Automated Worm Fingerprinting

Paper By: Sumeet Singh, Cristian Estan, George Varghese and Stefan Savage
Department of Computer Science and Engineering
University of California, San Diego

Published: Operating Systems Design and Implementation ’04 (OSDI’04)

Presented By: Dan DeBlasio for CAP 6133 Spring 2008

Overview

- Developed a system called **Earlybird** at UCSD
- Implemented for 8 months.
- Able to detect, and create signatures for major outbreaks during this period
  - Blaster
  - MyDoom
  - Kibuv.B

Motivation

- Need to be able to identify a worm quickly and with regularity with some low tolerance for false positives.
- Need to be able to quickly extract a signature to effectively combat the spread of the worm.
    - Slow Moving: (Code Red): 60 Min
    - Fast Moving: (Slammer): 5 Min - 60 Sec
- Need to be able to contain the worm once it is identified.

Background/Observations

- Code Invariance
  - Some part of the worm code will be static across all copies.
- Content Prevalence
  - Due to worm dynamics, many copies of the worm will be floating around on the network.
- Address Dispersion
  - As the worm infects more host, there will be more host/destination combinations for the same data.

Content Sifting

- Idealized would track the exact matches for every packet.
- Keep track of all source and destinations.
- Analyzes packets above certain thresholds to identify them as worms.

```
ProcessTraffic(payload,srcIP,dstIP)
 prevalence[payload]++
 Insert(srcIP,dispersion[payload].sources)
 Insert(dstIP,dispersion[payload].dests)
 if (prevalence[payload]> T1
   and size(dispersion[payload].sources)> T2
   and size(dispersion[payload].dests)> T3
   if (payload in knownSignatures)
     return
   endif
   Insert(payload,knownSignatures)
   NewSignatureAlarm(payload)
 endif
```

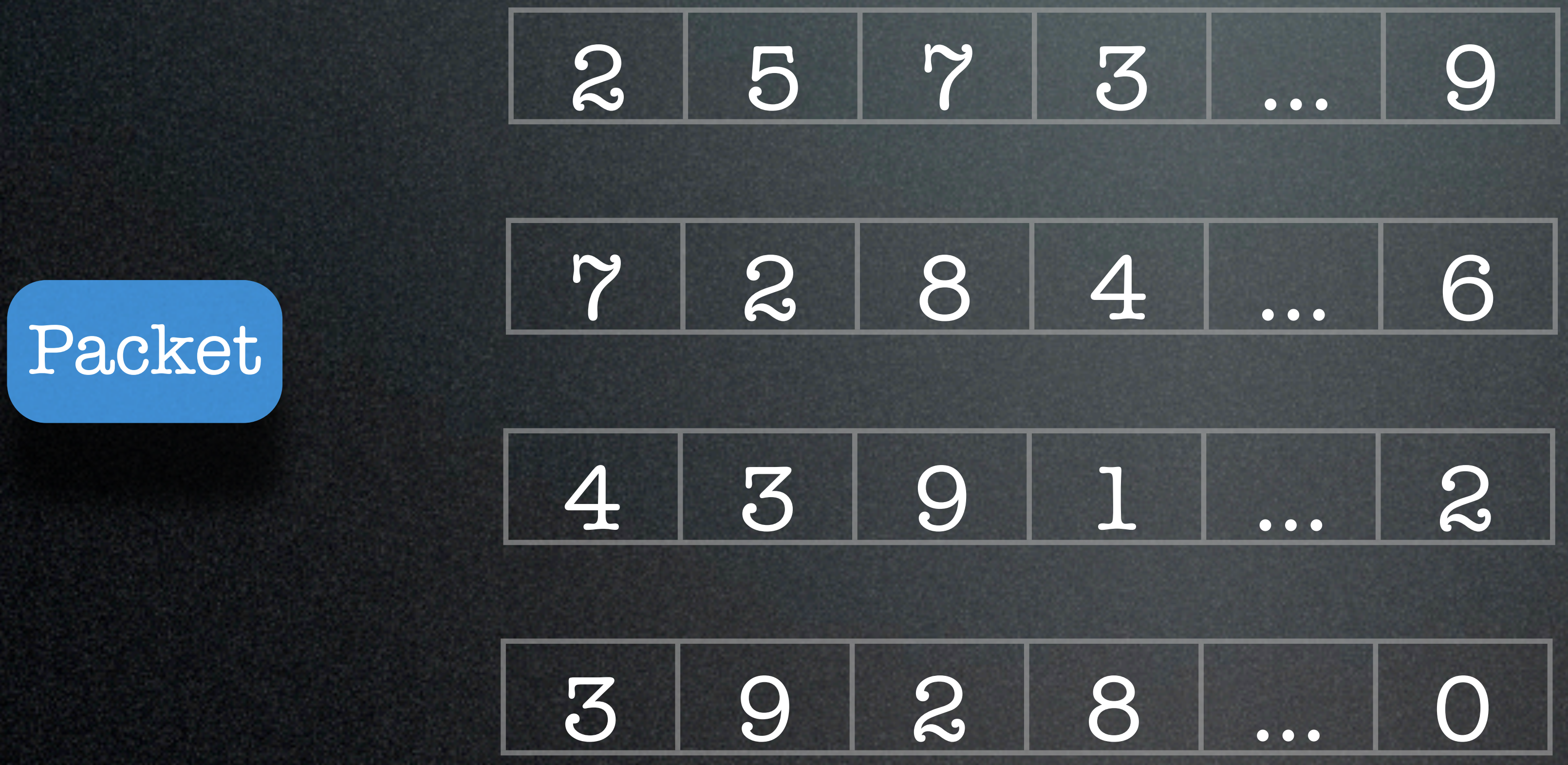
Content Sifting

- Memory and processing requirements would be too high.
- Hashing provides a solution but too many collisions.
- Multi-stage filters provide the answer.
  - Each packet is hashed multiple times.
  - A counter is kept at each hashing stage.
  - Kept if hash count for all is above a threshold.

Multi-Stage Filtering

2	5	7	3	...	9
7	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

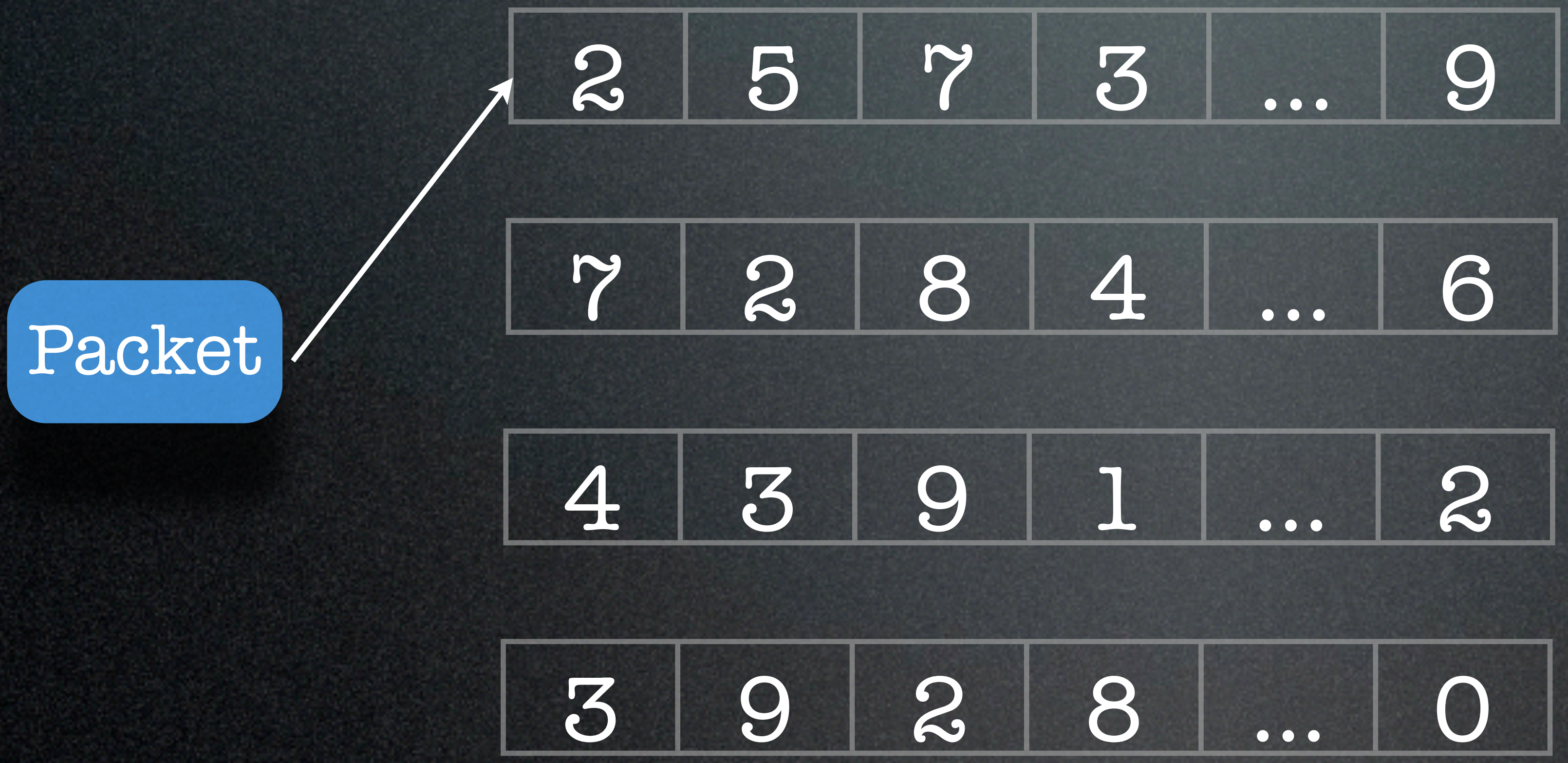
Multi-Stage Filtering



Packet

2	5	7	3	...	9
7	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

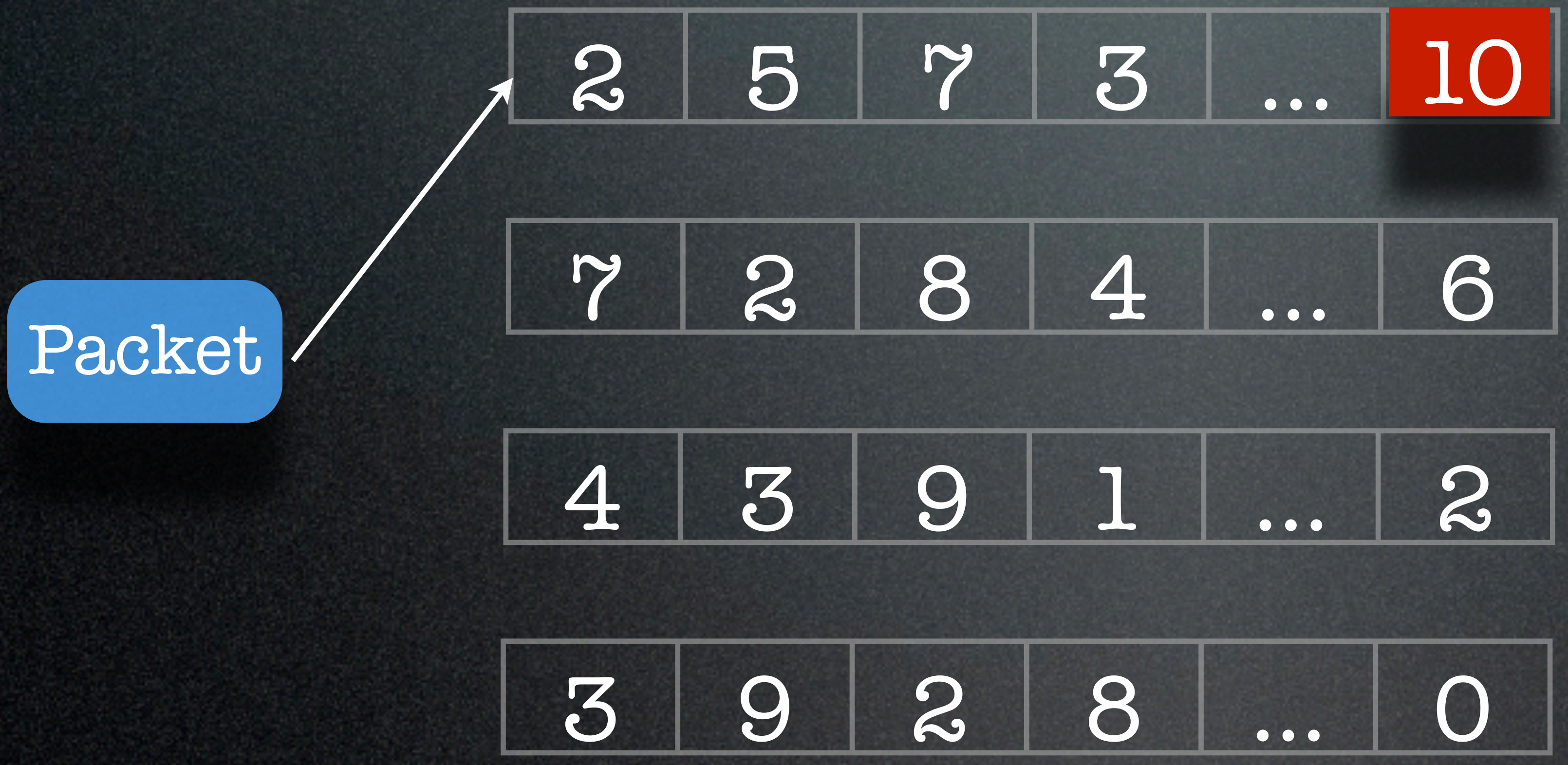
Multi-Stage Filtering



Packet

2	5	7	3	...	9
7	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

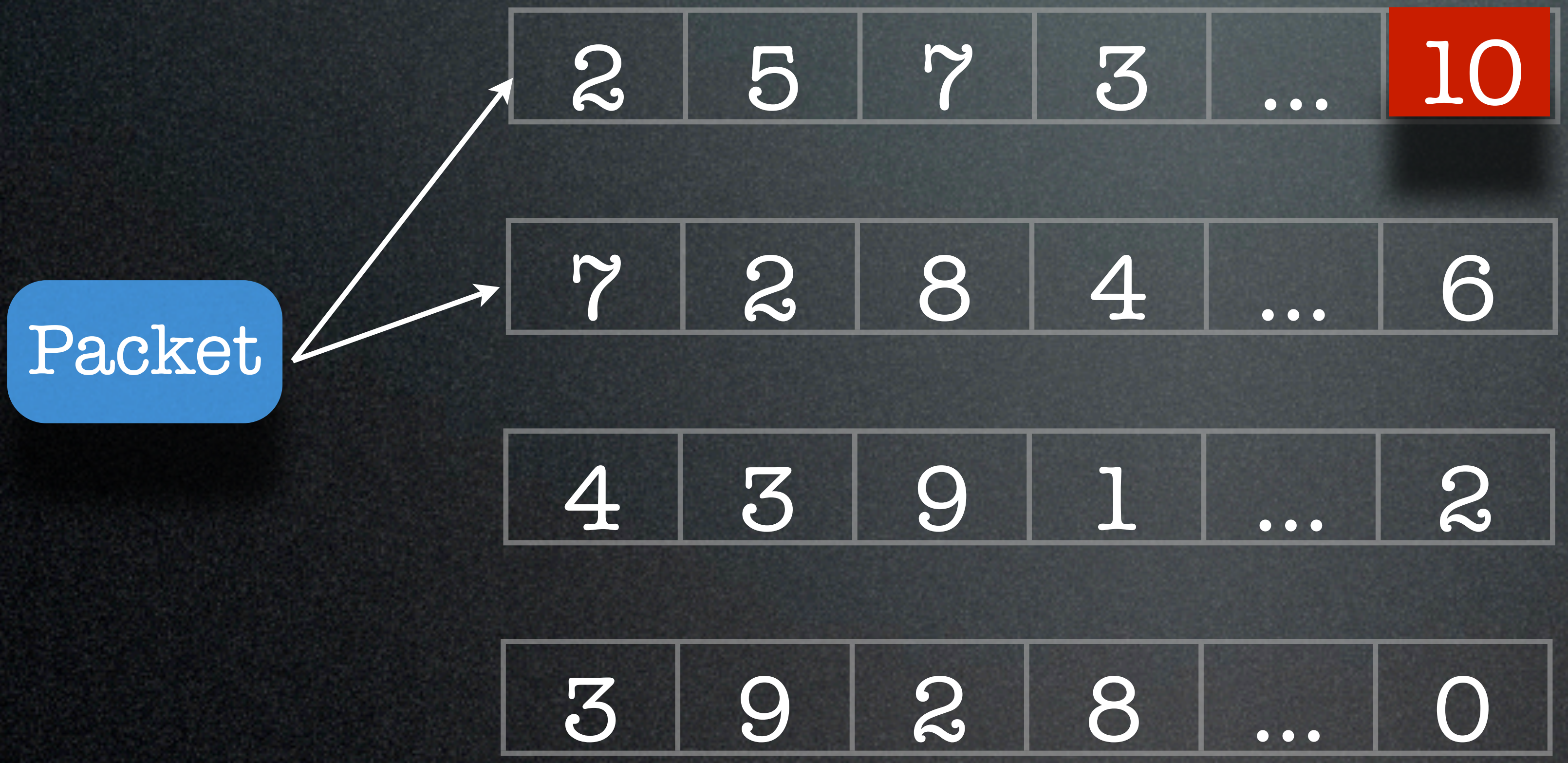
Multi-Stage Filtering



Packet

2	5	7	3	...	10
7	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

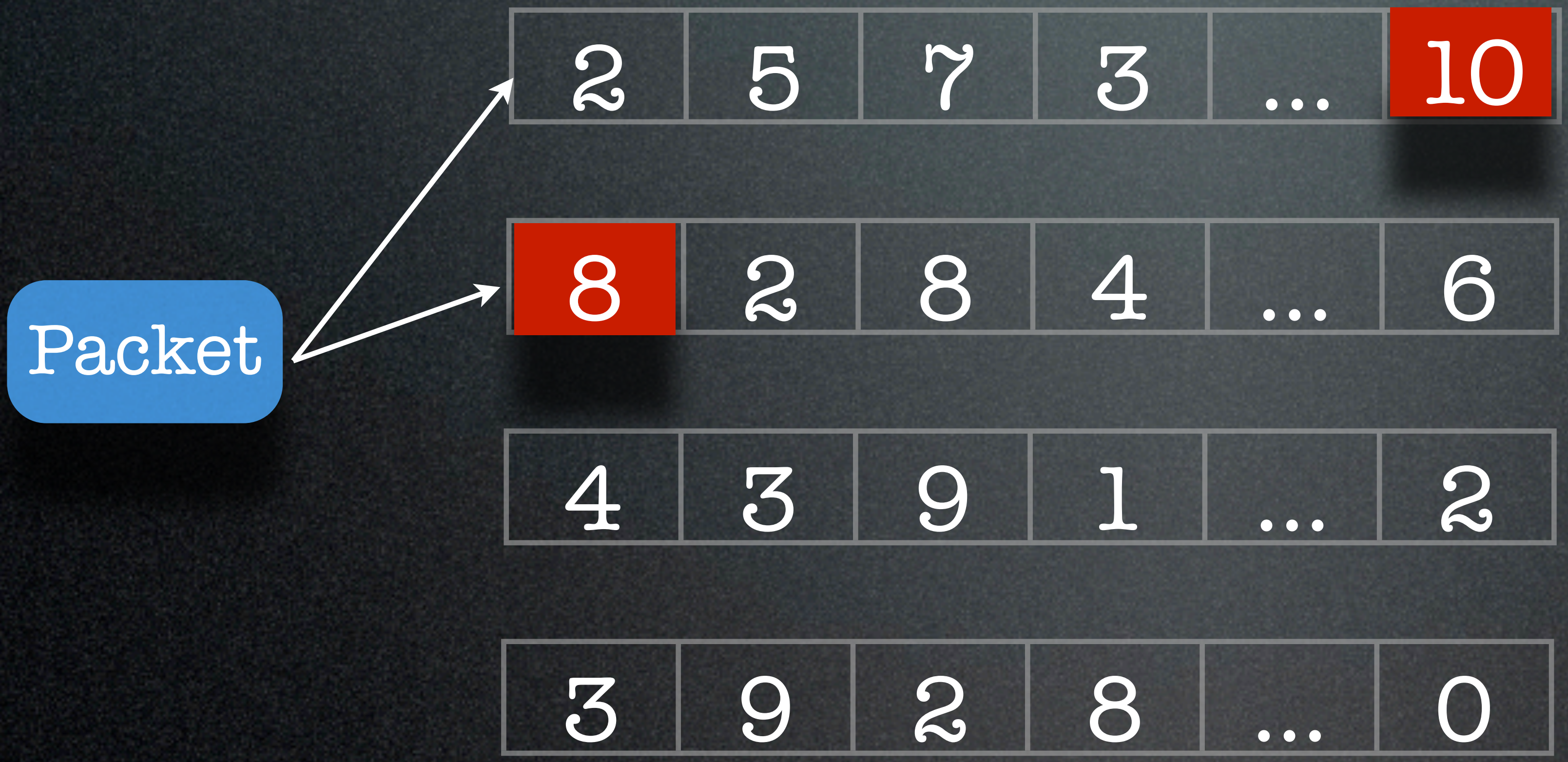
Multi-Stage Filtering



Packet

2	5	7	3	...	10
7	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

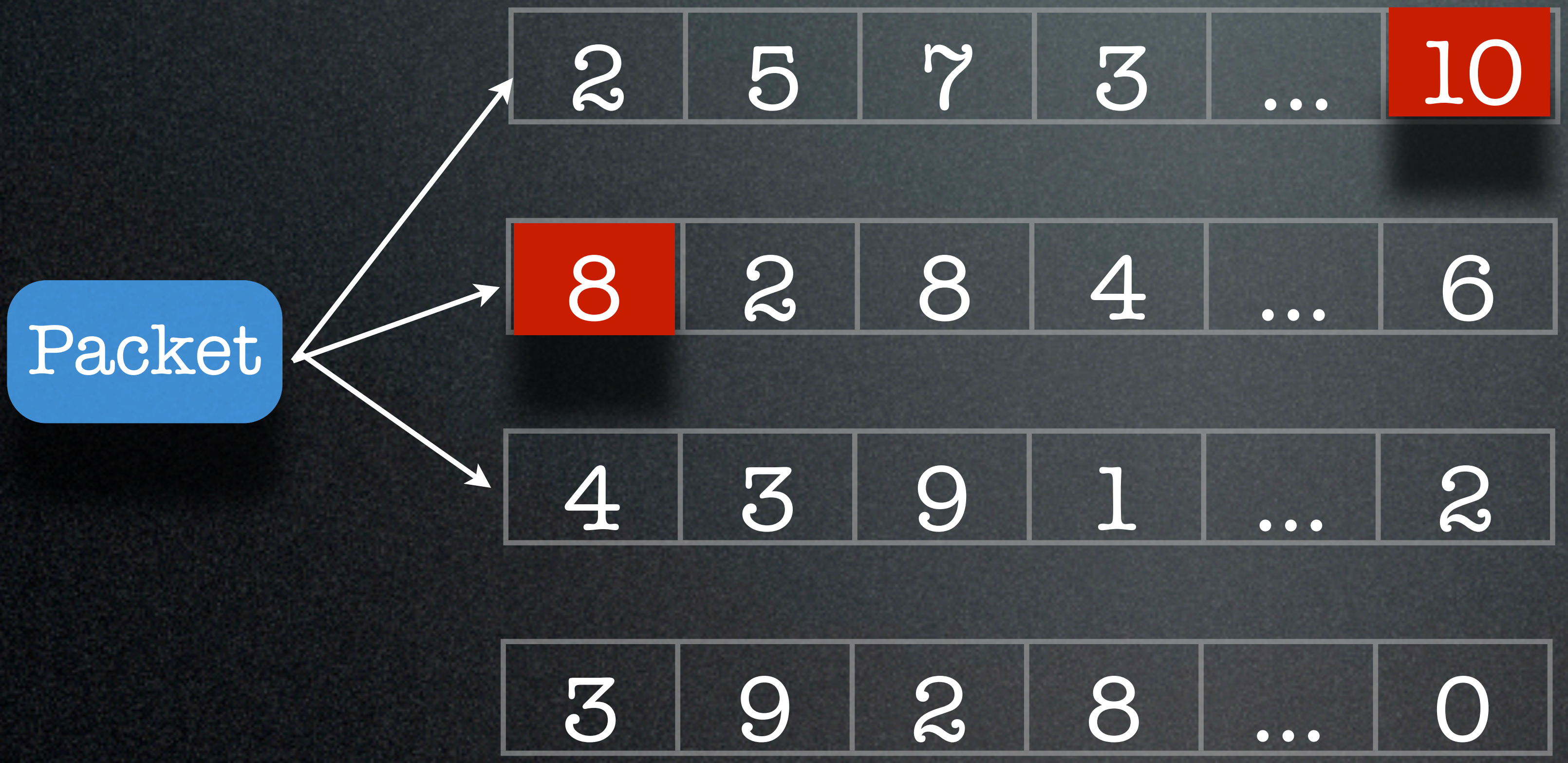
Multi-Stage Filtering



Packet

2	5	7	3	...	10
8	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

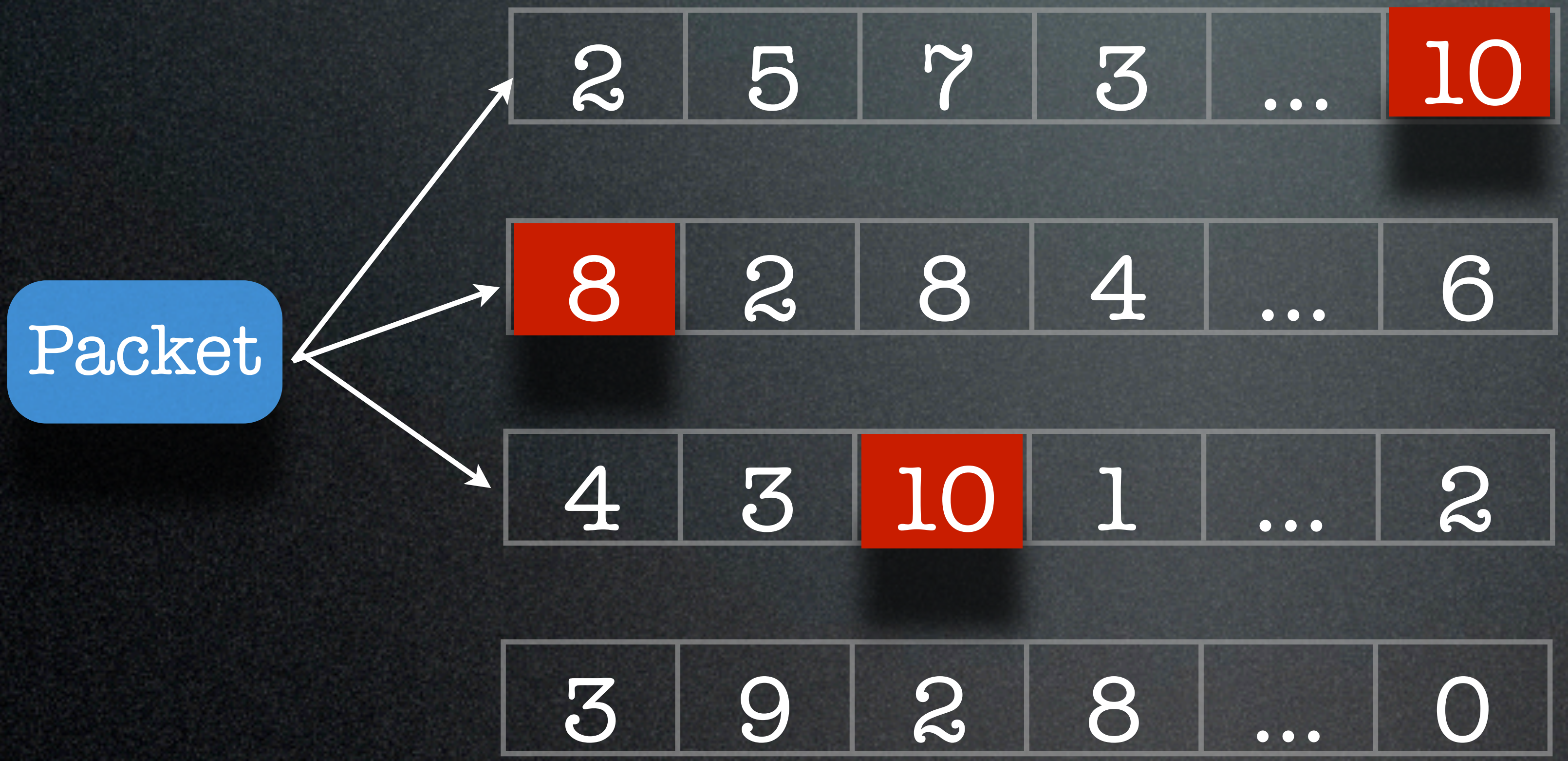
Multi-Stage Filtering



Packet

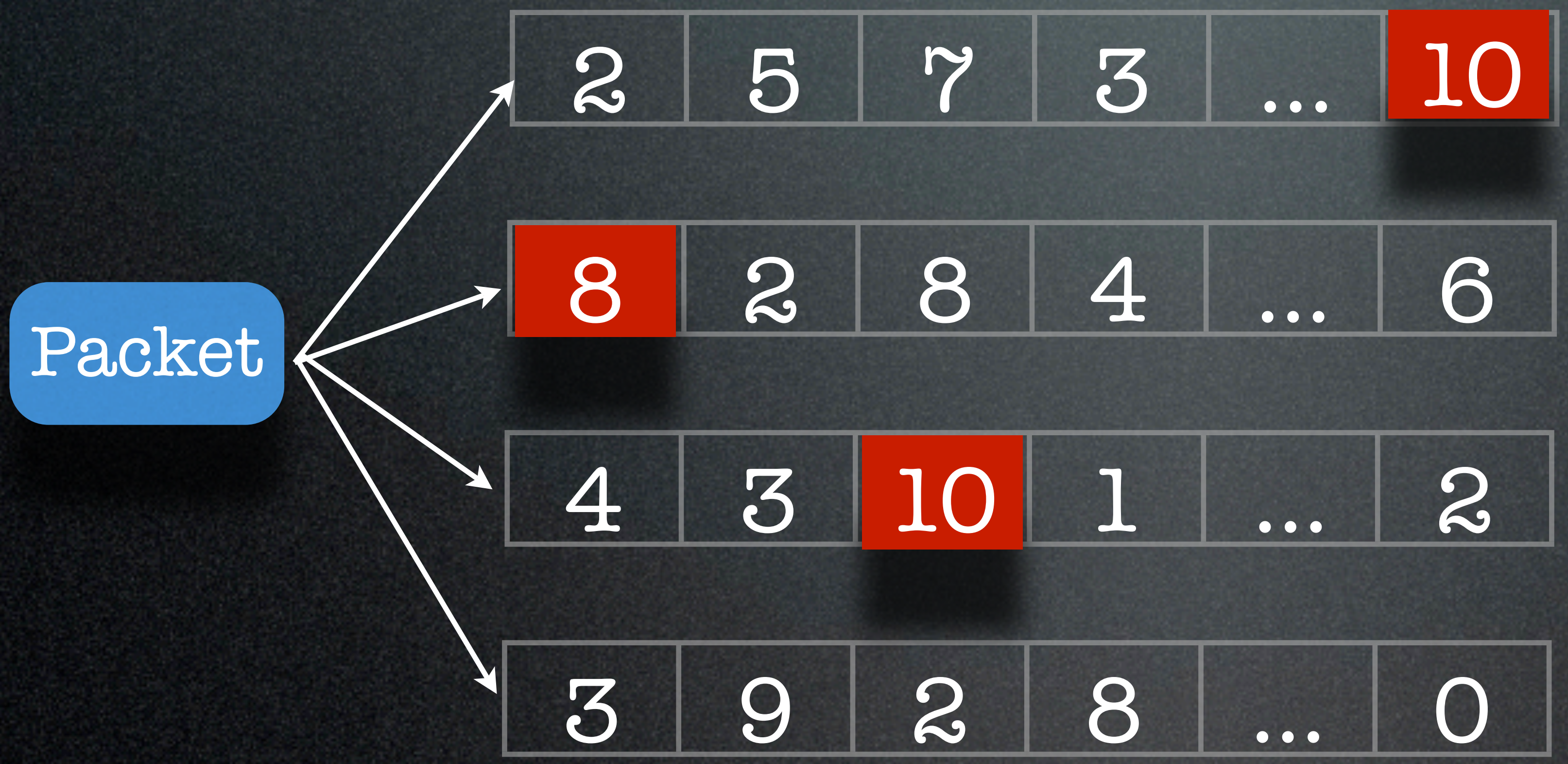
2	5	7	3	...	10
8	2	8	4	...	6
4	3	9	1	...	2
3	9	2	8	...	0

Multi-Stage Filtering



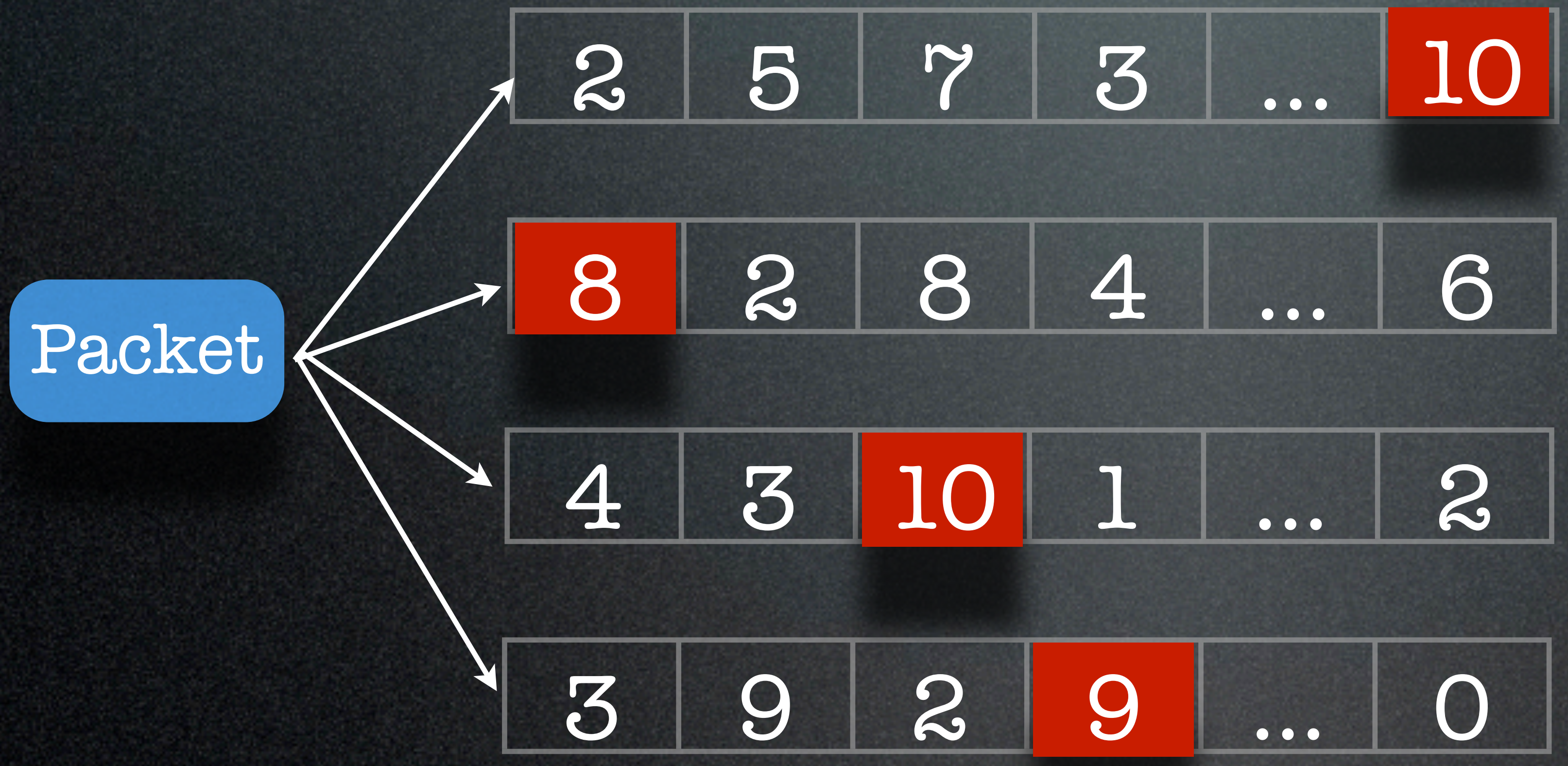
2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	8	...	0

Multi-Stage Filtering



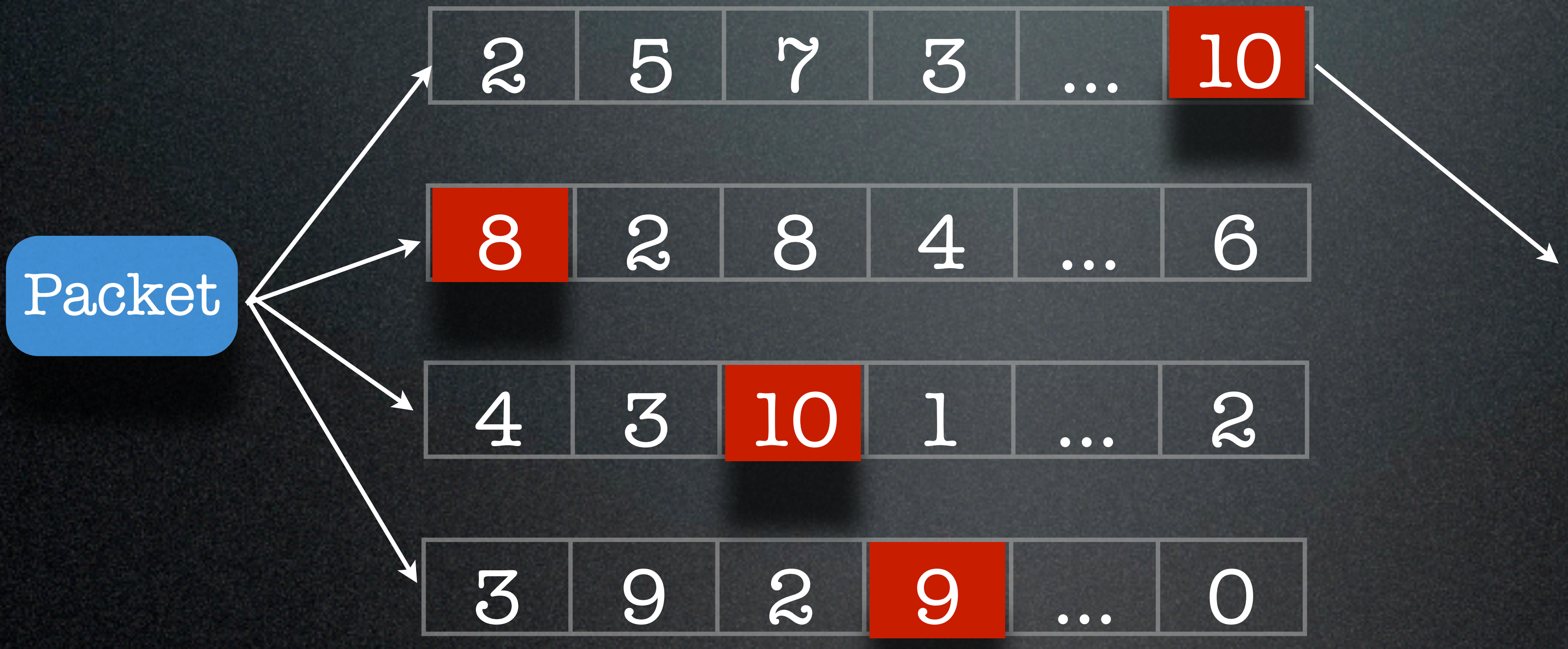
2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	8	...	0

Multi-Stage Filtering



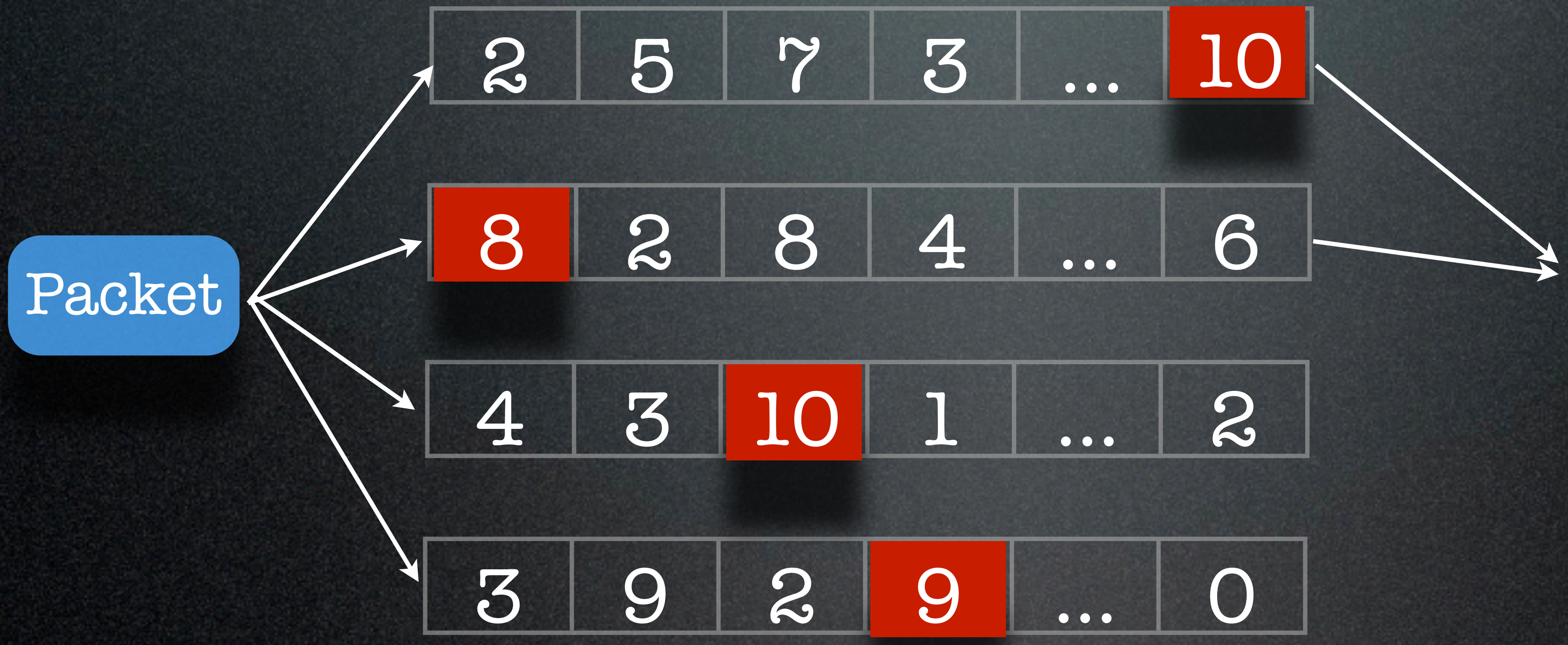
2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

Multi-Stage Filtering



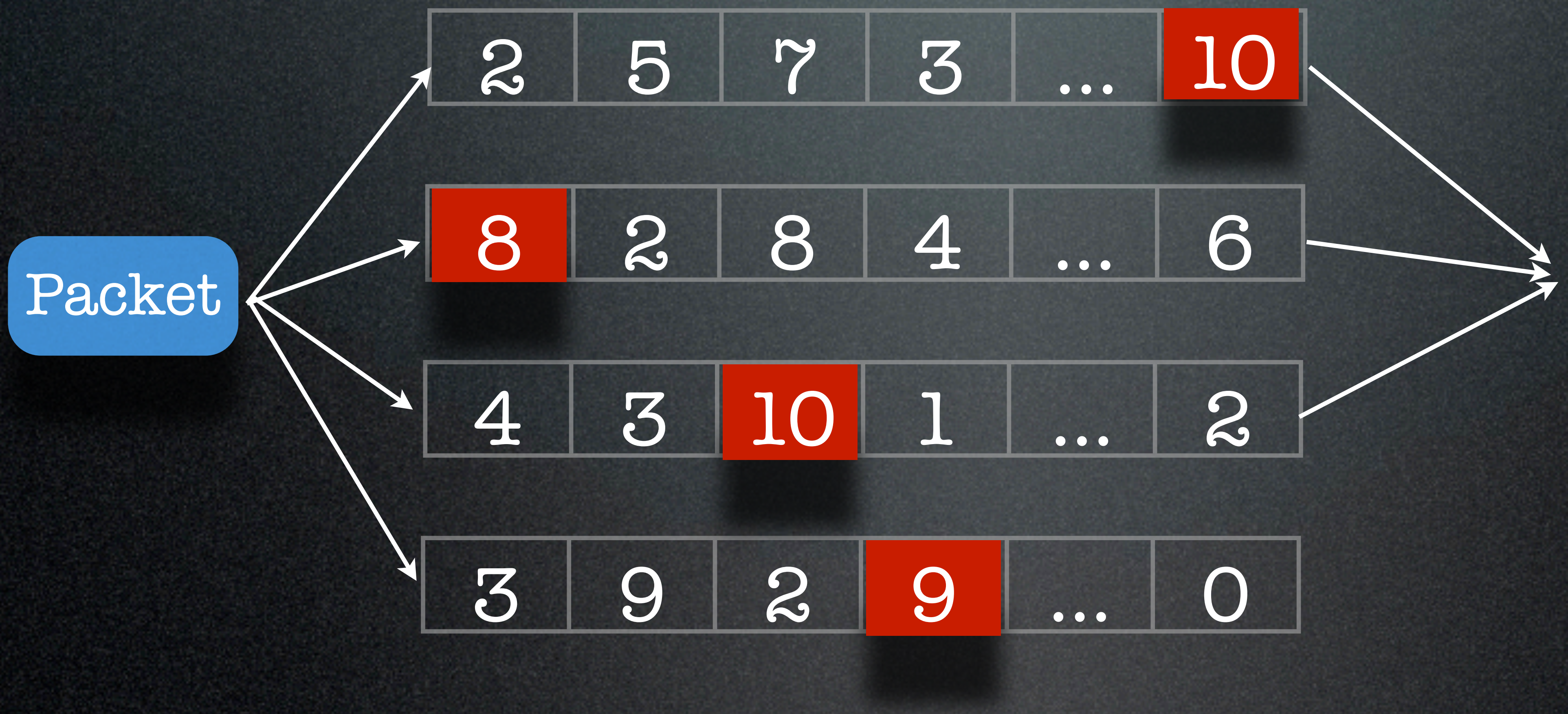
2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

Multi-Stage Filtering



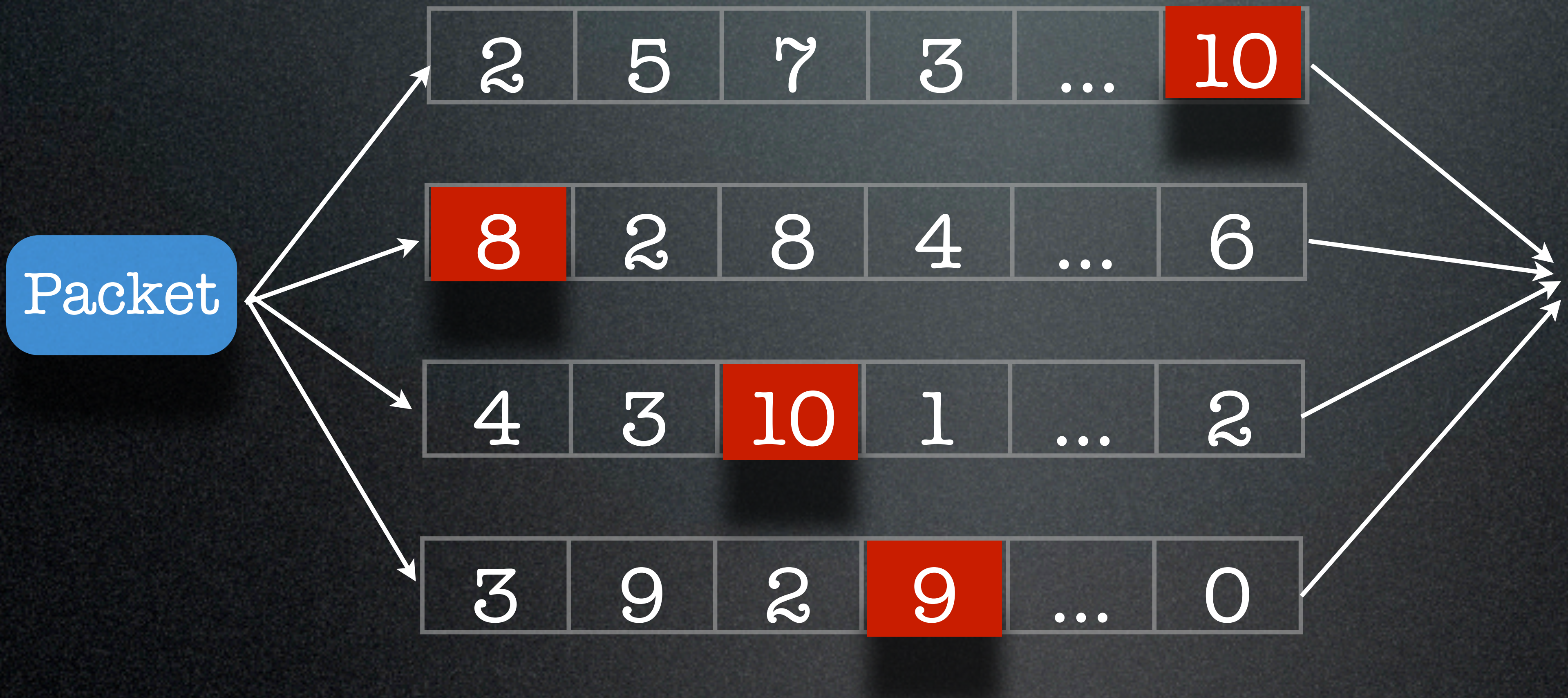
2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

Multi-Stage Filtering



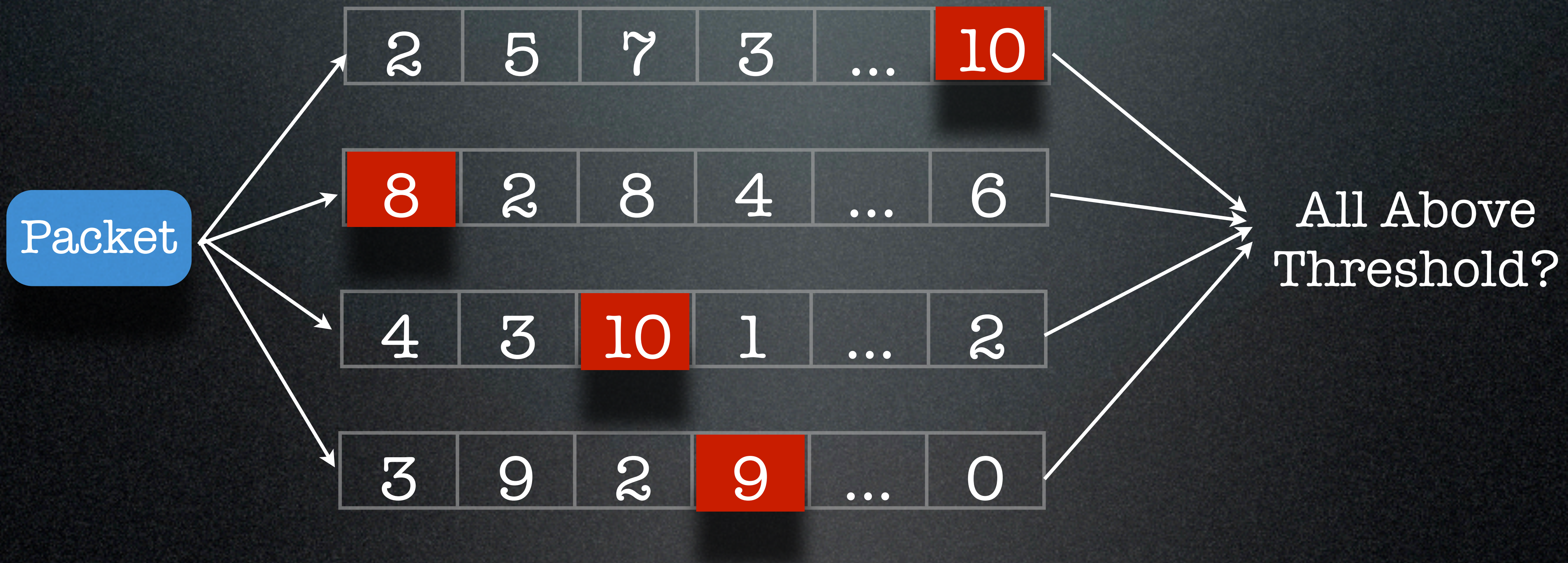
2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

Multi-Stage Filtering



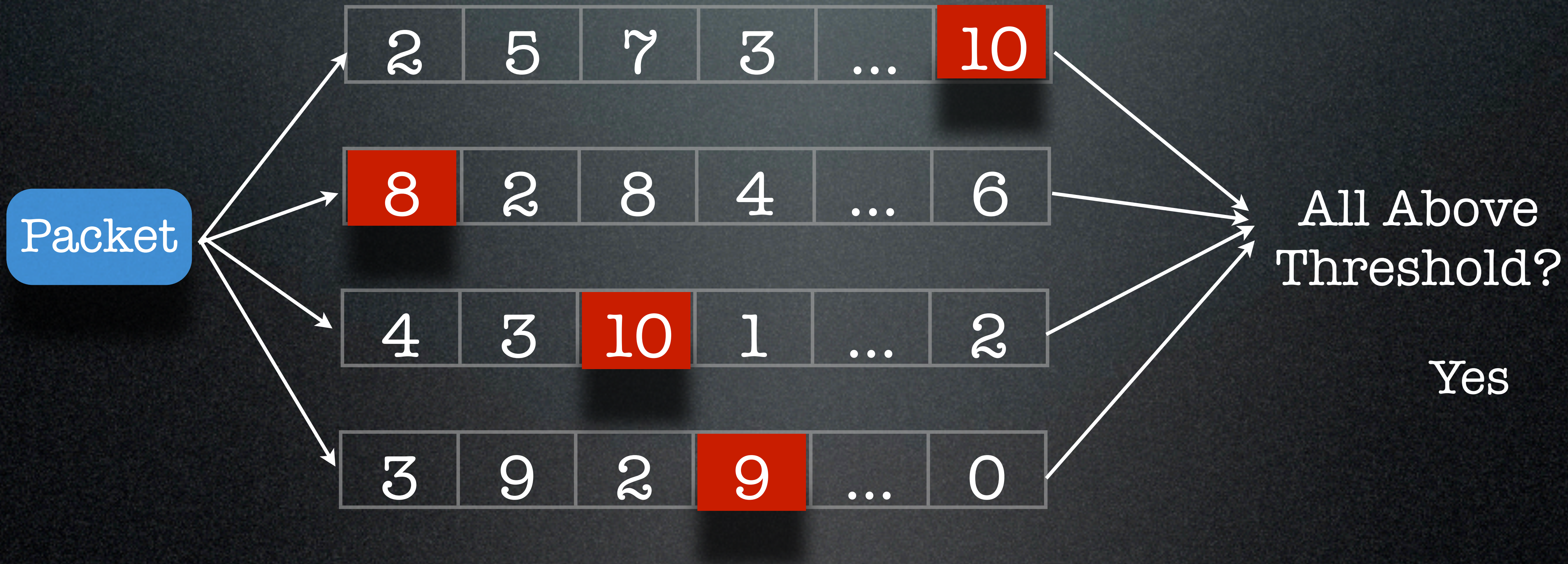
2	5	7	3	...	**10**
8	2	8	4	...	6
4	3	**10**	1	...	2
3	9	2	**9**	...	0

Multi-Stage Filtering



2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

Multi-Stage Filtering



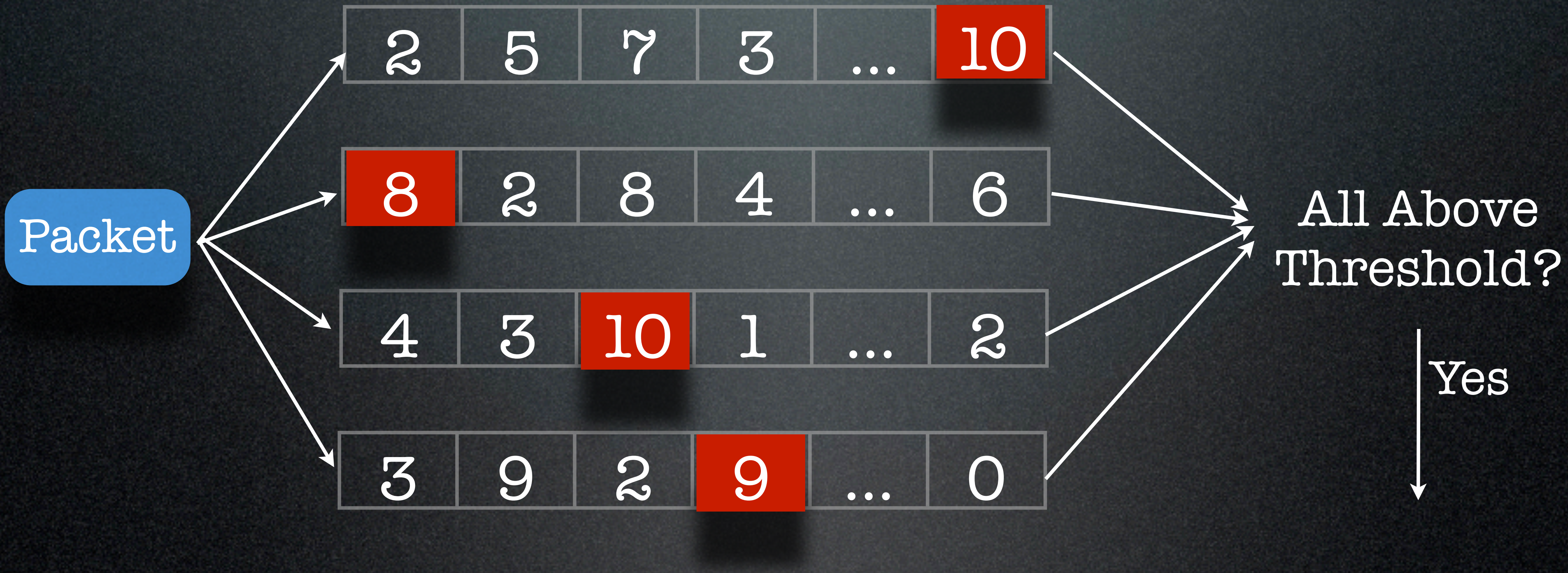
Packet

2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

All Above Threshold?

Yes

Multi-Stage Filtering



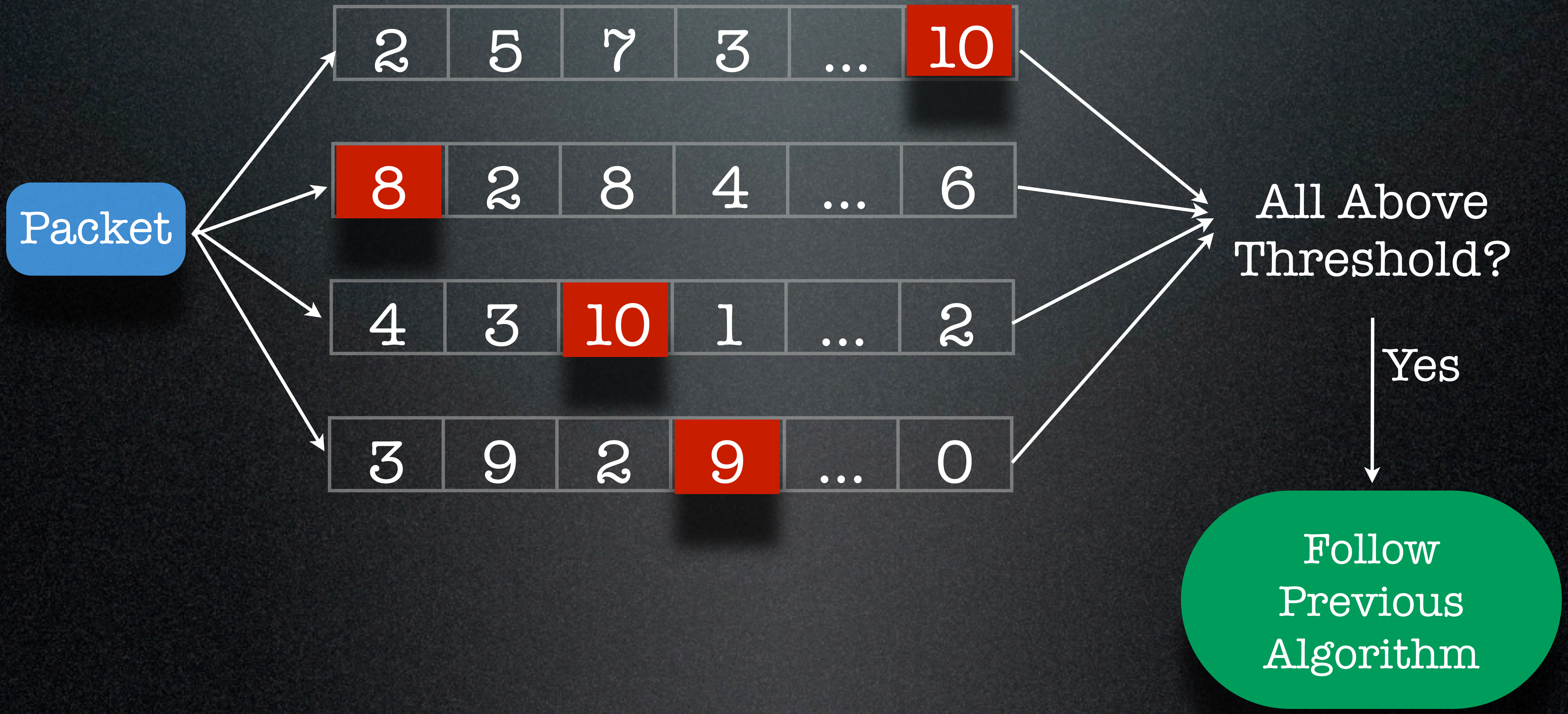
Packet

2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

All Above Threshold?

Yes

Multi-Stage Filtering



Packet

2	5	7	3	...	10
8	2	8	4	...	6
4	3	10	1	...	2
3	9	2	9	...	0

All Above Threshold?

Yes

Follow Previous Algorithm

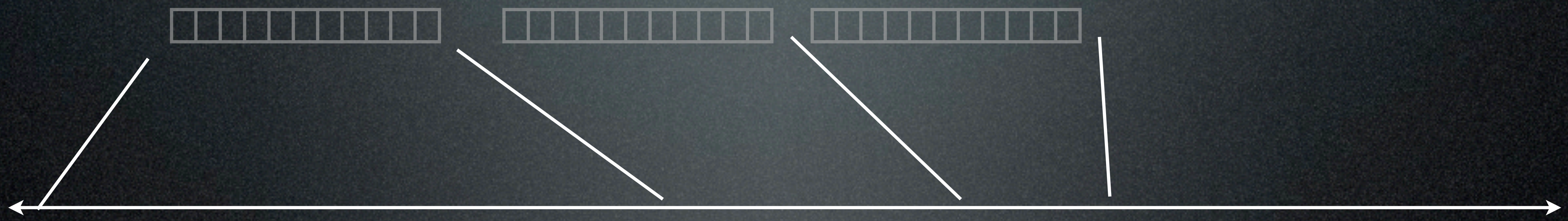
Rabin Fingerprints

- Worms may shift code though several packets or within a packet to disguse it.
- Use a fingerprint smaller than a whole packet, thus many in one packet.
- Analise a while stream, not just a single packet.
- Use a fingerprint of size β, thus a stream of s bytes would have s-β+1 fingerprints.

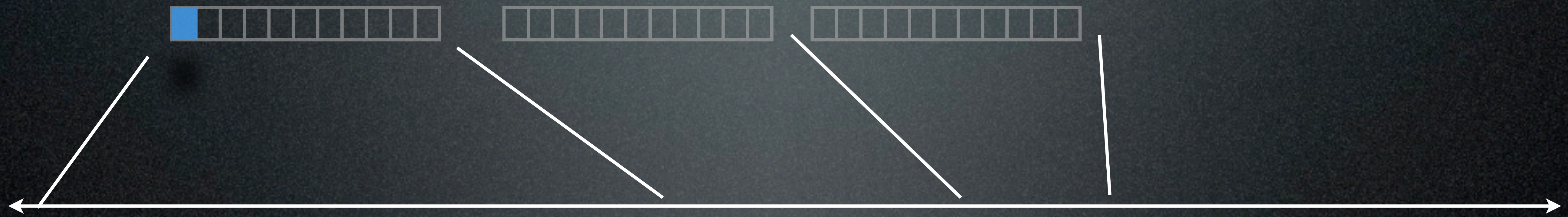
IP-Address Bit-mapping

- Storing all IP addresses after the preveleance thresholds are met would be memory intensive.
- Use a constant size maping of IP address hashes to keep track of the number and extrapilate a count of IP addresses.
- Not robust enough to get granularity as the number of infected machines and prevelance of packets increases.
- Use a multi-level bit mapping to keep track at a higher granularity.

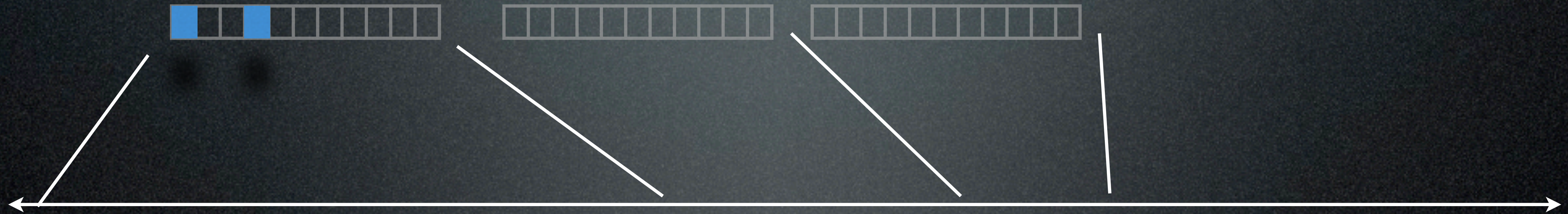
IP-Address Bit-mapping



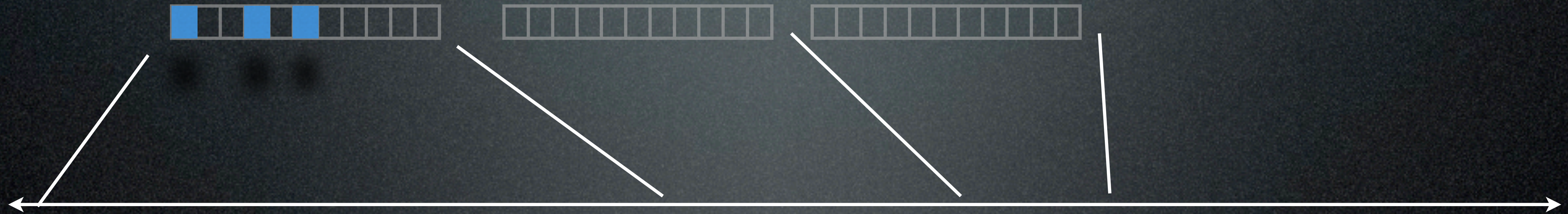
IP-Address Bit-mapping



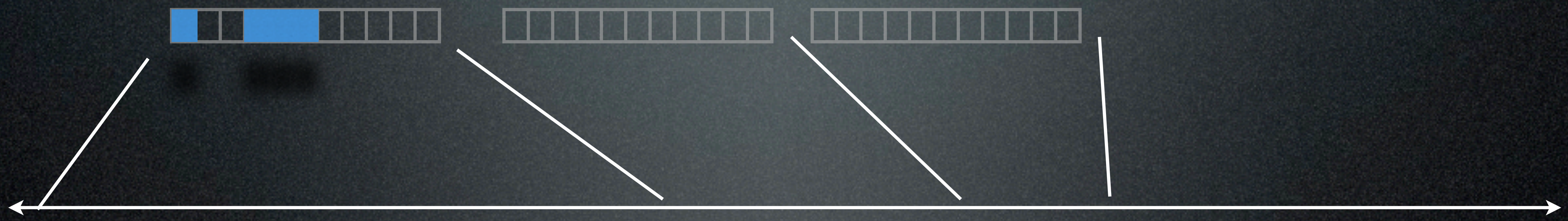
IP-Address Bit-mapping



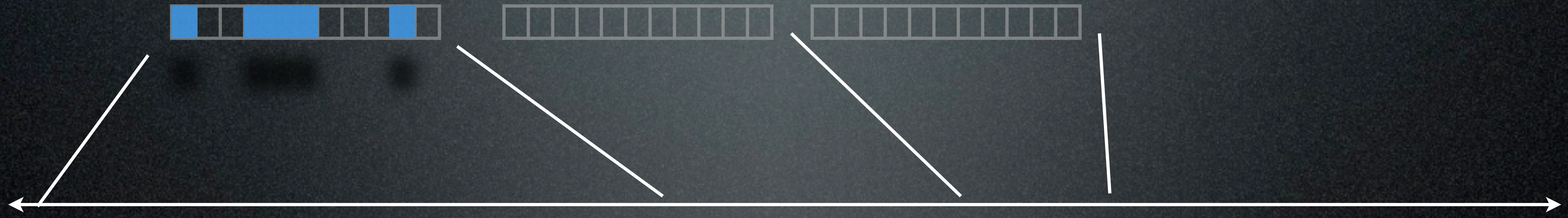
IP-Address Bit-mapping



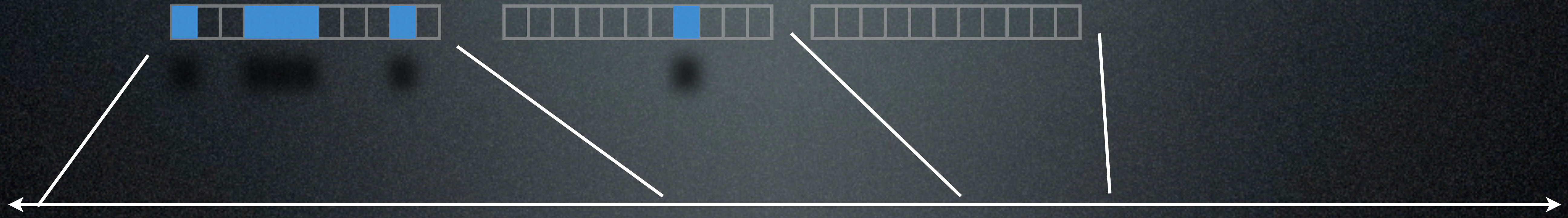
IP-Address Bit-mapping



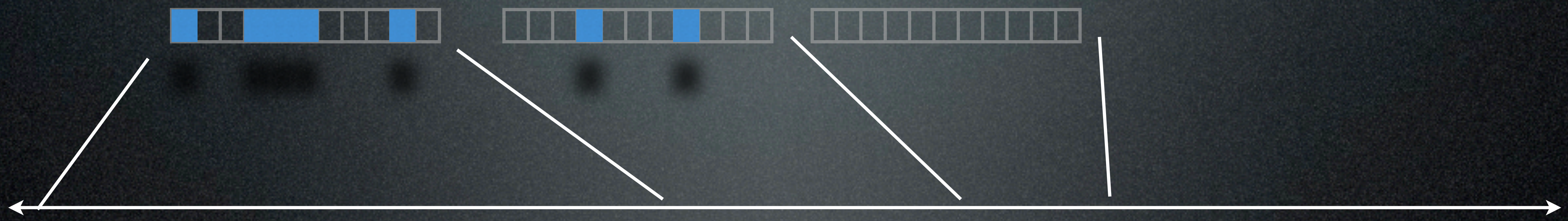
IP-Address Bit-mapping



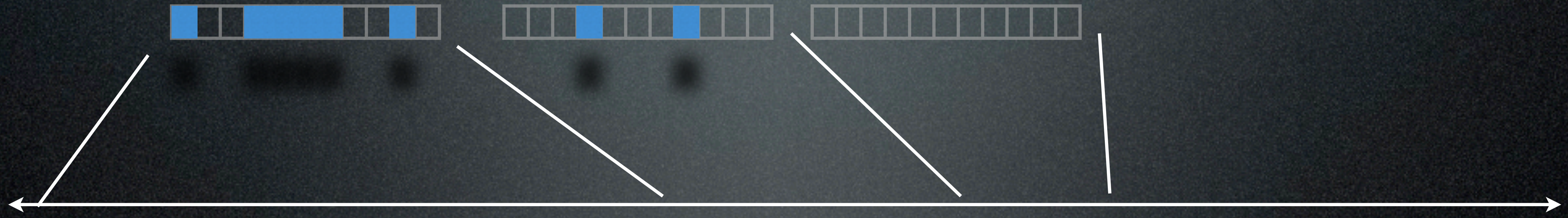
IP-Address Bit-mapping



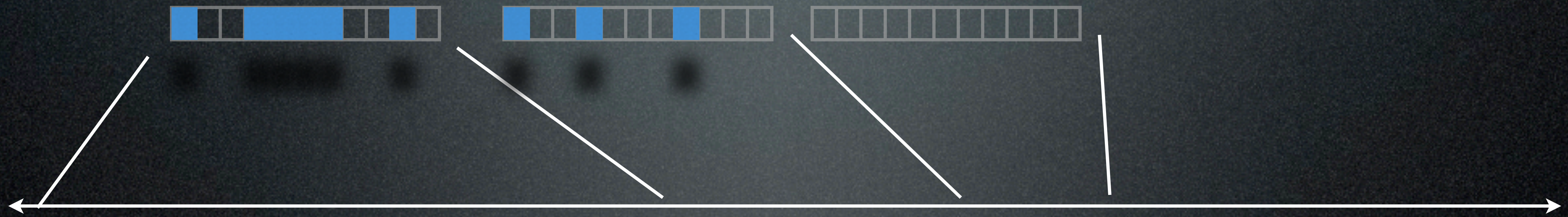
IP-Address Bit-mapping



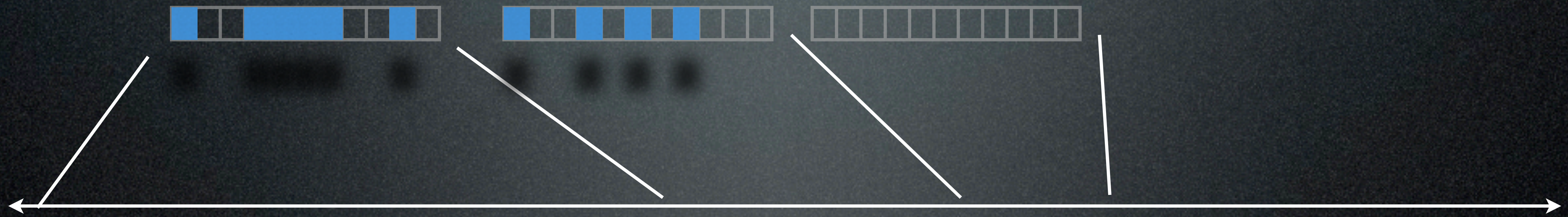
IP-Address Bit-mapping



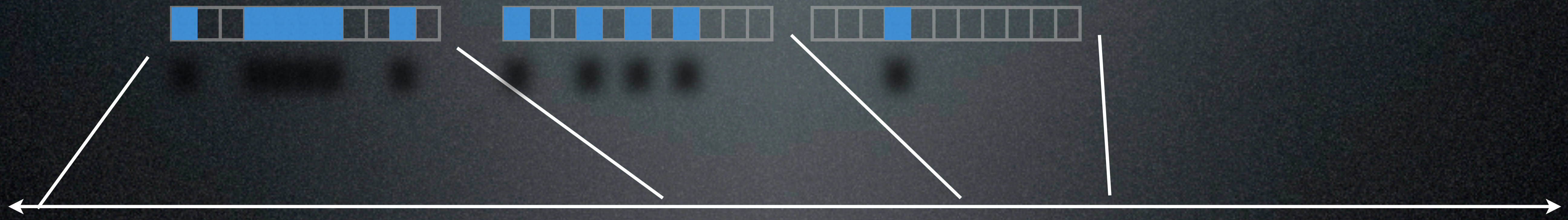
IP-Address Bit-mapping



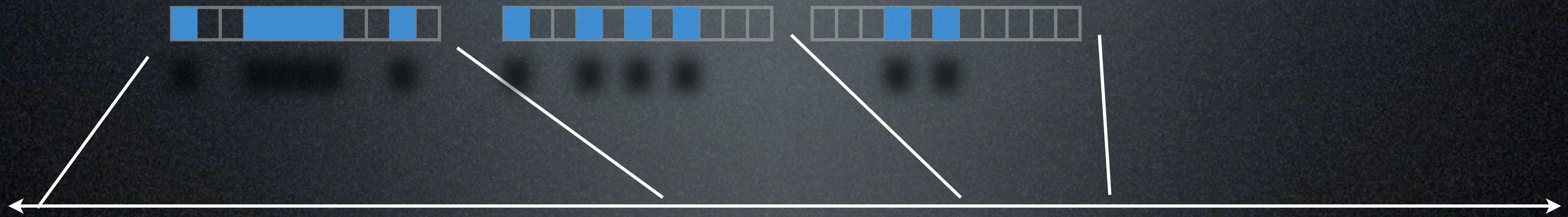
IP-Address Bit-mapping



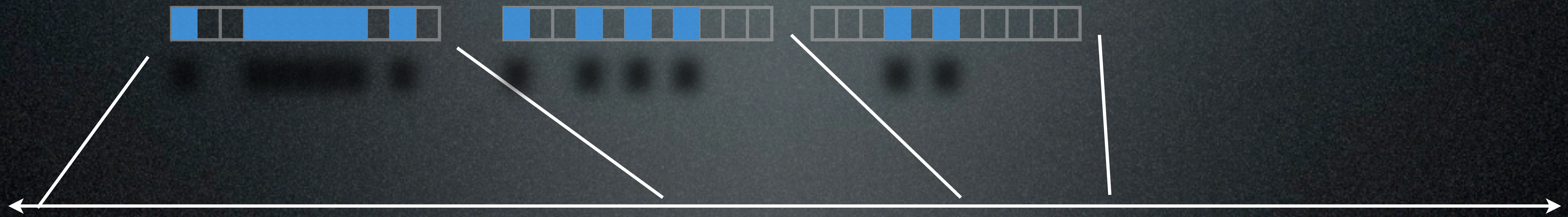
IP-Address Bit-mapping



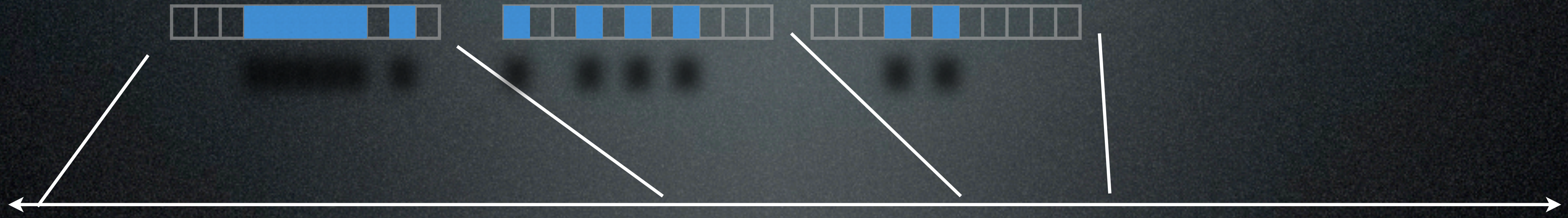
IP-Address Bit-mapping



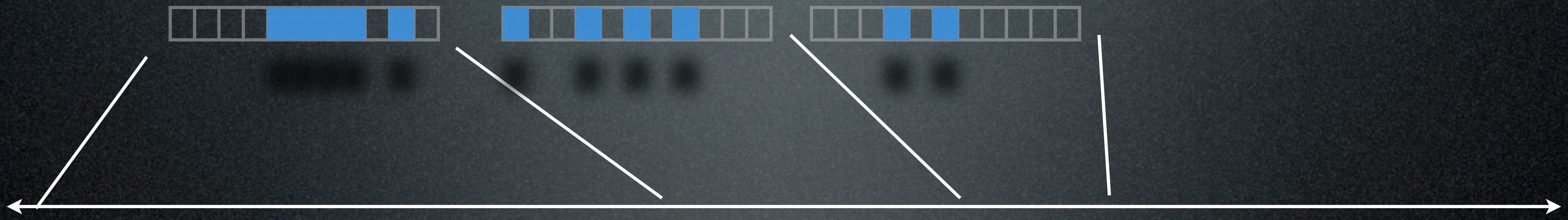
IP-Address Bit-mapping



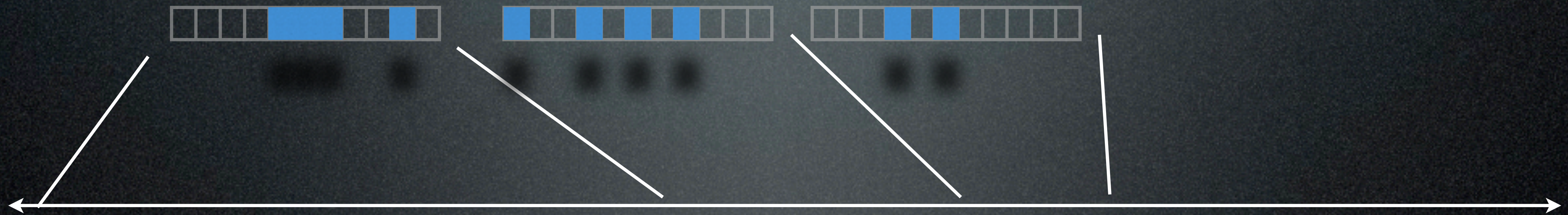
IP-Address Bit-mapping



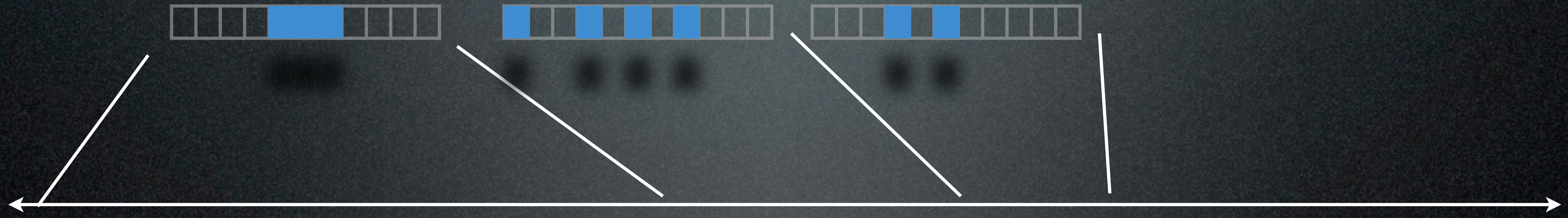
IP-Address Bit-mapping



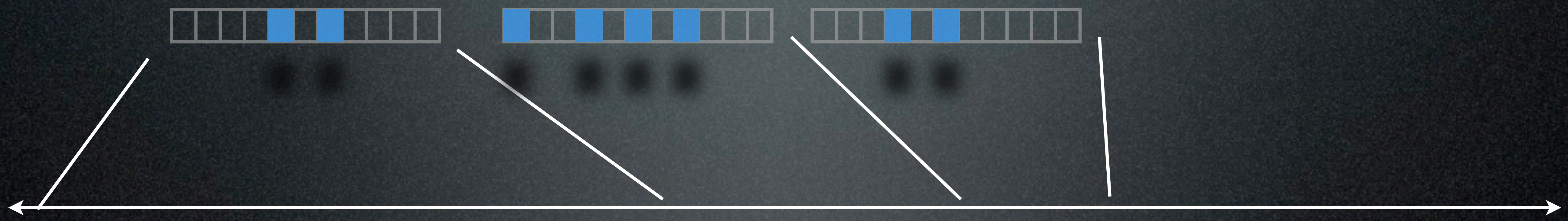
IP-Address Bit-mapping



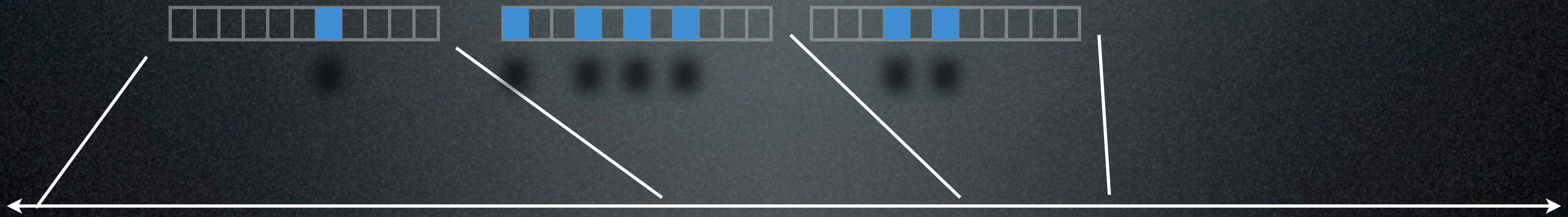
IP-Address Bit-mapping



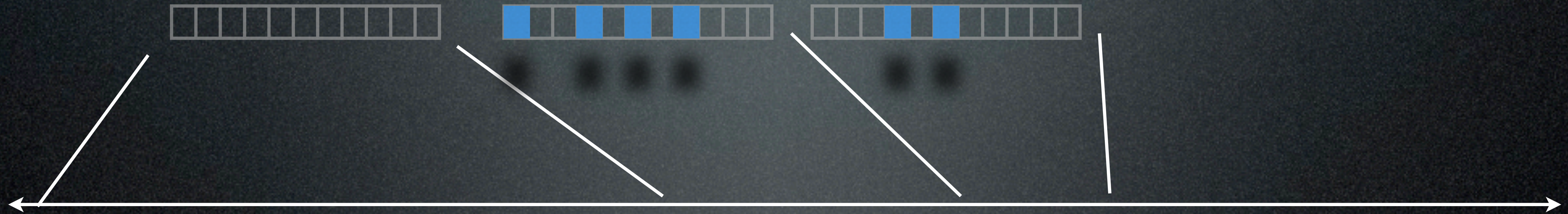
IP-Address Bit-mapping



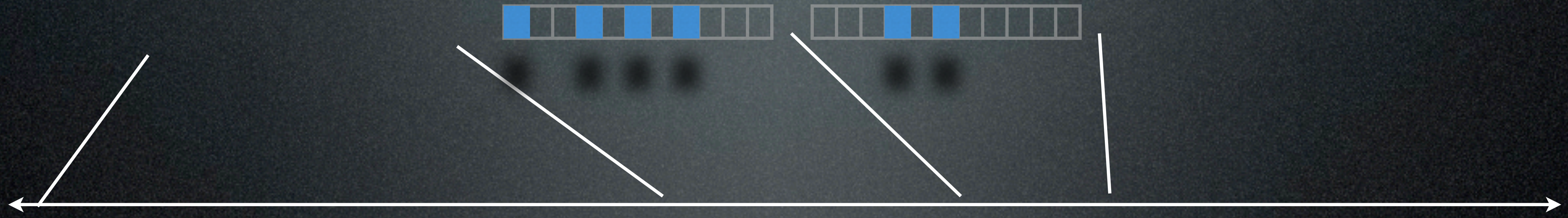
IP-Address Bit-mapping



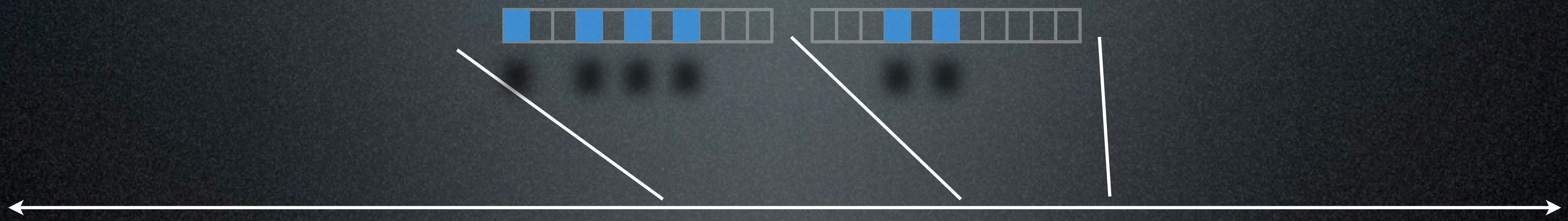
IP-Address Bit-mapping



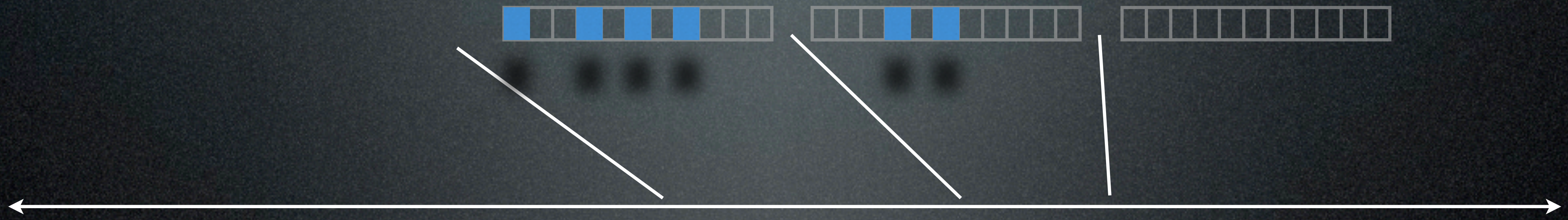
IP-Address Bit-mapping



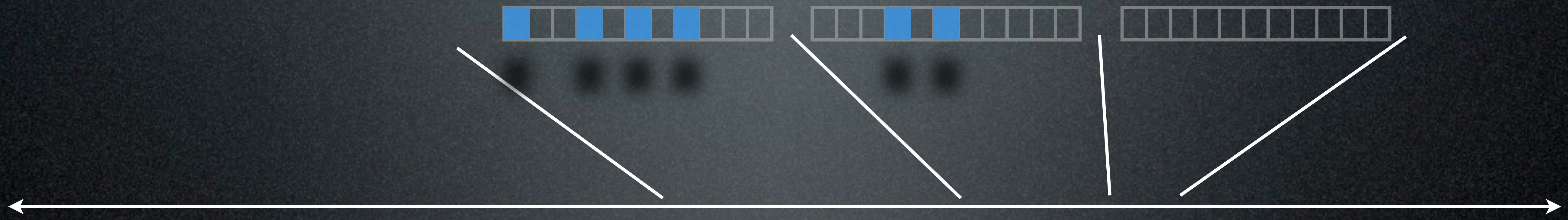
IP-Address Bit-mapping



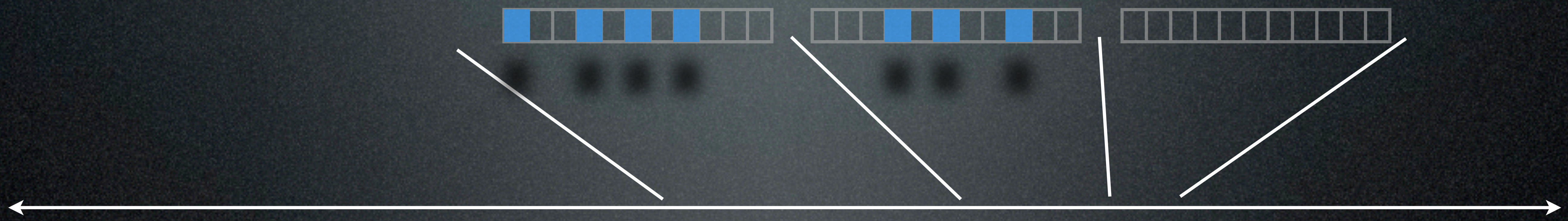
IP-Address Bit-mapping



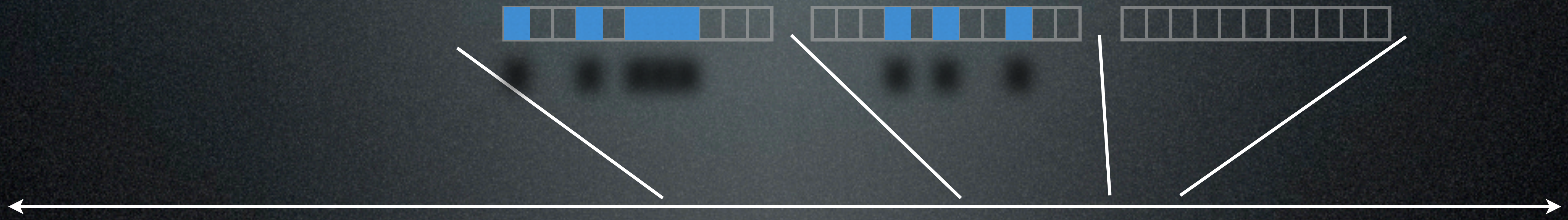
IP-Address Bit-mapping



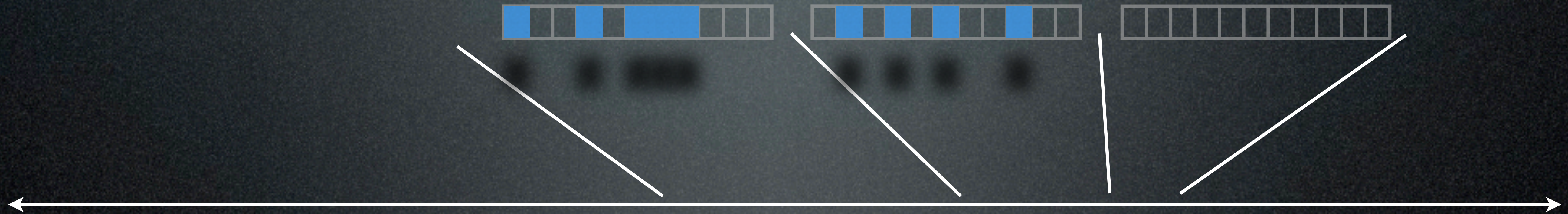
IP-Address Bit-mapping



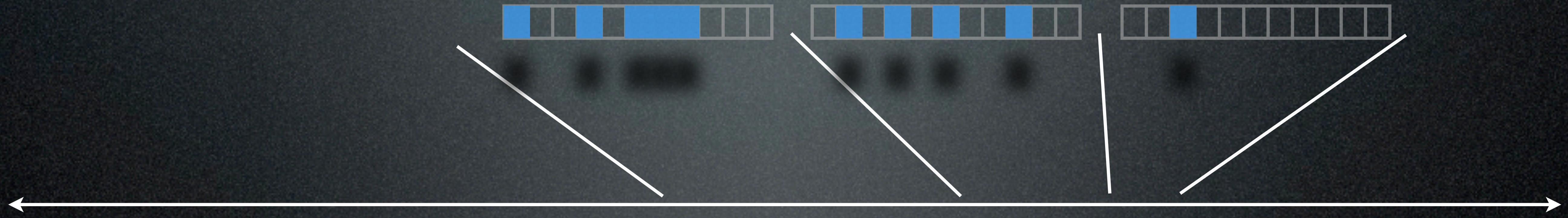
IP-Address Bit-mapping



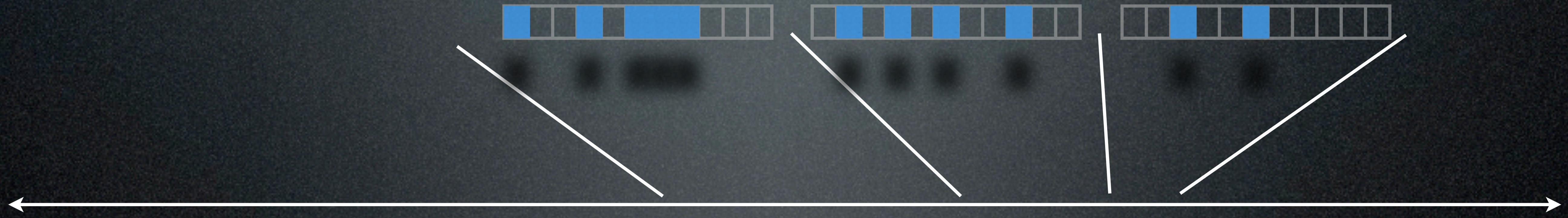
IP-Address Bit-mapping



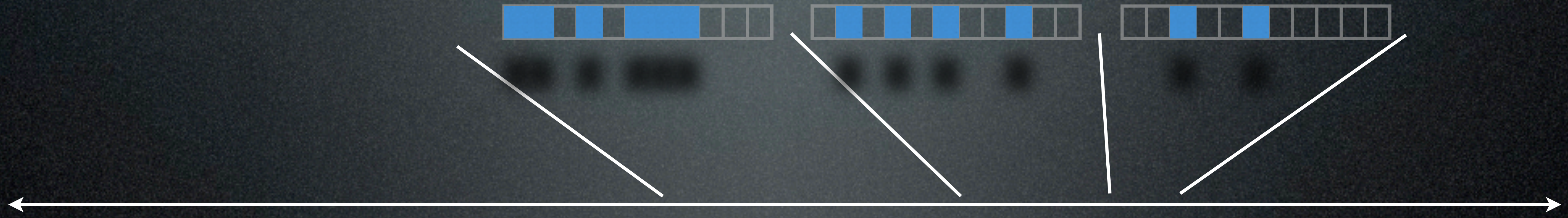
IP-Address Bit-mapping



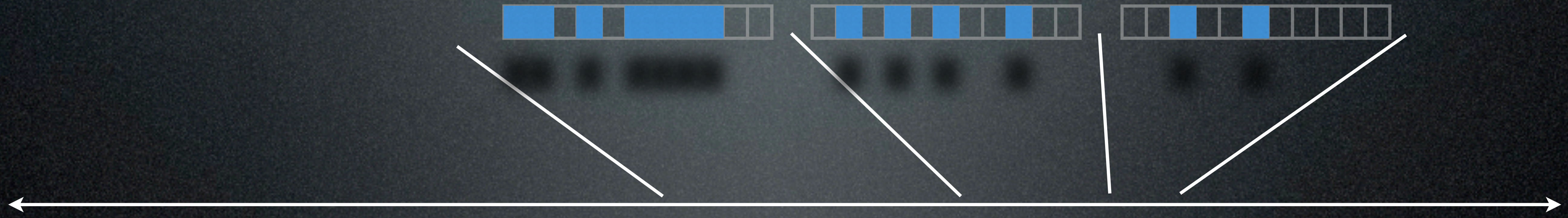
IP-Address Bit-mapping



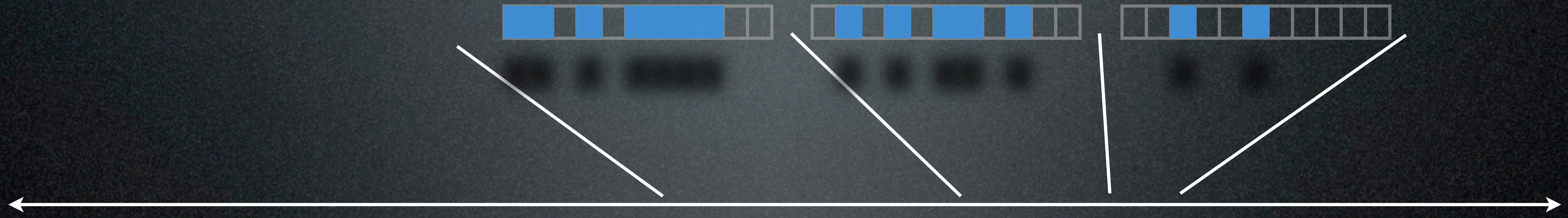
IP-Address Bit-mapping



IP-Address Bit-mapping



IP-Address Bit-mapping



Summary

- Track network traffic, if a lot of traffic looks very similar (maps to the same hash) pay attention to it.
- Keep track of how many unique paths the data that is being observed, if the traffic is suspicious analise it.
- Extract the key of the worm if it shows all the signs of a worm.

Contributions

- Proof of concept that a system can be created to identify worms on a reliable basis.
- Was able to identify all worms that appeared in the sampling time, much faster than then the rest of the industry.
- Later arguments in the paper show how it can be expanded to a larger system.

Weaknesses

- If there is a invariant that is smaller than β then this system would not catch it.
- Reassembling worms might evade the system.
- Encrypted code, (SSL, SSH, or VPN).
- Has a hard time filtering BitTorrent.

How to Improve

- Test on hardware, or router level detection.
- Be able to dynamically change thresholds depending on traffic fluctuations.