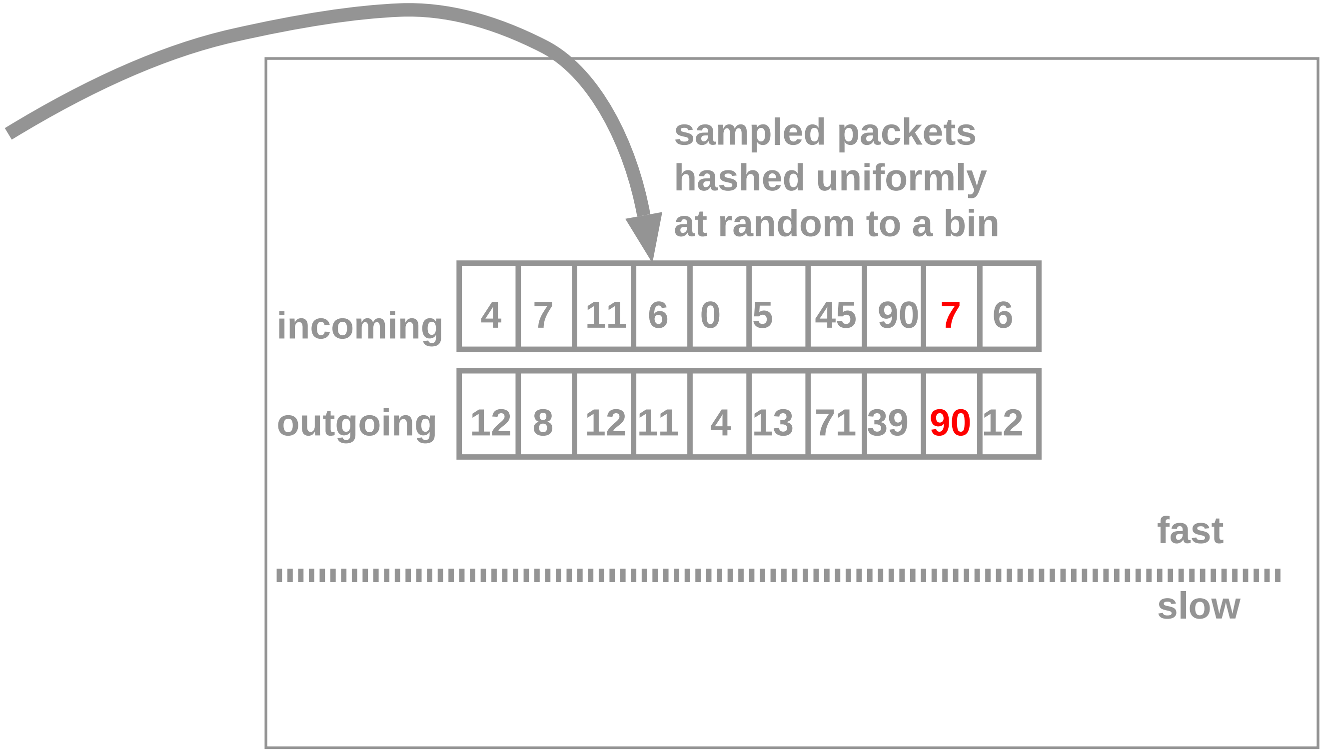
- Sampled packets are binned and counted



- Binning and counting at line speeds (modulo sampling)

DDoS Test — phase 1

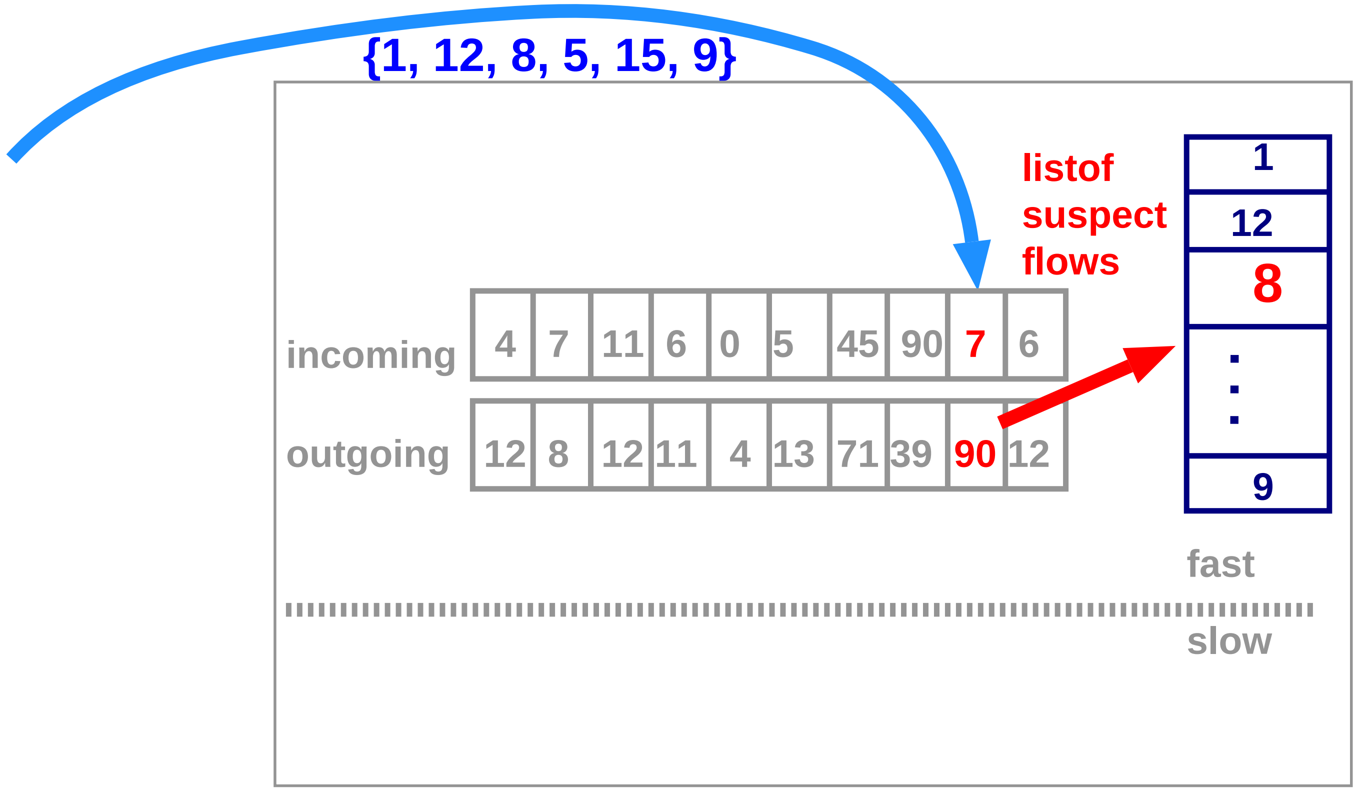
- Simple ratio-based test signals bin overflow



- Counters are periodically zeroed to “reset” memory

DDoS Test — phase 1

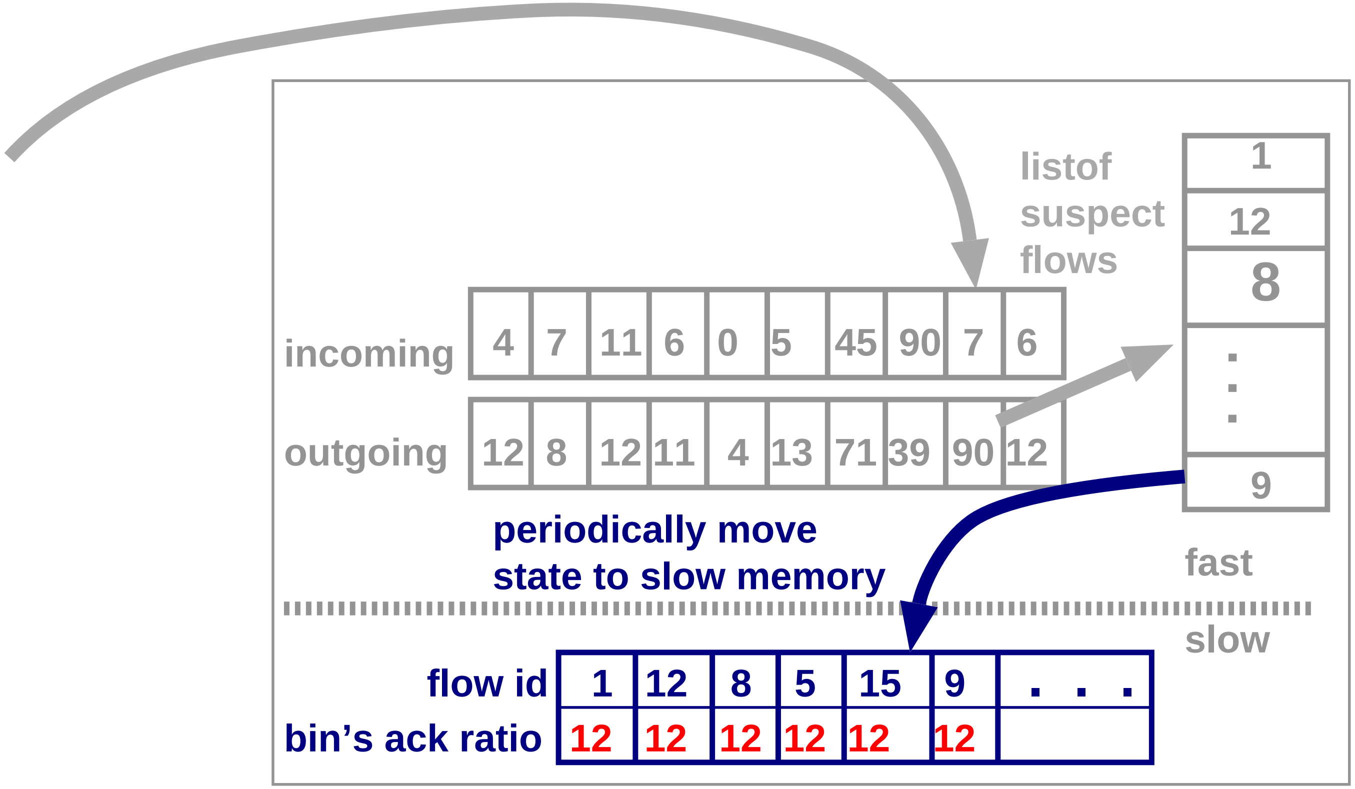
- Flows (destinations) that map to overflowing bins are logged



- The suspect log is temporarily maintained fast memory cache

DDoS Test — phase 2

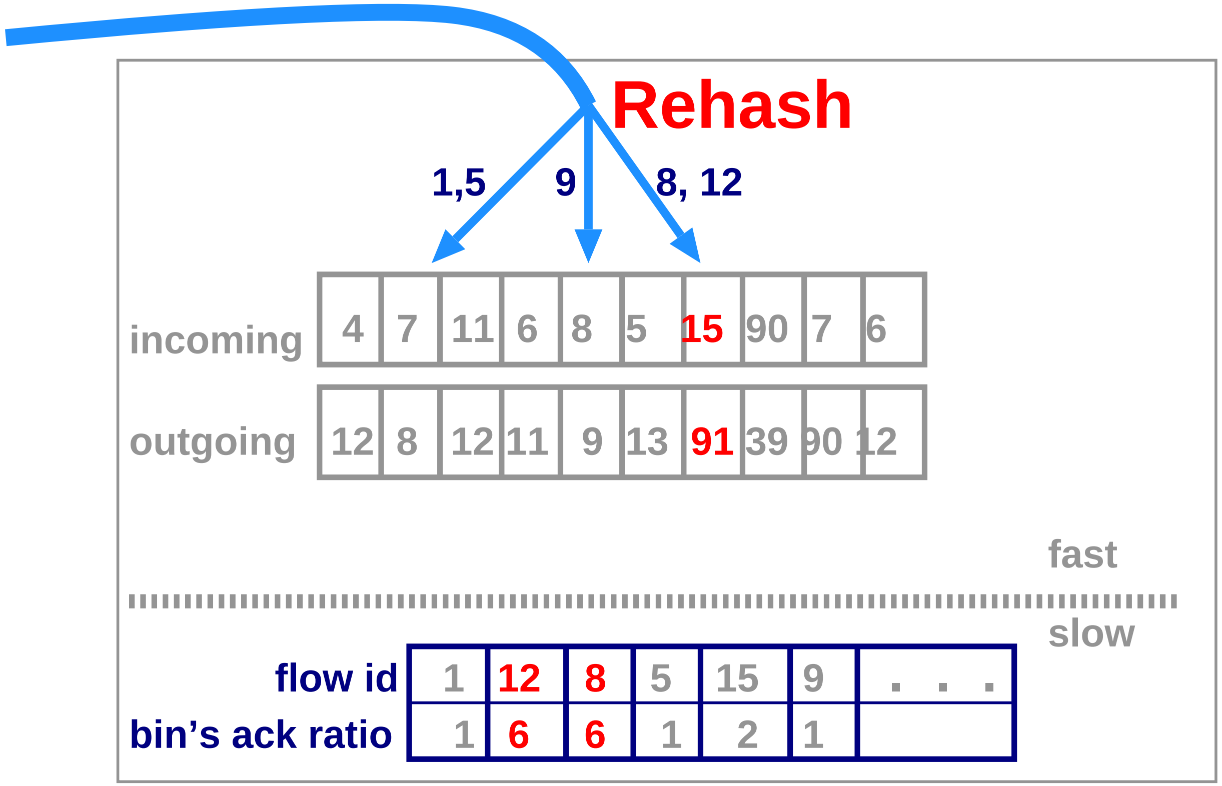
- State is periodically transferred to slow memory



- A flow score is computed for each suspect flow

DDoS Test — phase 2

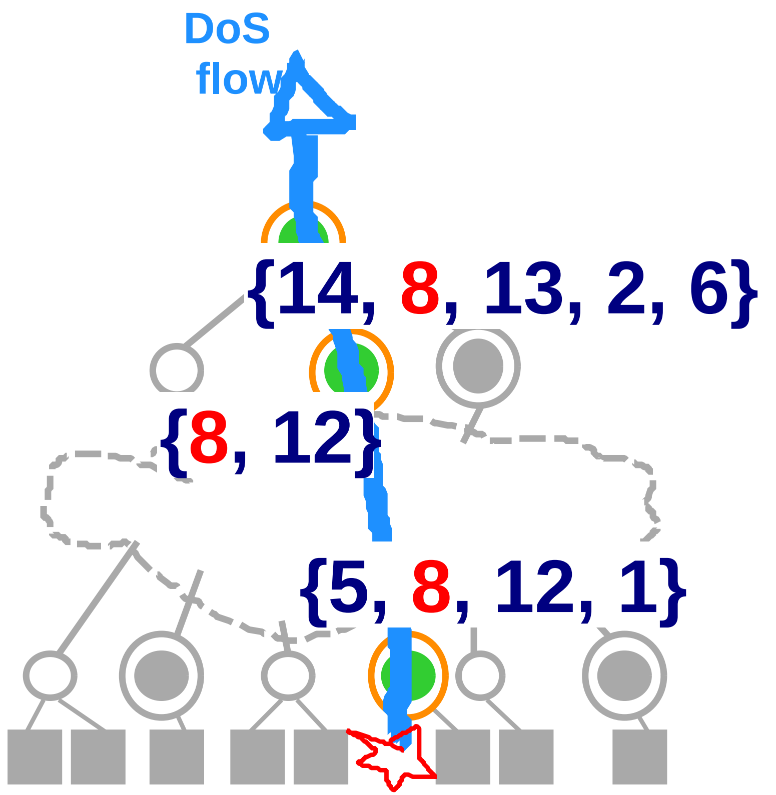
- The suspect flows at each monitor may contain false positives



- The flows are locally rehashed to reduce false positives

DDoS Test — distributed component

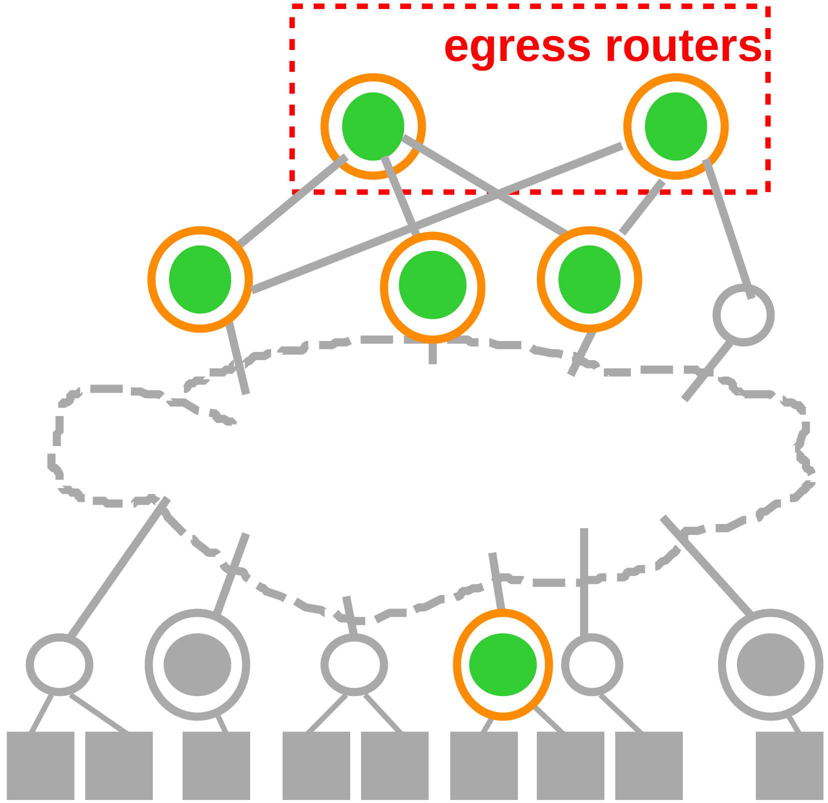
- Each monitor publishes list of suspect flows upstream



- Distributed voting protocol used to nominate attack flows

Multi-homed Domains

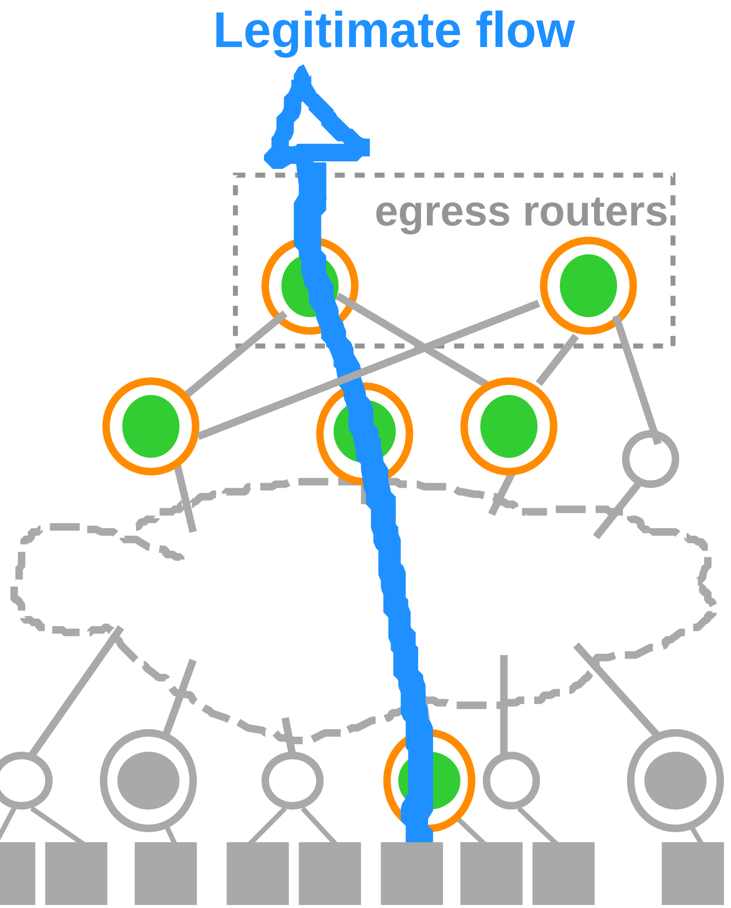
- Many (large) domains are now multi-homed



- No other source-based DDoS systems handle multi-homing

Multi-homed Domains

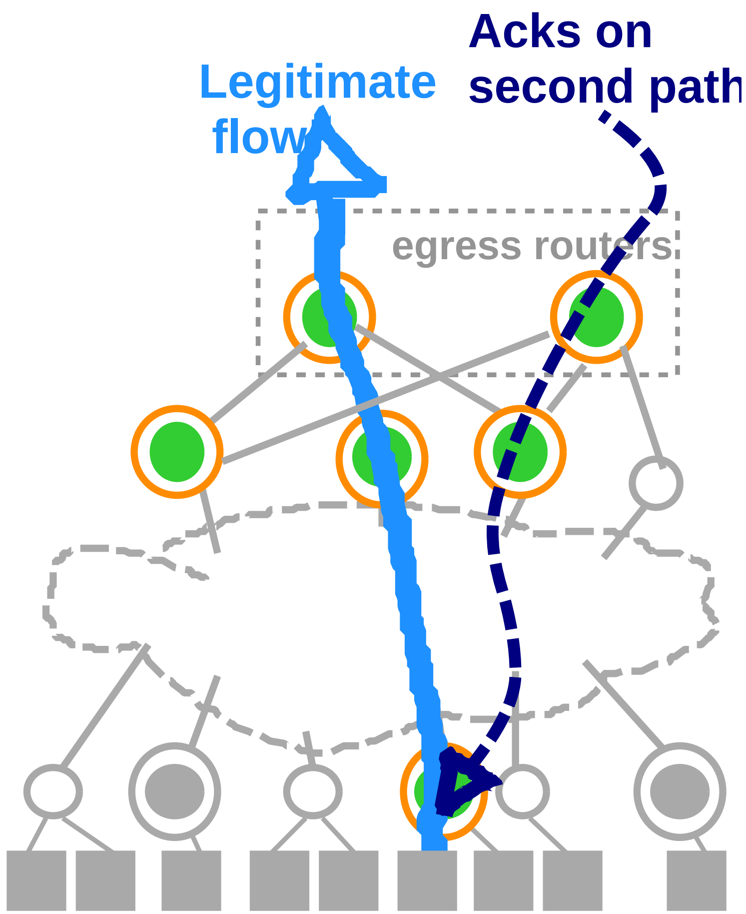
- Unfortunately, much more difficult problem. . .



- . . . and can lead to errors

Multi-homed Domains

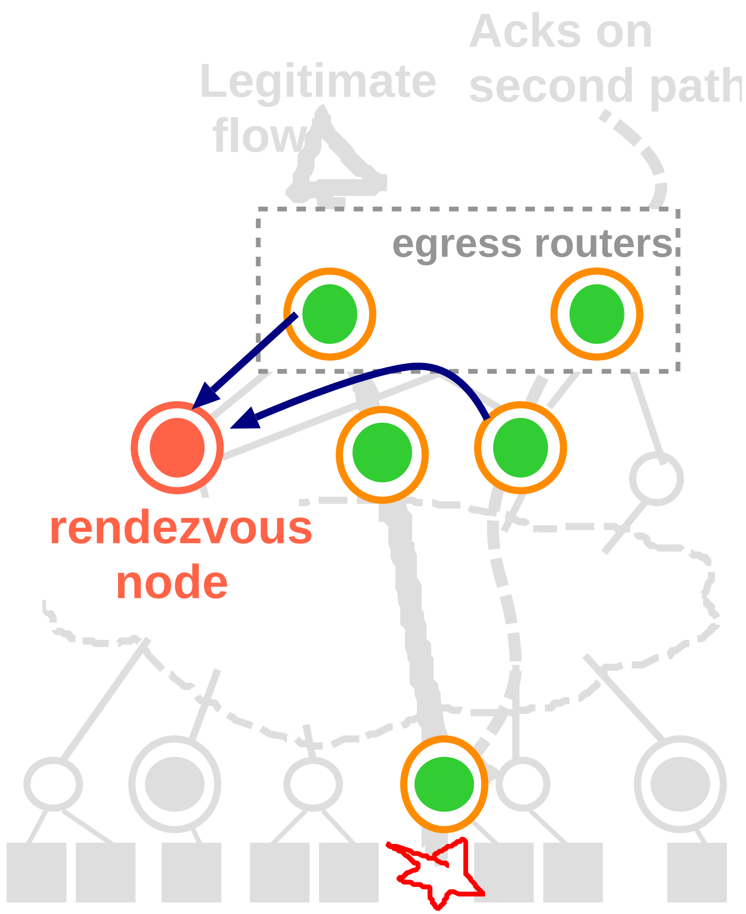
- Data and Acks can traverse disjoint routers



- Leads to more false positives

Multi-homed Domains

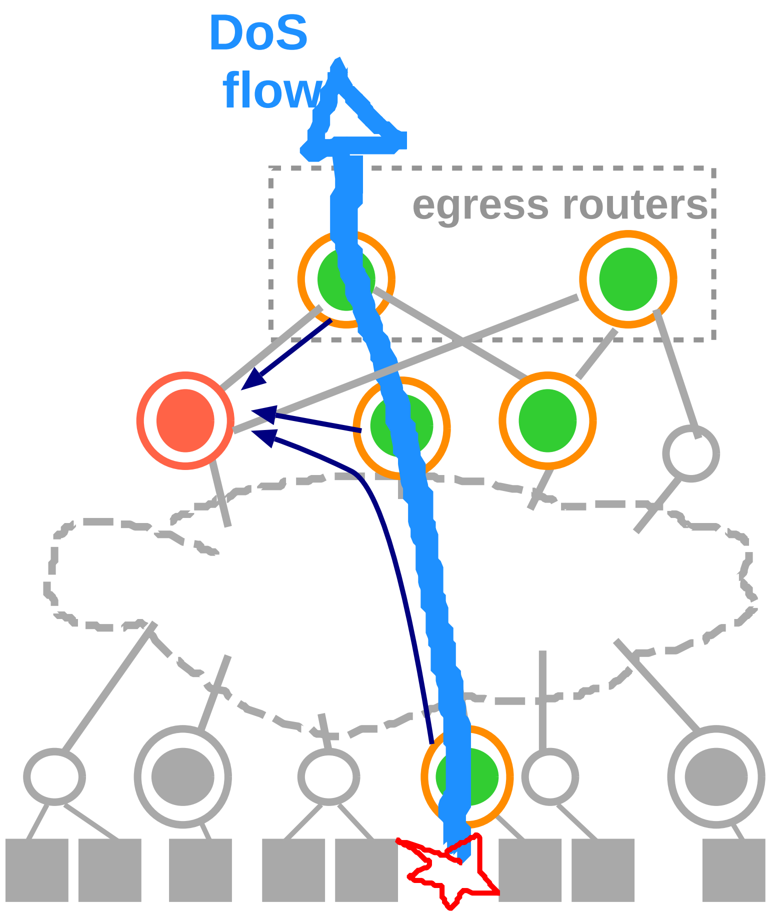
- Data for suspicious flows reconciled at rendezvous nodes



- Tests have account for asymmetry in packet rates

Multi-homed Domains

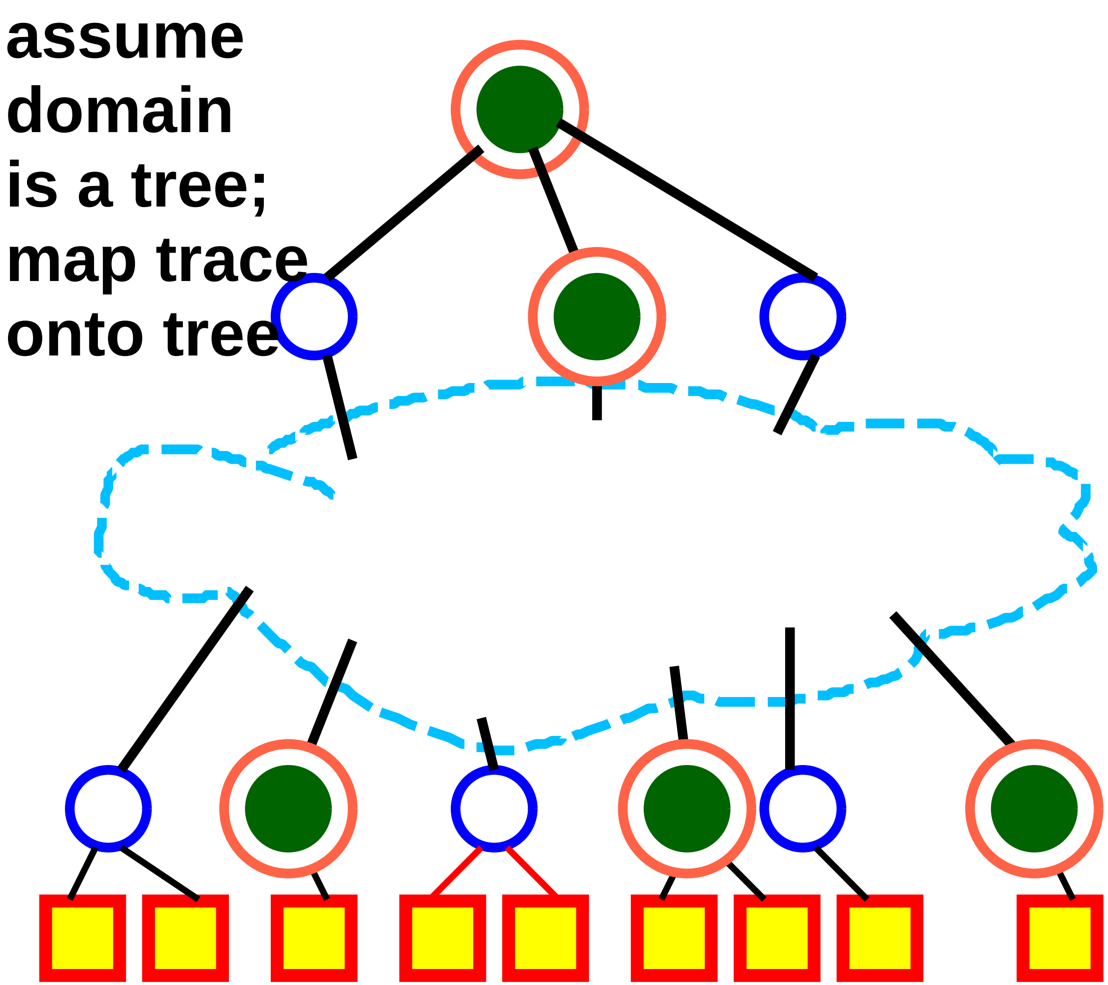
- Rendezvous node gathers data from routers on flow path. . .



- . . . and can classify a flow as an attack

Experiments — Set up

- Different types of attacks with varying number of attackers



- Trace-driven

Details of Traces

	Bell Labs	Abilene
trace duration	25 min	10 min
number of flows	65,000	235,000
pkt rate per sec (in/out)	1194/1586	55,583/45,867
number of addresses (int/ext)	1291/3445	24,257/23,647
avg # active flows per sec	200	3500

Detection Accuracy vs. Number of bins

Normalized # of bins	*Avg. # of false positives*	*Detection Rate (%)*	*Detection Time Time (seconds)*
0.05	0.00	89	97.95
0.10	0.00	100	27.25
0.20	0.00	100	15.28
0.40	0.00	100	12.47
0.60	0.12	100	11.00

- Bell Labs trace, single attacker, 20 pps attack rate
- 0.20 NB $\Rightarrow$ 40 bins

Accuracy vs. Sampling Rate

Sampling Rate (%)	*Avg # of false positives*	*Detection Rate (%)*	*Detection Time Time (seconds)*
2.5	0.00	72	98.21
5	0.07	99	52.00
10	0.00	100	15.28
20	0.00	100	12.04
40	0.00	100	9.95
60	0.00	100	10.15

- Bell Labs trace, single attacker, 20 pps attack rate
- 10% sampling rate $\Rightarrow$ 110 pps

More complicated attacks

- Test different scenarios on Abilene Trace

  100K pps at root

  3500 active flows on average

  Average flow: 34 pps

- Deployment Scope [15 monitors] $\Rightarrow$ top 4 levels of domain
- Normalized number of bins [0.2] $\Rightarrow$ 700 bins/monitor
- Sampling rate [0.1] $\Rightarrow$ 10K pps at each monitor

Attack Rate vs. Detection Accuracy

Attack Rate (pps)	*Avg # of False Pos.*	*Detection Rate (%)*	*Detection Time (sec)*	*Overhead (Bps)*
10	0.25	99	106.25	77.50
20	0.12	100	27.88	43.75
50	0.25	100	13.35	39.85
100	0.25	100	10.14	44.52

- Eight simultaneous attacks; average regular flow rate: 34 pps

  Attacks start every 15 seconds; last for 8 minutes

Multiple Attackers

Aggregate Attack Rate	*# of Attackers*	*Avg # of False Pos.*	*Detect. Rate (%)*	*Detect. Time (sec)*	*Overhead (Bps)*
20	1	0.12	100	27.89	43.75
100	5	0.25	100	12.38	45.38
200	10	0.25	100	10.21	73.84

- Average flow rate: 34 pps

Multiple Attackers

Aggregate Attack Rate	*# of Attackers*	*Avg # of False Pos.*	*Detect. Rate (%)*	*Detect. Time (sec)*	*Overhead (Bps)*
100	1	0.25	100	10.14	44.52
100	5	0.25	100	12.38	45.38
100	10	0.12	99	14.71	72.02

- Average flow rate: 34 pps

Pulse Attacks

	Det. Rate (%)			*Det. Time (sec)*			*Overhead (Bps)*		
Rate (pps)	1/1	1/3	1/5	1/1	1/3	1/5	1/1	1/3	1/5
20	94	5	2	130.04	91.88	58.00	90.66	118.23	74.90
40	100	99	47	31.38	145.69	240.25	43.39	85.46	103.74
60	100	100	97	19.32	53.07	119.43	38.25	51.90	68.20
80	100	100	100	15.93	33.75	67.88	40.16	47.98	51.20
100	100	100	100	13.82	29.03	47.55	38.27	41.42	47.04

- 1/x $\Rightarrow$ pulse with 1 second on time, x seconds off time

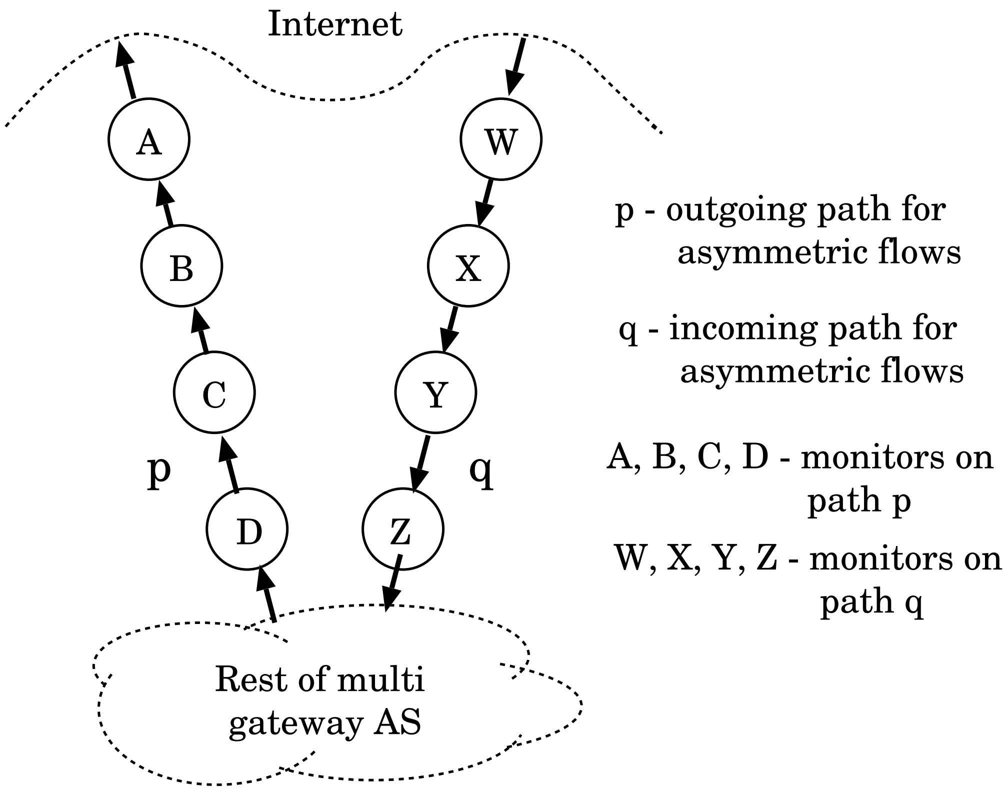
Multi-homed domain experiments

- A^p_{out} $\equiv$ frac. of all outgoing addresses that use path p
- A^p_{in} $\equiv$ frac. of all incoming addresses that use path p

– Example: $A^p_{out} = 50\%$ and $A^p_{in} = 20\%$

$\Rightarrow$ 30% of the flows are asymmetric and use p as the outgoing path (and q as incoming)

$\Rightarrow$ 20% of the symmetric flows in the domain use path p for both incoming and outgoing packets



Multi-homed Domains: Accuracy vs. Flow Asymmetry

A^p_{out}	A^p_{in}	# False Pos.	Detect. Time (sec)	Overhead (Bps)
	0%	1.12	53.60	7434.7
	20%	0.00	37.19	7829.8
10%	40%	0.00	29.61	10797.6
	60%	0.00	27.34	13575.2
	80%	0.00	28.36	16263.6
	100%	0.12	33.05	18536.0
	0%	1.38	56.57	12671.2
	20%	0.25	35.32	10586.1
	40%	0.00	27.81	8256.7
50%	60%	0.00	26.48	8301.4
	80%	0.25	28.98	10687.8
	100%	0.38	43.83	12676.1

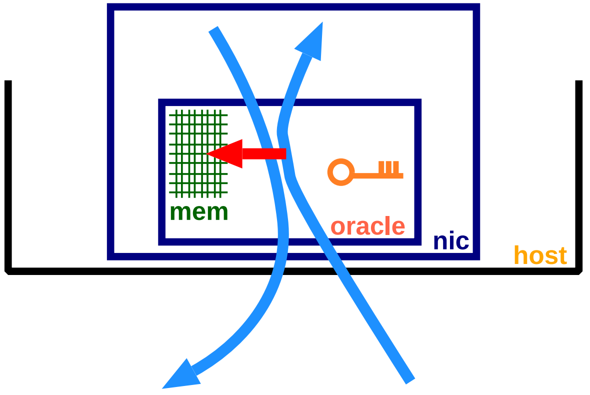
Local Oracle (Hardware)

- Pass-through processor on NIC with a physically secure key $\mathcal{K}$

  Cannot be controlled via host software

- Passive monitor of all network traffic

  Logs all headers+packet snippet



- Can also be deployed per subnet

Local Oracle (Hardware)

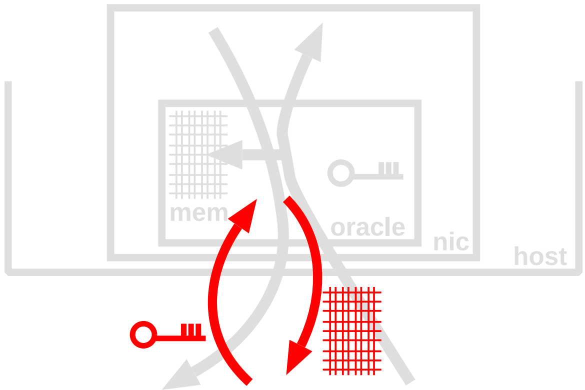
- Pass-through processor on NIC with a physically secure key $\mathcal{K}$

  Cannot be controlled via host software

- Passive monitor of all network traffic

  Log requires 10 MB storage/minute (avg. for 100Mb link)

  worst case 1 order of magnitude worse.



- Log dumped to sender when packet with $\mathcal{K}$ intercepted

Local Oracle (Hardware)

- Pass-through processor on NIC with a physically secure key $\mathcal{K}$
  Cannot be controlled via host software
- Passive monitor of all network traffic

Attackers (can) know of the oracle, but cannot modify its operation

What can such a complete detection system do . . . ?

- Detect different attacks — DDoS, malicious packets, worms, intrusion detection, . . .

  More capable than single node systems

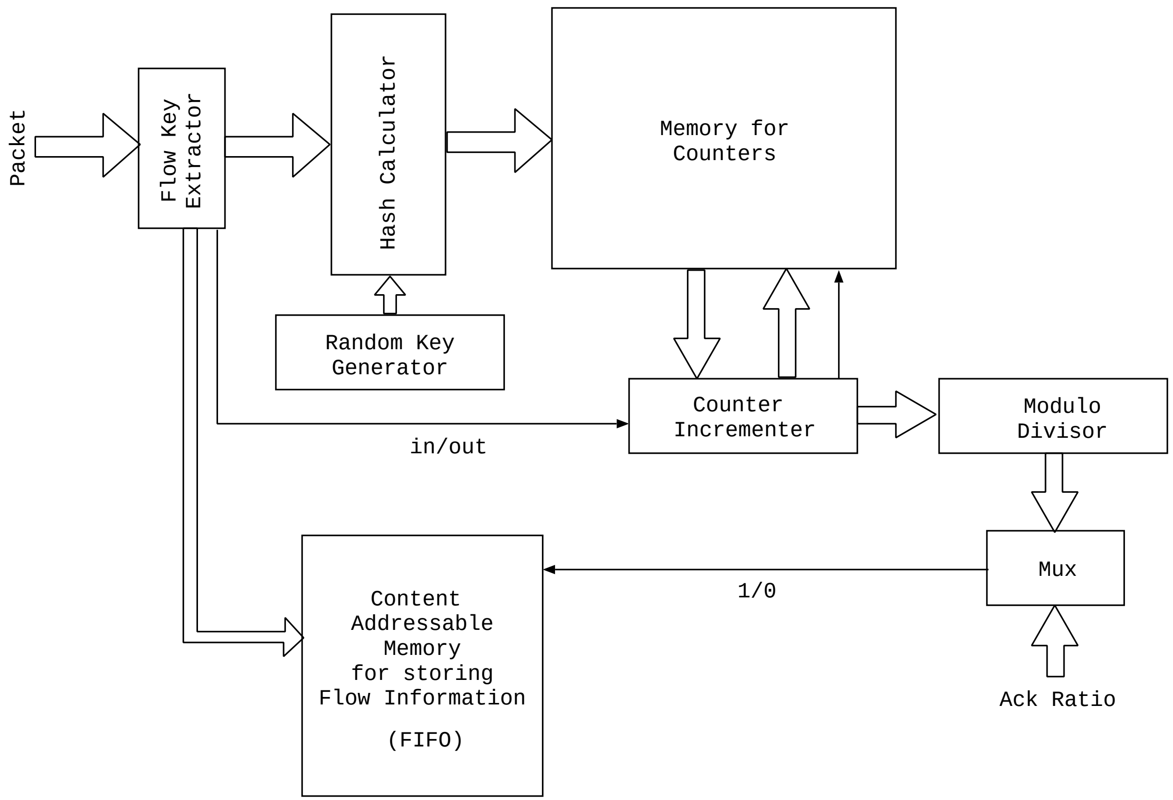
  Incrementally deployable

- Complete single packet traceback (using local oracle)

  Post-mortem of attacks

Implementation

- Detailed packet level simulations complete
- Partial in-kernel Linux implementation
- FPGA based hardware implementation



Current hardware would process 2.4 Gbps links at line rates

20% sampling would allow implementation on 10Gbps links

Future work

- Extend tests to include more attack types

    UDP, ICMP traffic

- History-based attack detection

    Current system is entirely stateless

- Better compression algorithms for logger

- Distributed PKI work with Mike Marsh