Eclipse Attacks on Bitcoin's Peer-to-Peer Network

Ethan Heilman
Computer Science, Boston University
August 2015

Work with Alison Kendler (BU), Aviv Zohar (Hebrew University), & Sharon Goldberg (BU)

BOSTON UNIVERSITY

usenix THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

24th USENIX Security Symposium

Symantec.

Introduction

Bitcoin is often thought to be secure as long as 51% of the mining power is honest,

Introduction

Bitcoin is often thought to be secure as long as 51%
of the mining power is honest,
but this assumes all parties see all valid blocks/transactions.

Bitcoin Consensus

P2P Network

Bitcoin relies on its P2P network to deliver this information.

Introduction

Bitcoin is often thought to be secure as long as 51%
of the mining power is honest,
but this assumes all parties see all valid blocks/transactions.

Bitcoin relies on its P2P network to deliver this information.

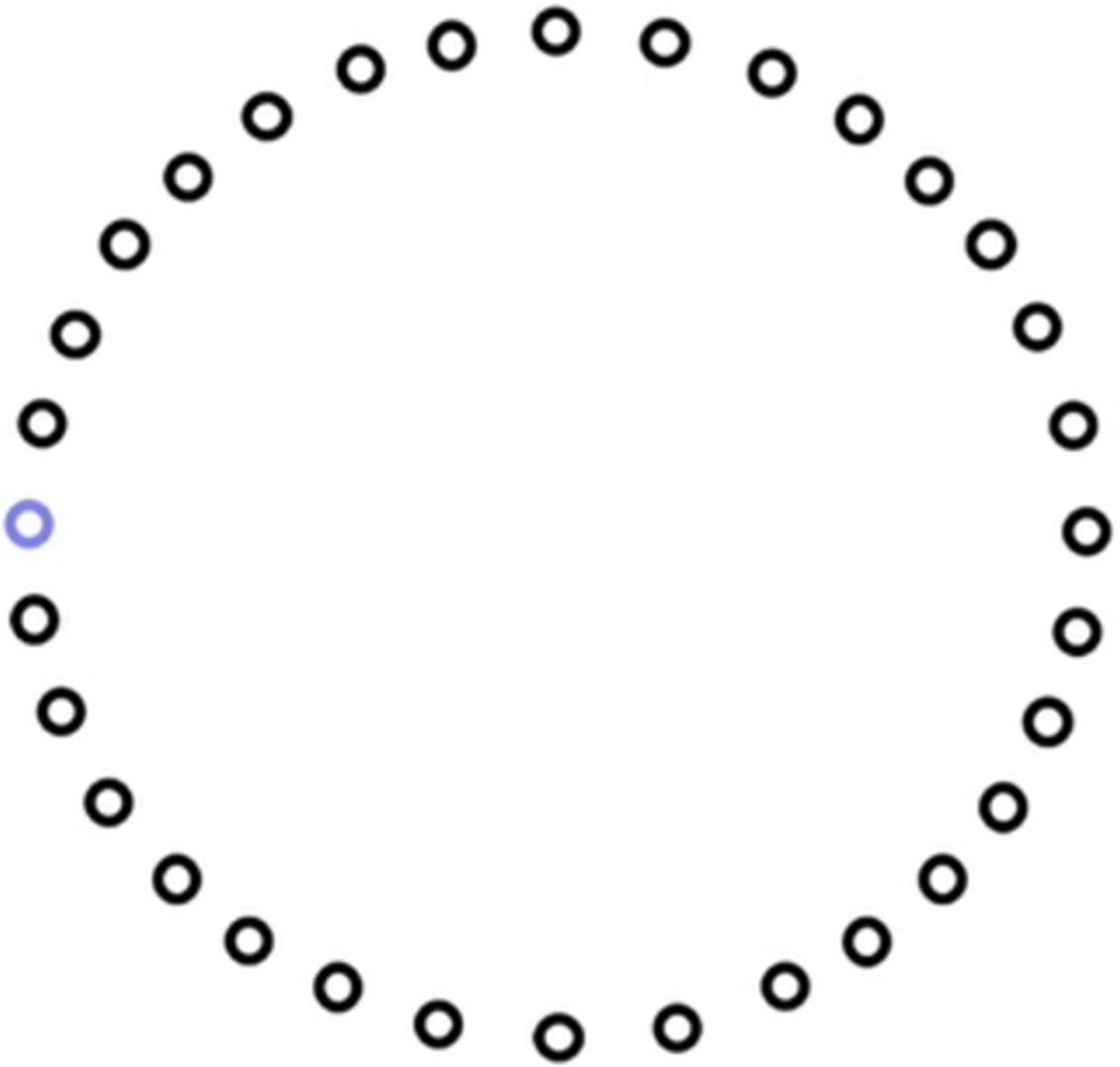
Thus,
control the P2P network → control info flow → control the blockchain.

We attack the P2P network & use info eclipsing to subvert Bitcoin's security.

Outline

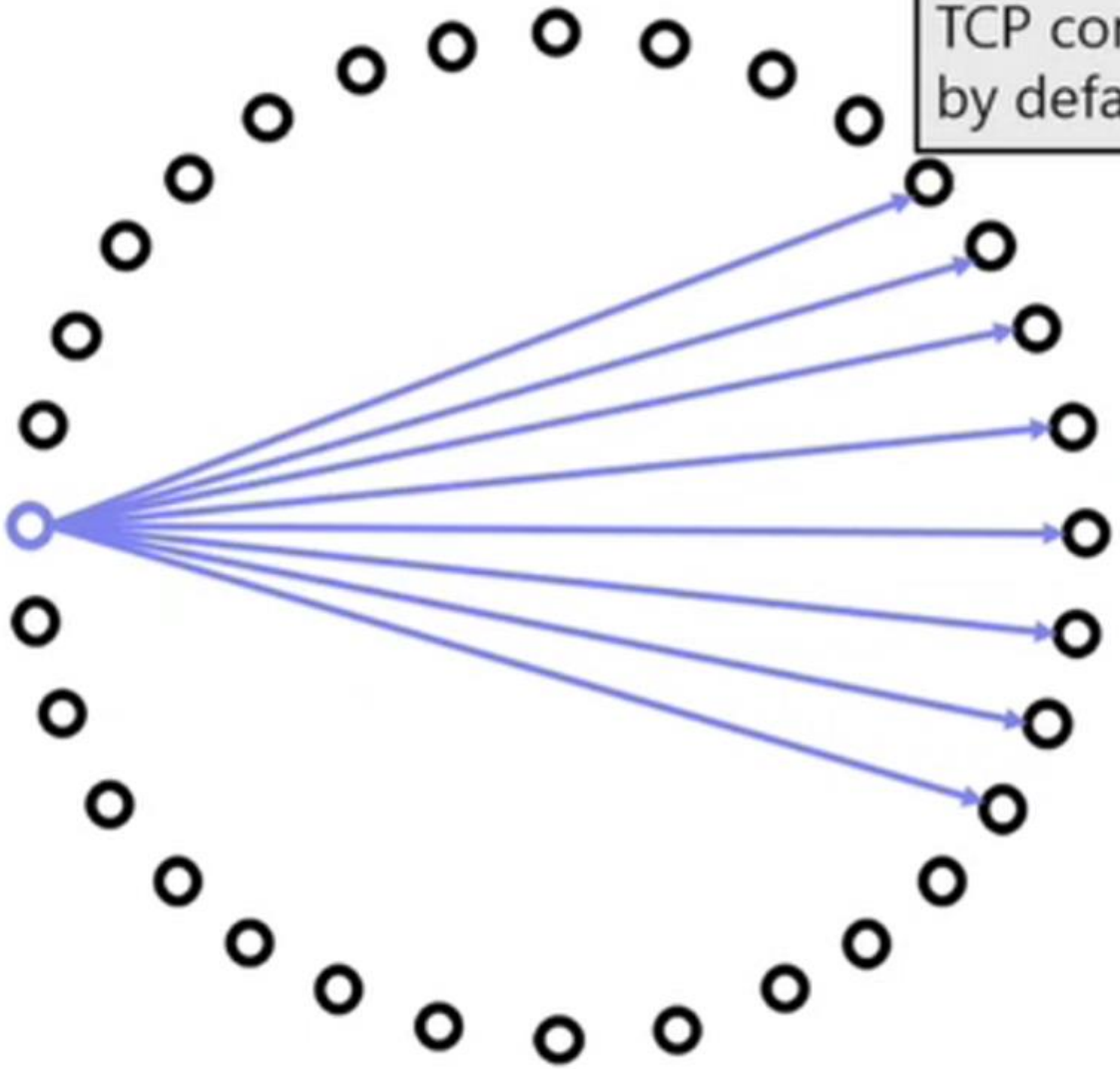
- Eclipse attacks & implications
  - What is an eclipse attack?
  - 51% attacks with far less than 51% of the mining power
  - N-confirmation double spending
- How to eclipse a Bitcoin node
  - P2P network details
  - How to exploit it
- How many IPs does the attacker need?
  - Models & Experimental Results
  - Botnets
- Countermeasures
  - Current deployment
  - Effectiveness of countermeasures

Bitcoin's Peer-to-Peer Network



Bitcoin's Peer-to-Peer Network

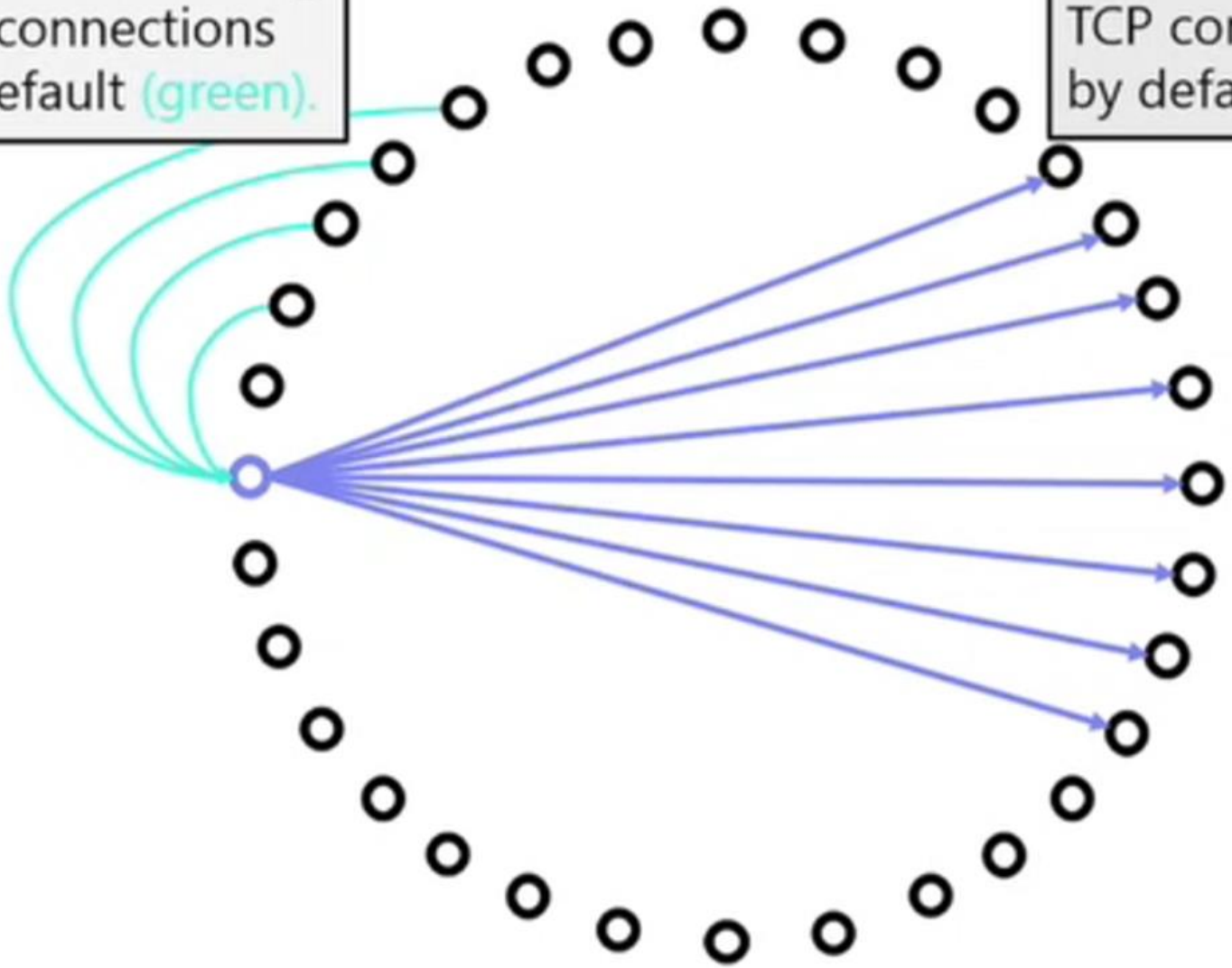
Max 8 outgoing TCP connections by default (purple).



Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

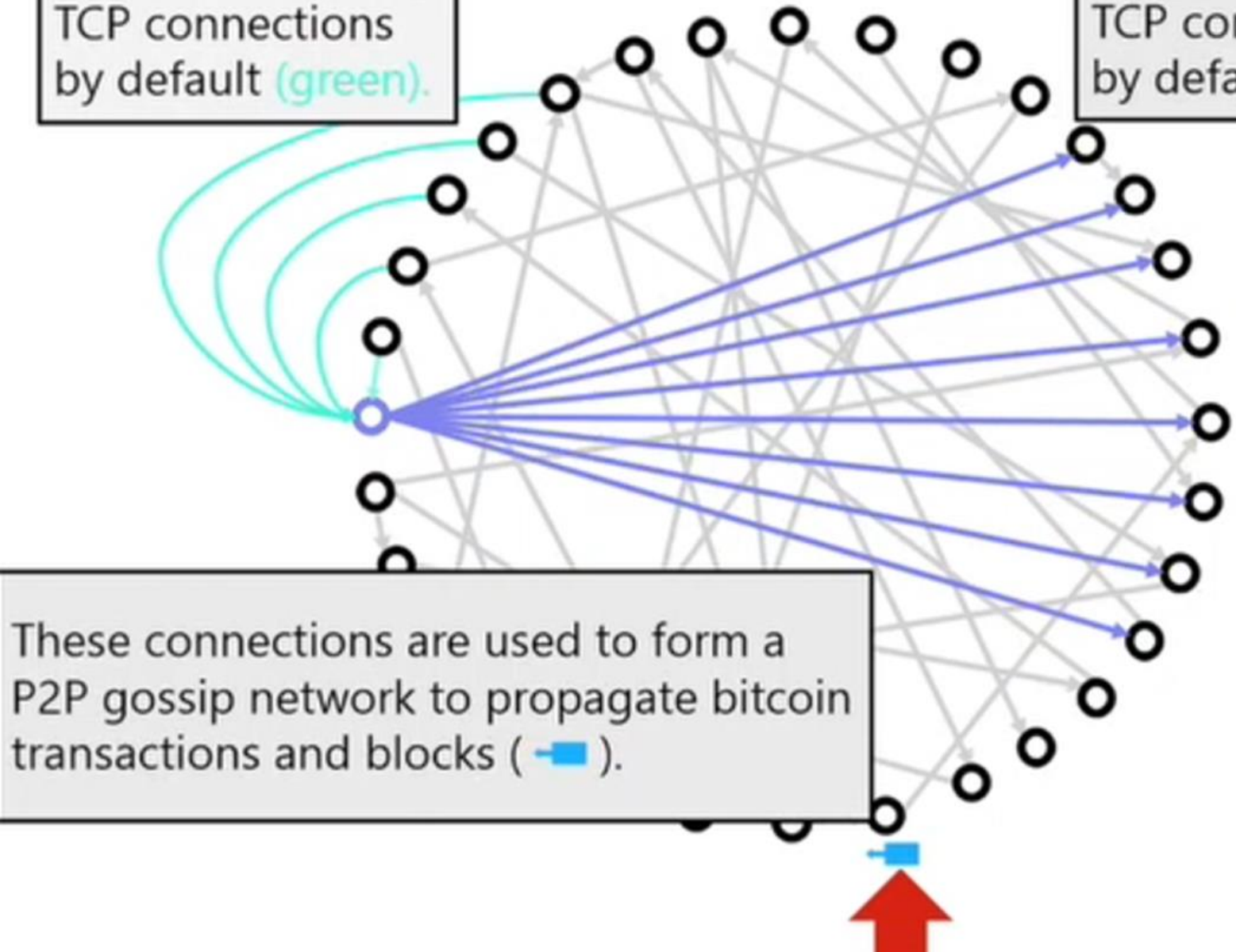
Max 8 outgoing TCP connections by default (purple).



Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

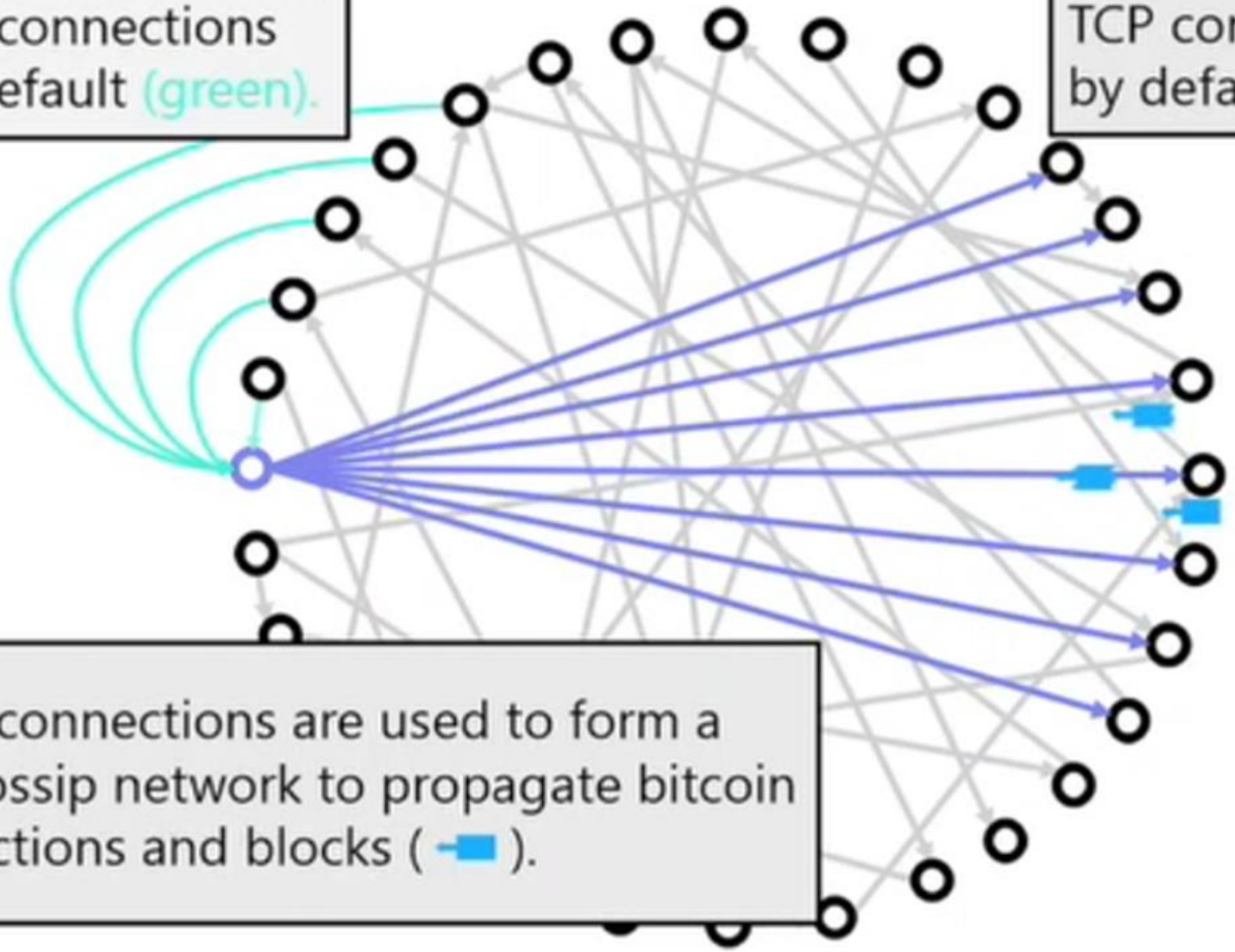


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

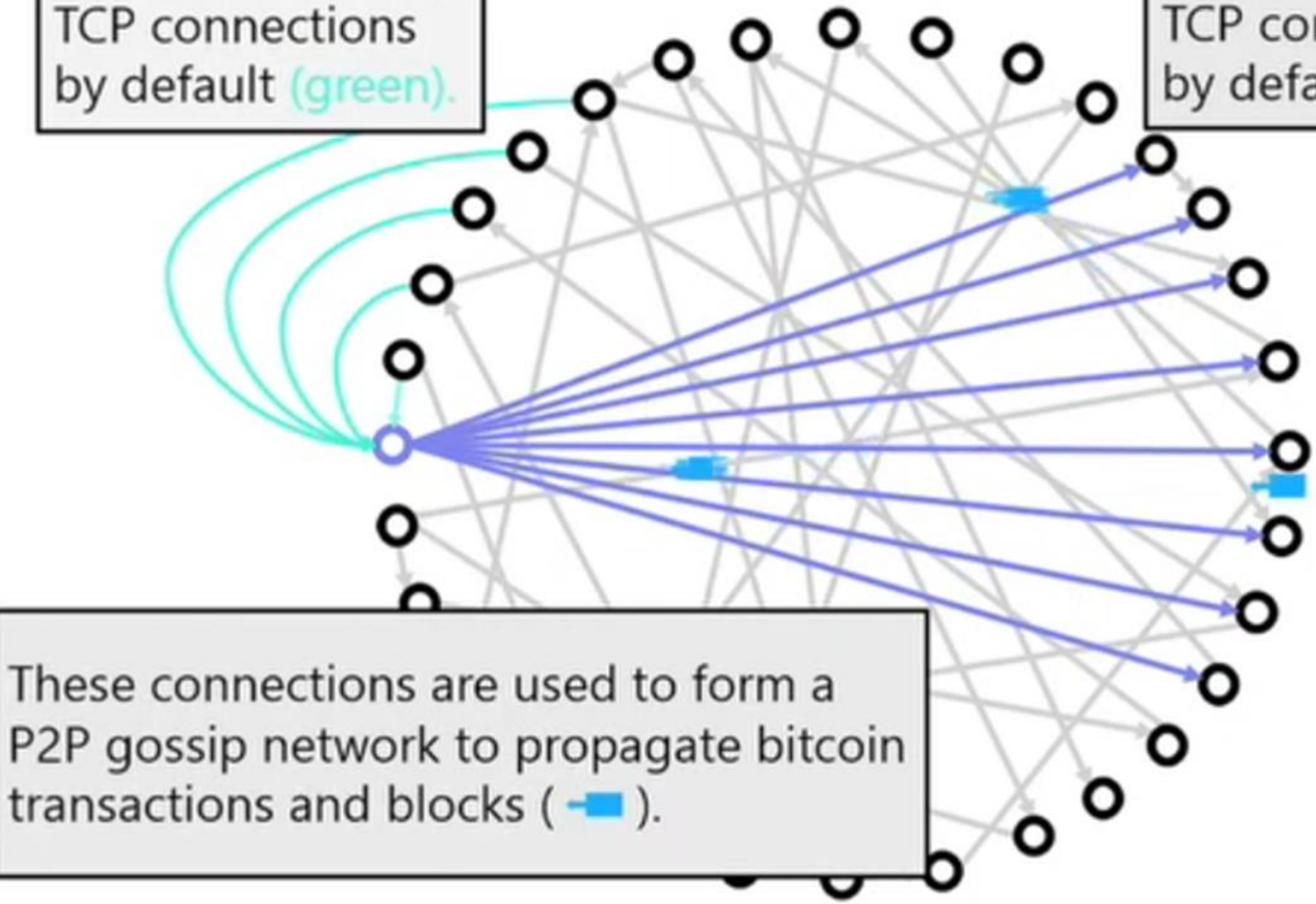


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

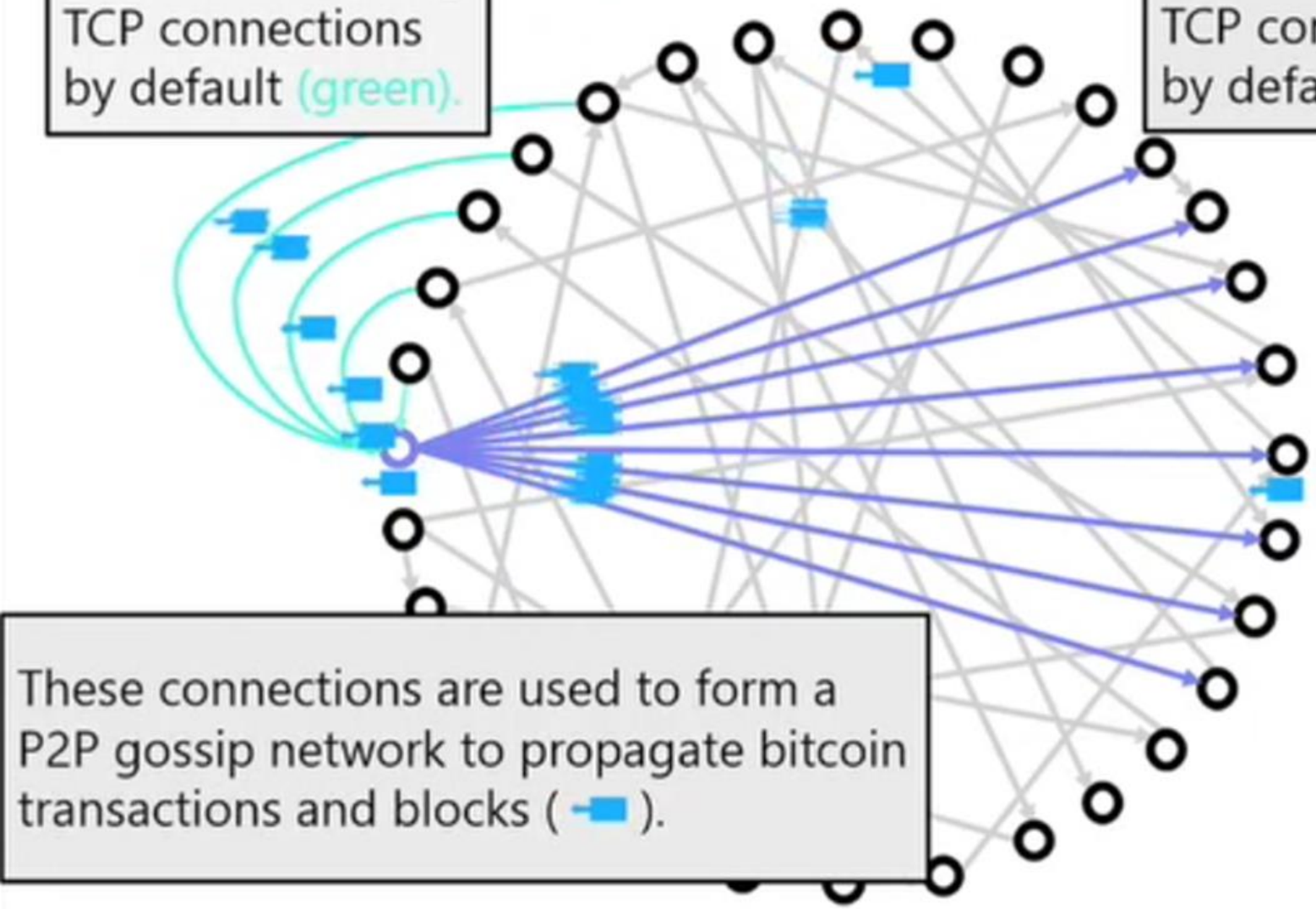


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

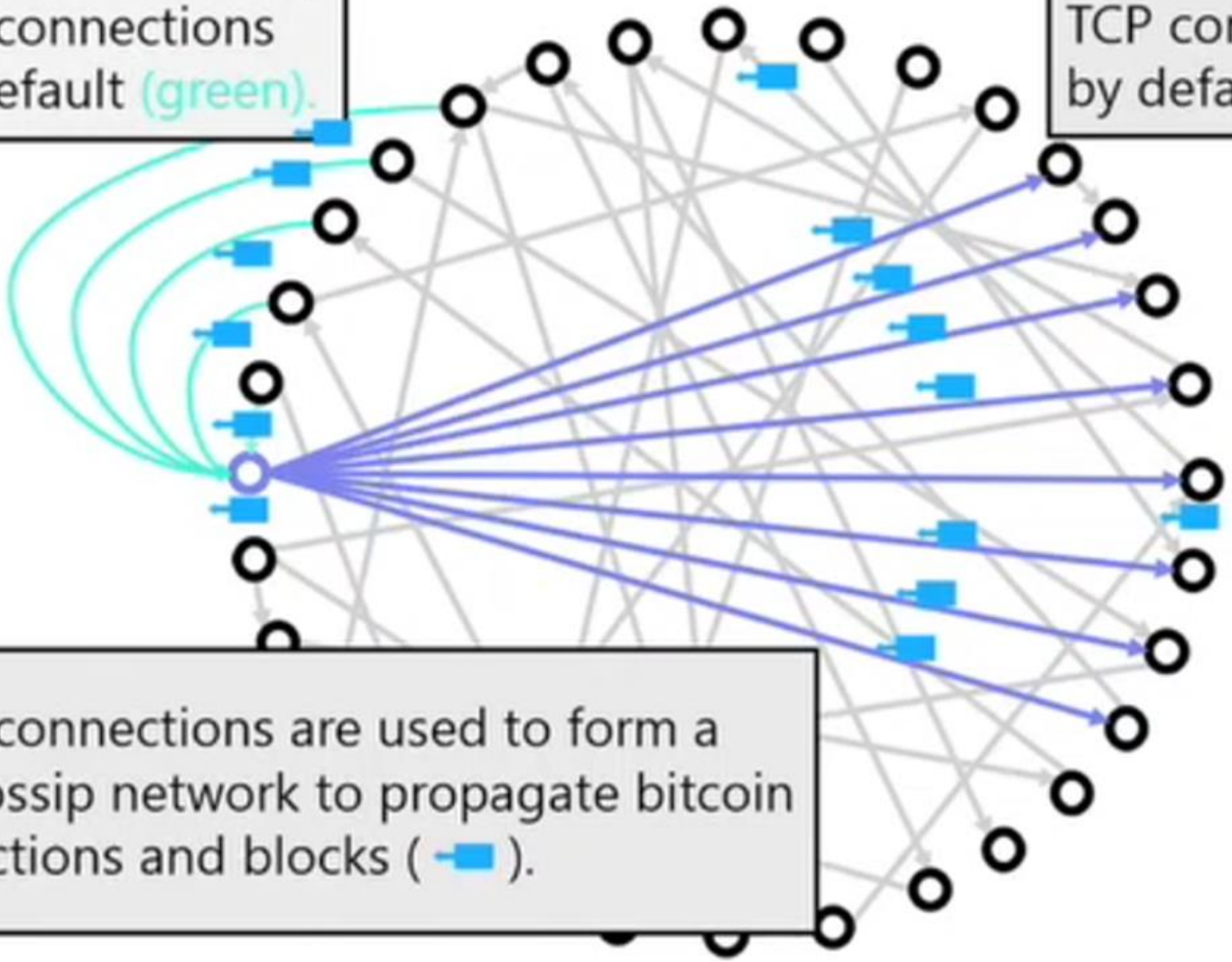


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

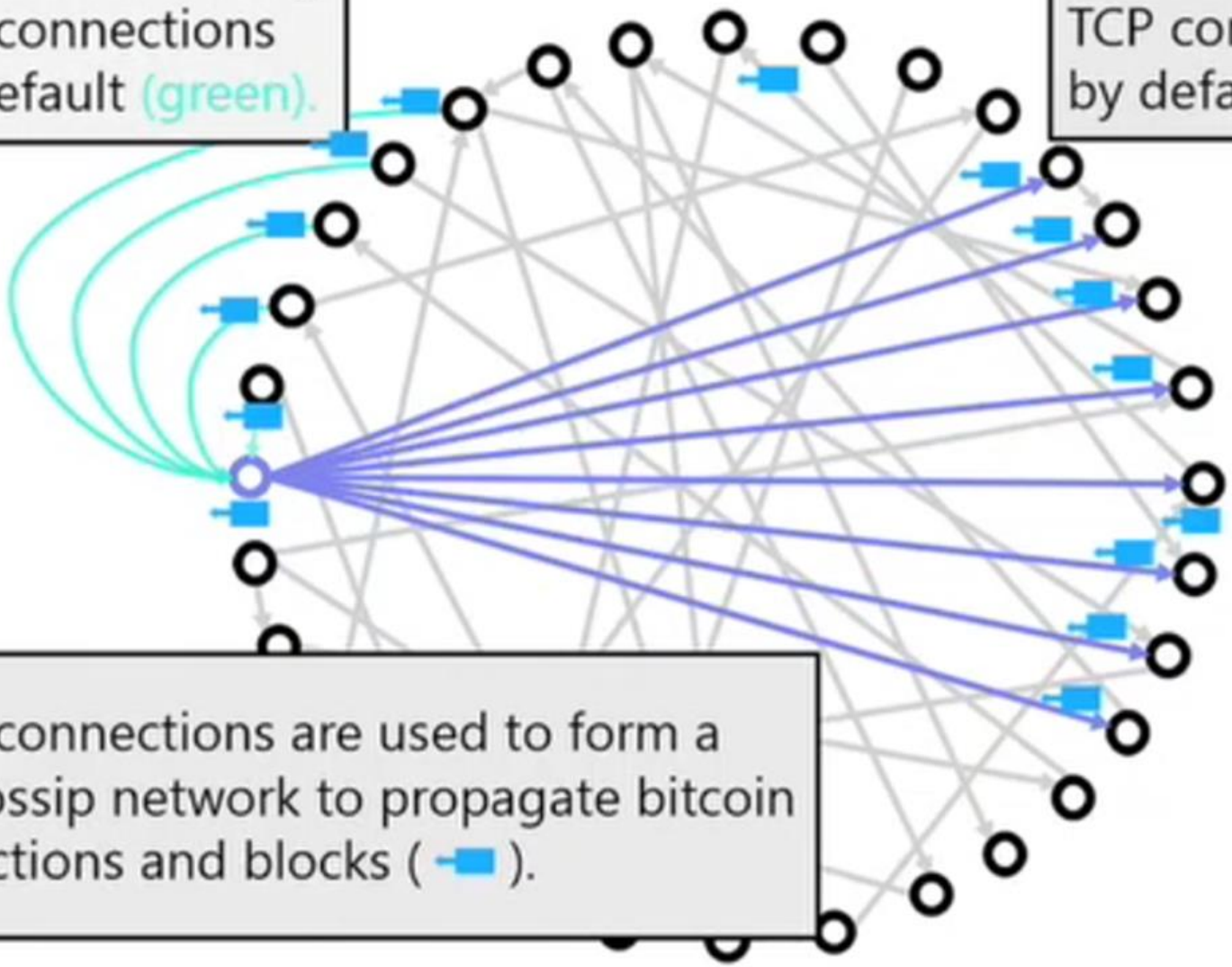


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

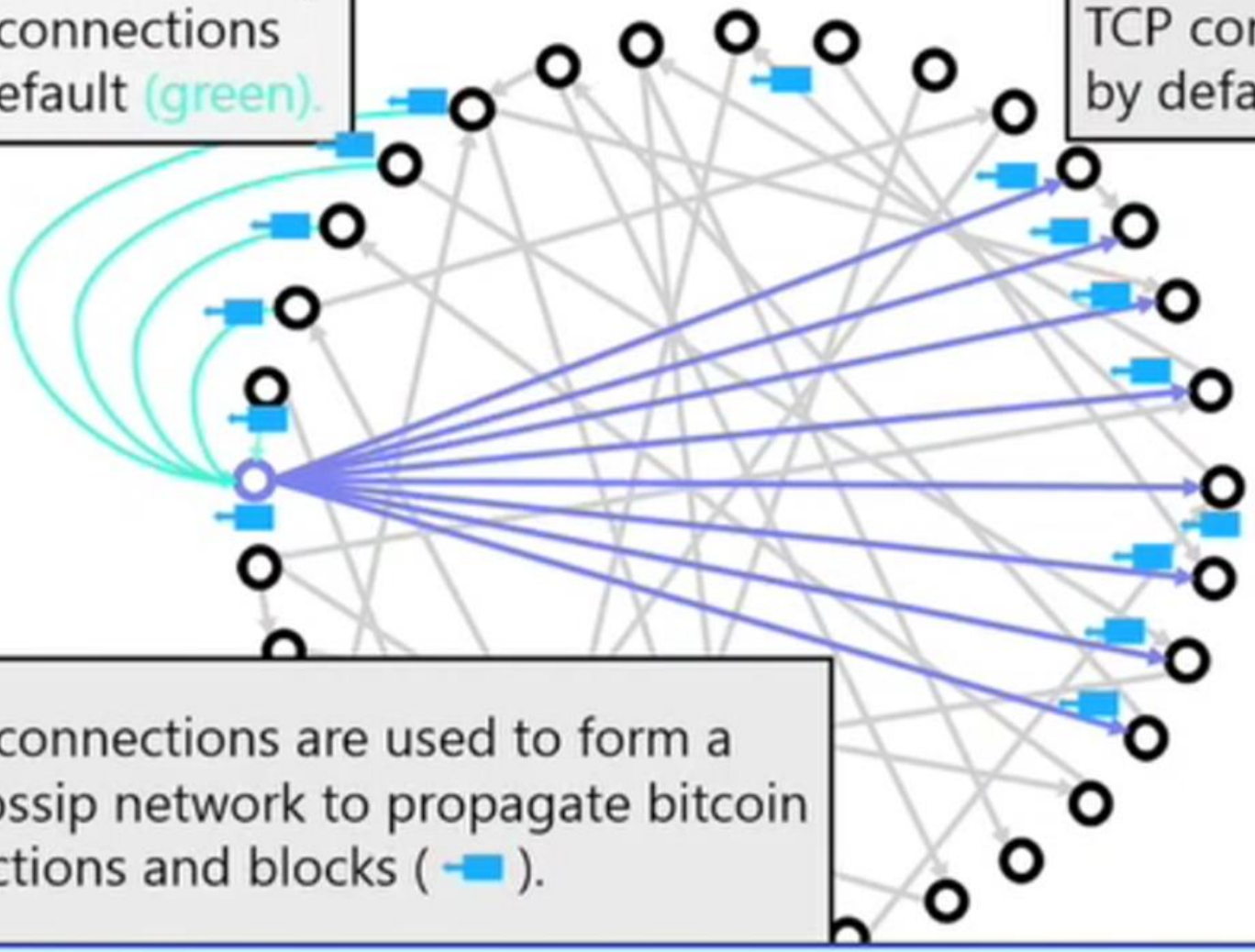


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).

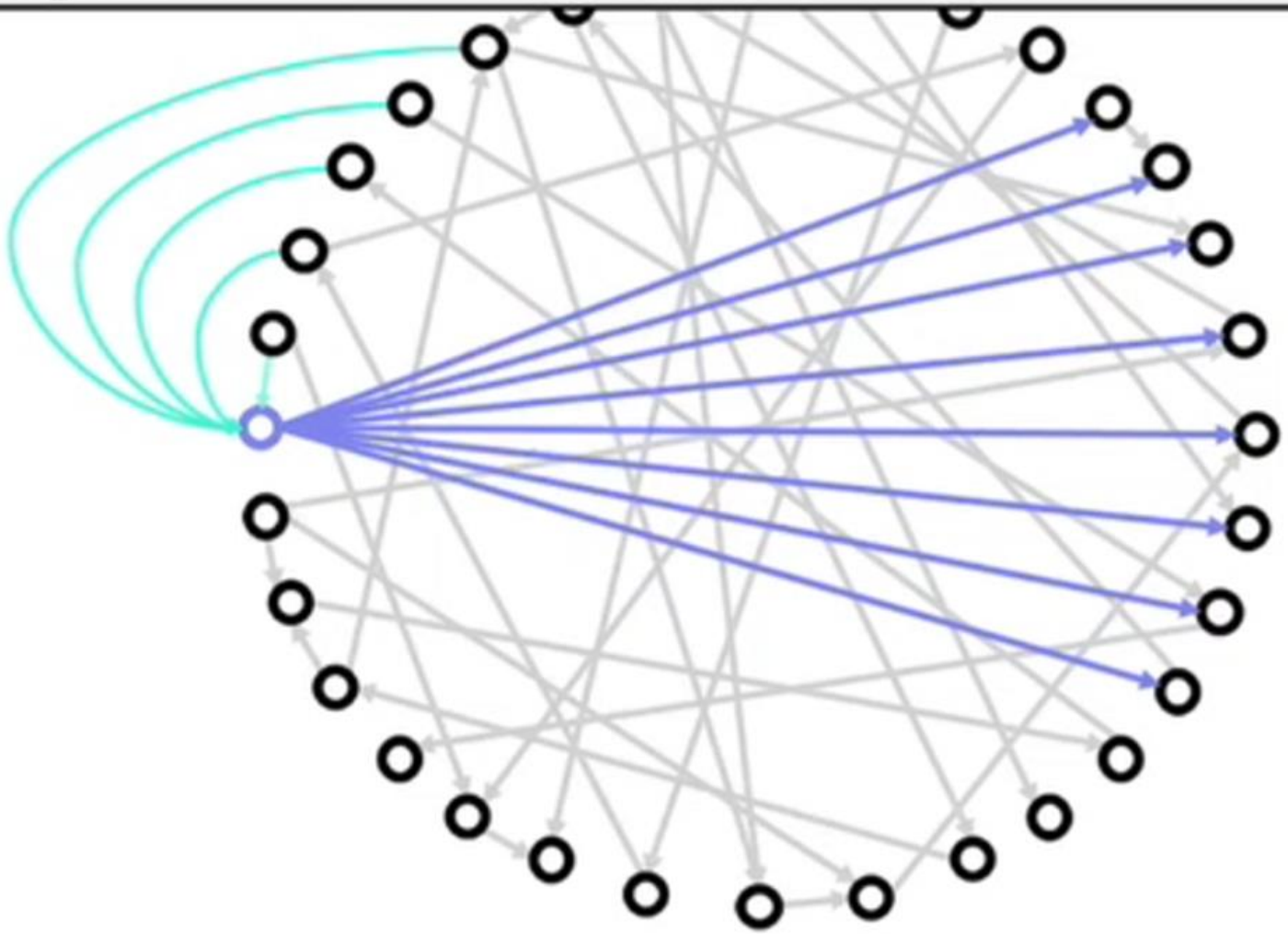


These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks ( ).

Our attack only targets Bitcoin nodes which accept incoming connections, not all nodes accept incoming connections.

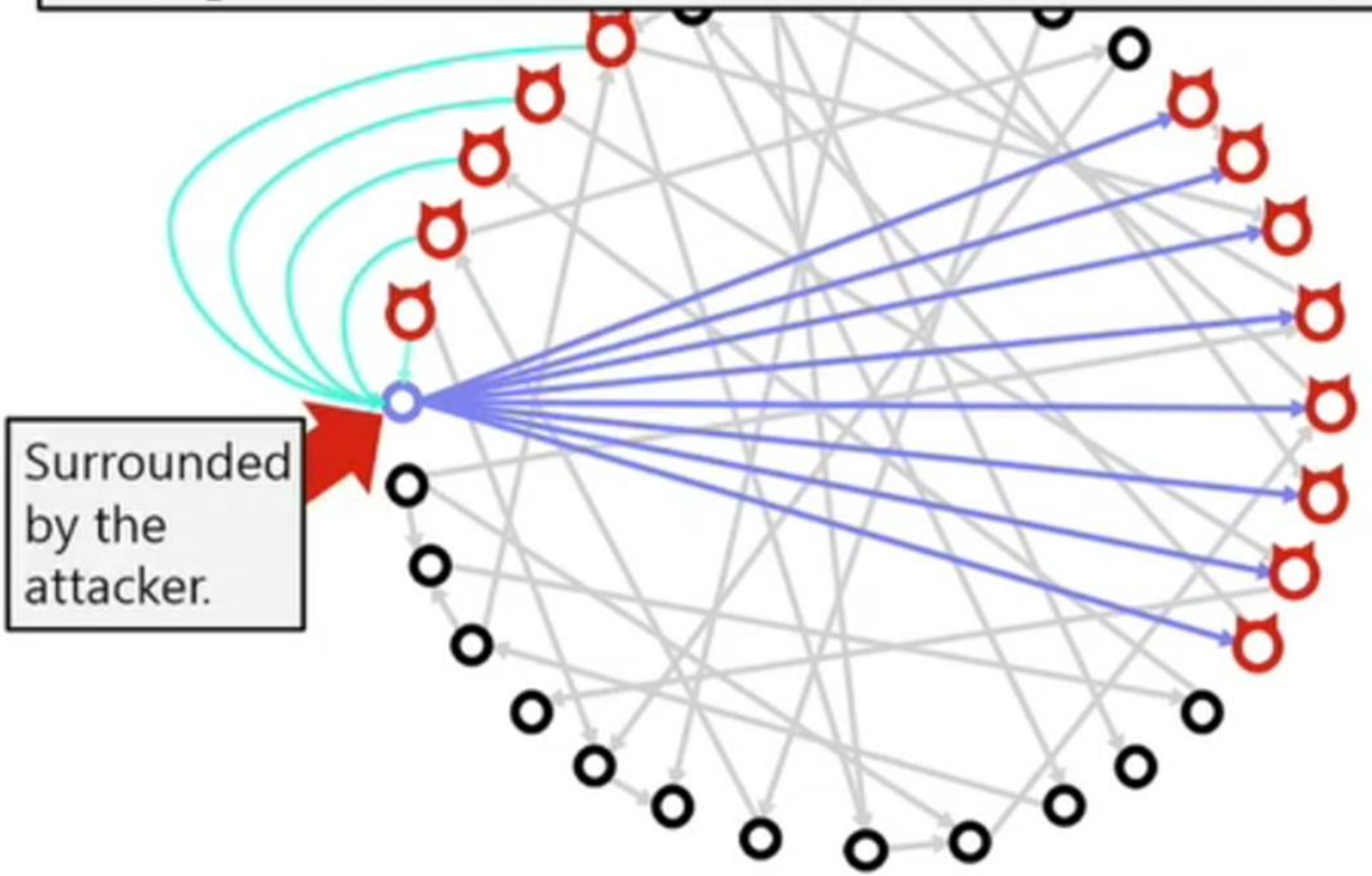
Eclipse Attacks

Information Eclipse Attack (def):
Gaining control over a node's access to information in a P2P network.



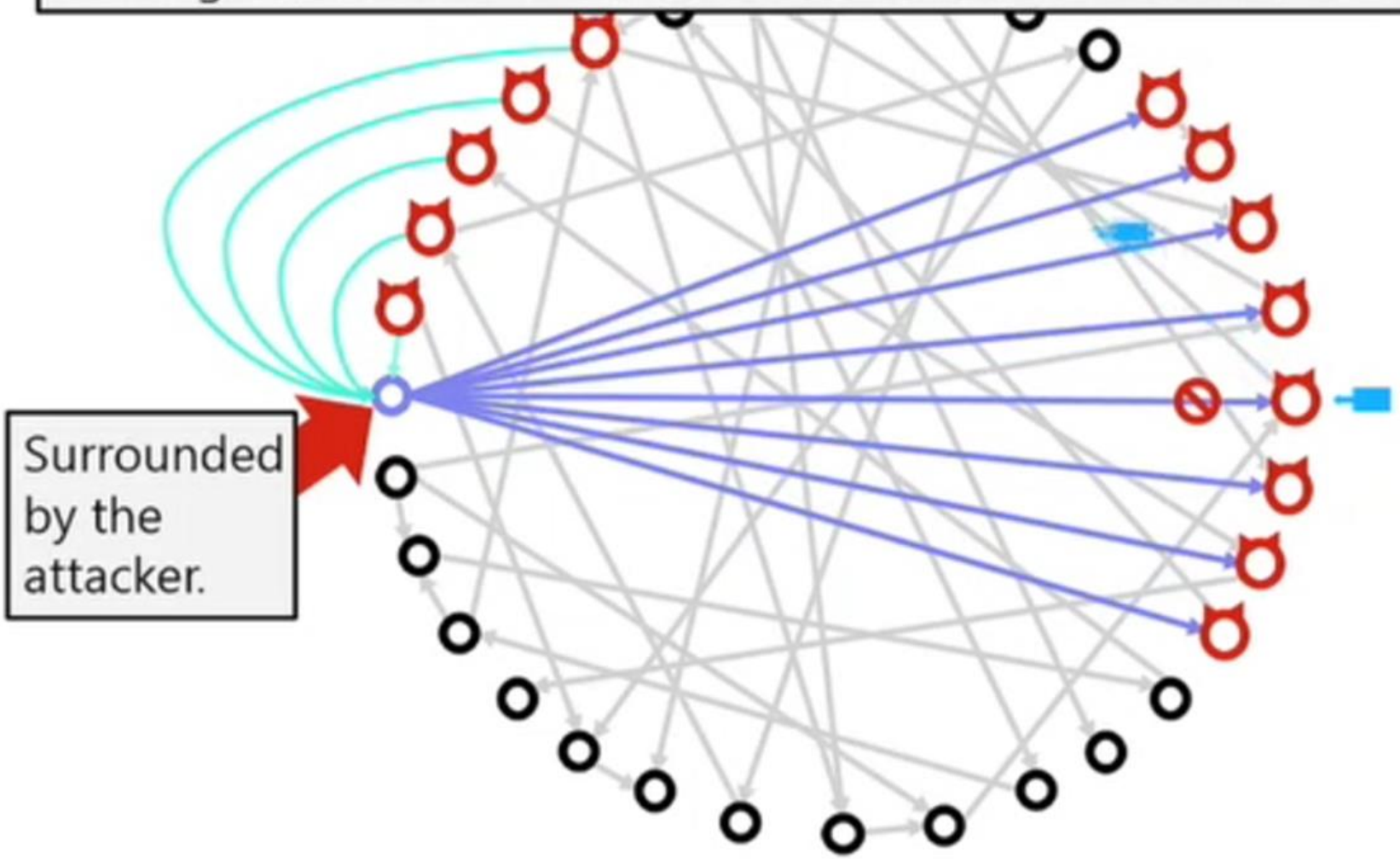
Eclipse Attacks

Information Eclipse Attack (def):
Gaining control over a node's access to information in a P2P network.



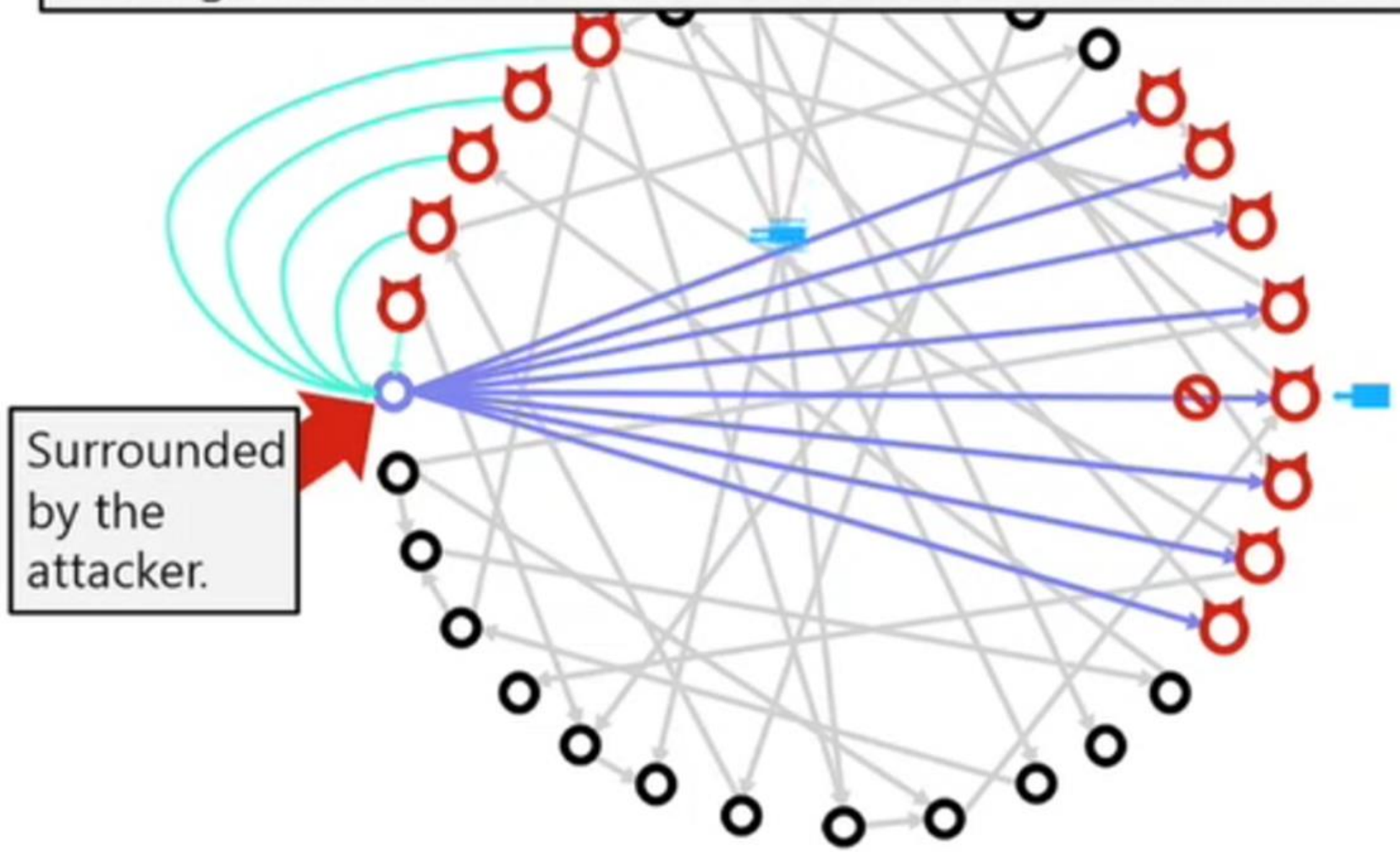
Eclipse Attacks

Information Eclipse Attack (def):
Gaining control over a node's access to information in a P2P network.



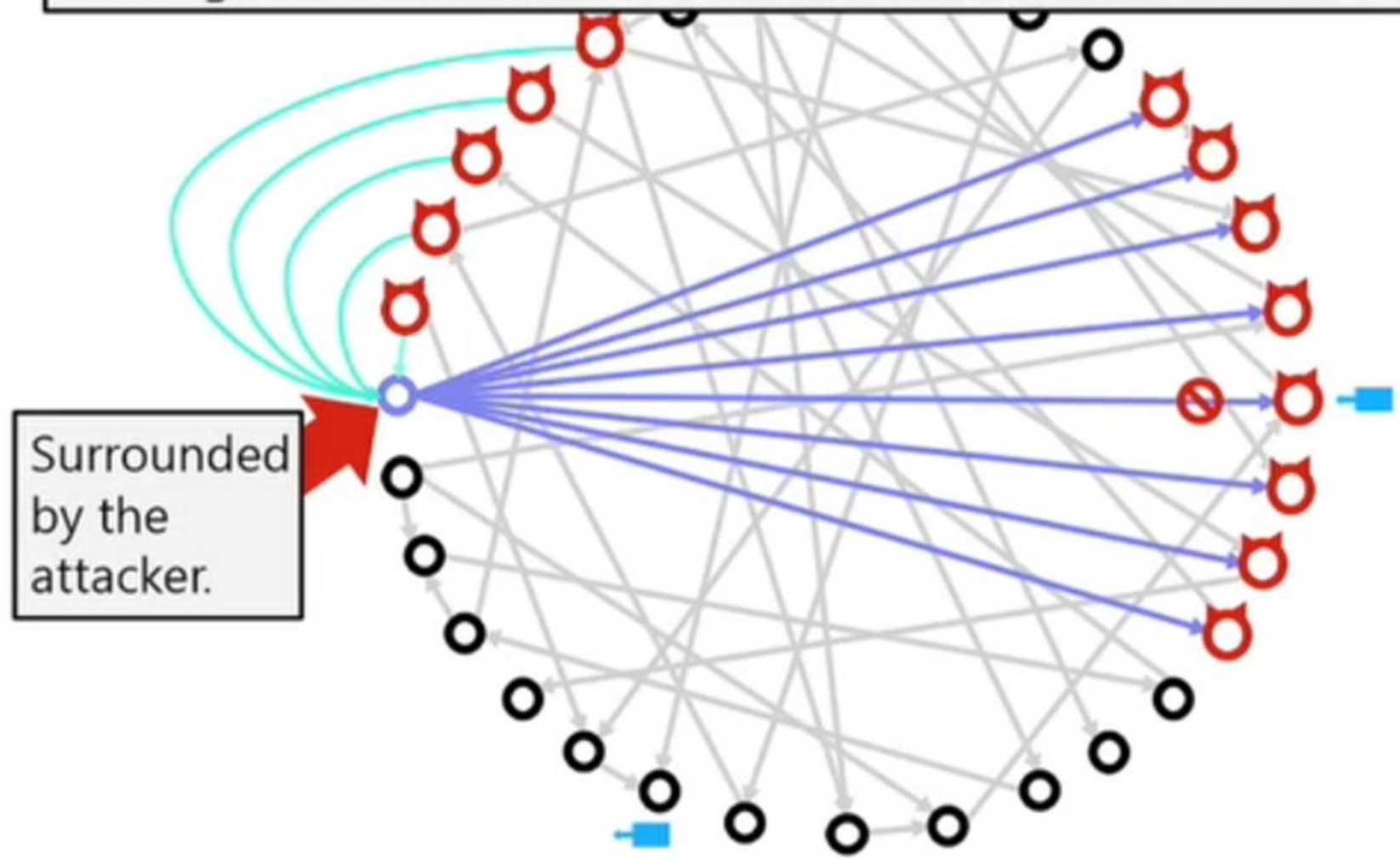
Eclipse Attacks

Information Eclipse Attack (def):
Gaining control over a node's access to information in a P2P network.



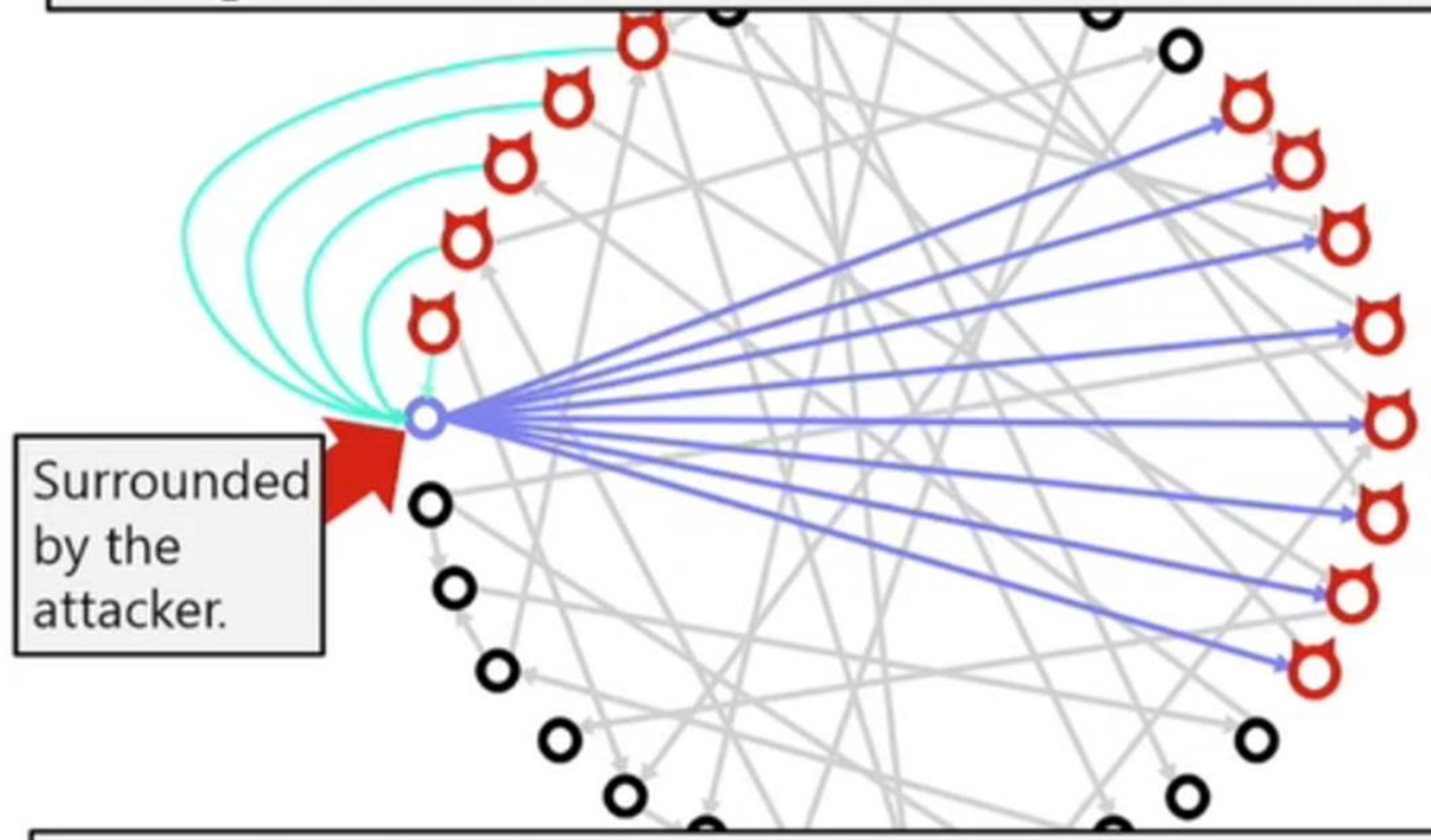
Eclipse Attacks

Information Eclipse Attack (def):
Gaining control over a node's access to information in a P2P network.



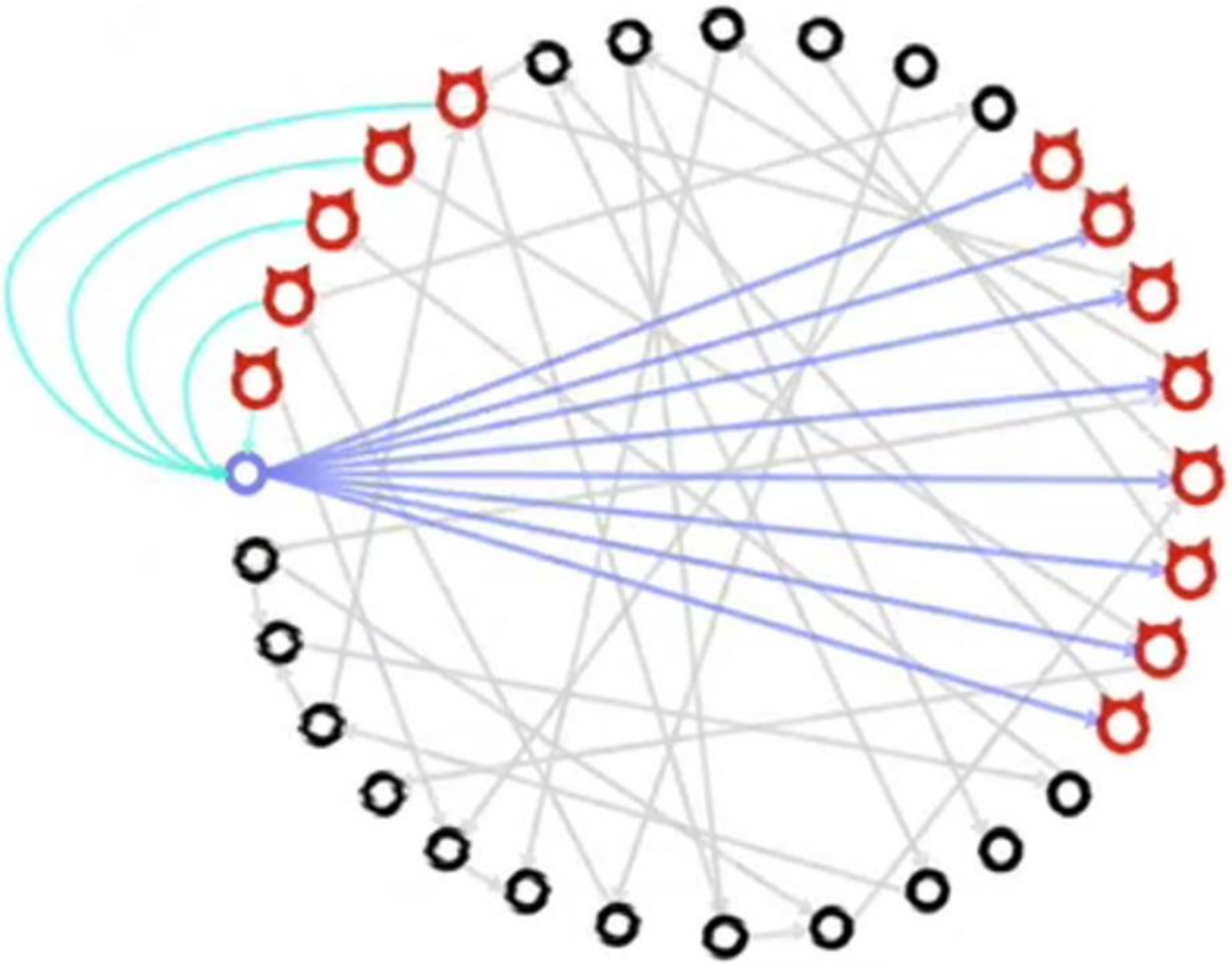
Eclipse Attacks

Information Eclipse Attack (def):
Gaining control over a node's access to information in a P2P network.

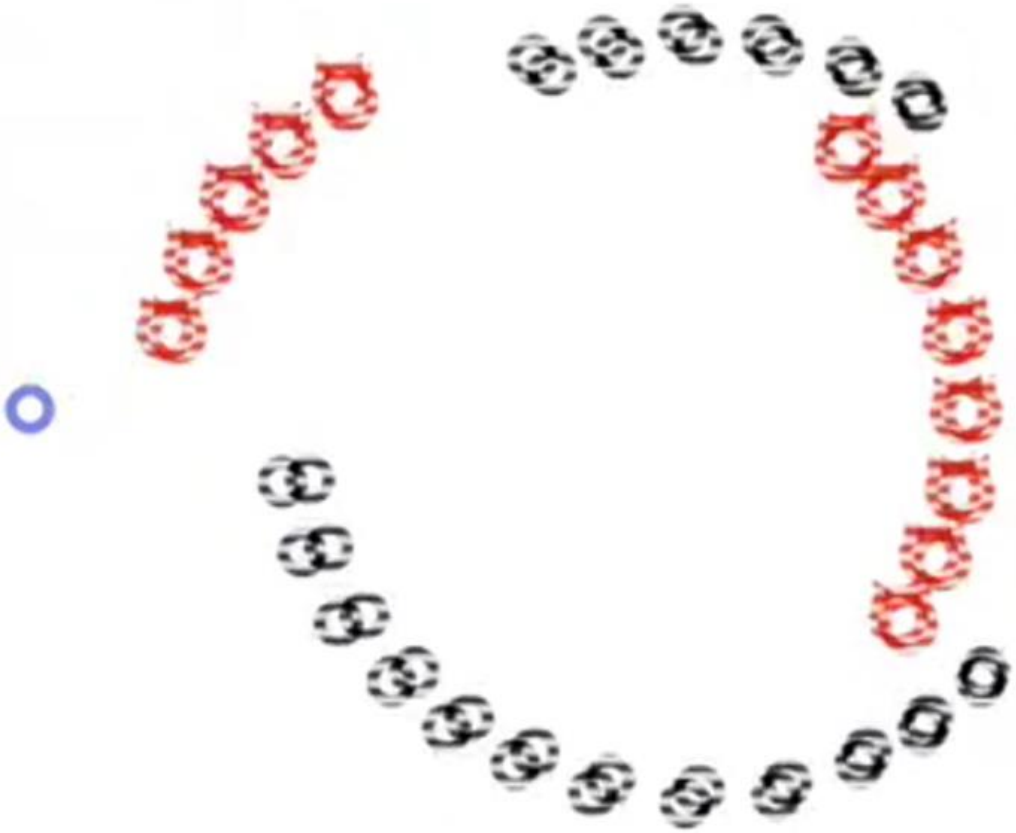


By manipulating the P2P network, the attacker eclipses the node.

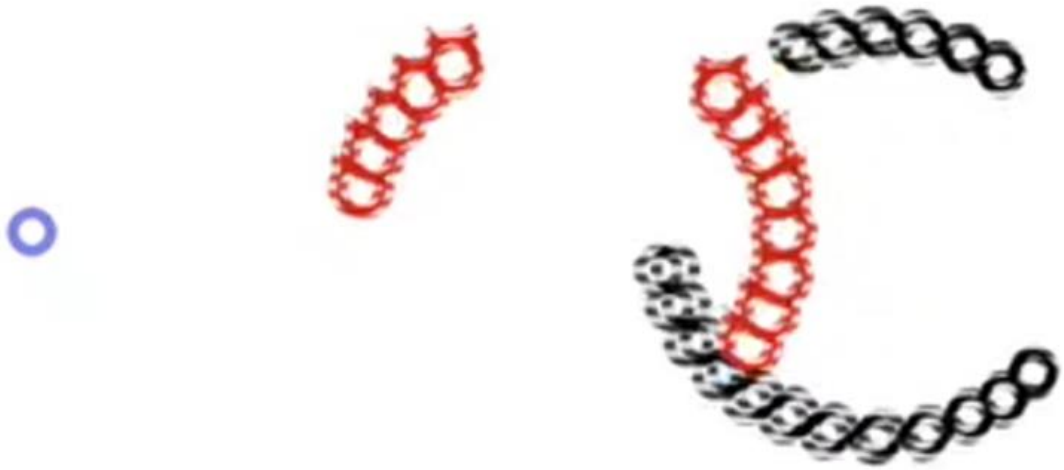
Eclipse Attacks



Eclipse Attacks



Eclipse Attacks



Eclipse Attacks

Eclipse Attacks

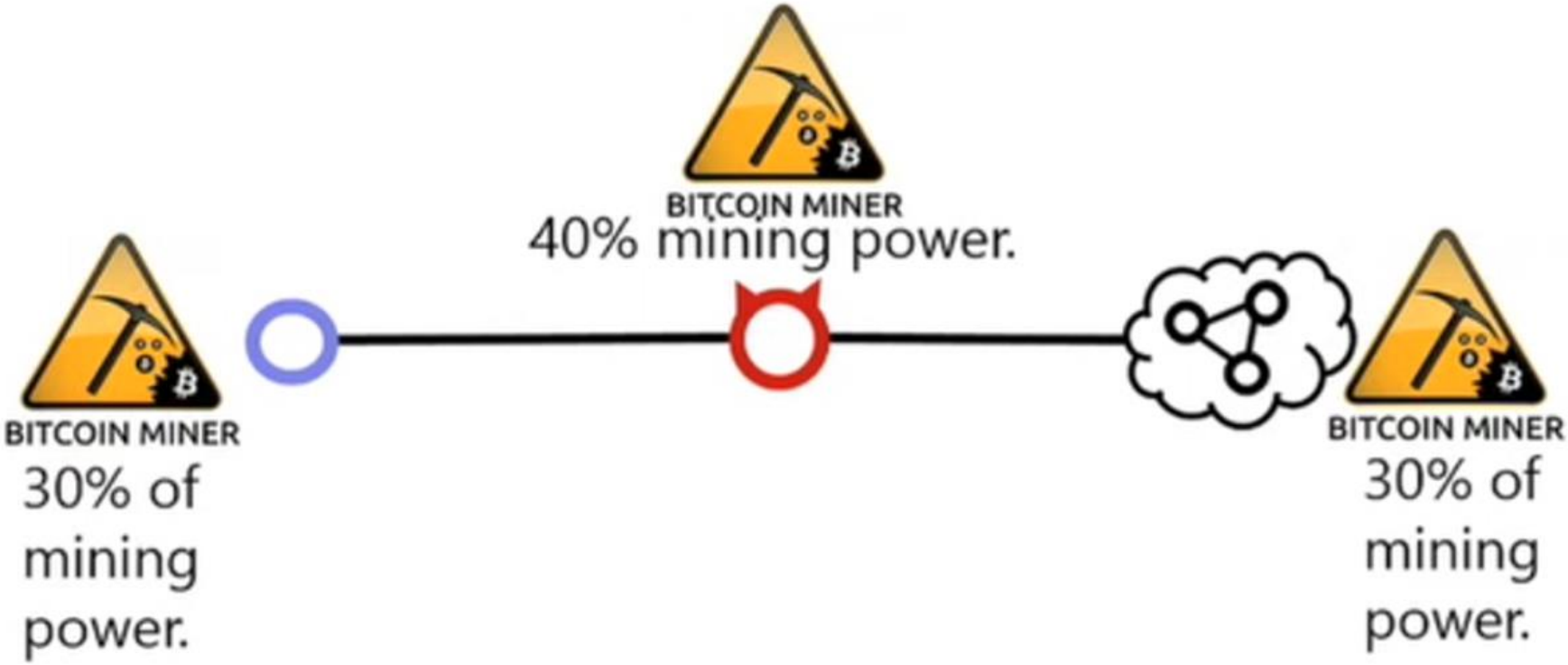
Eclipse Attacks

The attacker is **off-path**,
but all of the nodes P2P connections are made to the attacker.

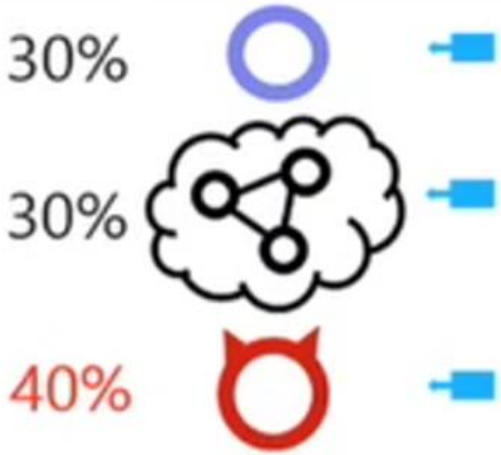
What can an attacker do if they can eclipse a node?

Example 1: 51% attack with 40% mining power

Example 1: 51% attack with 40% mining power

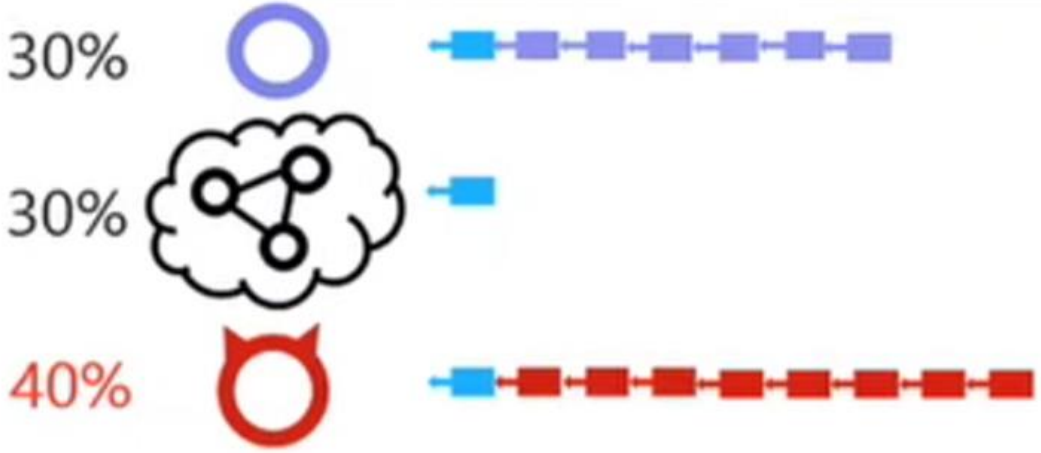


Example 1: 51% attack with 40% mining power



Attacker partitions miners so they can't build on each others blocks.

Example 1: 51% attack with 40% mining power



30% < 40%

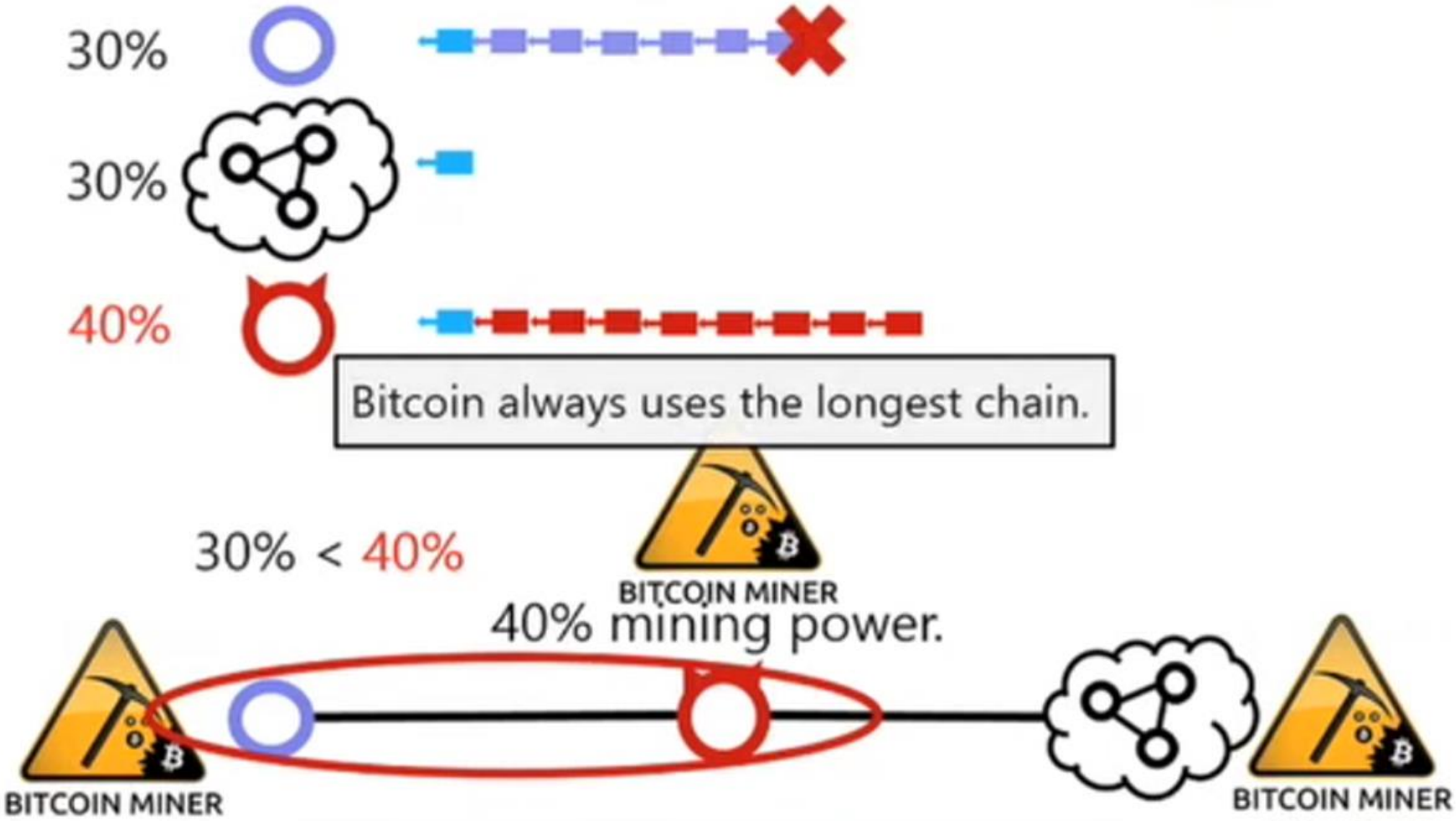
30% of mining power.

Attacker partitions miners so they can't build on each others blocks.

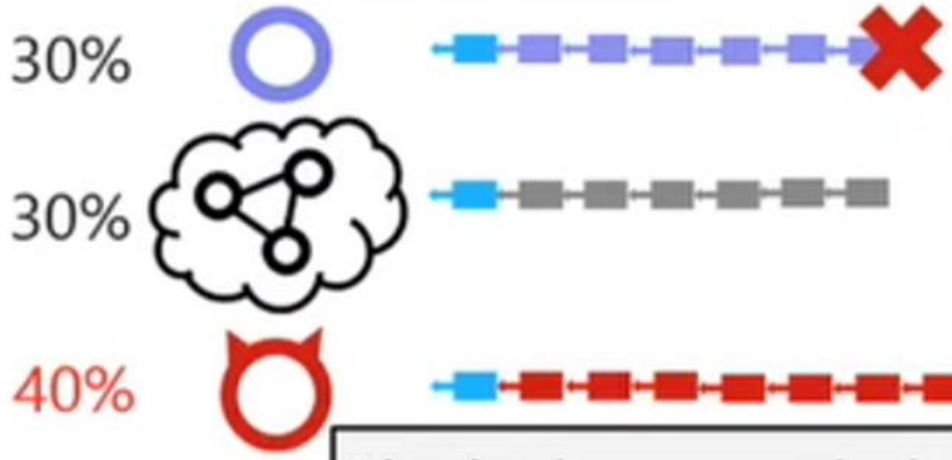
30% of mining power.

The attacker then outcompetes each partitioned miner.

Example 1: 51% attack with 40% mining power



Example 1: 51% attack with 40% mining power



Bitcoin always uses the longest chain.

30% < 40%

40% > 30%

BITCOIN MINER

40% mining power.

BITCOIN MINER

30% of mining power.

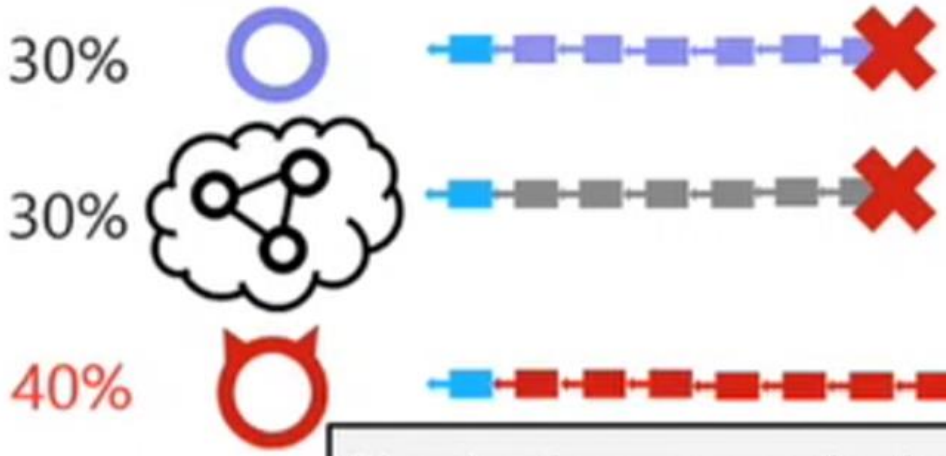
BITCOIN MINER

30% of mining power.

Attacker partitions miners so they can't build on each others blocks.

The attacker then outcompetes each partitioned miner.

Example 1: 51% attack with 40% mining power



Attacker chain becomes the consensus chain.

Bitcoin always uses the longest chain.

30% < 40%

40% > 30%

BITCOIN MINER

40% mining power.

BITCOIN MINER

30% of mining power.

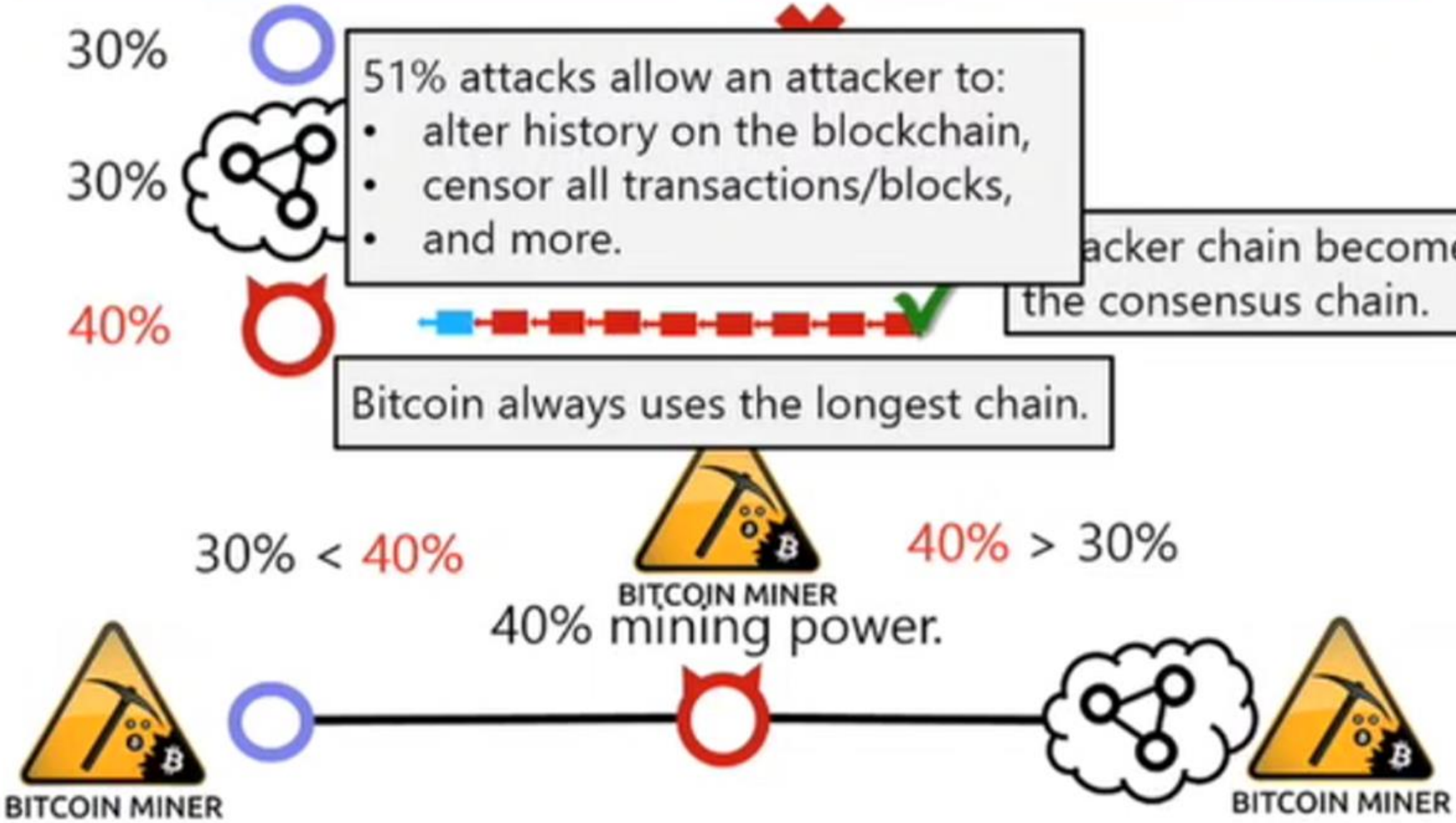
BITCOIN MINER

30% of mining power.

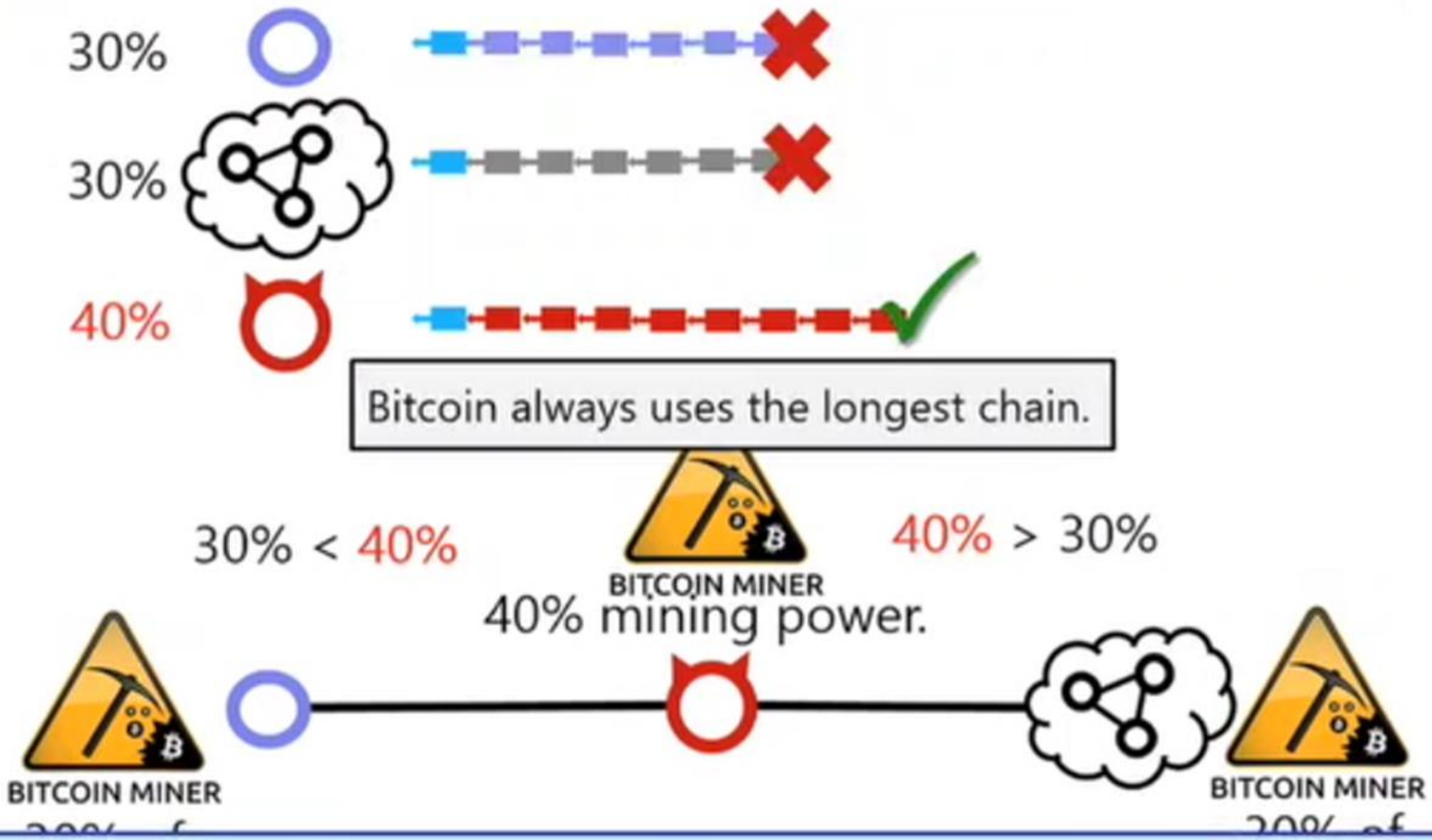
Attacker partitions miners so they can't build on each others blocks.

The attacker then outcompetes each partitioned miner.

Example 1: 51% attack with 40% mining power



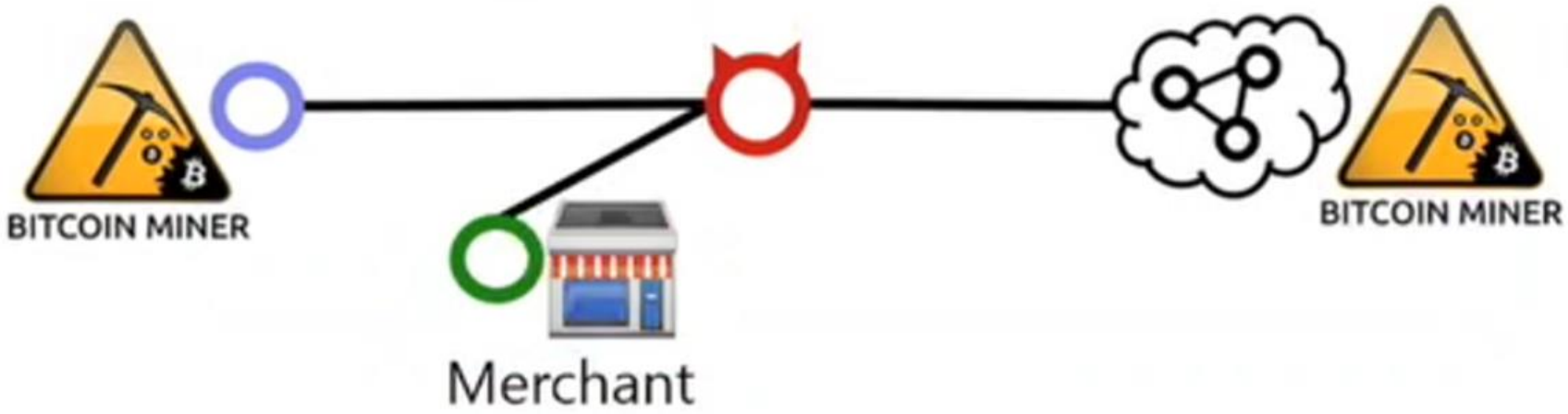
Example 1: 51% attack with 40% mining power



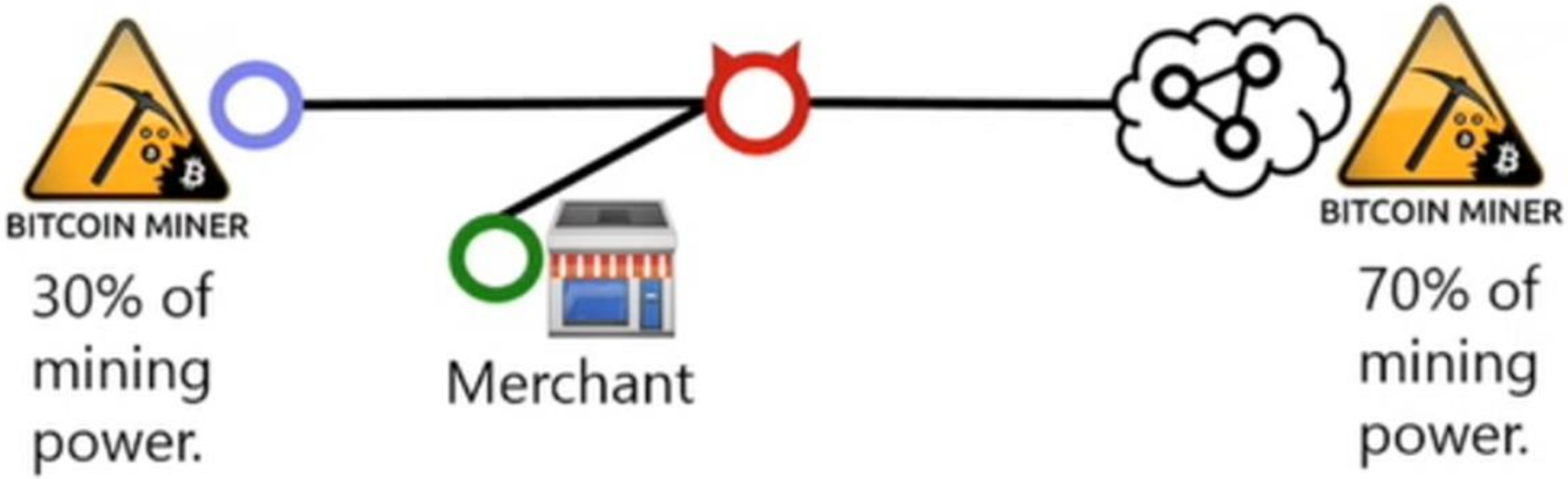
An eclipsing attacker with mining power can also perform an improved selfish mining attack[1].

[1]: Eyal, I., Majority is not enough: Bitcoin mining is vulnerable. (2014)

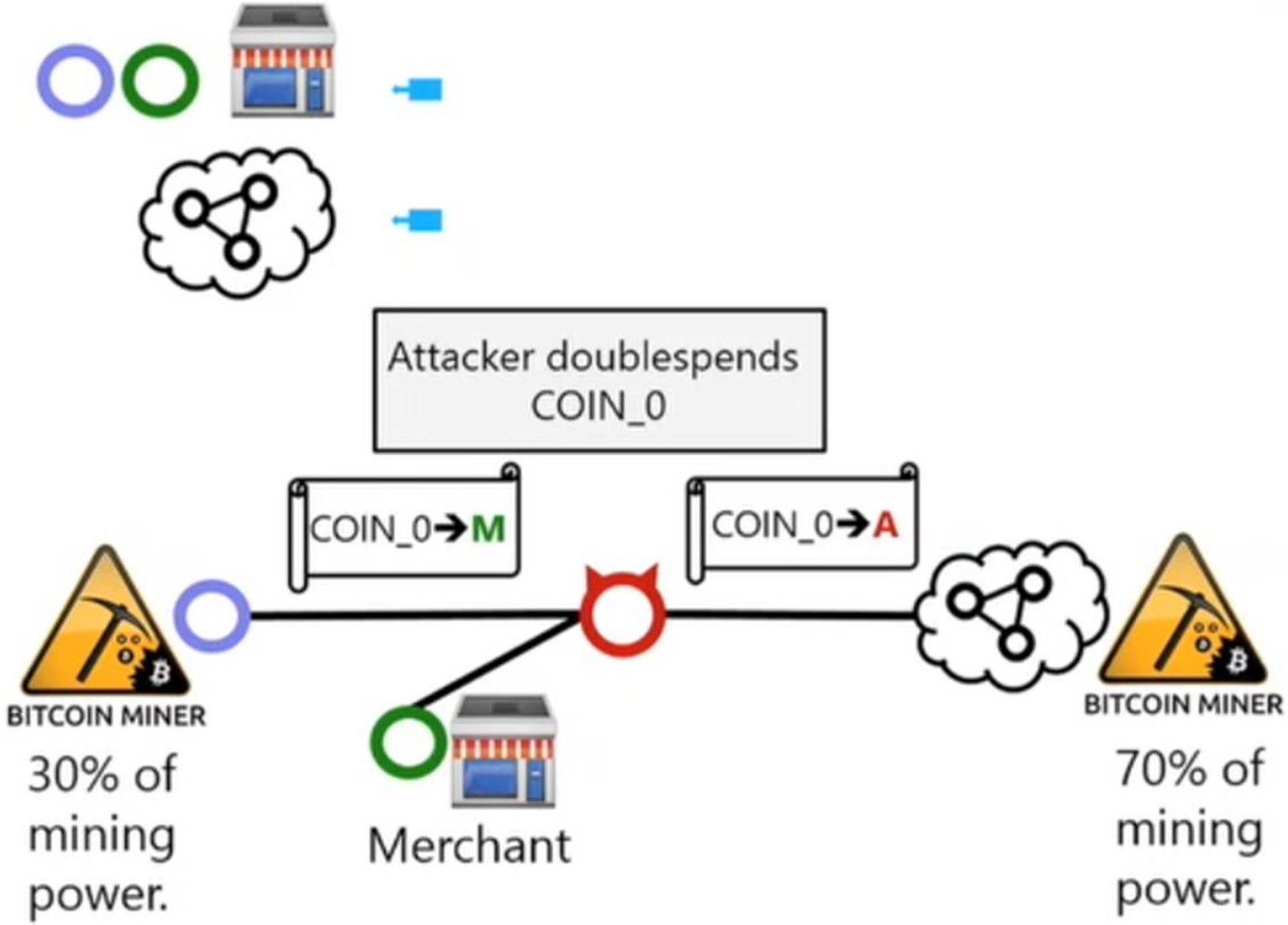
Example 2: N-Confirmation Double Spending



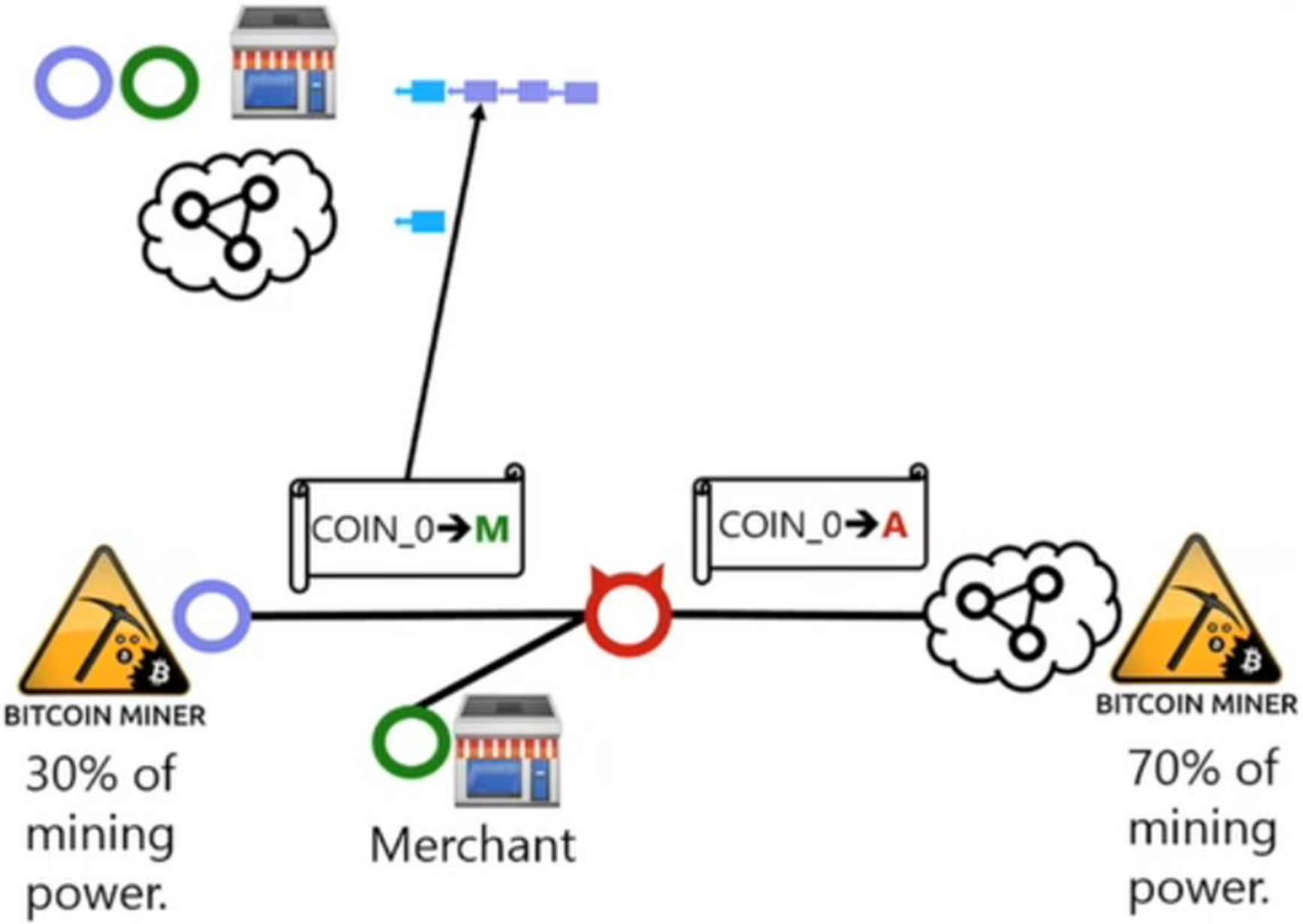
Example 2: N-Confirmation Double Spending



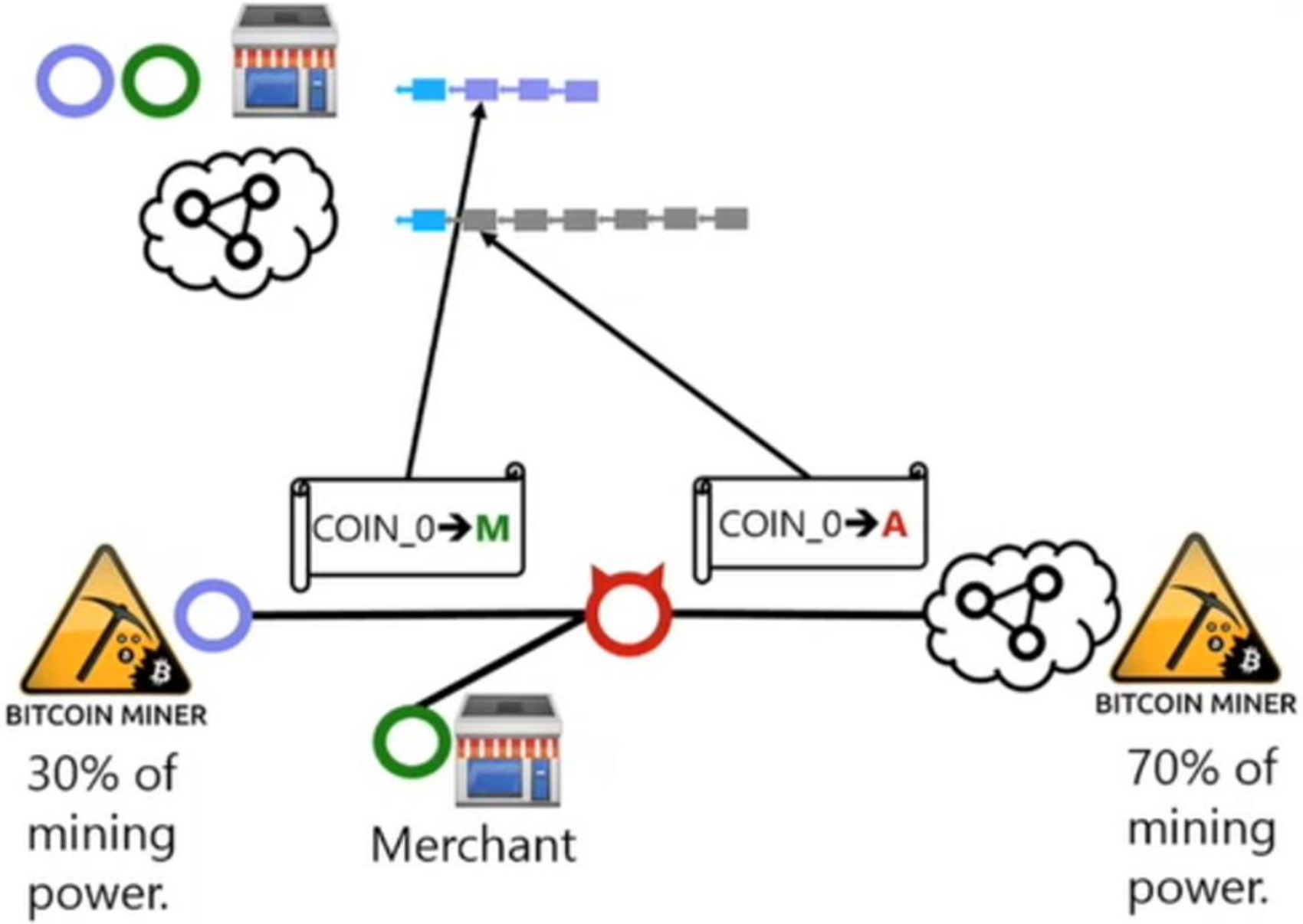
Example 2: N-Confirmation Double Spending



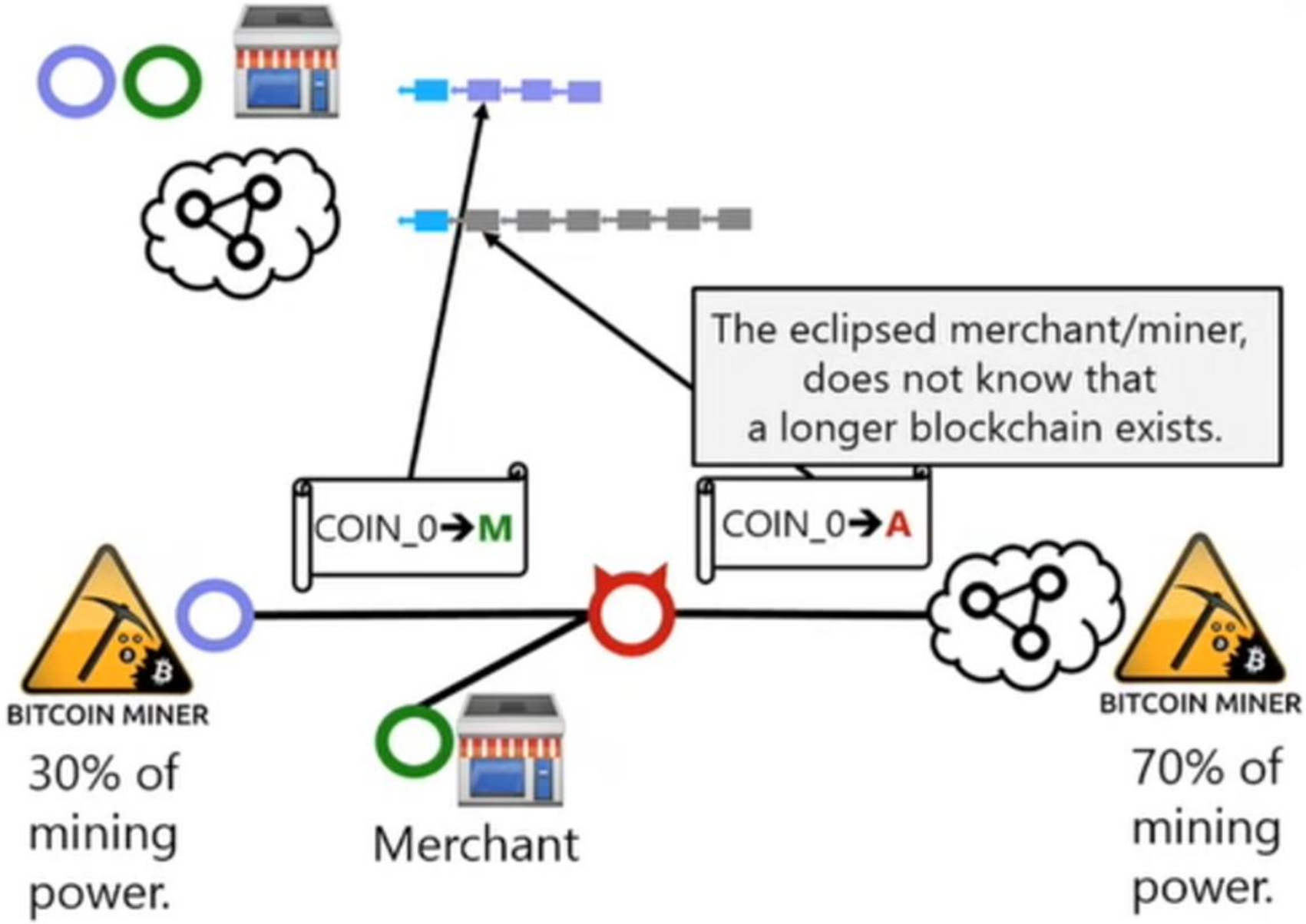
Example 2: N-Confirmation Double Spending



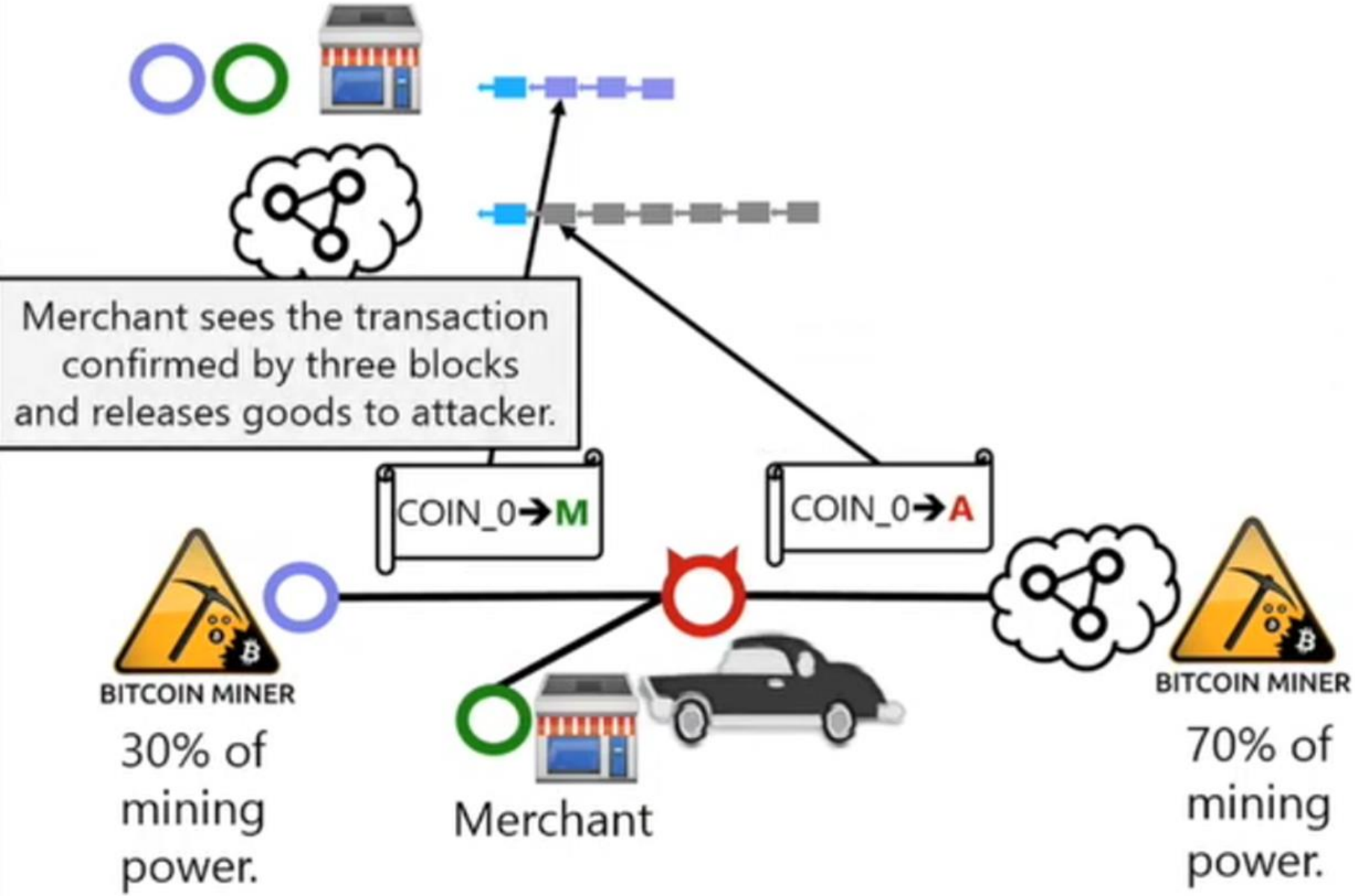
Example 2: N-Confirmation Double Spending



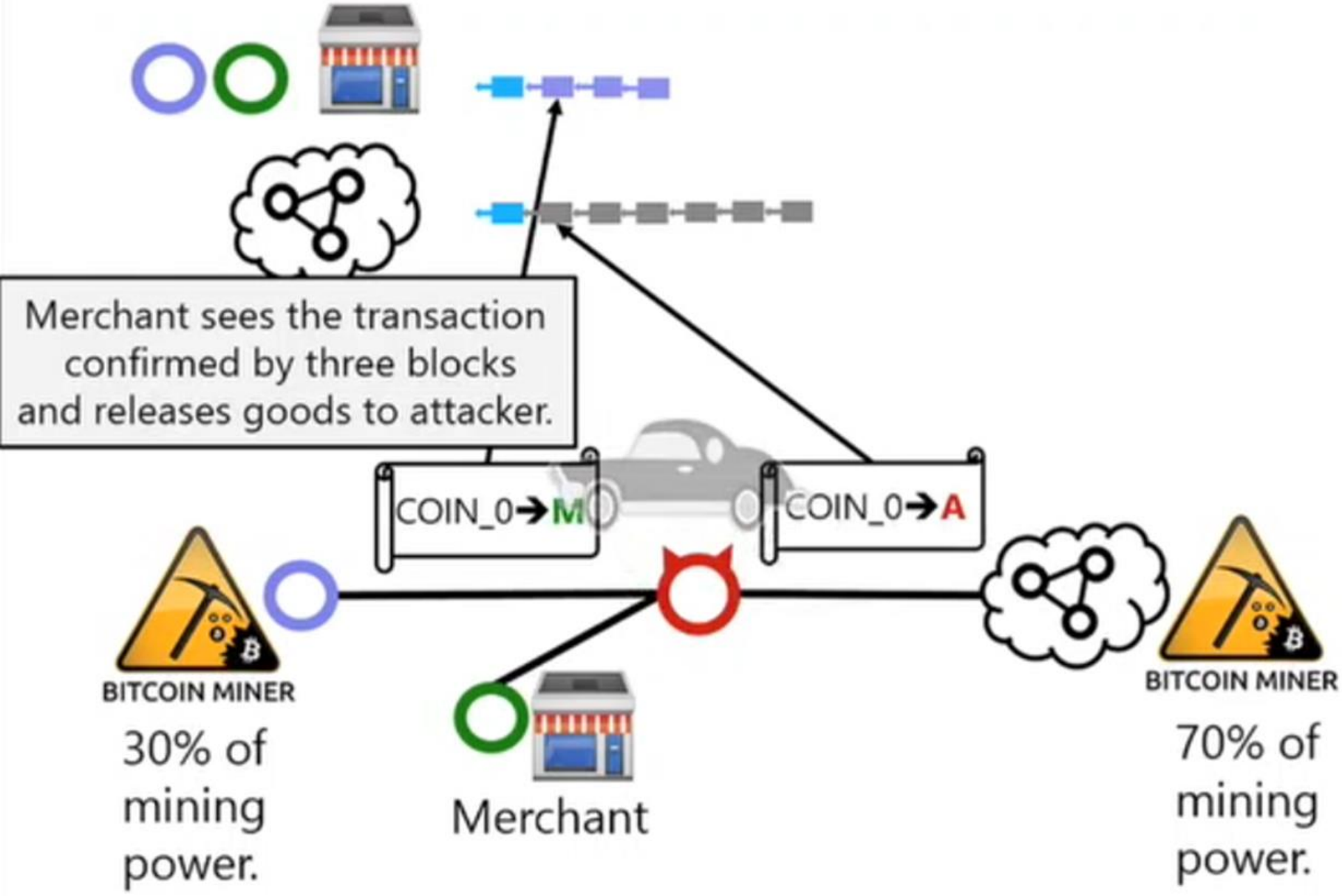
Example 2: N-Confirmation Double Spending



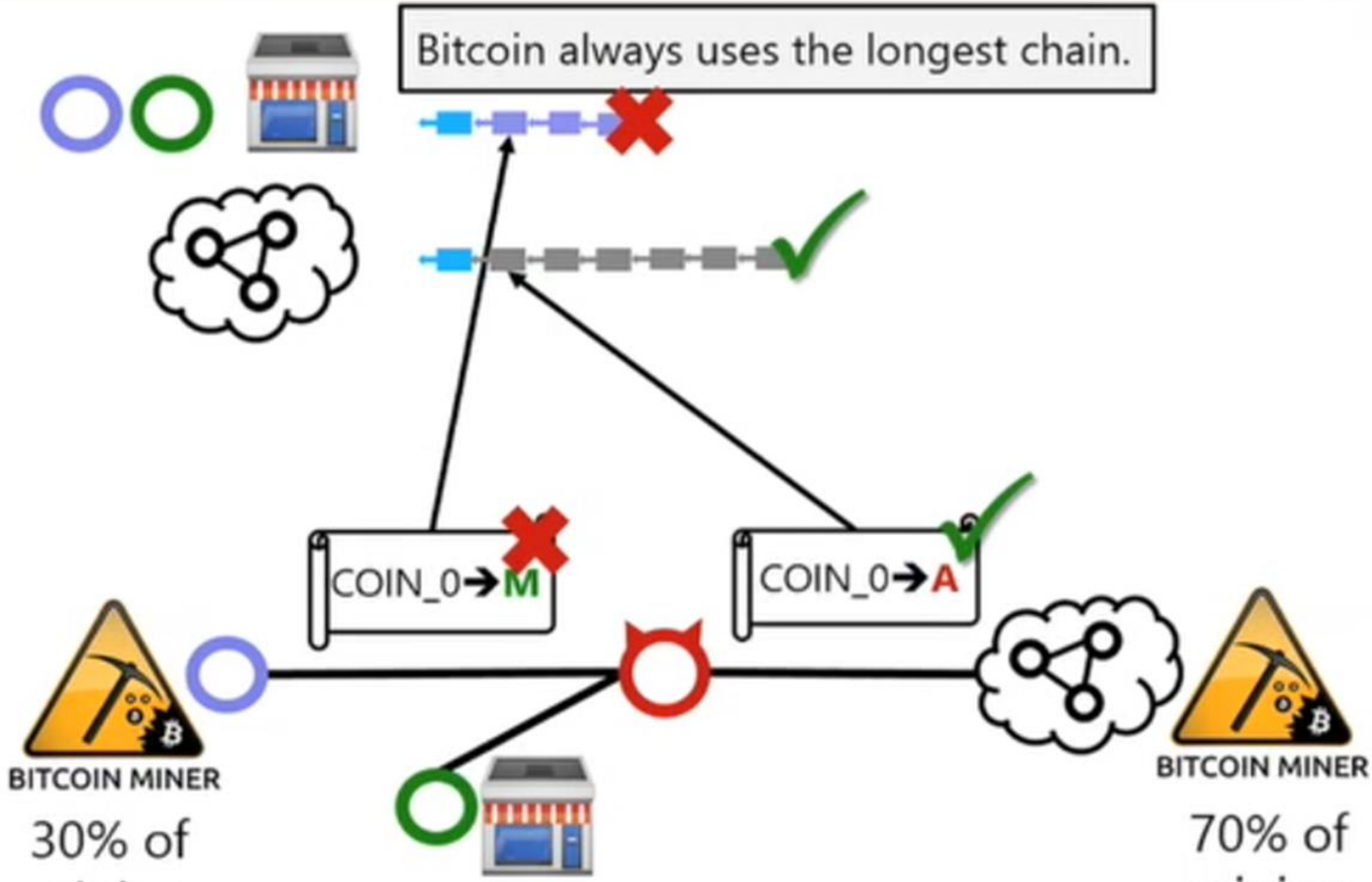
Example 2: N-Confirmation Double Spending



Example 2: N-Confirmation Double Spending

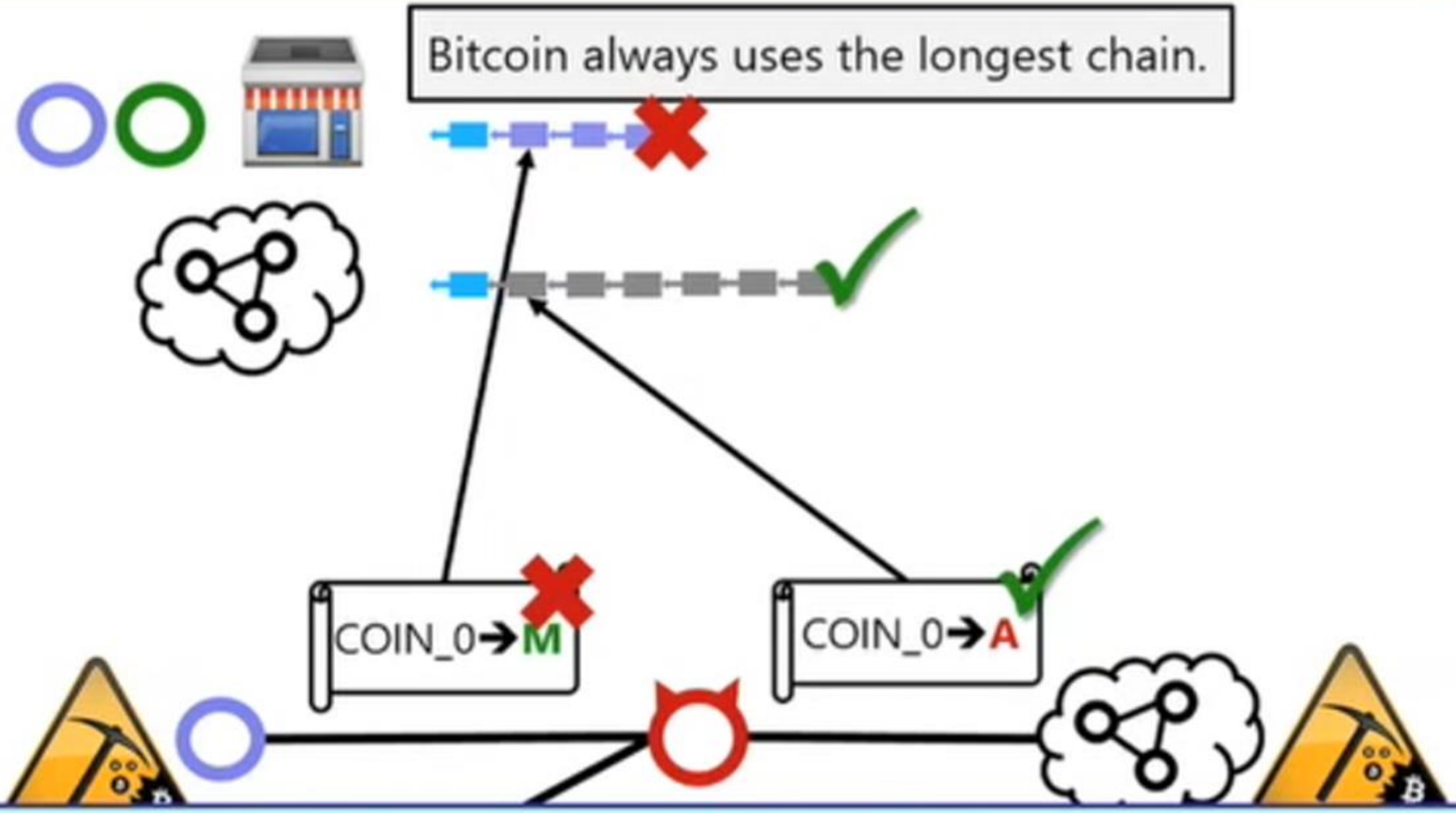


Example 2: N-Confirmation Double Spending



Merchant sees the transaction confirmed by three blocks so it appears safe, but it is double spent.

Example 2: N-Confirmation Double Spending



Attacker **did not** own mining power, but manipulated 3rd party's mining power.

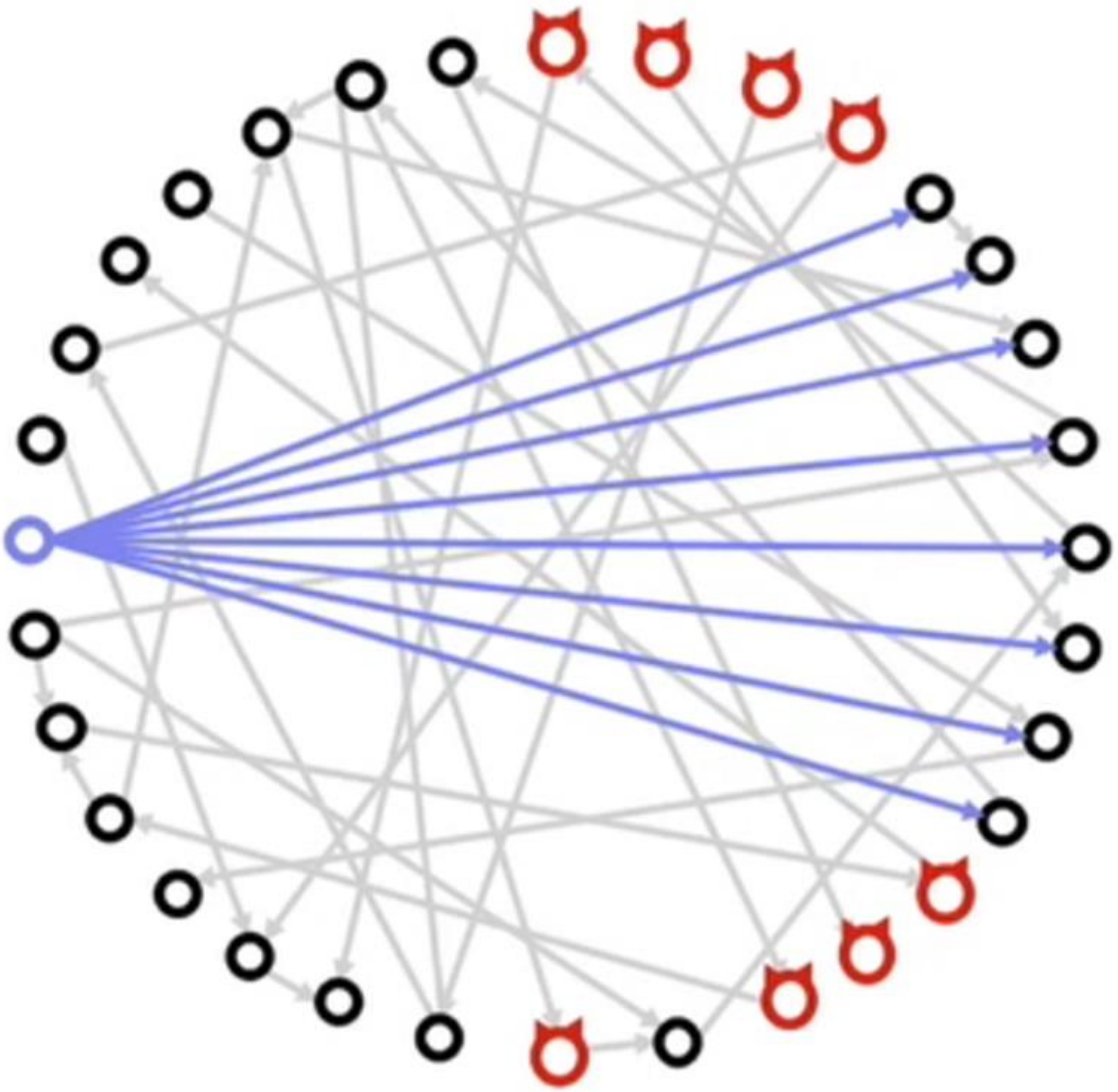
Similarly there are other "no mining power" eclipsing attacks:

- Finney-like 0-confirmation attacks,
- SPV transaction hiding attacks.

Outline

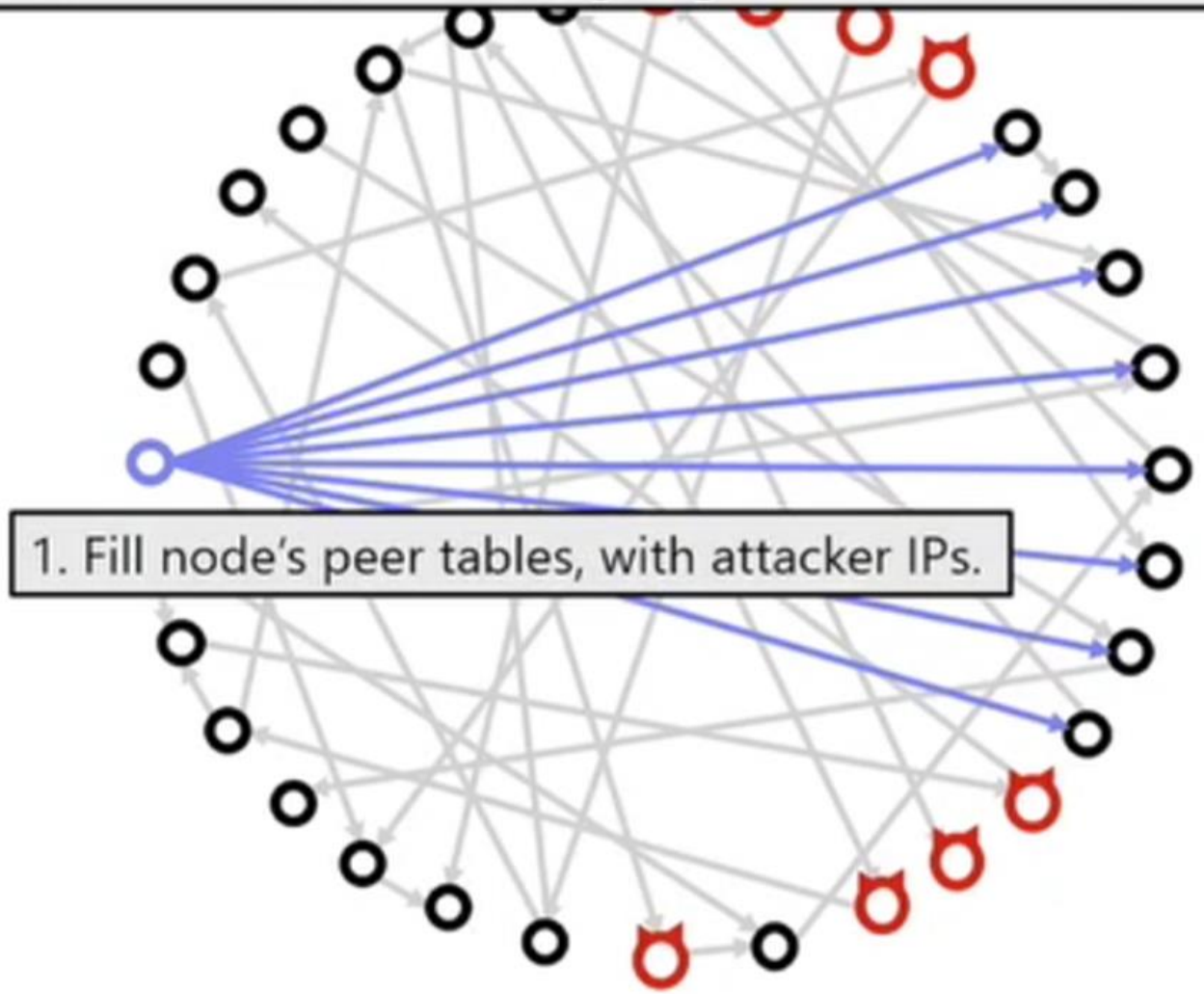
- Eclipse attacks & implications
  - What is an eclipse attack?
  - 51% attacks with far less than 51% of the mining power
  - N-confirmation double spending
- How to eclipse a Bitcoin node
  - P2P network details
  - How to exploit it
- How many IPs does the attacker need?
  - Models & Experimental Results
  - Botnets
- Countermeasures
  - Current deployment
  - Effectiveness of countermeasures

How to eclipse a node?



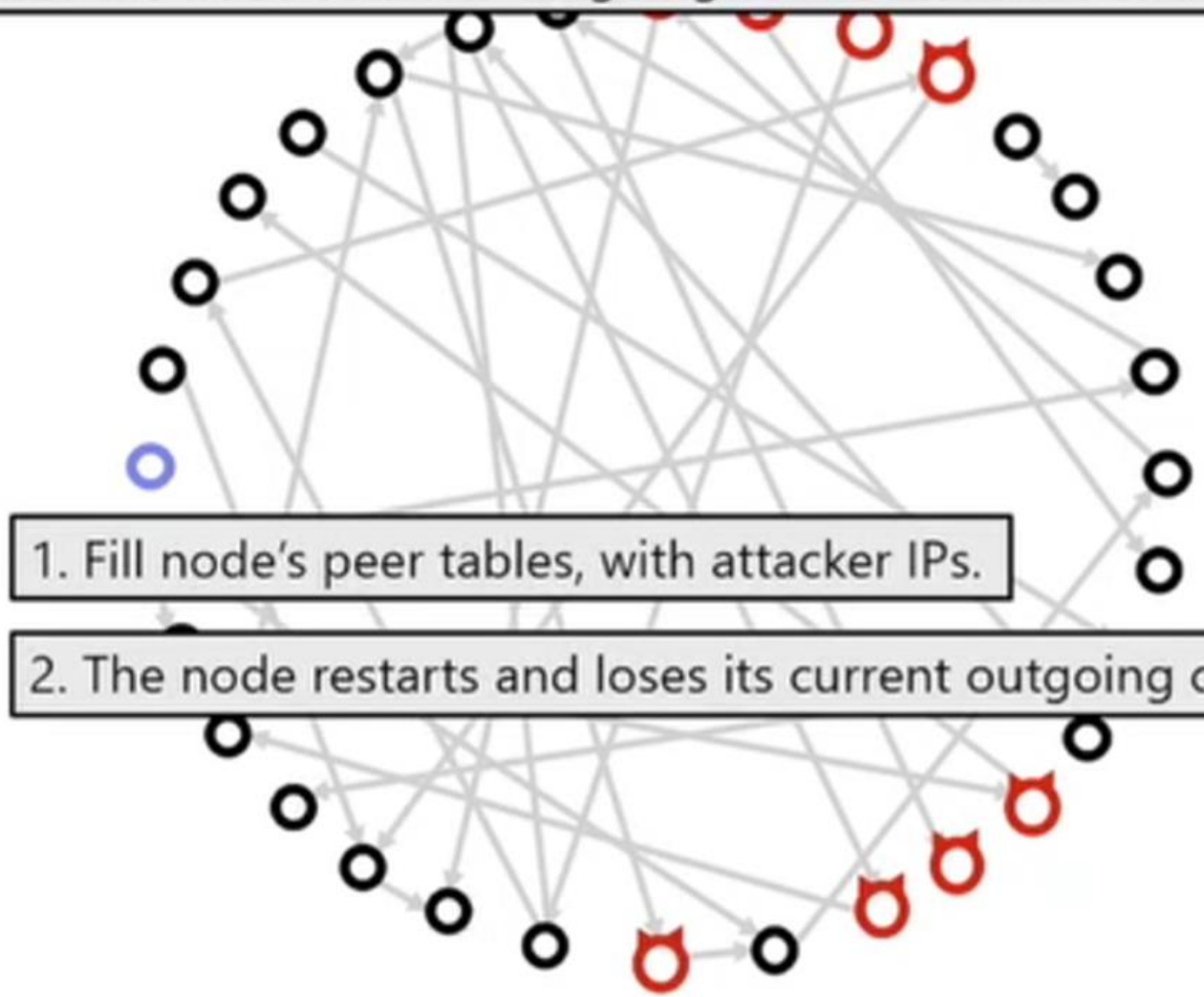
How to eclipse a node?

We manipulate the node so all its outgoing connections are to attacker IPs.



How to eclipse a node?

We manipulate the node so all its outgoing connections are to attacker IPs.

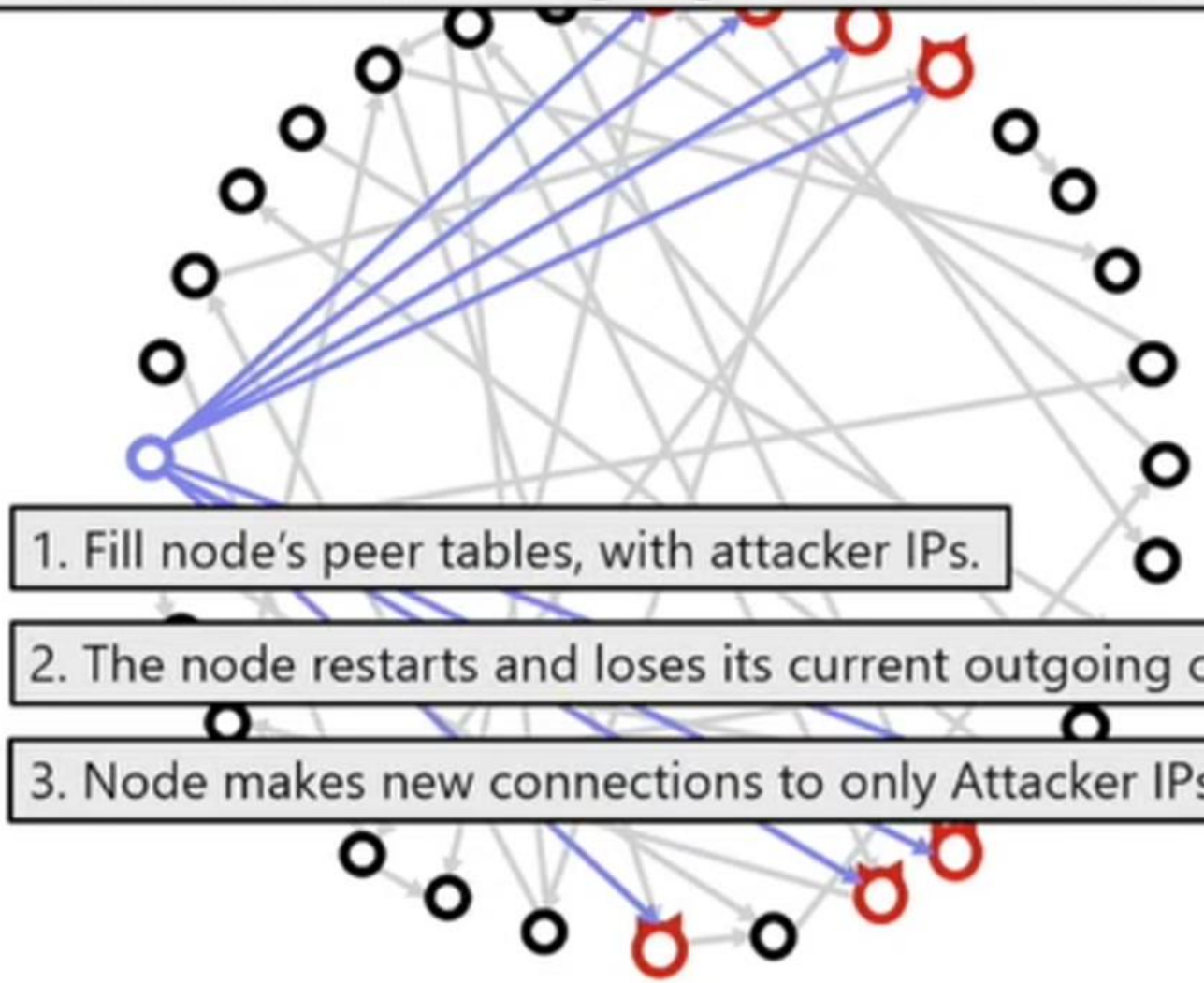


1. Fill node's peer tables, with attacker IPs.

2. The node restarts and loses its current outgoing connections.

How to eclipse a node?

We manipulate the node so all its outgoing connections are to attacker IPs.



1. Fill node's peer tables, with attacker IPs.

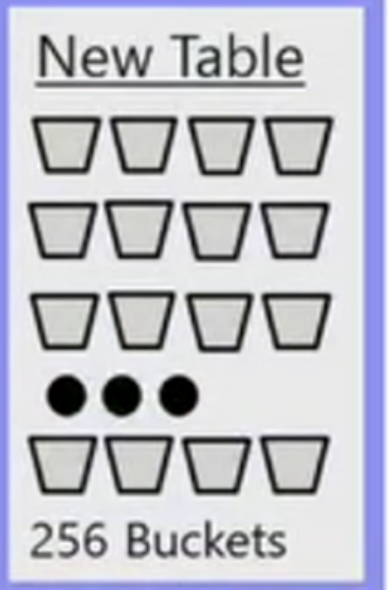
2. The node restarts and loses its current outgoing connections.

3. Node makes new connections to only Attacker IPs.

Storing and Selecting IPs.

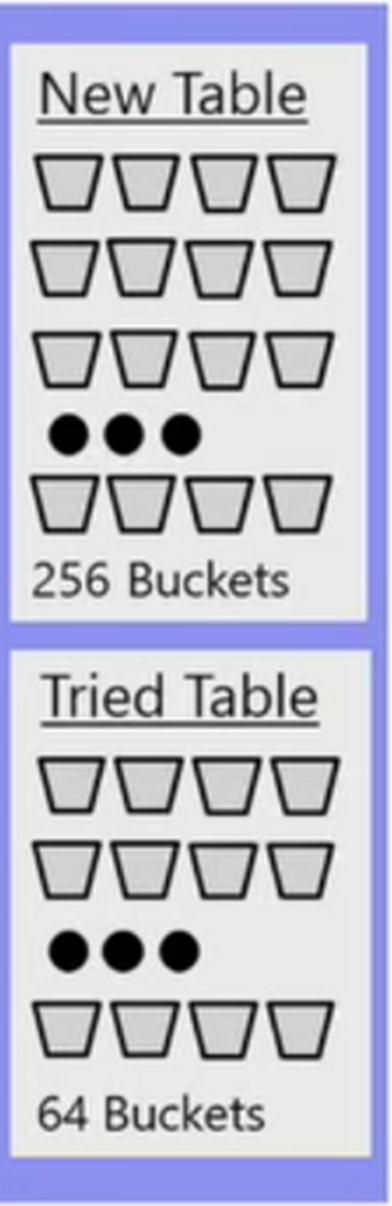
- Each node picks its peers from IP addresses stored in two tables:

Storing and Selecting IPs.



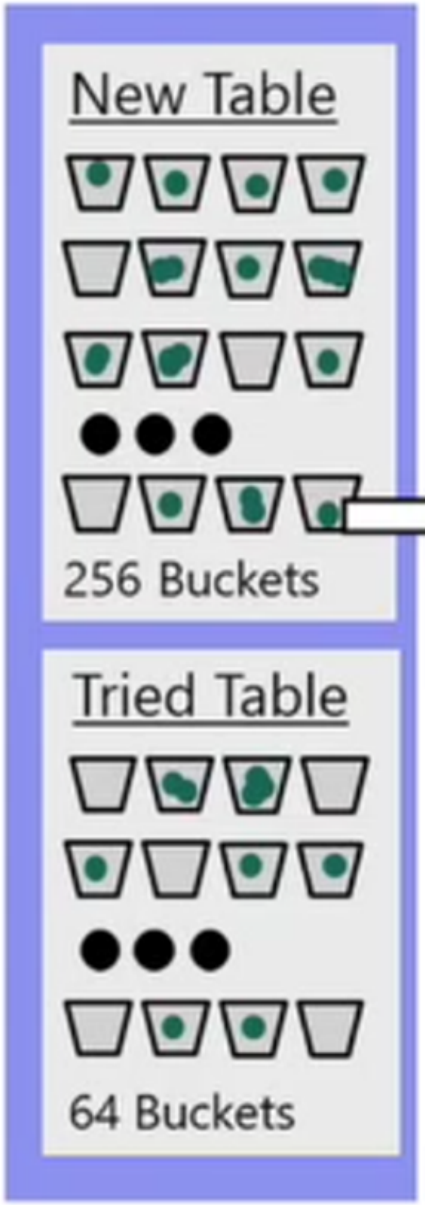
- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.

Storing and Selecting IPs.

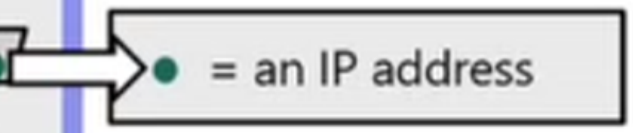


- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.

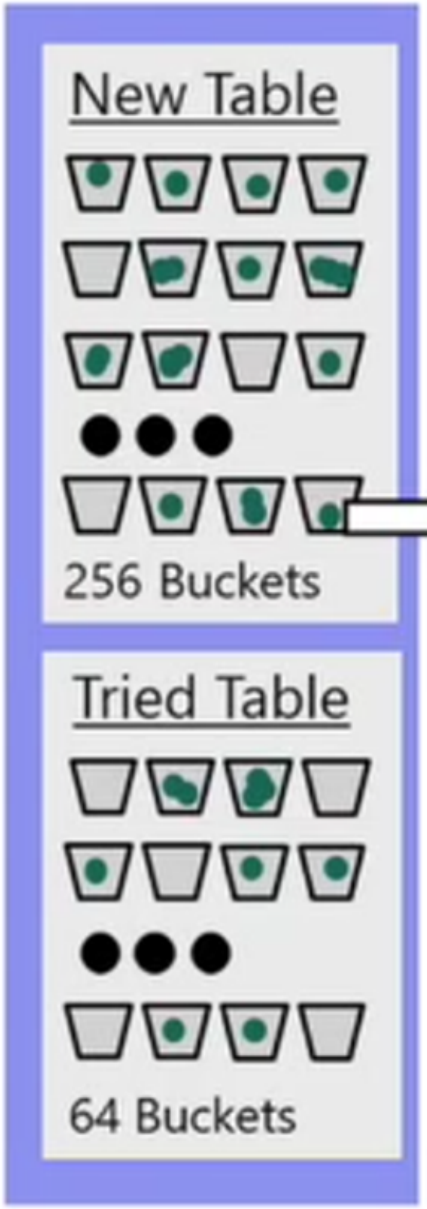
Storing and Selecting IPs.



- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.

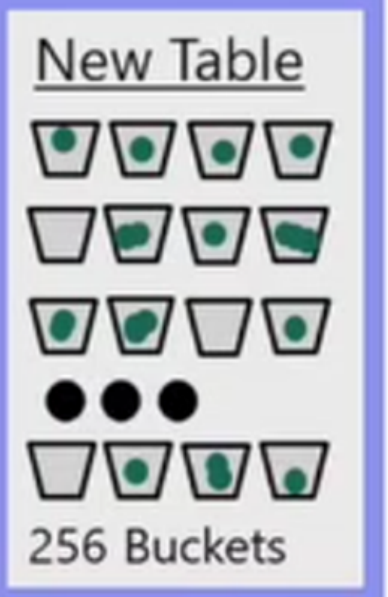


Storing and Selecting IPs.



- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.
- The tables also store a timestamp for each IP.

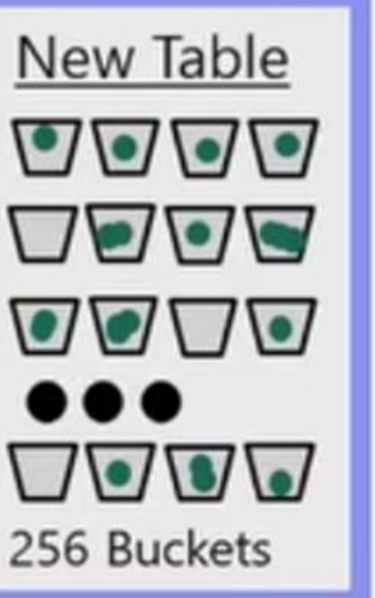
Storing and Selecting IPs.



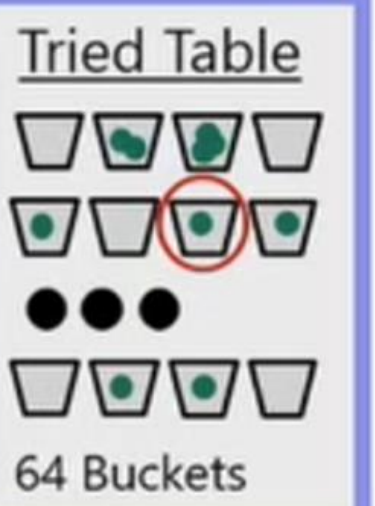
- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.
- The tables also store a timestamp for each IP.

- To find an IP to make an outgoing connection to:
  1. Choose new or tried tables to select from.

Storing and Selecting IPs.

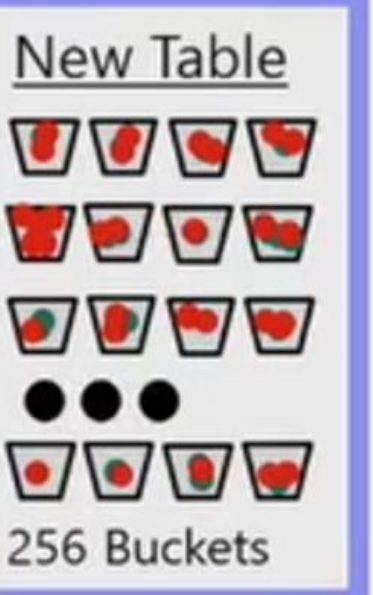


- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.
- The tables also store a timestamp for each IP.



- To find an IP to make an outgoing connection to:
  1. Choose new or tried tables to select from.
  2. Select an IP (biased toward "fresher" timestamps).

Storing and Selecting IPs.



- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.
- The tables also store a timestamp for each IP.

In our attack we fill the tables with attacker IPs (●), so that the node will only connect to attacker IPs.

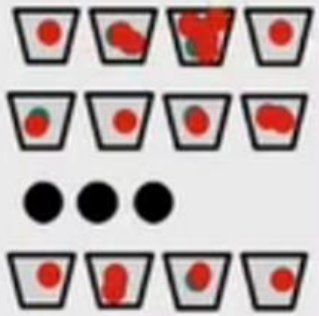
- To find an IP to make an outgoing connection to:
  1. Choose new or tried tables to select from.
  2. Select an IP (biased toward "fresher" timestamps).
  3. Attempt an outgoing connection to that IP.

Storing and Selecting IPs.

New Table

256 Buckets

Tried Table



64 Buckets

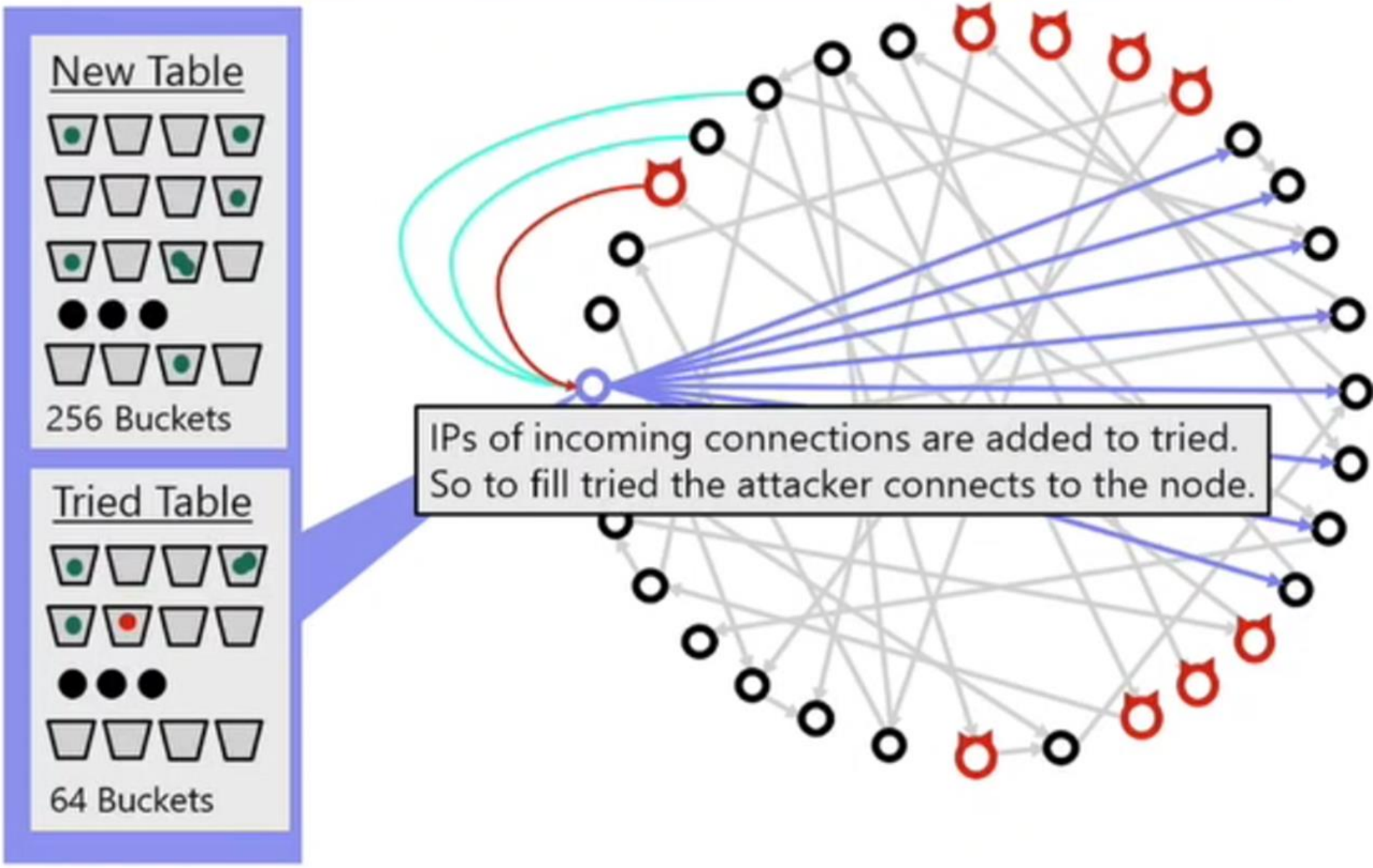
- Each node picks its peers from IP addresses stored in two tables:
  - **New table:** IPs the node has heard about.
  - **Tried table:** IPs the node peered with at some point.
- The tables also store a timestamp for each IP.

In our attack we fill the tables with attacker IPs (●), so that the node will only connect to attacker IPs.

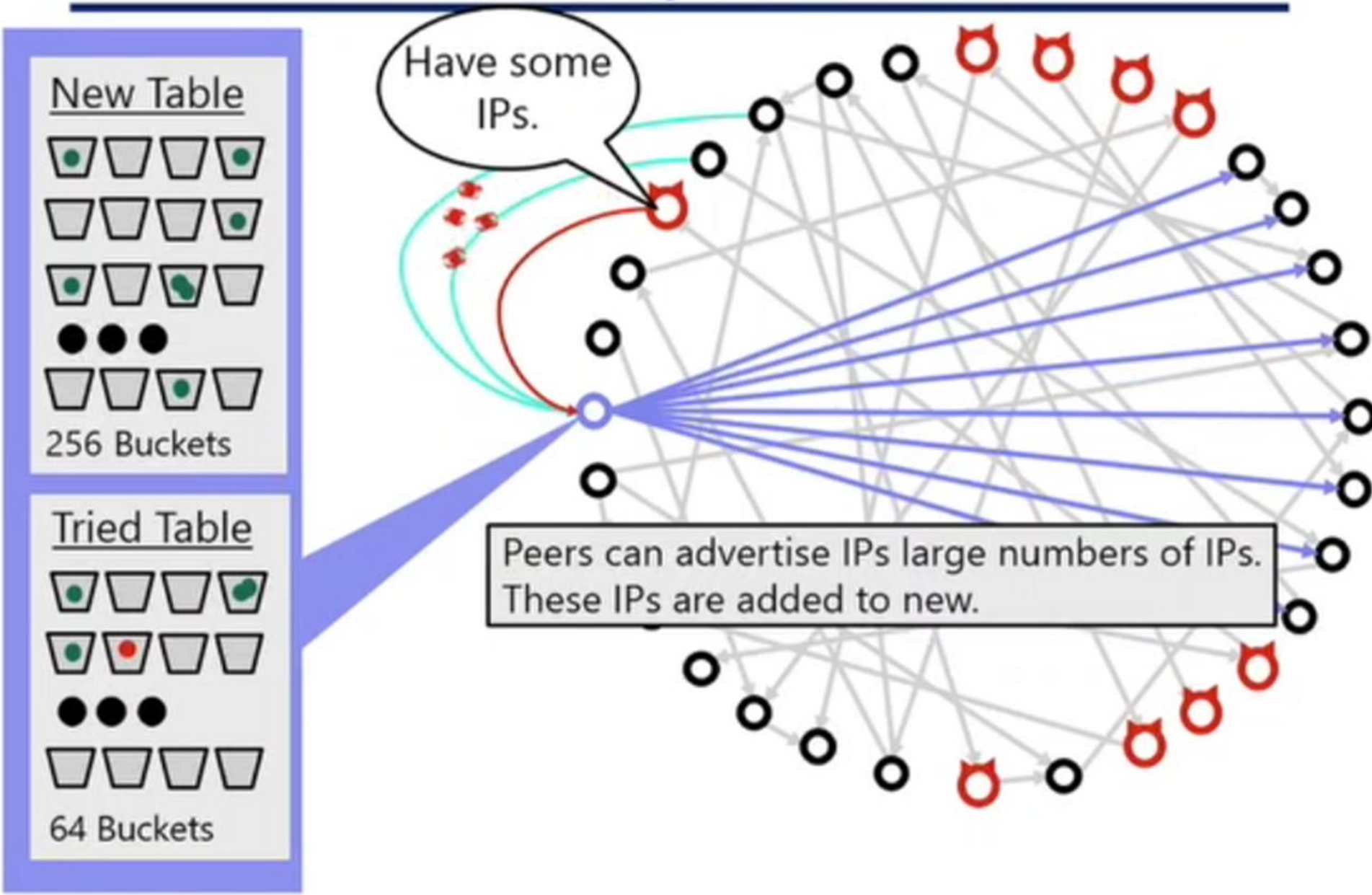
- To find an IP to make an outgoing connection to:
  1. Choose new or tried tables to select from.
  2. Select an IP (biased toward "fresher" timestamps).
  3. Attempt an outgoing connection to that IP.

Vulnerability 1 – Selection Bias:
Attacker ensures its IPs are fresher,
so they are more likely to be selected.

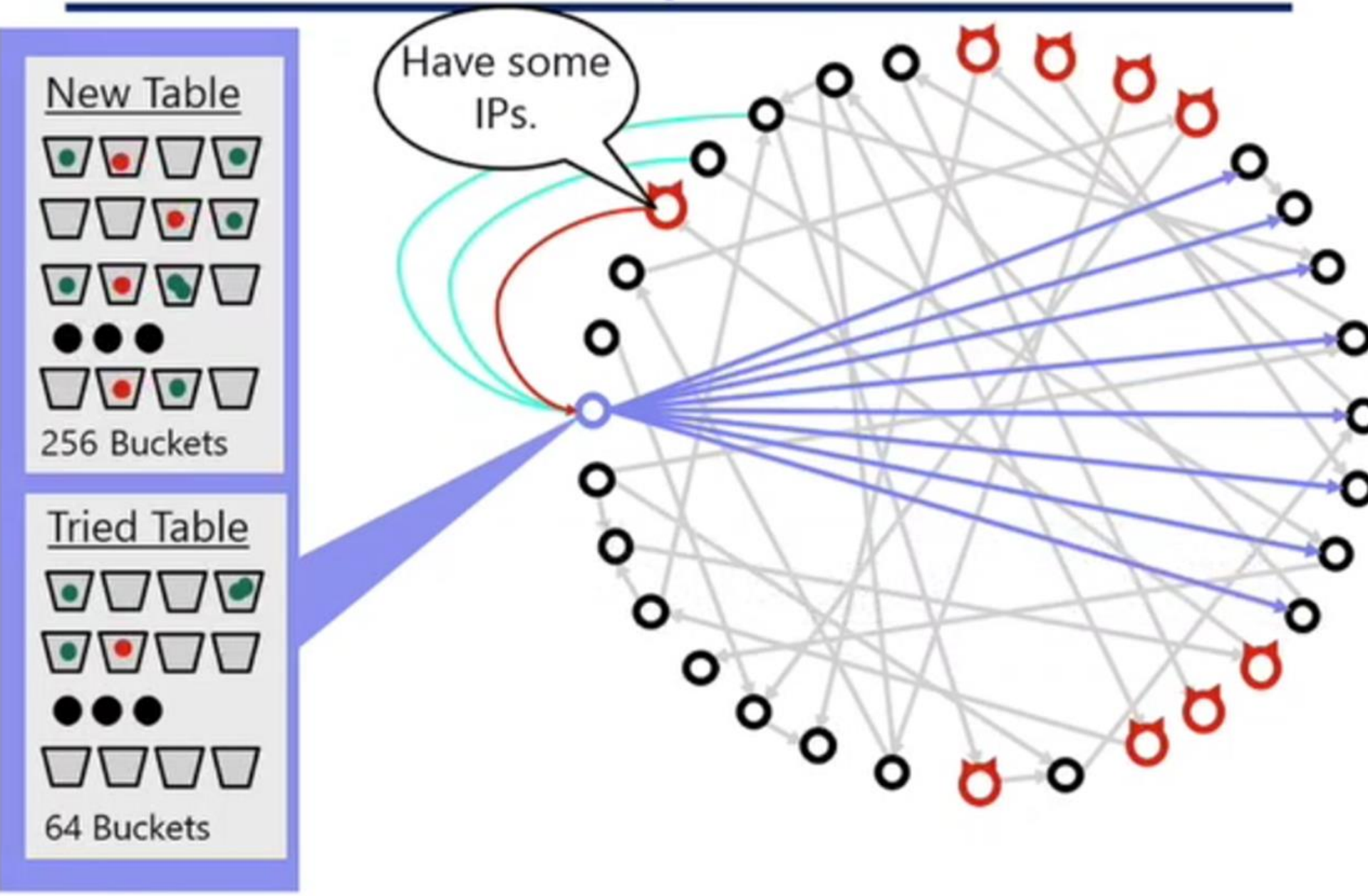
How to eclipse a node?



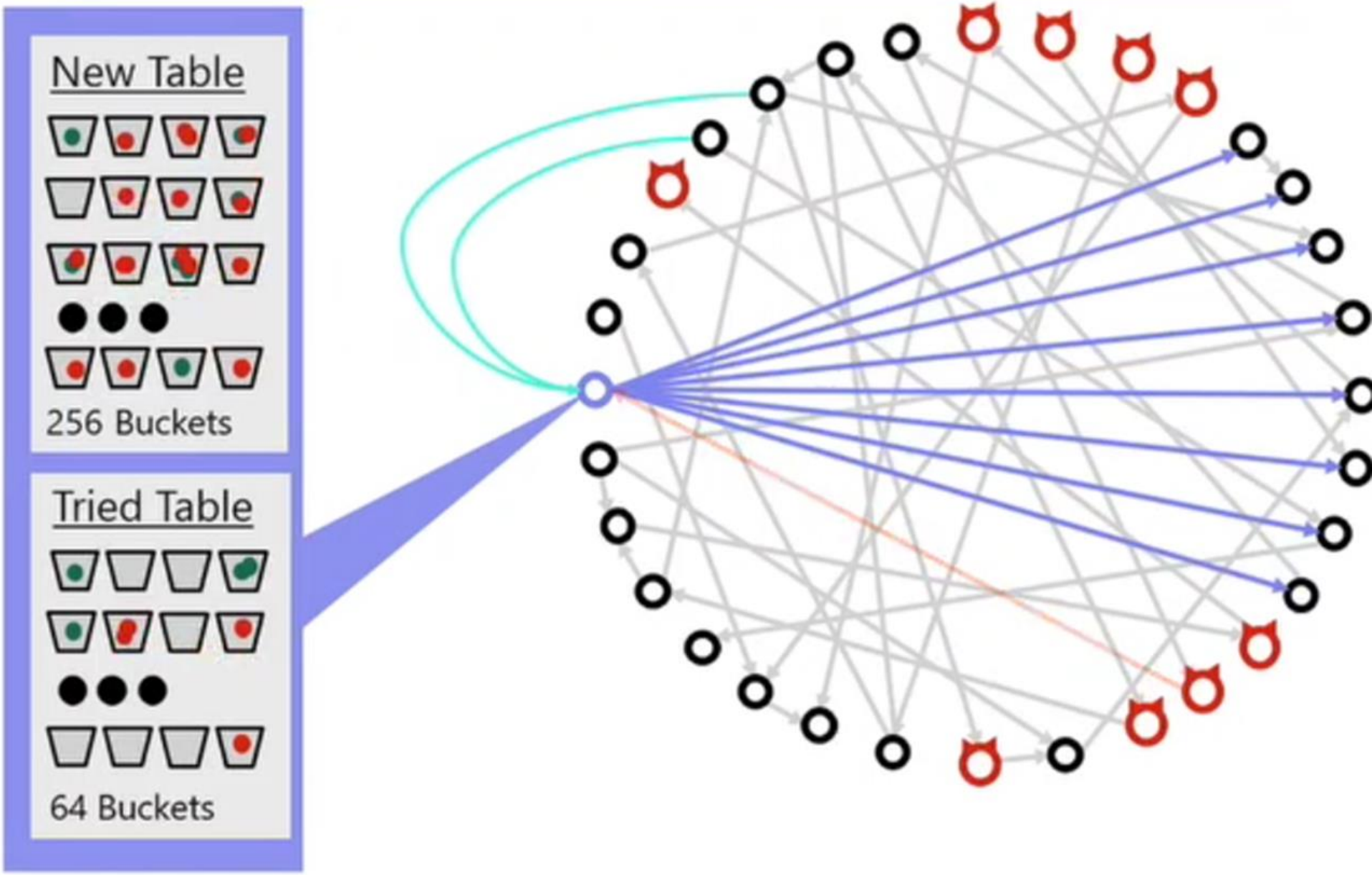
How to eclipse a node?



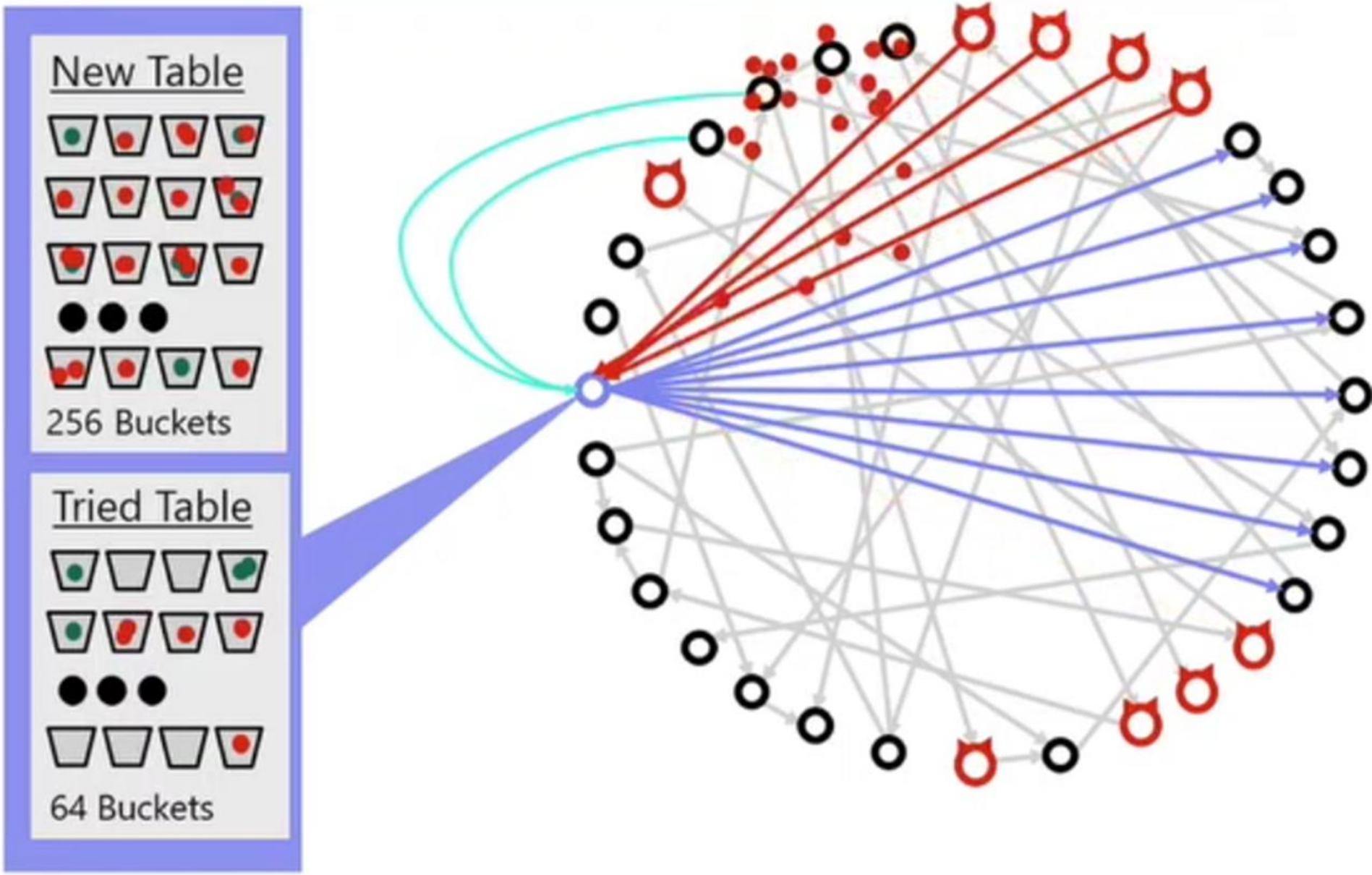
How to eclipse a node?



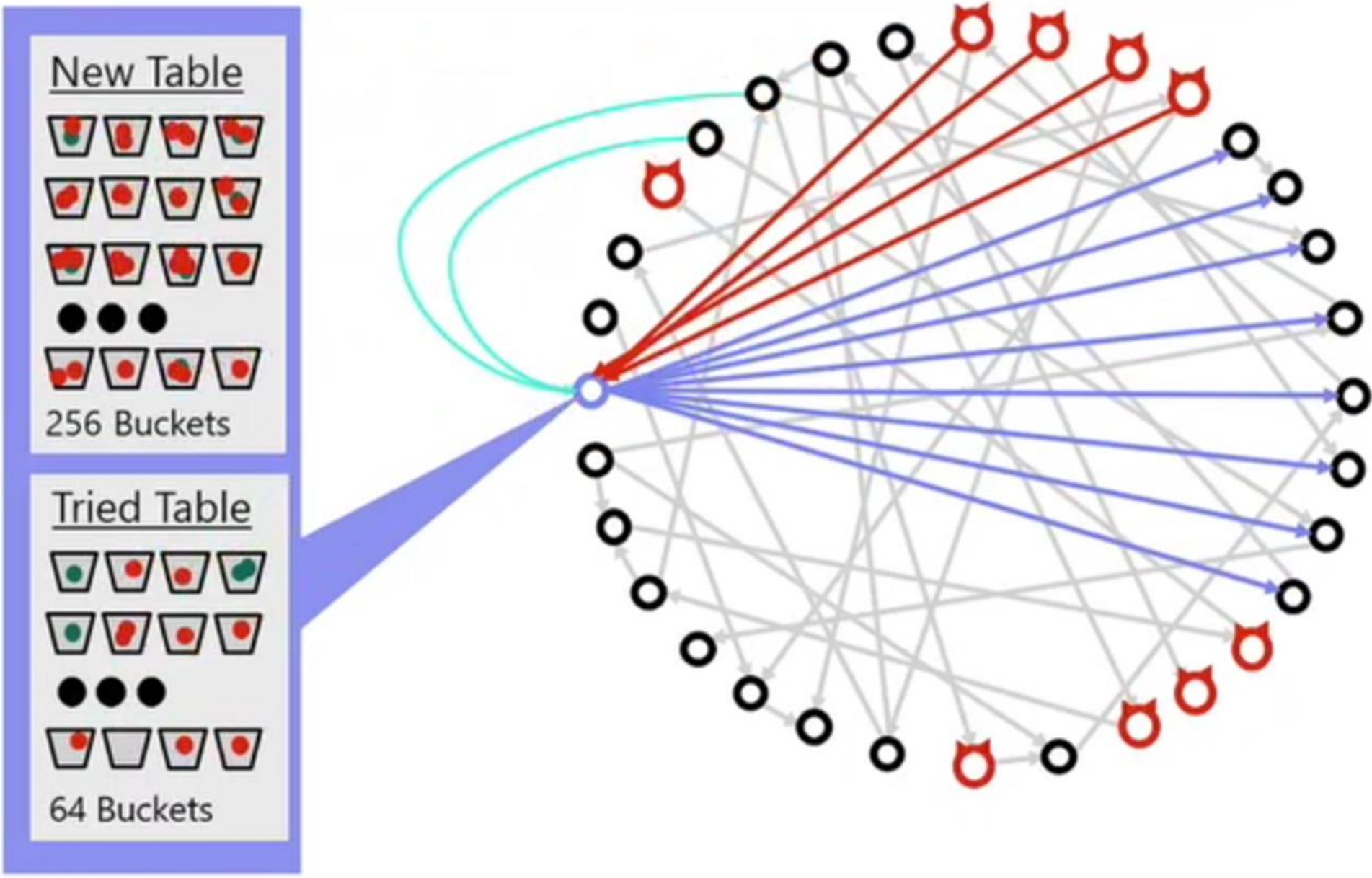
How to eclipse a node?



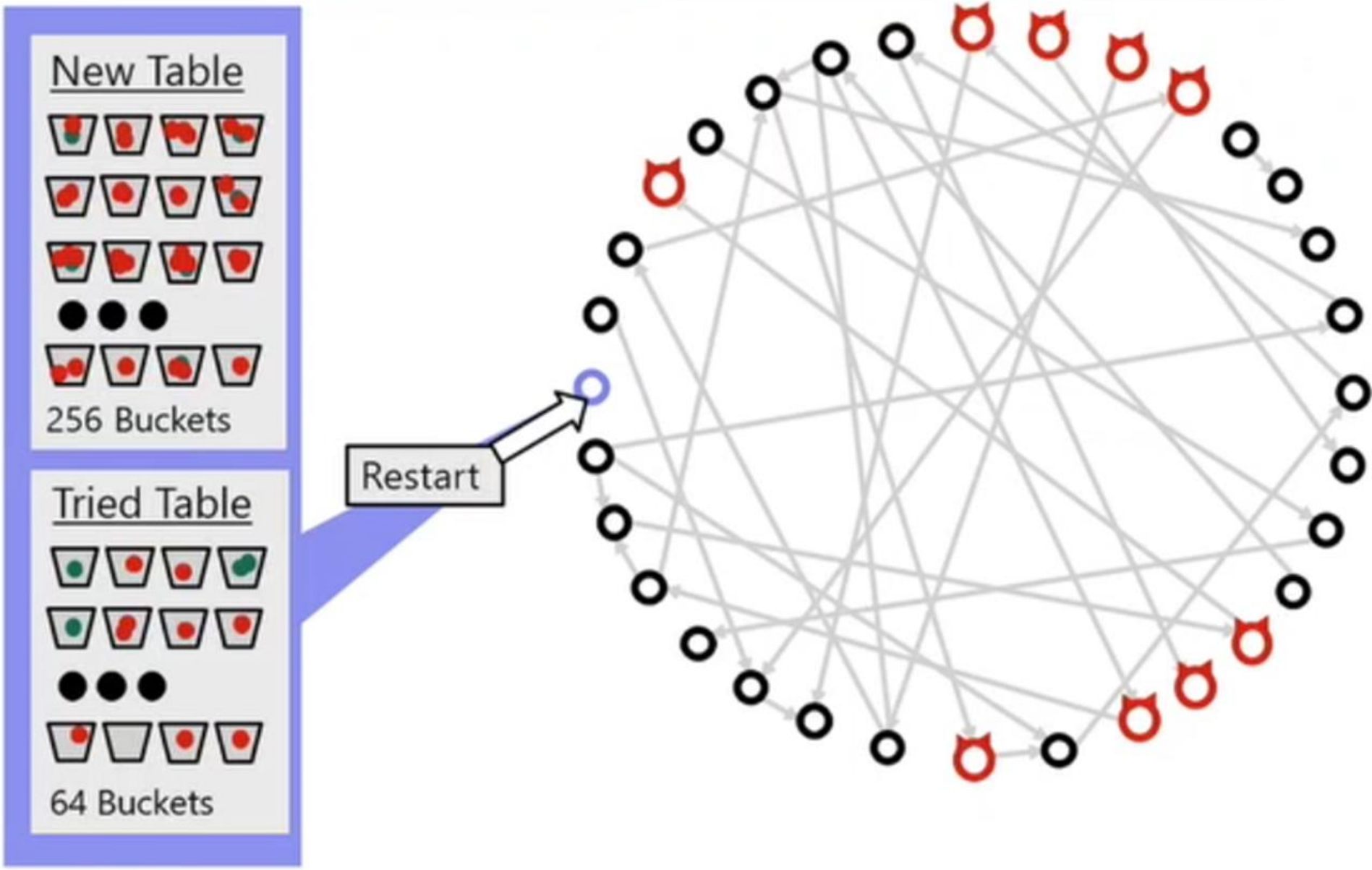
How to eclipse a node?



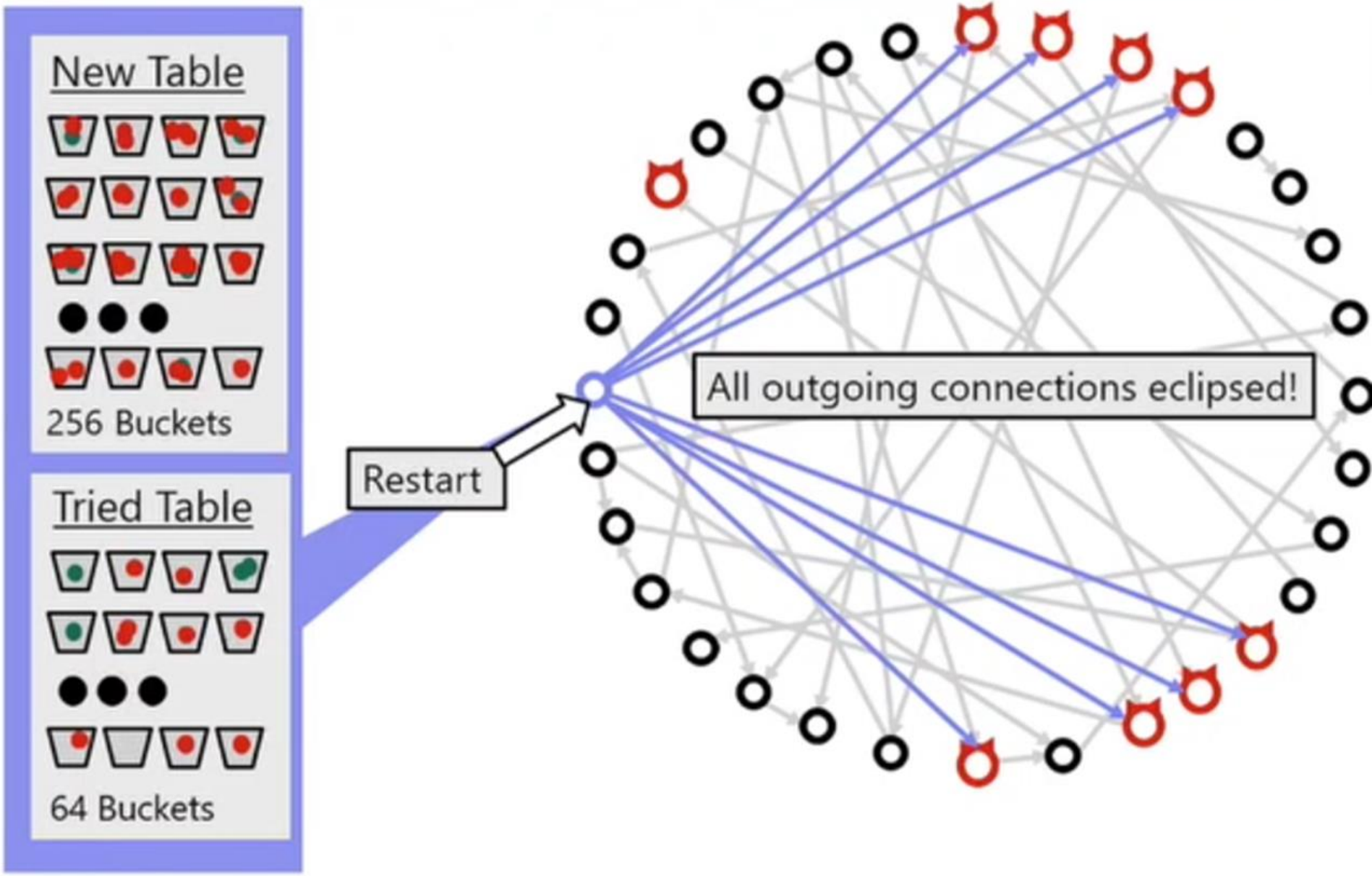
How to eclipse a node?



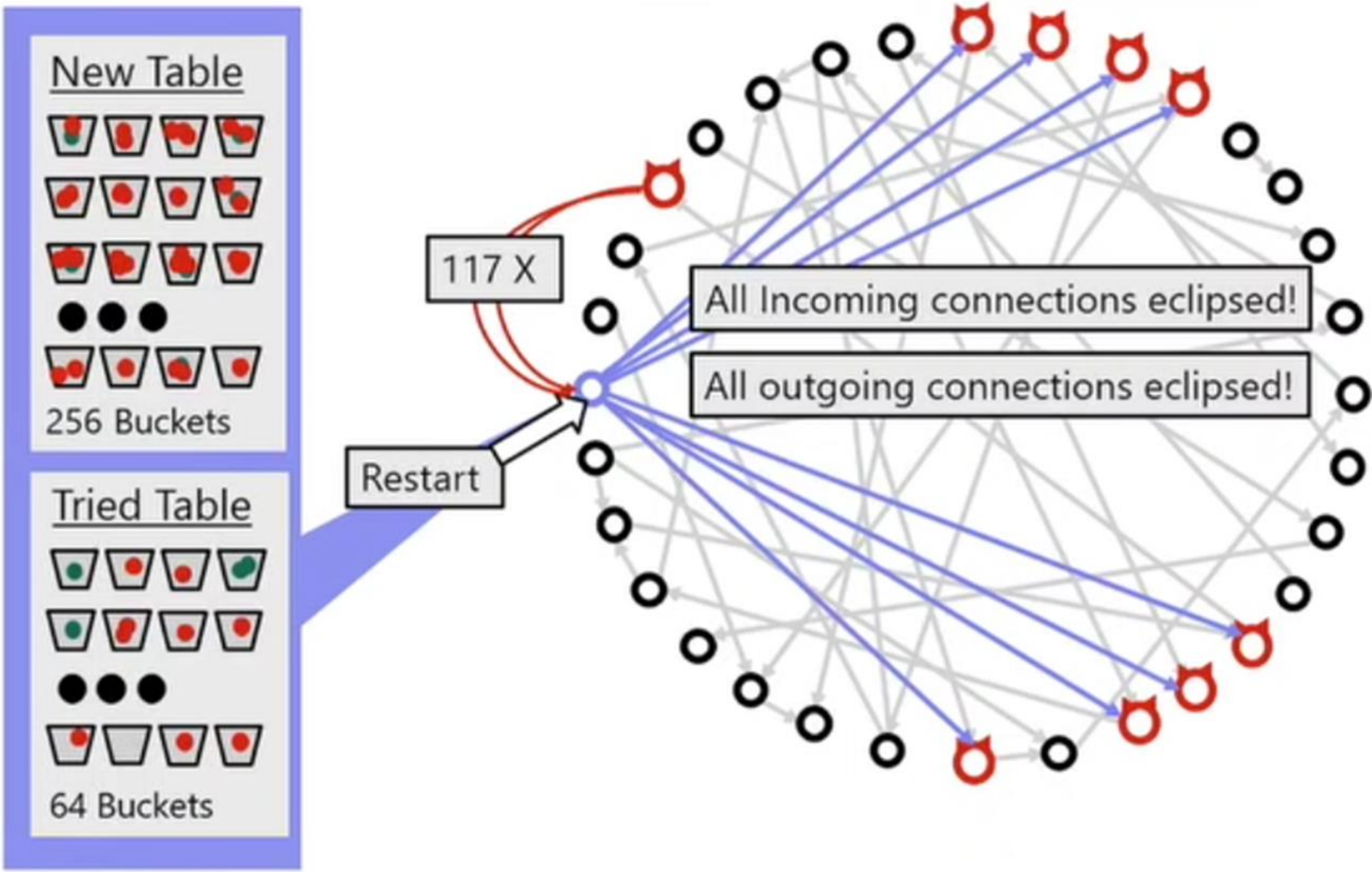
How to eclipse a node?



How to eclipse a node?



How to eclipse a node?



How easy are restarts?

Our Attack requires the victim node restarts; how can this happen?

- **Software/security updates:**
  - predictable for the attacker, users are notified of upcoming updates,
  - lose/lose for the victim, restart or remain vulnerable.
- **Packets of death/DoS Attacks:**
  - Ten DoS CVEs in Bitcoin[1], many more on underlying OSes.
- **Power/network failures:**
  - Bitcoin nodes have 25% chance of going offline within 10 hours[2].

[1]: https://en.bitcoin.it/wiki/Common_Vulnerabilities_and_Exposures

[2]: Biryukov, A. et al., Deanonymisation of clients in Bitcoin P2P network.

How easy are restarts?

Our Attack requires the victim node restarts; how can this happen?

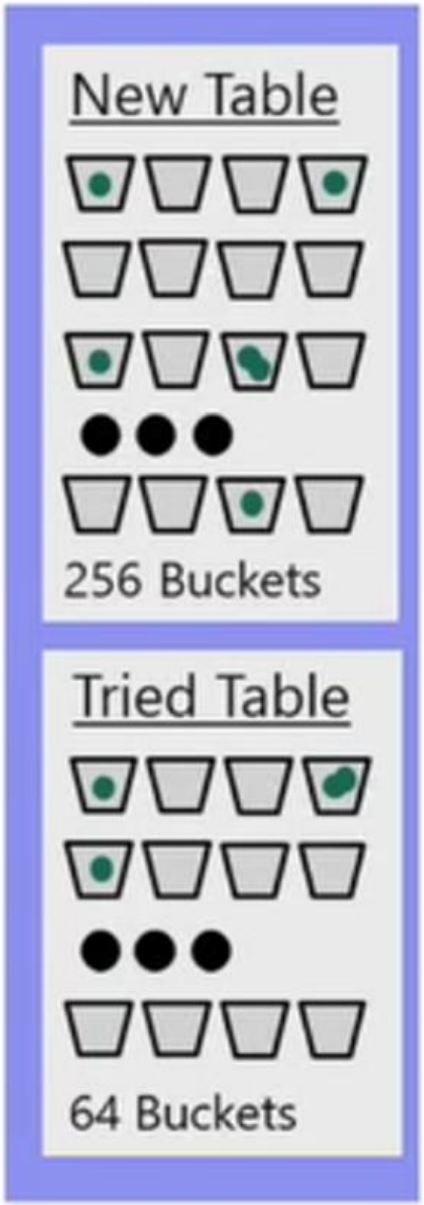
- **Software/security updates:**
  - predictable for the attacker, users are notified of upcoming updates,
  - lose/lose for the victim, restart or remain vulnerable.
- **Packets of death/DoS Attacks:**
  - Ten DoS CVEs in Bitcoin[1], many more on underlying OSes.
- **Power/network failures:**
  - Bitcoin nodes have 25% chance of going offline within 10 hours[2].

The security of the peer-to-peer network should not rely on 100% node uptime!

[1]: https://en.bitcoin.it/wiki/Common_Vulnerabilities_and_Exposures

[2]: Biryukov, A. et al., Deanonymisation of clients in Bitcoin P2P network.

Table Details: Hashing IPs into the Tried Table



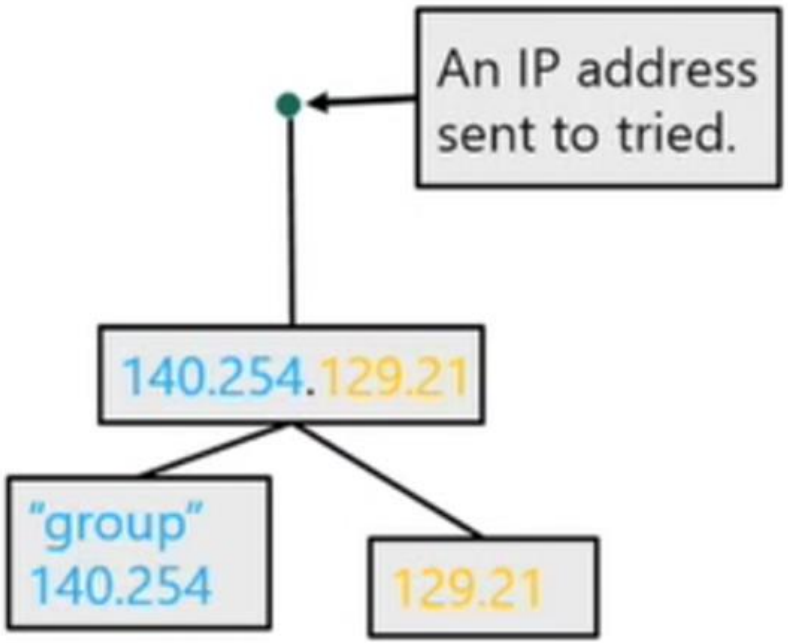


Table Details: Hashing IPs into the Tried Table

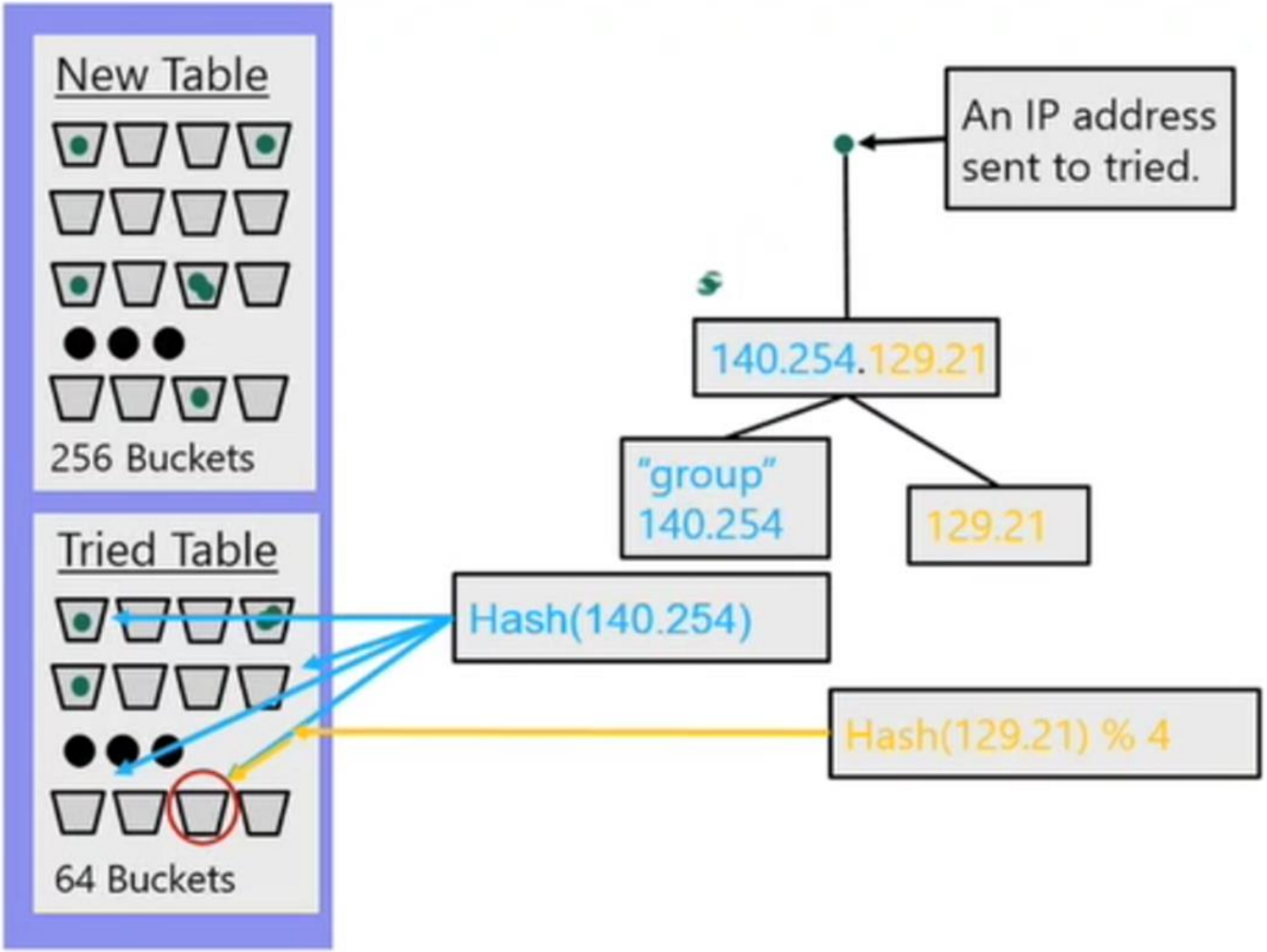
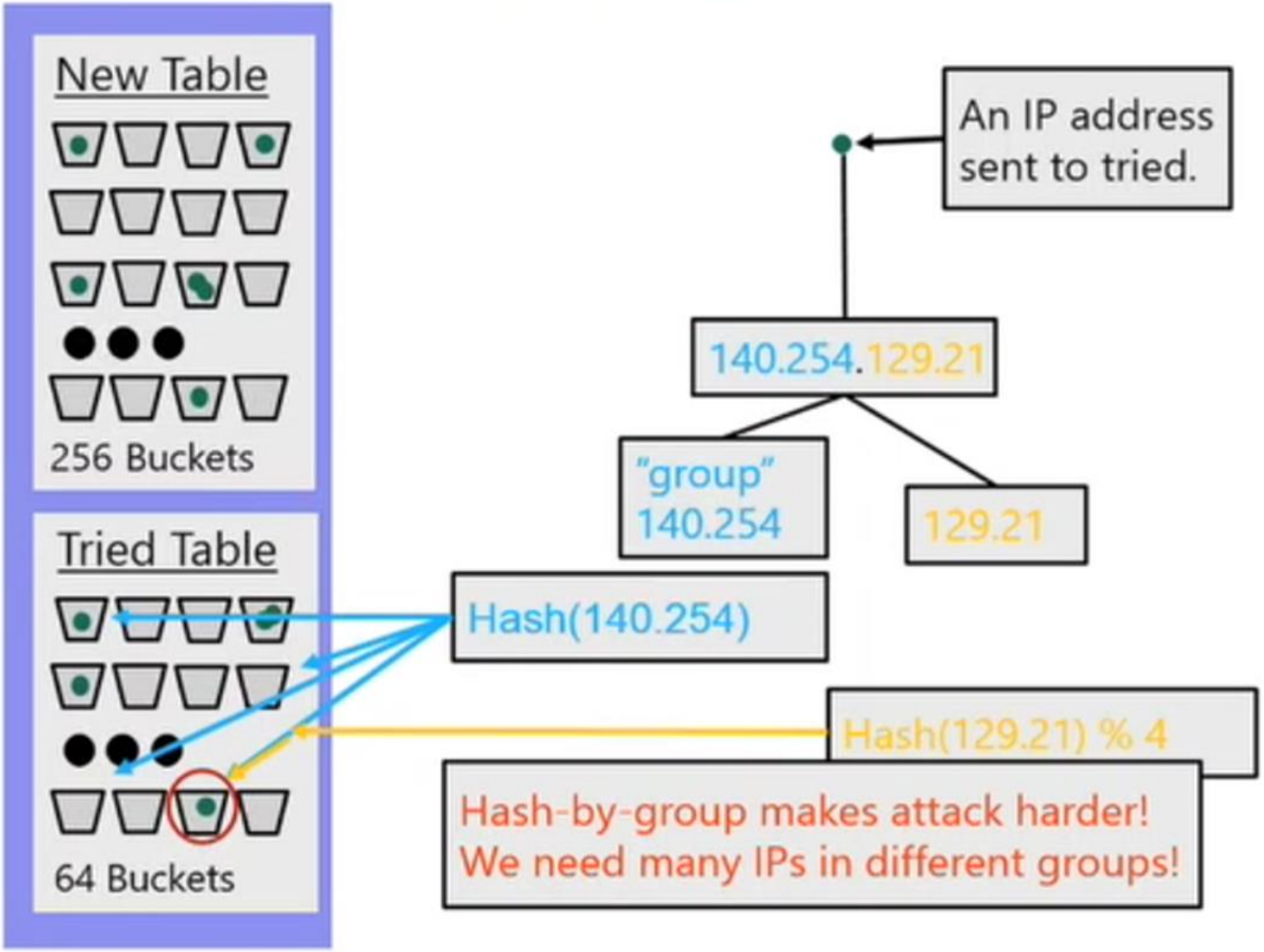


Table Details: Hashing IPs into the Tried Table



How to eclipse with a limited number of IPs?

Hash-by-group makes attack harder!
We need many IPs in different groups!

The attack gets easier if

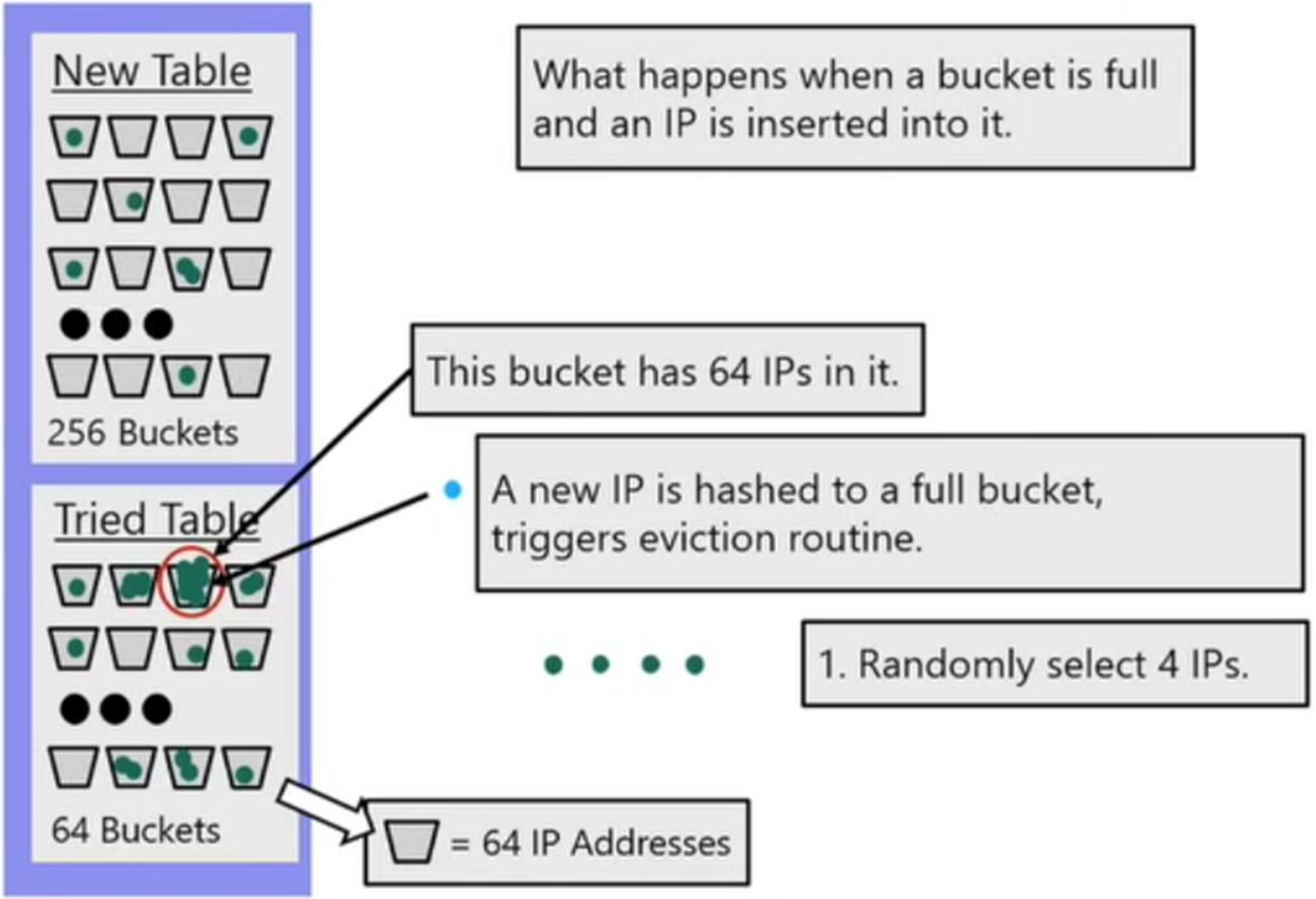
- More attacker IPs in distinct groups.
- Few honest IPs in tried table.

- Stale honest IPs in the tried table.
- Fresh attacker IPs in the tried table.

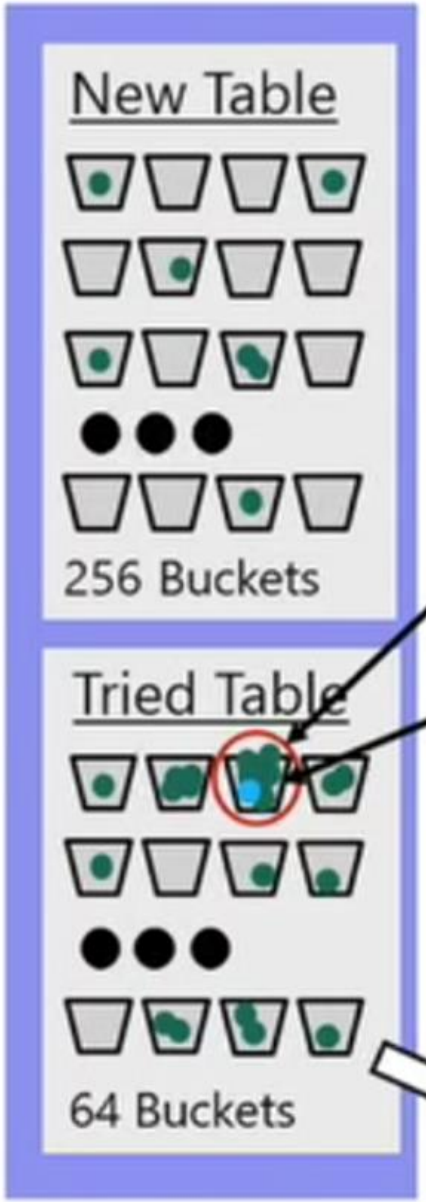
Vulnerability 1 – Selection Bias:
Attacker ensures its IPs are fresher,
so they are more likely to be selected.

Attacker can age honest IPs by investing more time in the attack.
Attacker can ensure fresh IPs by continually filling the new table.

Exploiting Bucket Eviction by Investing Time



Exploiting Bucket Eviction by Investing Time



What happens when a bucket is full and an IP is inserted into it.

This bucket has 64 IPs in it.

A new IP is hashed to a full bucket, triggers eviction routine.

1. Randomly select 4 IPs.

2. Delete oldest IP.

3. Insert new IP.

= 64 IP Addresses

Exploiting Bucket Eviction by Investing Time

Vulnerability 2 - Try-Try-Again:
If an attacker IP replaces another attacker IP, we can resend the evicted IP and eventually replace an honest IP.

1. Randomly select 4 IPs.

Vulnerability 3 – Eviction Bias:
Attack IPs will always have the most recent timestamps.

2. Delete oldest IP.

3. Insert new IP.

Outline

- Eclipse attacks & implications
  - What is an eclipse attack?
  - 51% attacks with far less than 51% of the mining power
  - N-confirmation double spending
- How to eclipse a Bitcoin node
  - P2P network details
  - How to exploit it
- How many IPs does the attacker need?
  - Models & Experimental Results
  - Botnets
- Countermeasures
  - Current deployment
  - Effectiveness of countermeasures

How Many IP Addresses does a botnet need?

Our Approach:

1. Model Bitcoin with probability analysis & Monte-Carlo simulations.
2. Use these models to determine effective attack parameters.
3. Experimentally verify these parameters against Bitcoin nodes.

Results: Worst Case for Attacker (Artificially filled)

Before attack:
Artificially fill a node's tried table.
Tried table is 99.9% full of honest IPs (4093 IPs).

Attacker resources:
Botnet of 4600 IPs, 2 IPs per group.
5 hours invested.

Walowdac Botnet: ~160,000 IPs, ~25,000 groups[1]

After attack:
Node's tried table is now 98.8% attacker IPs.

Experimental results:
100% attacker success rate (all 8 outgoing connections eclipsed).

[1]: Stock, B., et al Walowdac: Analysis of a peer-to-peer botnet. (2009)

Results: Live experiment

Before attack:
Connected a node to the bitcoin network for +43 days.
Node's tried table has 298 honest IPs.

Attacker resources:
Botnet of 400 IPs = 400 groups, 1 IPs per group.
1 hour invested.

Walowdac Botnet: ~160,000 IPs, ~25,000 groups[1]
Carna Botnet: ~420,000 IPs[2]
1250 random IPs from Carna spans 402 groups on average

After attack:
Tried table still mostly empty, but 57% of IPs are attacker IPs.

Experimental results:
84% attacker success rate (all 8 outgoing connections eclipsed).

[1]: Stock, B., et al Walowdac: Analysis of a peer-to-peer botnet. (2009)
[2]: Carna Botnet. Internet census 2012.

Outline

- Eclipse attacks & implications
  - What is an eclipse attack?
  - 51% attacks with far less than 51% of the mining power
  - N-confirmation double spending
- How to eclipse a Bitcoin node
  - P2P network details
  - How to exploit it
- How many IPs does the attacker need?
  - Models & Experimental Results
  - Botnets
- Countermeasures
  - Current deployment
  - Effectiveness of countermeasures

Countermeasures: Random Selection

Vulnerability 1 – Selection Bias:

Attacker ensures its IPs are fresher so they are more likely to be selected.

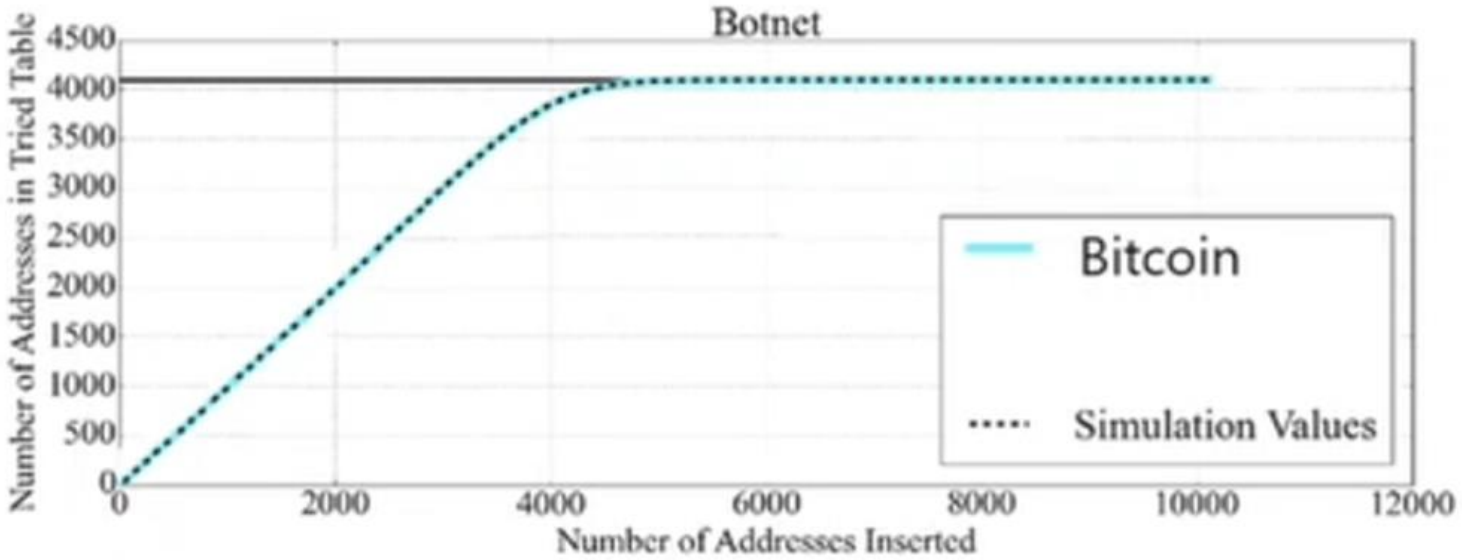
Countermeasure 2: Random Selection:

Randomly select IPs with no bias toward fresher timestamps.

Countermeasures: Deterministic Random Eviction

Vulnerability 2 – Eviction Bias:
Attacker exploited Eviction bias toward older IPs.

Vulnerability 3 – Try Try Again:
Attacker exploited randomness in eviction process to improve odds of stuffing tried table by running the attack multiple times.

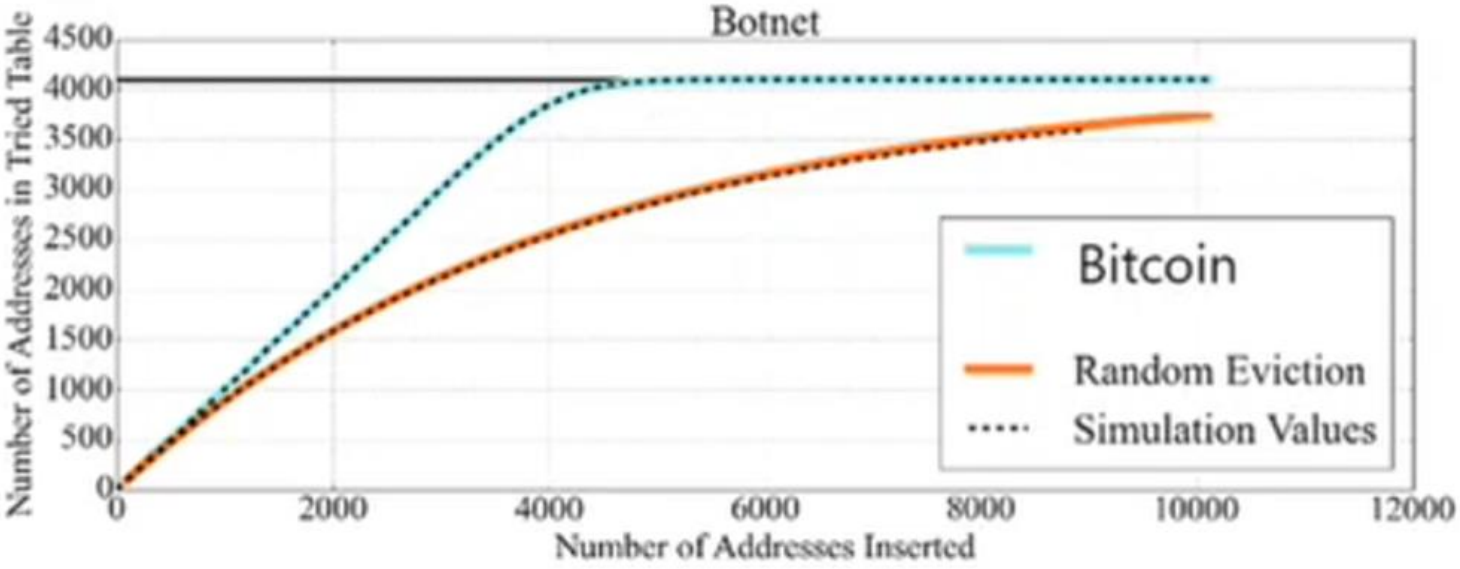


Countermeasure 1: Deterministic Random Eviction:
Deterministically map IPs to buckets and positions in buckets, evicting whatever happens to be in that position.

Countermeasures: Deterministic Random Eviction

Vulnerability 2 – Eviction Bias:
Attacker exploited Eviction bias toward older IPs.

Vulnerability 3 – Try Try Again:
Attacker exploited randomness in eviction process to improve odds of stuffing tried table by running the attack multiple times.



Countermeasure 1: Deterministic Random Eviction:
Deterministically map IPs to buckets and positions in buckets, evicting whatever happens to be in that position.

Countermeasures: Feelers & Test-Before-Evict

Problem:
Tried table fills up very slowly and contains mostly dead IPs.
The fewer honest IPs in tried, the easier the attack is.

Countermeasure 4: Feeler Connections:
Make test connections to IPs in new to fill tried table faster.

Problem:
Good IP addresses from tried table get evicted.

Countermeasure 3: Test Before Evict:
Test IPs in tried before evicting them, if online do not evict.

Inspired by the Storm Botnet's anti-peer-poisoning system.

Countermeasures: Deployment

10.1 1. Deterministic Random Eviction
10.1 2. Random Selection
patch 3. Test-Before-Evict
patch 4. Feeler connections
5. Anchor connections
10.1 6. More buckets
...and more!

We disclosed our attack to the Bitcoin developers,
they took quick action
In Bitcoind v0.10.1 they implemented countermeasures 1,2,6.

We implemented countermeasures 3,4 in a patch,
it is awaiting review.

Countermeasures: How effective?

	No countermeasures	1,2,6 countermeasures Bitcoin 0.10.1	1,2,3,4,6 countermeasures Bitcoin 0.10.1 + patch
Full tried table (worst case)	**4600** IPs **100%** attacker success	**41,000** IPs (model) **~50%** attacker success	test-before-evict keeps attacker IPs out. **0%** attacker success
Live node (298 IPs)	**400** IPs **84%** attacker success	**3,700** IPs (model) **~50%** attacker success	

Better Security

Objections...?

"Miners shouldn't use the P2P network,
they should only connect to trusted parties"

But:

- how do you decide who to trust?
- how do you maintain decentralization?

Our goal:

Make Bitcoin robust to P2P attacks
while preserving decentralization.

Summary

- **Eclipse attacks violate Bitcoin's core security guarantees.**
  - N-Confirmation double spends,
  - 51% attacks with less than 51% of the mining power,
  - and more.

- **We develop practical attacks:**
  - A Botnet of 400 IPs is sufficient to eclipse our live Bitcoin nodes
  - In an attackers worst case a botnet of 4600 IPs will eclipse a node

- **We have effective countermeasures to resist these attacks:**
  - Some of the countermeasures have already been deployed in Bitcoin
  - Others are awaiting review