TRAFFIC CONFIRMATION ATTACKS DESPITE NOISE

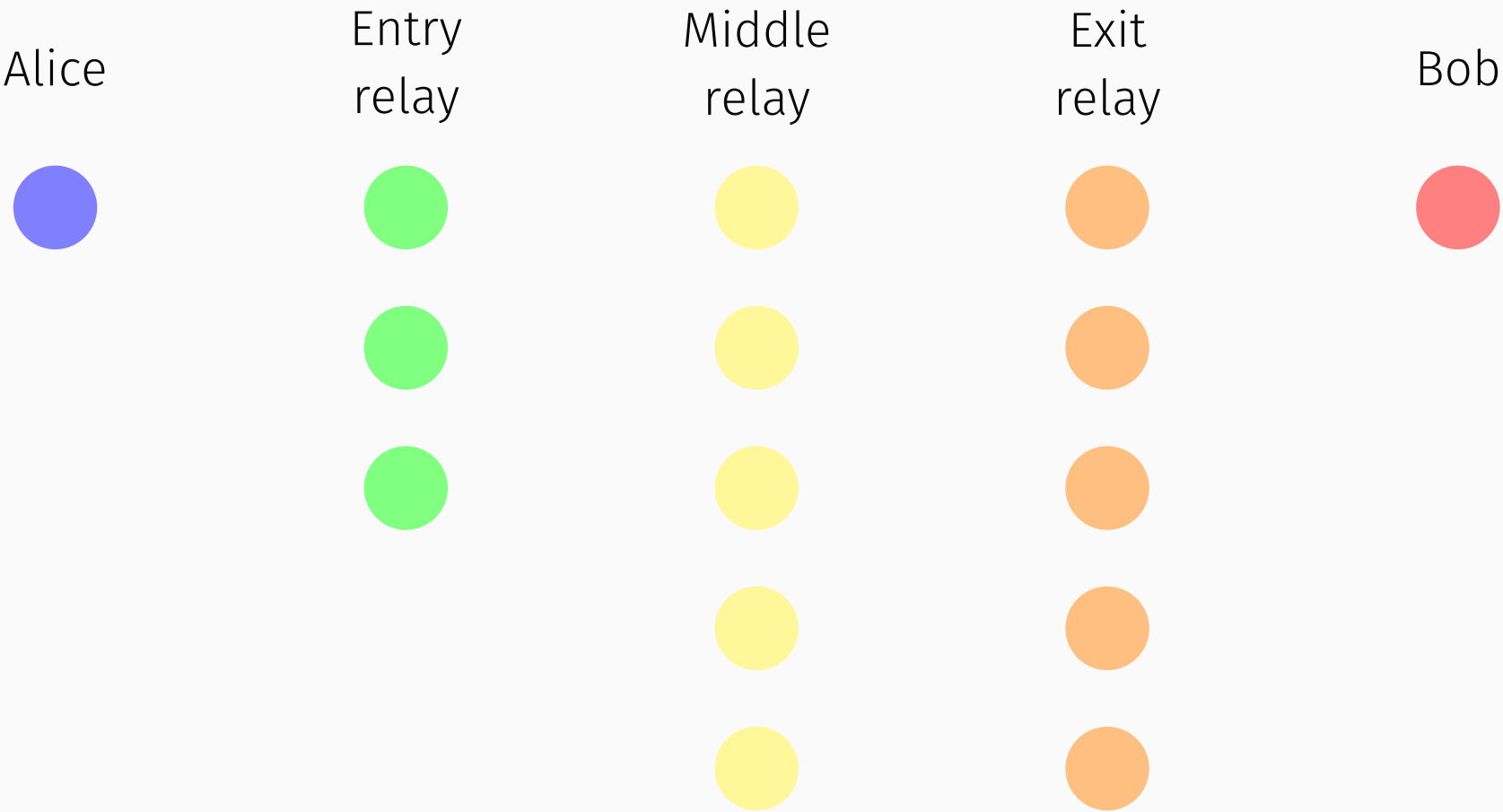
Jamie Hayes

University College London
j.hayes@cs.ucl.ac.uk

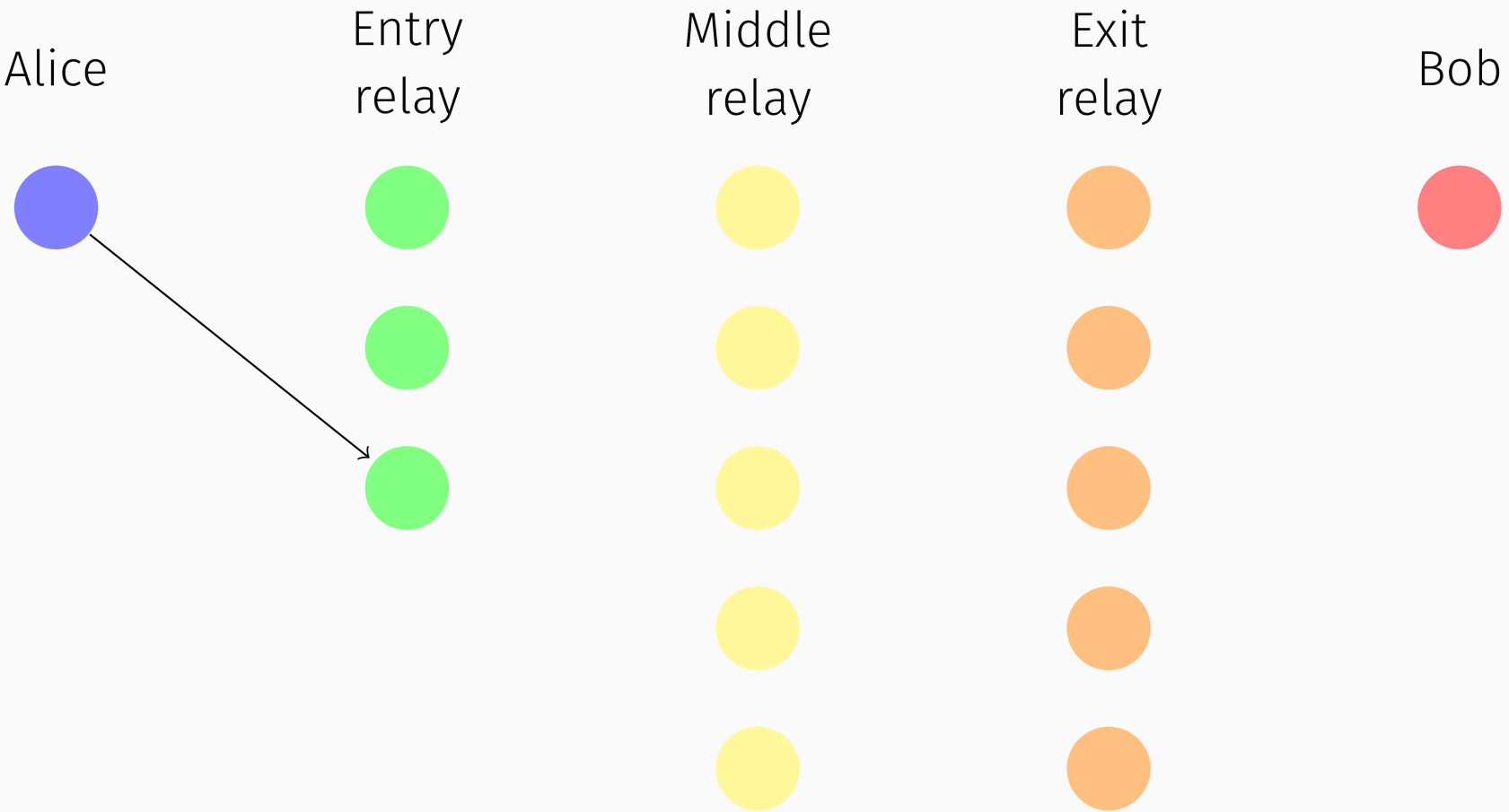
ANONYMOUS COMMUNICATION

- ”We kill people based on the metadata.” - Michael Hayden (Former NSA Director) - Sometimes encryption isn’t good enough.
- Onion Routing attempts to hide this metadata by obscuring communication patterns by sending traffic through relays.
- Low latency - trade off between usability and security.

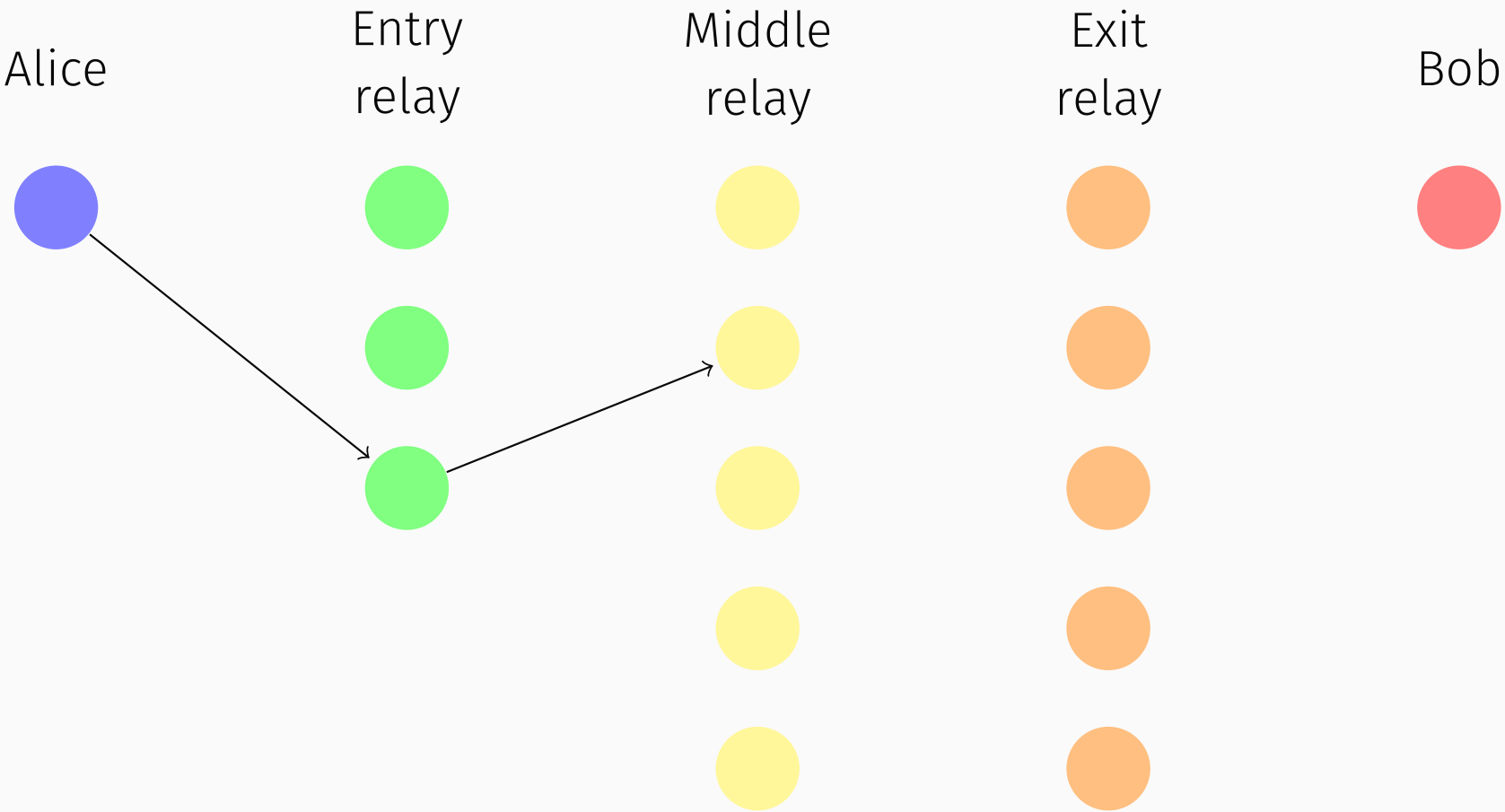
HOW DOES TOR WORK?



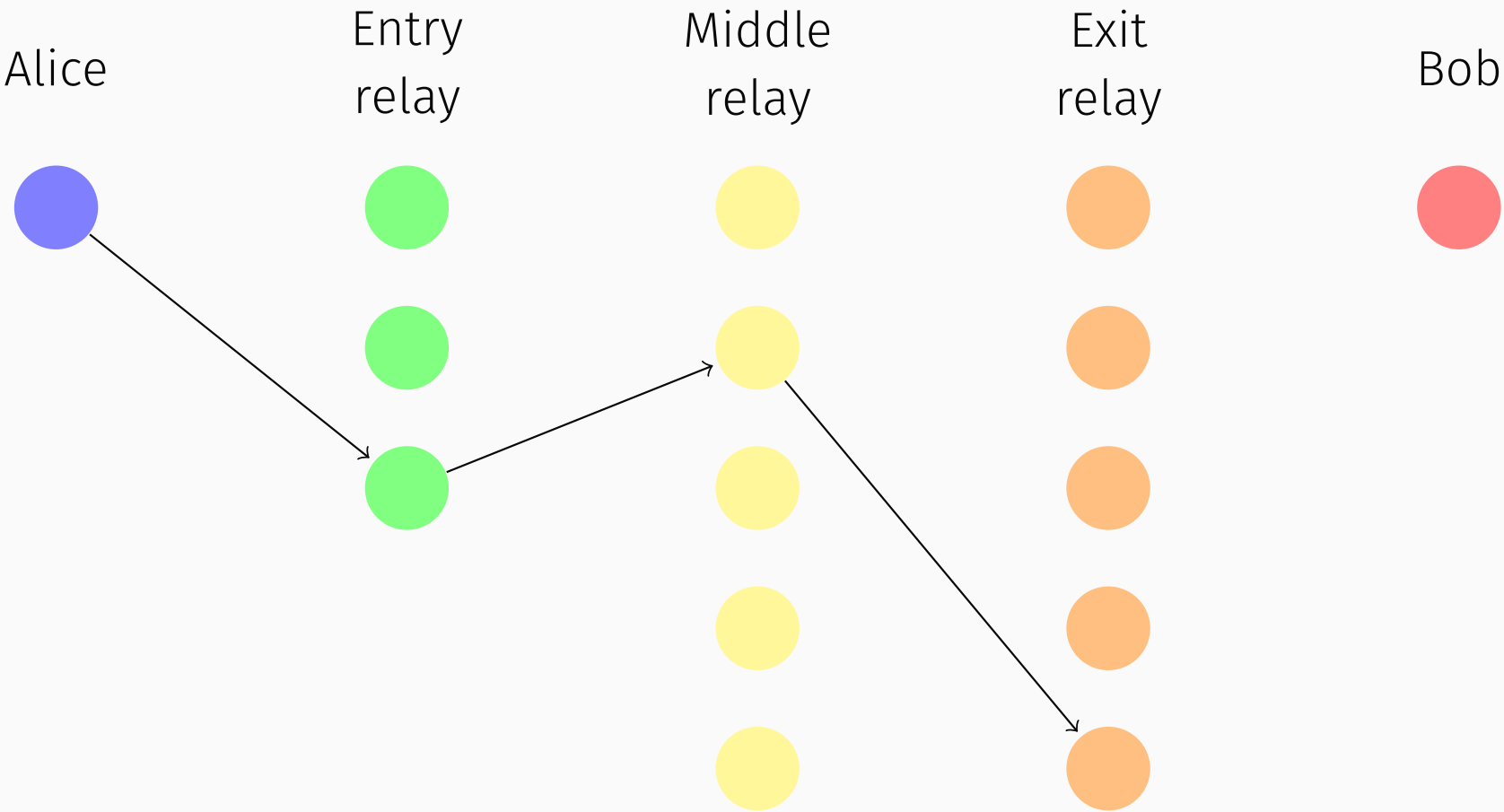
HOW DOES TOR WORK?



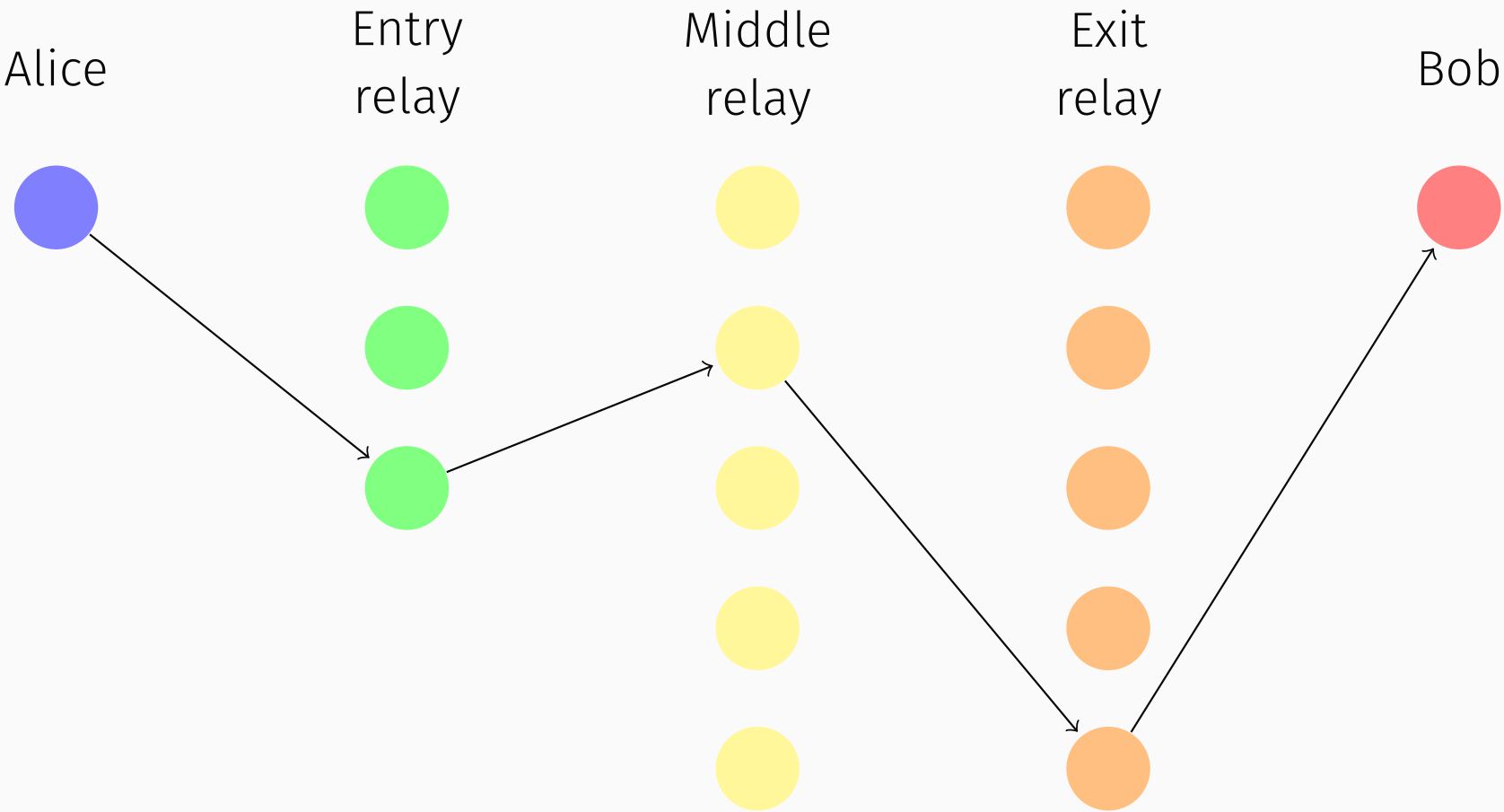
HOW DOES TOR WORK?



HOW DOES TOR WORK?



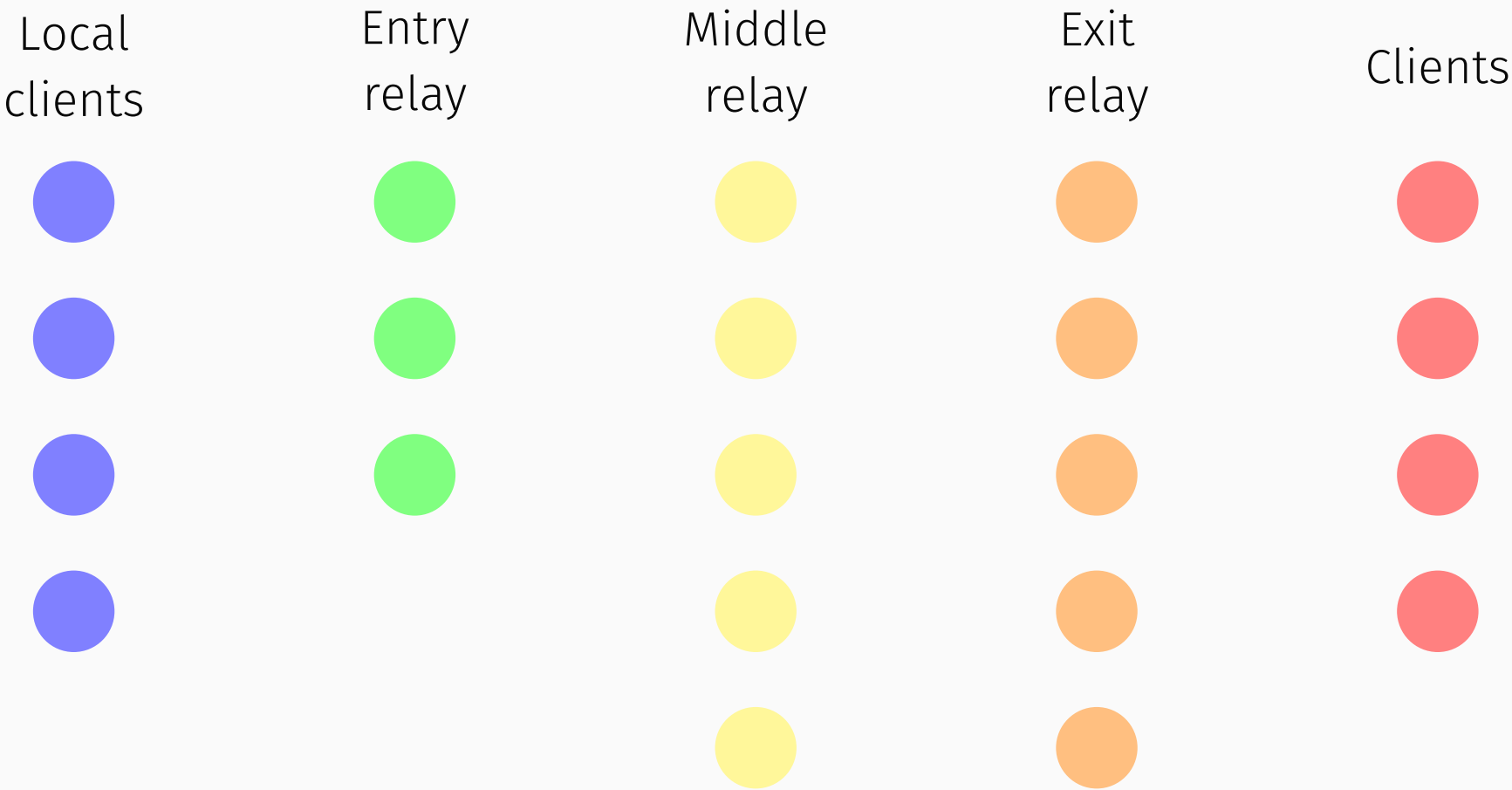
HOW DOES TOR WORK?



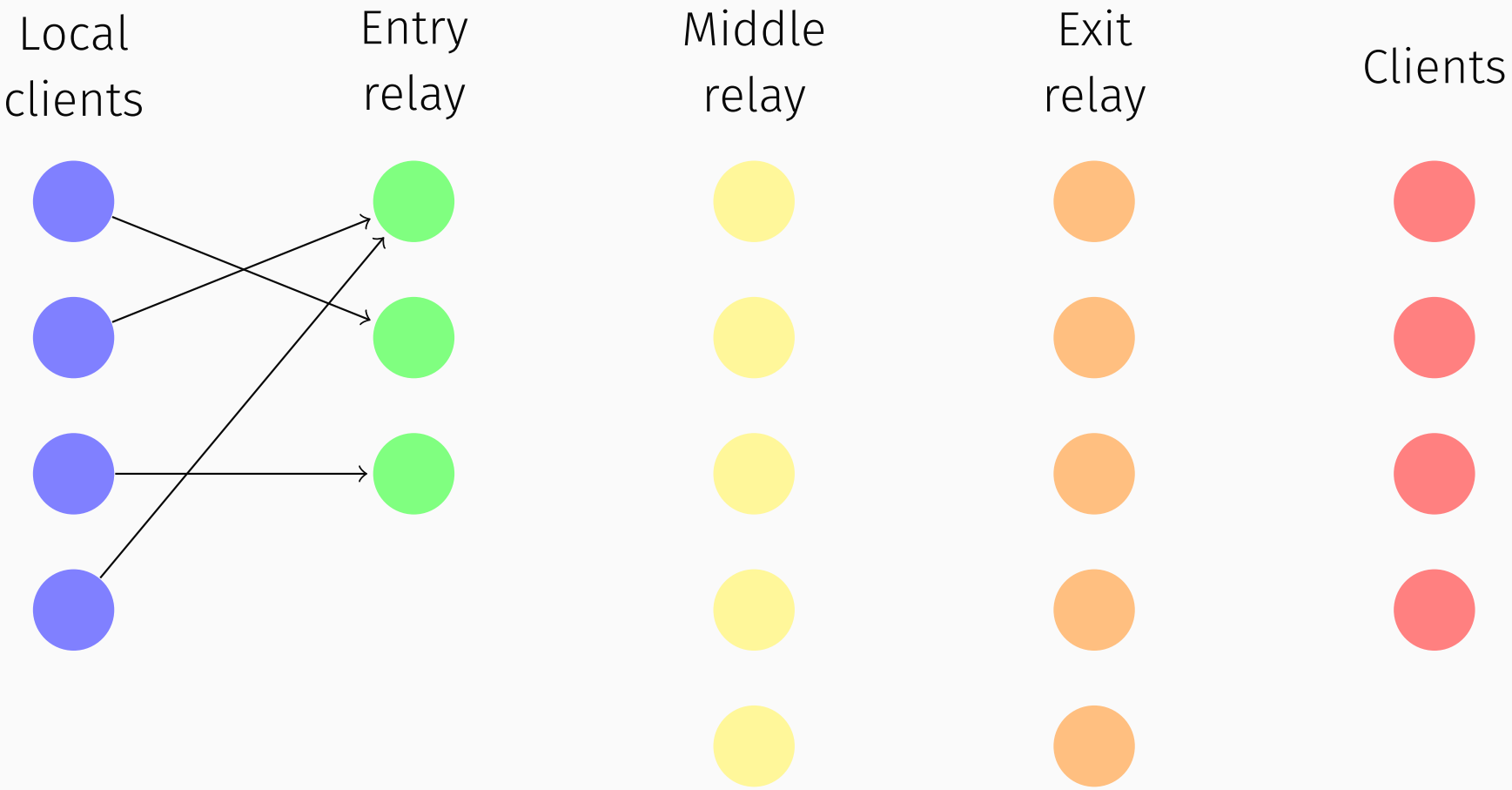
TRAFFIC CONFIRMATION ATTACKS

- Active attacks purposefully inject watermarks in to traffic. The adversary modifies network traffic flows introducing patterns that be can observed at another location in the network, allowing the adversary to de-anonymize clients.
- Passive attack captures traffic and creates a fingerprinting without injecting or modifying.
- Fingerprints created by exploiting unique information from network traffic traces like unique packet lengths and burst patterns.
- Easier when packet lengths are available but Tor pads all cells to a fixed size of 512 bytes.

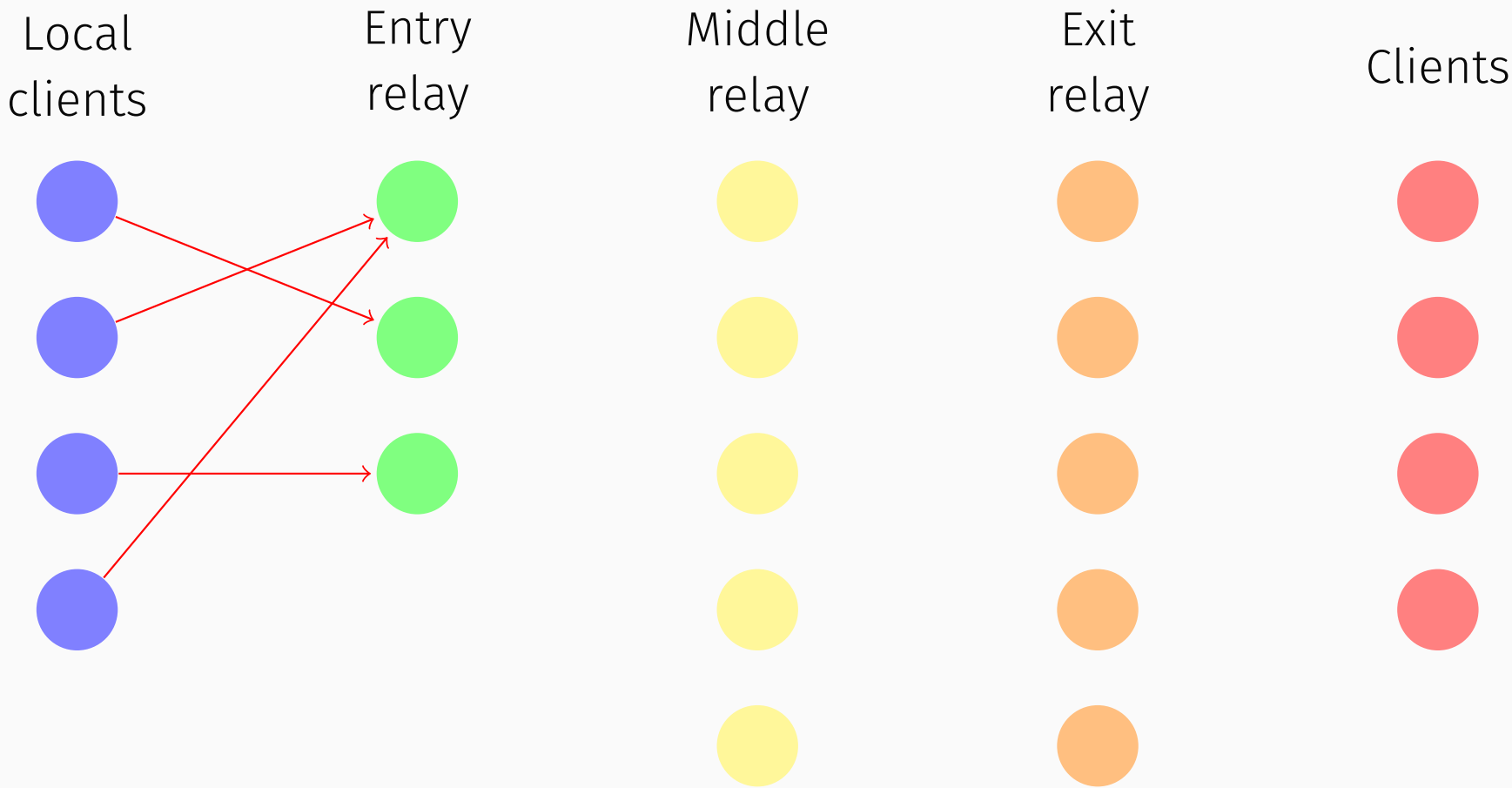
PASSIVE TRAFFIC CONFIRMATION ATTACK



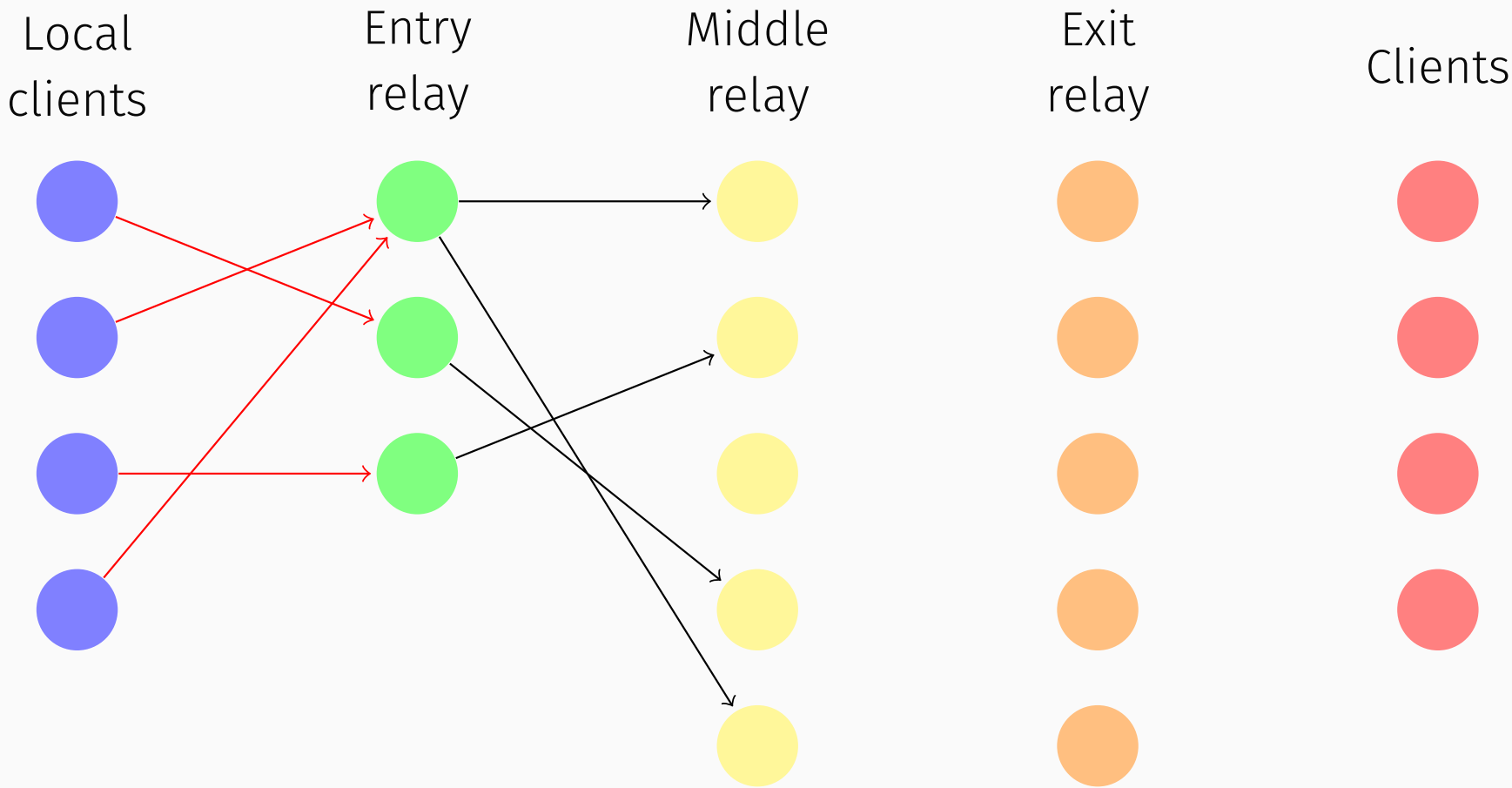
PASSIVE TRAFFIC CONFIRMATION ATTACK



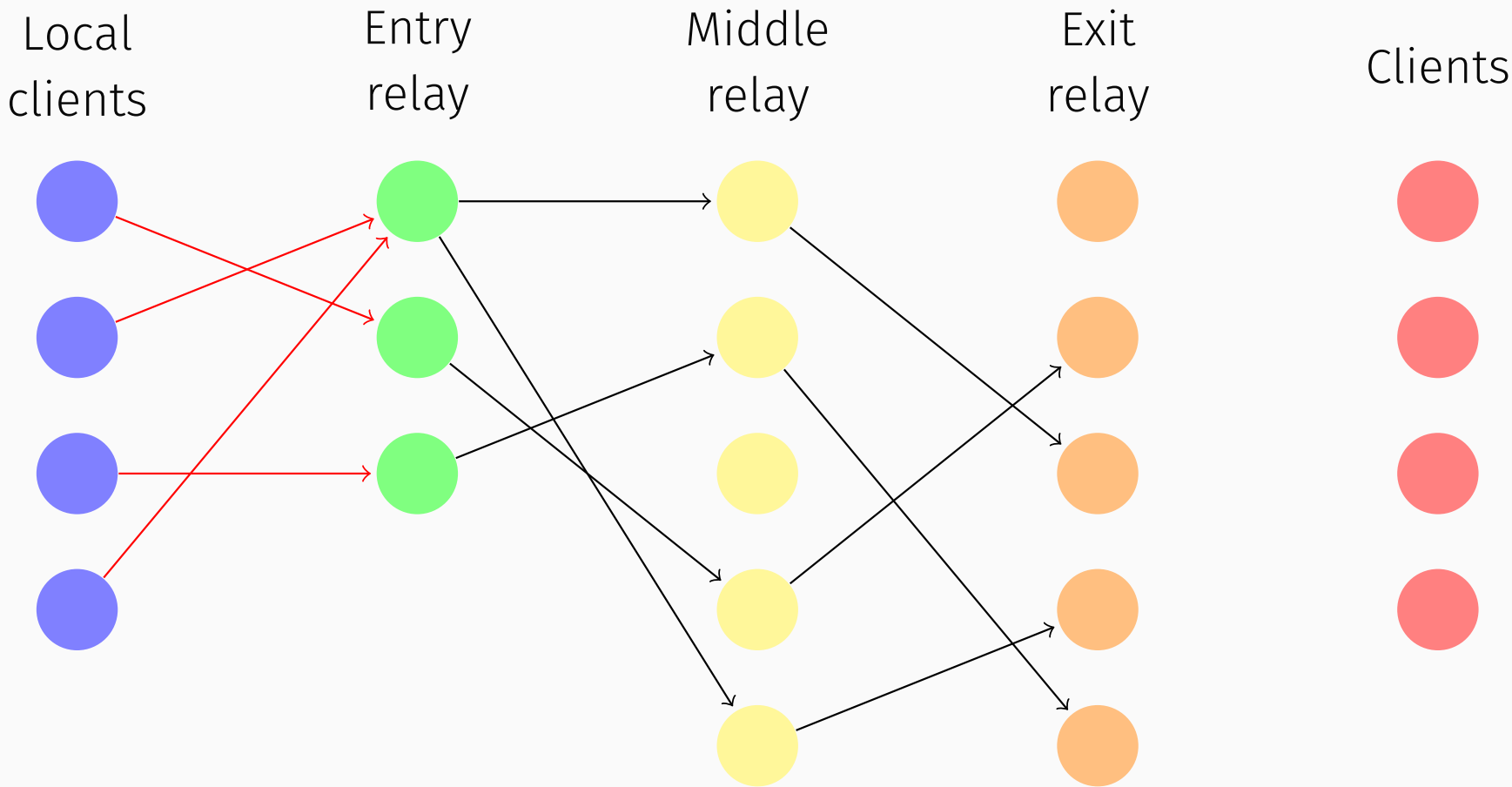
PASSIVE TRAFFIC CONFIRMATION ATTACK



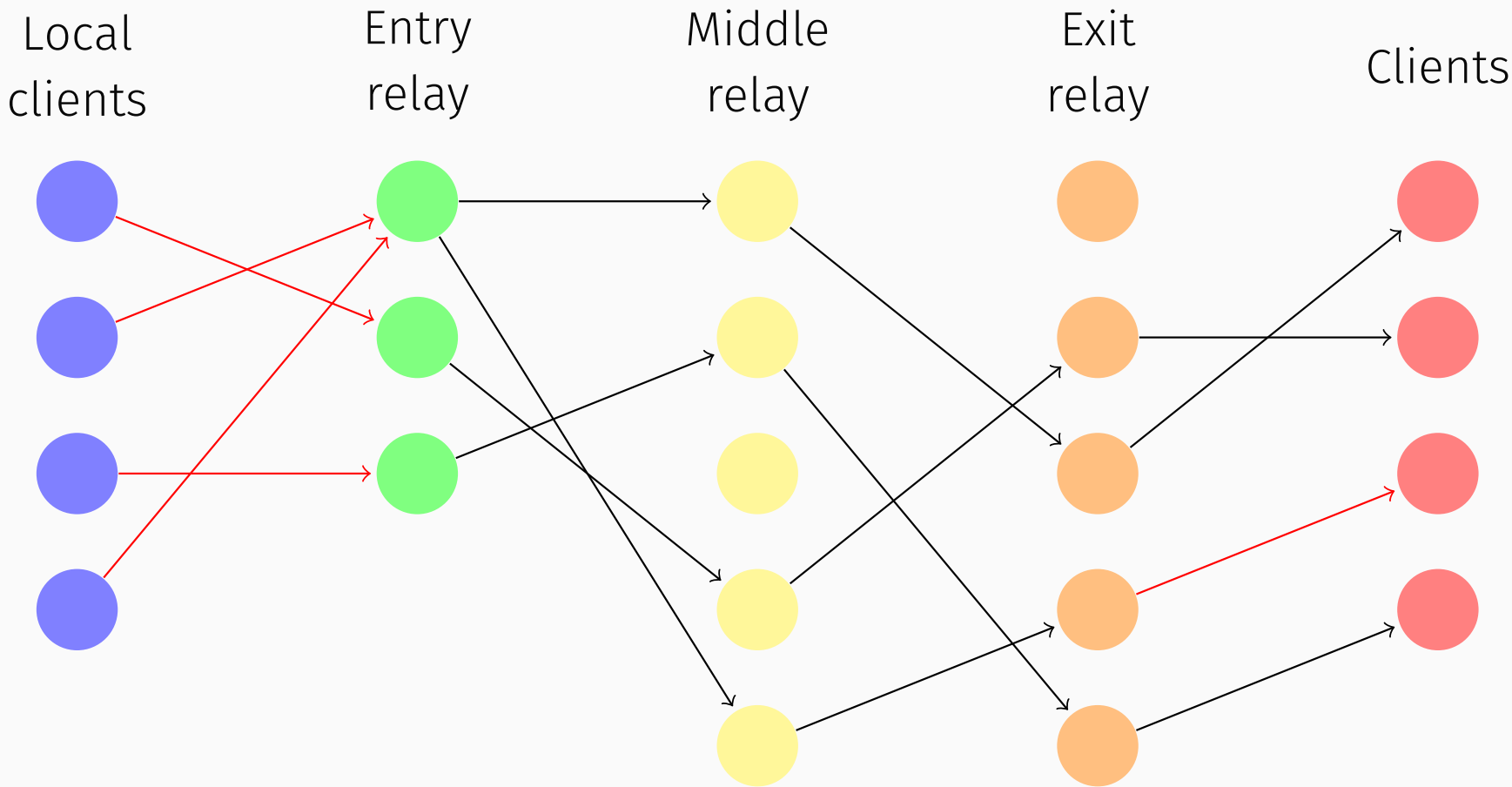
PASSIVE TRAFFIC CONFIRMATION ATTACK



PASSIVE TRAFFIC CONFIRMATION ATTACK



PASSIVE TRAFFIC CONFIRMATION ATTACK



THE THREAT IS REAL

- Long history of passive attacks on Tor.
- E.g. [Steven J. Murdoch et al. 2007] ”Sampled Traffic Analysis by Internet-Exchange-Level Adversaries”.
- Not in the Tor threat model.

TOR CONGESTION

- Currently packets are sent through the Tor network using TCP which guarantees reliable transmission.
- TCP flow control has been identified as a major cause of latency in Tor.
- If the reduction of latency in Tor or the use of VoIP and similar protocols is desired, it may be prudent to allow UDP over Tor.

DATA

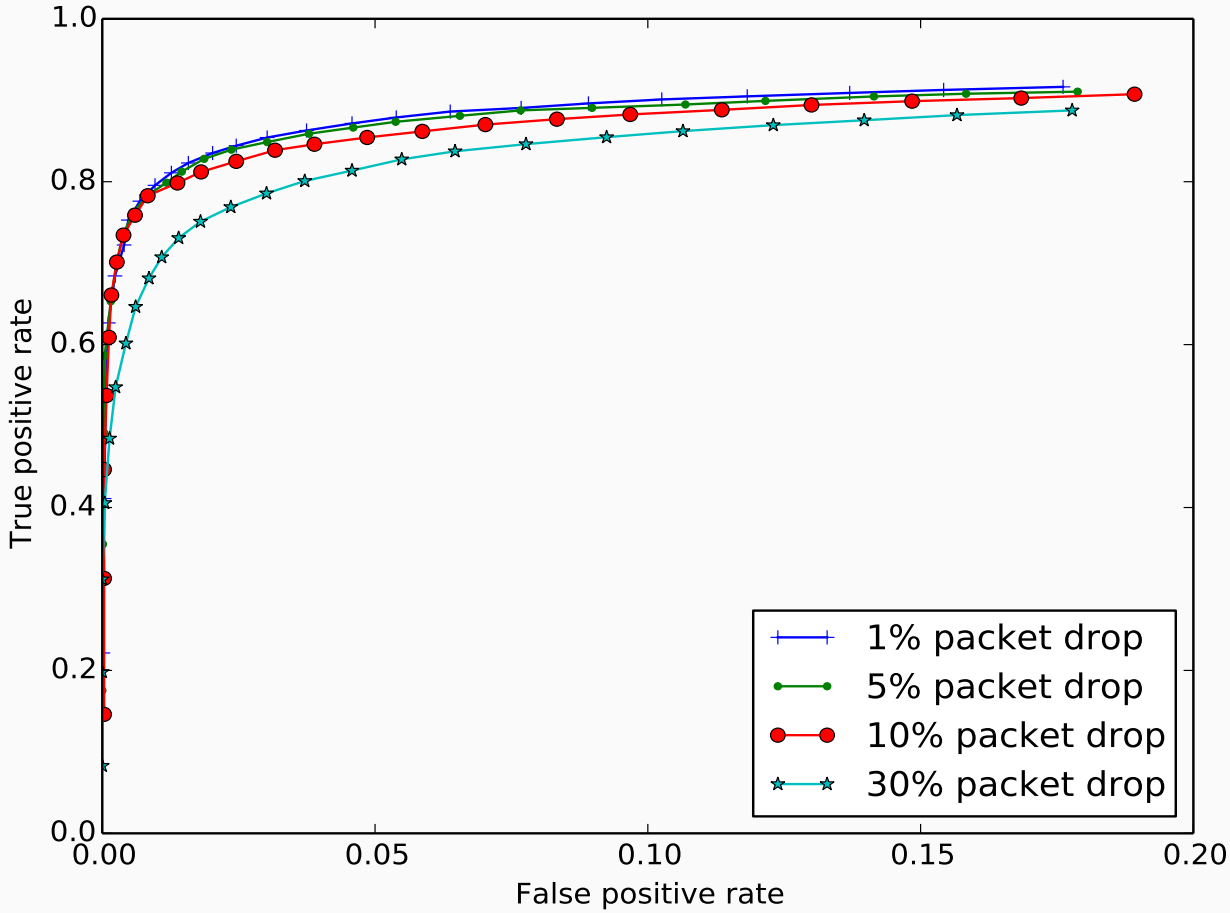
- 9000 traces of different website loads for client side [Wang et al. 2014], injected IPDV for server side.
- IPDV distribution collected over Tor.
- We set up a simple webpage hosted in five different regions. For each we loaded the webpage 100 times through Tor and recorded the packet time arrivals at both client and sever, giving the inter-packet delay variation that an adversary can expect between client and server when collecting traces over Tor.
- No live implementation.

HOW TO CONVERT A TRAFFIC TRACE TO A FINGERPRINT

1. Split the traffic trace in to N windows of equal time. Initialize fingerprint, H, of m integers all set at 0.
2. For each time window $T_0, .., T_{N-1}$ extract the max time, $\bar{T}_i$, and the number of packets, B_i.
3. H is a function of $\bar{T}_i$ and B_i onto m pseudo-random bases $R_a()$. The pseudo-random bases were chosen so that packets with similar timings will be projected to values on the bases that are close to one another, resulting in similar hashes for similar network traffic flows.
4. H is updated at each time window. With the final fingerprint:

$$H_a = sign(H_a) = \begin{cases} 1 & \text{if } H_a > 0 \\ 0 & \text{if } H_a \leq 0 \end{cases} \text{ for } a \in \{1, .., m\}.$$

RESULTS



At 10% drop, **TPR=0.7** and **FPR=0.005**. With 9,000 background traces 45 will be incorrectly flagged as match. Is this amount tolerable?

PADDING

- Currently Tor does not implement any padding.
- Constant padding could defeat this attack but is probably too expensive to implement.
- Probabilistic padding offers a softer alternative.
- Adaptive padding [Shmatikov et al. 2006] tries to unlink sender and receiver of a communication by injecting padding in to statistically unlikely gaps in network traffic. This limits the amount of extra bandwidth required and does not incur any latency costs as packets are forwarded as soon as they are received.

ADAPTIVE PADDING

● = real packet, ● = dummy packet

- Adaptive padding (**AP**) uses histograms of inter-arrival packet times to determine when a packet should be injected.
- **AP** notifices there is a natural delay –> enters gap mode –> increases the probability of injecting a dummy packet.
- **AP** notifices traffic is flowing at a high rate –> enters burst mode –> reduced the probability of injecting a dummy packet.
- **AP** destroys natural fingerprints created by the gaps in flows.

TRAFFIC CONFIRMATION ATTACKS ON AP

- All flows that are seen before the entry relay are padded by AP with padding removed by some trusted relay in the network before arriving at the destination of the communication.
- In the opposite direction we assume AP is applied before an adversary captures traffic (either by directly applying AP at the web sever, or at a trusted bridge) and is removed before an adversary captures traffic at the end of the circuit.

TRAFFIC CONFIRMATION ATTACKS ON AP

For both incoming and outgoing network flows, we created histograms of inter-arrival packet times by crawling the top 25K alexa websites. We then applied AP to each of the 9000 traces in our data set, to simulate the padded flows collected by the adversary at the start of communication. Note that the adversary is only concerned with classifying one unpadded network flow at any one time.

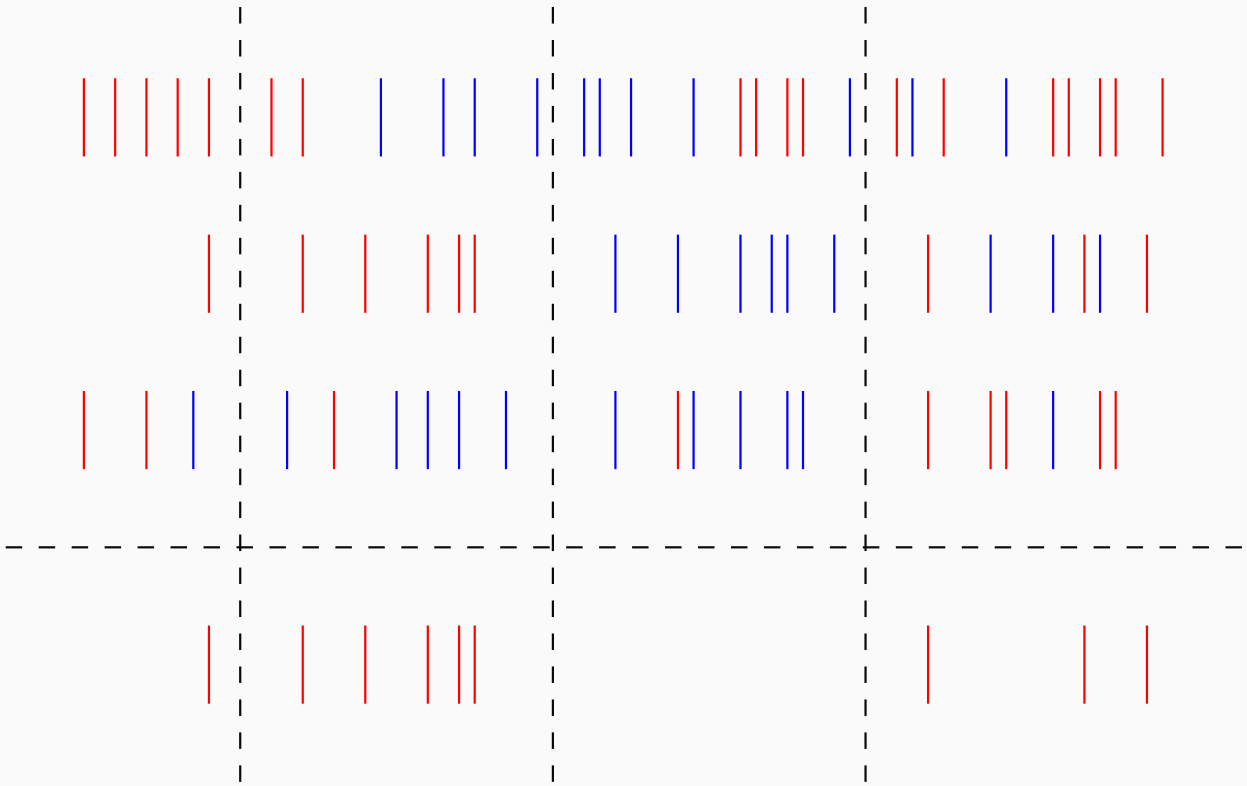
Since AP inserts dummy packets to gaps in a flow, the padded flow is a superset of the original flow.

Given one unpdadded flow and a bunch of candidate padded flows. To find a match:

- split a flow in to windows of time.
- Each padded flow is given a score that corresponds to the number of windows which share an equal number of packets with the unpadded flow.
- Take the score for both incoming and outgoing padded flows, if the highest score of the incoming and outgoing flow is the from the same network traffic flow we consider this padded flow as the match of the unpadded flow.

THE ATTACK

Padded traces

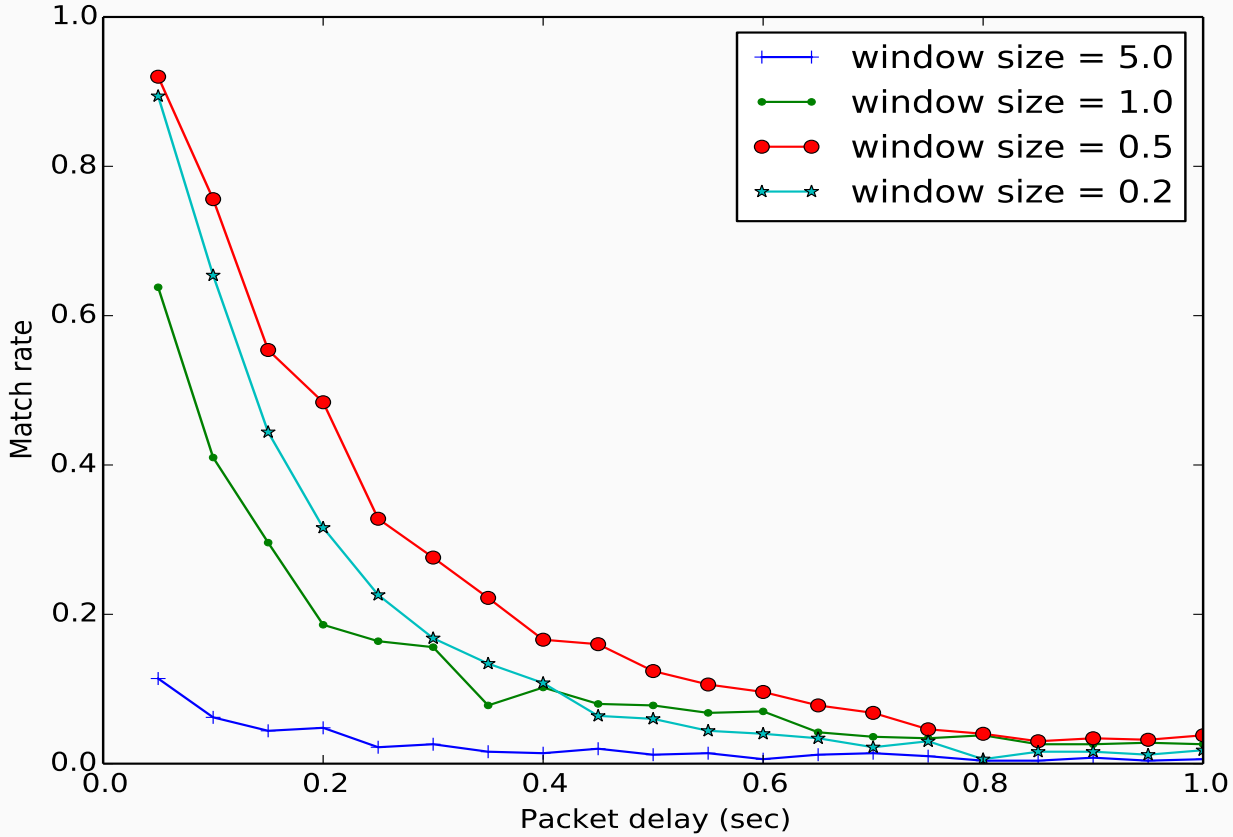


Original trace

= real packet, = dummy packet

RESULTS

- At the experimentally observed IPDV of 21ms, our attack is able to match over 90% of flows correctly.
- We observed almost no false positives.



PRACTICAL IMPLEMENTATION

- AP can only be applied to Tor in its current form in the forward direction.
- Relays do not have the ability to generate multi-hop dummy cells, instead clients must provide the dummy cells.
- The ability for relays to generate dummy cells would also violate Tor's integrity checks since the running digest must be synchronized between client and relays, but a client has no way to check if a relay has generated new dummy cells.
- Before AP can be applied in Tor either a protocol change (realistic) must be applied or web servers must be persuaded to generate dummy cells server side (unrealistic).

SUMMARY

- AP is vulnerable to simple yet powerful timing attacks.
- Any padding scheme that doesn't intentionally add some delay will be vulnerable.
- Our attack only works well for small amounts of jitter and becomes ineffective as the volatility of jitter increases. Adding small random packet delays could foil timing attacks without incurring a high latency overhead.

THANKS

Questions?

j.hayes@cs.ucl.ac.uk or @_jamiedh