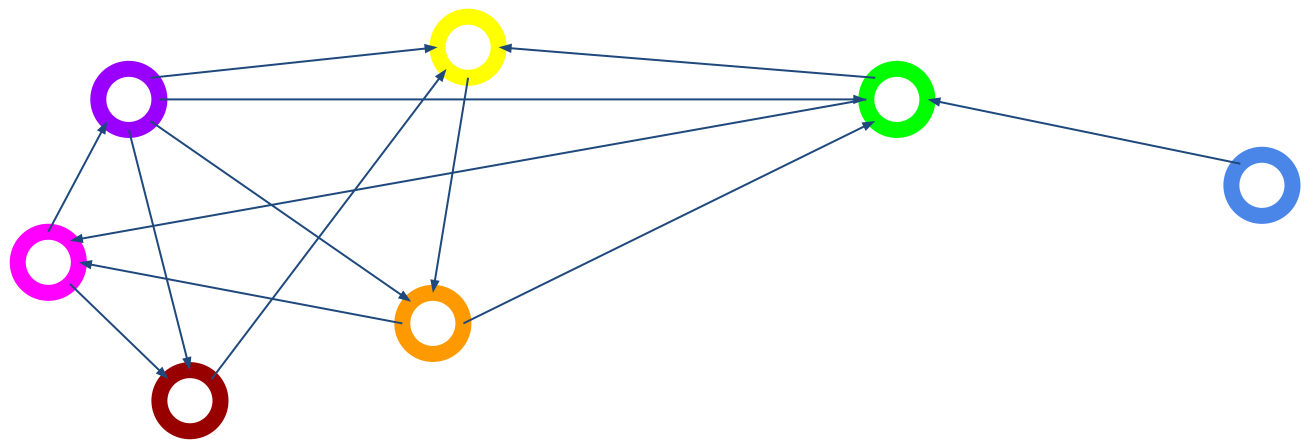
Bitcoin’s P2P Network

- Bitcoin nodes use a peer-to-peer network to announce and learn about transactions and blocks?

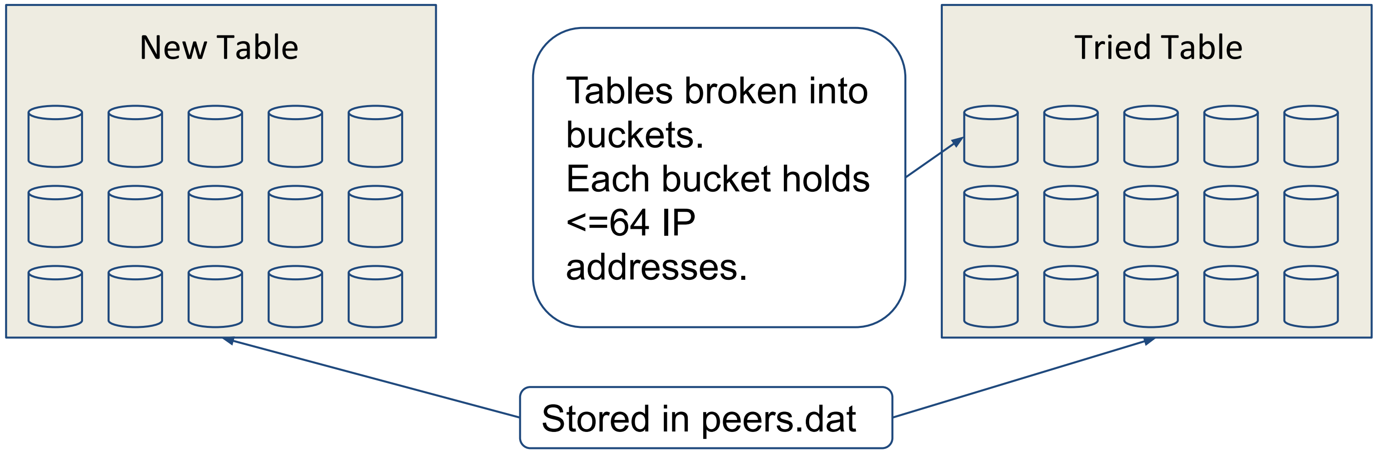
- By default a Bitcoin node has:
  - **8 Outgoing connections** (TCP conn initiated by the node)
  - **Up to 116 Incoming connections** (TCP conn initiated by another node)
  - **1 Feeler connection** (short-lived outgoing connection)

- Every outgoing connection is another nodes incoming connection



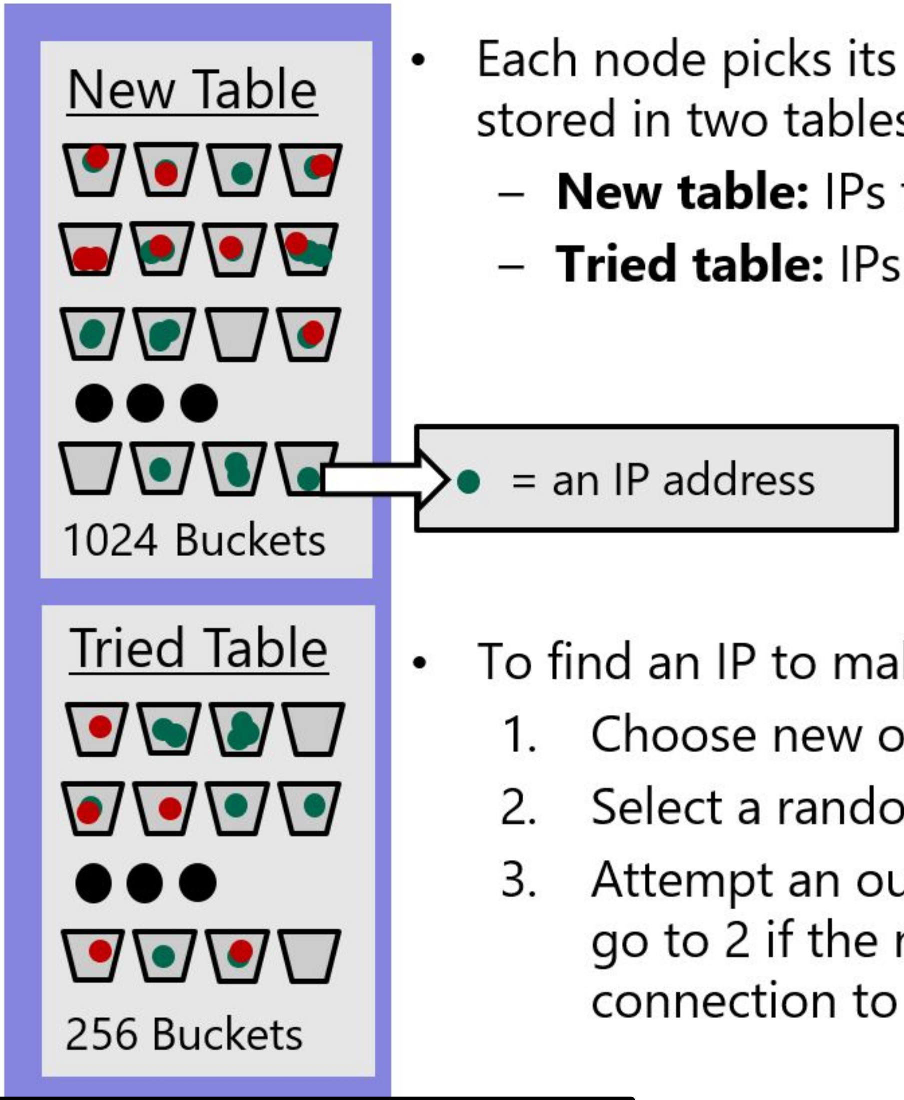
Bitcoin's P2P Network

- Nodes store the IP addrs of other nodes in two tables:
  - **New table** - IP addrs heard about but not connected to
  - **Tried table** - IP addrs we have connected to


New Table
Tables broken into buckets.
Each bucket holds <=64 IP addresses.
Tried Table
Stored in peers.dat

- **The node will:**
  1. randomly choose the new or tried table
  2. sample a node from the chosen table and connect to it
  3. if successful and the node has 8 outgoing connections halt, otherwise go to 1.

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.

- To find an IP to make an outgoing connection to:
  1. Choose new or tried tables to select from.
  2. Select a random bucket and IP from that bucket
  3. Attempt an outgoing connection to that IP go to 2 if the node already has an outgoing connection to that /16 or x.x.*.*

This is managed by addrman

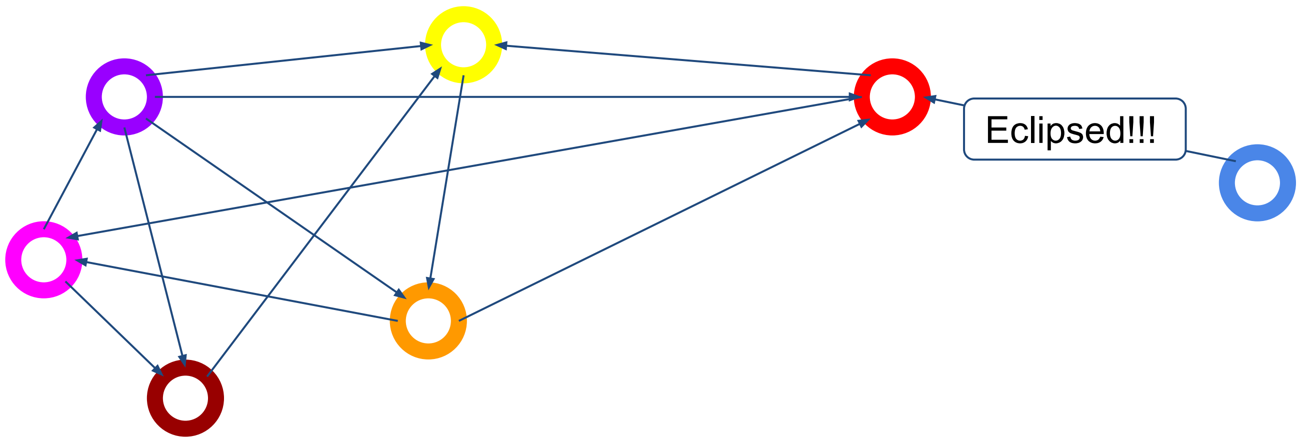
Bitcoin's P2P Network

Why should we care about what the P2P network?

Bitcoin assumes that users have an accurate view of the blockchain via the P2P network

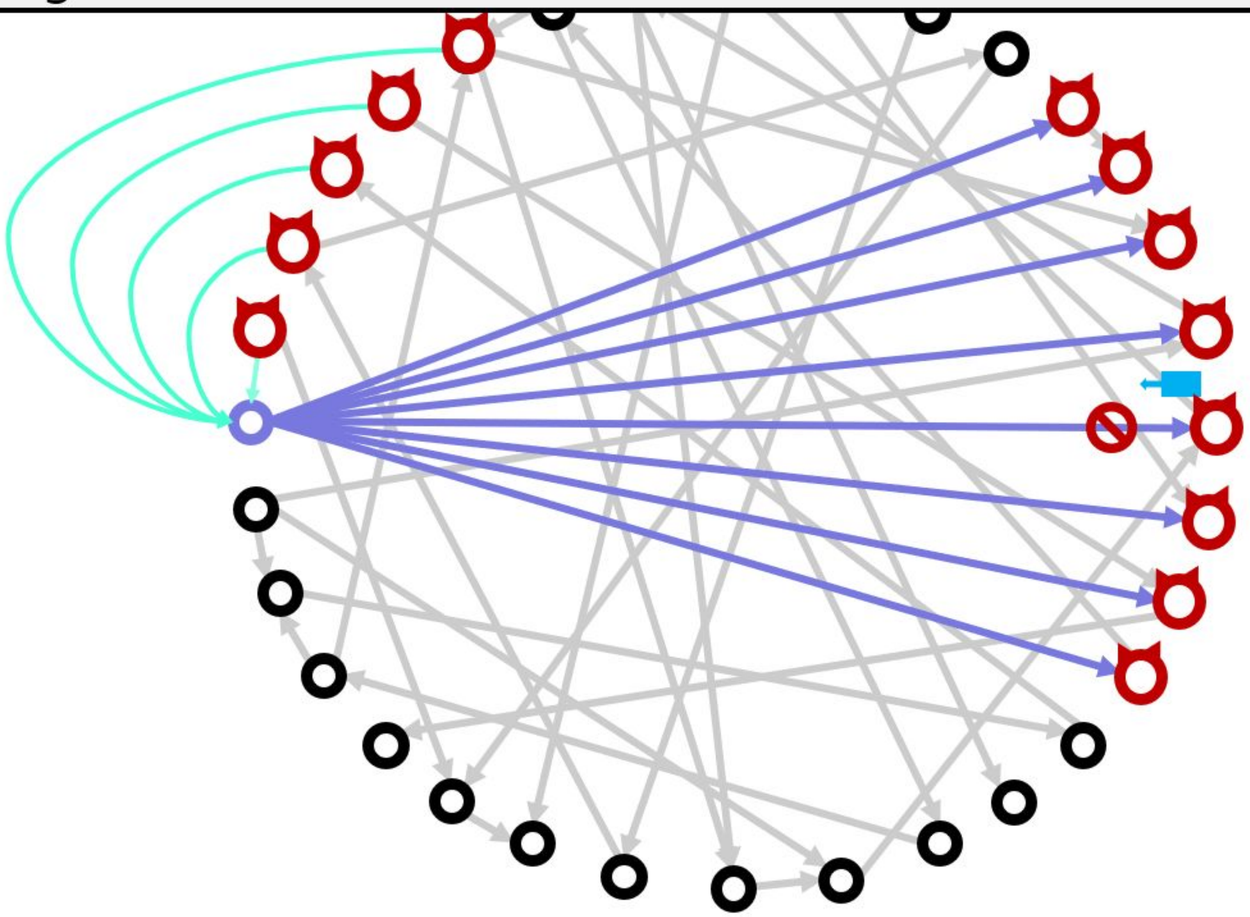
Let's look at what happens when this assumption is violated

Attacks in which a malicious party controls a users access to information in a P2P network is called an eclipse attack.



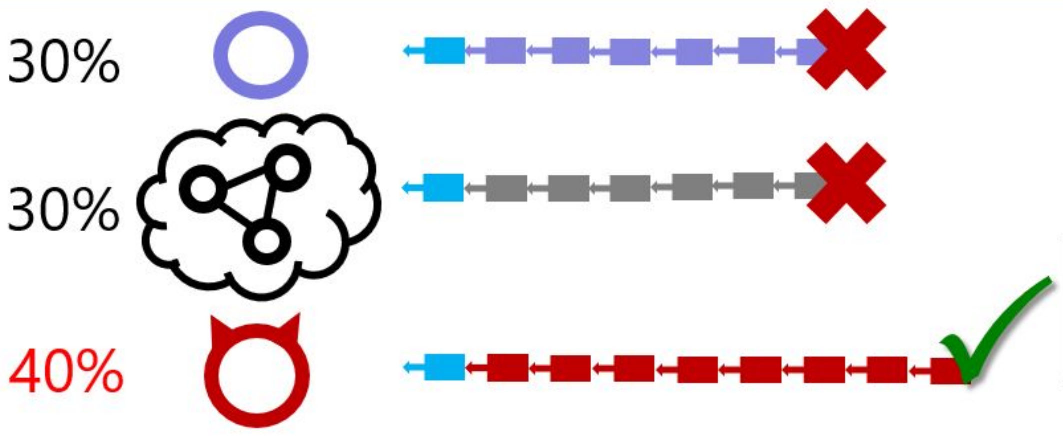
Eclipse Attacks

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

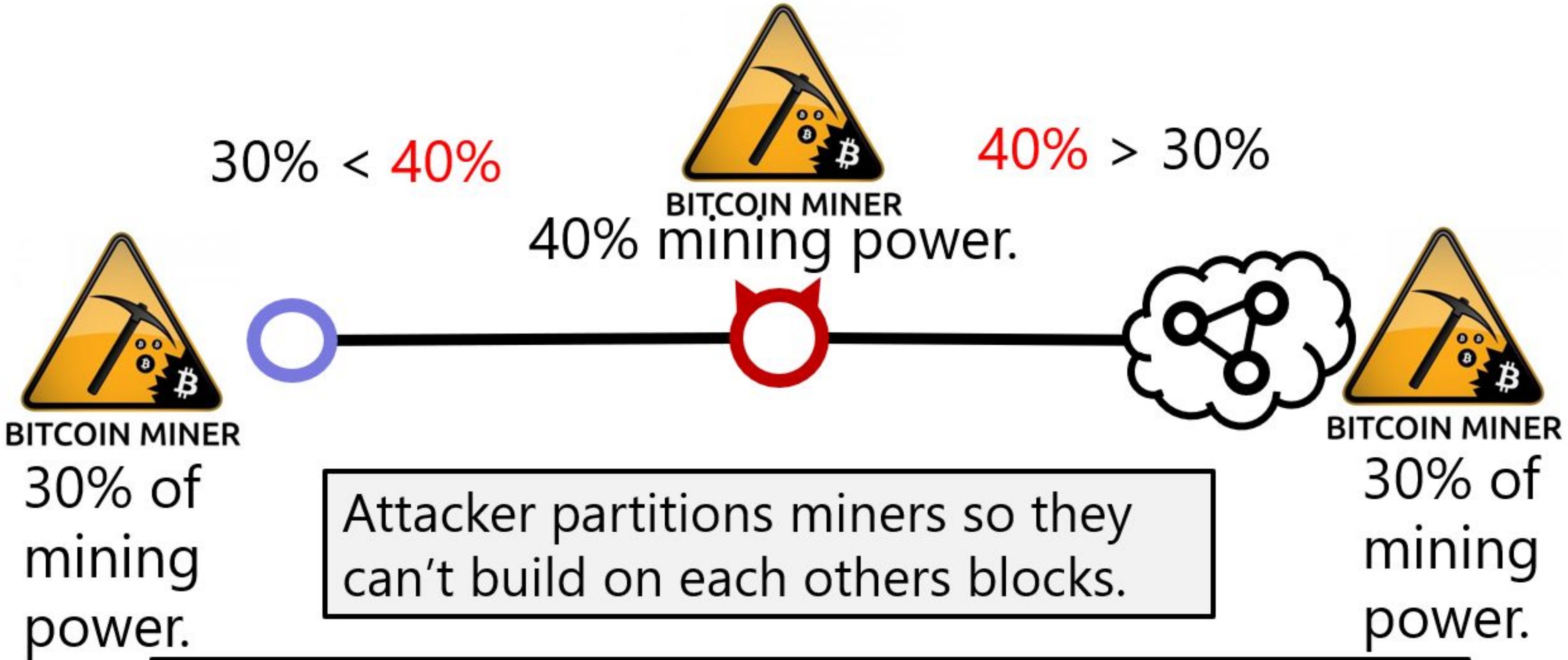


By manipulating the P2P network, the attacker eclipses the node.

Example 1: 51% attack with 40% mining power



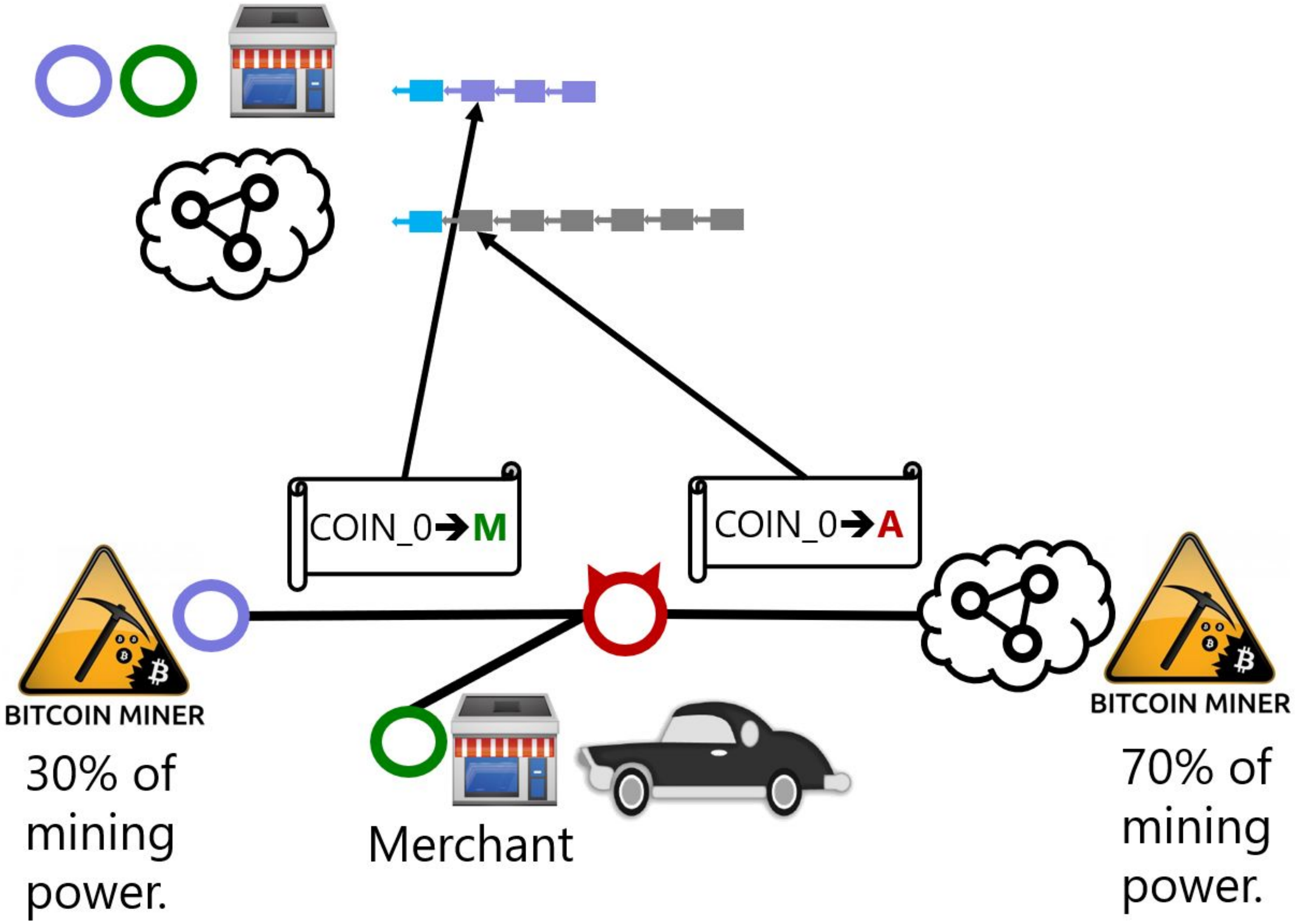
Attacker chain becomes the consensus chain.



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

The attacker then outcompetes each partitioned miner.

Example 2: N-Confirmation Double Spending



Eclipse attacks: Other ramifications

- Attacks on Layer 2 protocols (Lightning):
  Censor a breach-remedy transaction to steal funds.

- Privacy:
  Determine if a node originated a transaction

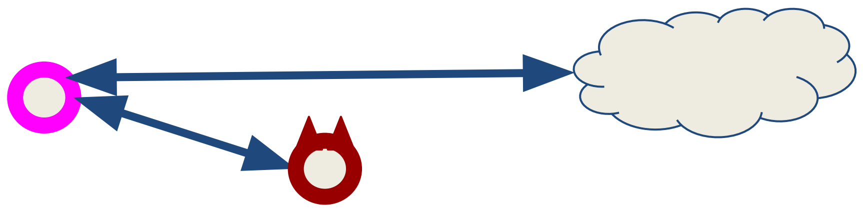
- Forks:
  If a fork occurs the attacker can double spend by showing the victim the losing side of the fork.

How can this happen?

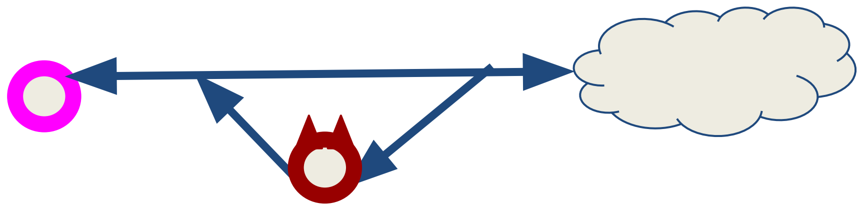
- On-path or in-path attacks
- Off-path attacks that manipulate the P2P network
- DNS attacks poison the tables of a new Bitcoin node
- **...Then** we will look at other bad things
  that can happen

What about on/in-path attackers

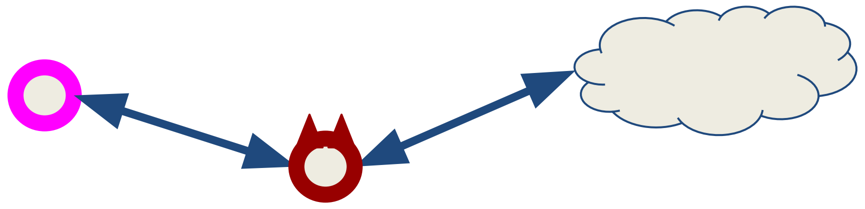
- **Off-path:** Attacker can send messages to victim and receive **<u>only</u>** messages sent to attacker



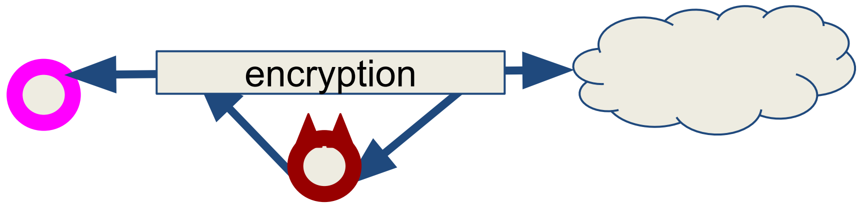
- **On-path:** Attacker can send messages to victim, and receive **<u>all</u>** messages victim sends to anyone.



- **In-path:** Attacker can send messages to victim, receive all messages victim sends to anyone, and selectly drop messages to/from victim.



Security against on-path attackers



- **On-path defense:** encrypt the connection?

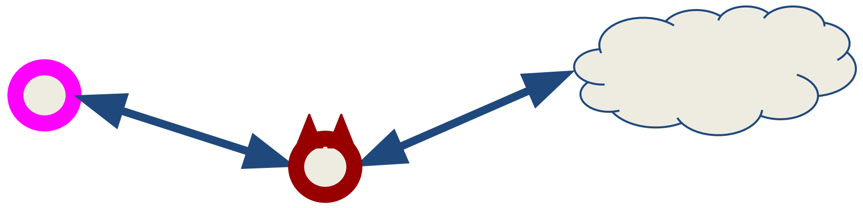
Let’s talk about this a bit

GFW and NSA’s QUANTUM limited to this.
<u>On-path attacker can send pkts that interfere/drop other parties TCP pkts</u>

Bitcoin currently has no on-path mitigations

And on-path attacker can manipulate the P2P network to become in-path

Security against in-path attackers



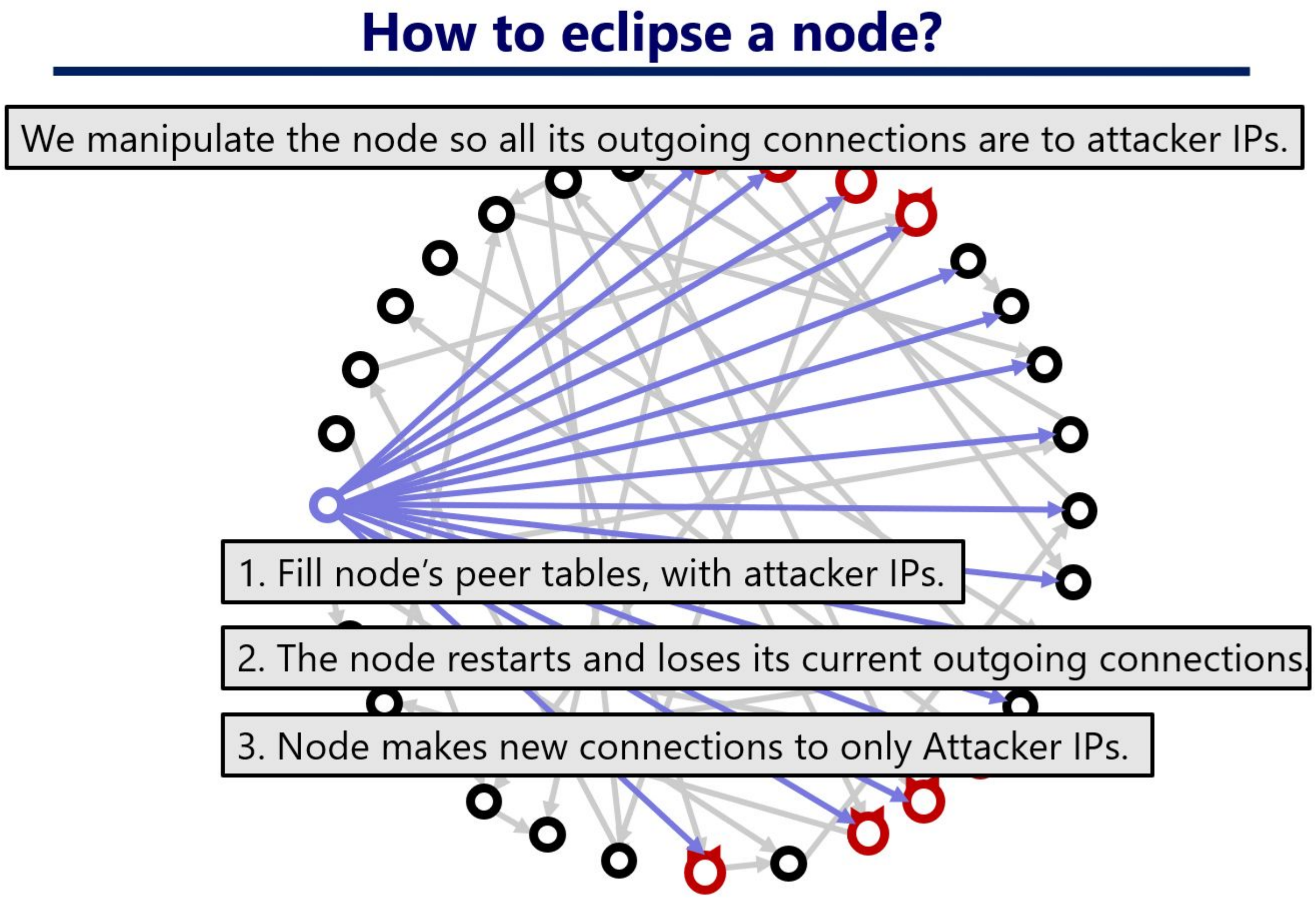
- **In-path:** Attacker can send messages to victim, receive all messages victim sends to anyone, and selectly drop messages to/from victim.

In-path is more expensive than on-path

Bitcoin currently has no in-path mitigations.
Mitigating in-path attacks is extremely hard.

"The battle between intelligence and deception is the efforts of one side to establish as many channels as possible through which to observe the opponent, in the hope that he may fail to block at least some of these channels, while the opponent may in addition try to send false, and preferably consistent, signals in as many channels as possible." R.V Jones

Off-path attacks: table stuffing



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.

How easy are restarts?

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

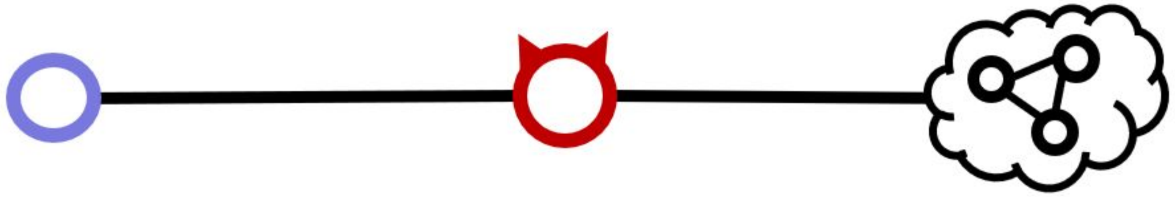
- **Software/security updates:**
  - predictable for the attacker, users are notified of upcoming updates,
  - lose/lose for the victim, restart or remain vulnerable.
- **Packets of death/DoS Attacks:**
  - Ten DoS CVEs in Bitcoin[1], many more on underlying OSes.
- **Power/network failures:**
  - Bitcoin 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.

Eclipse Attacks



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

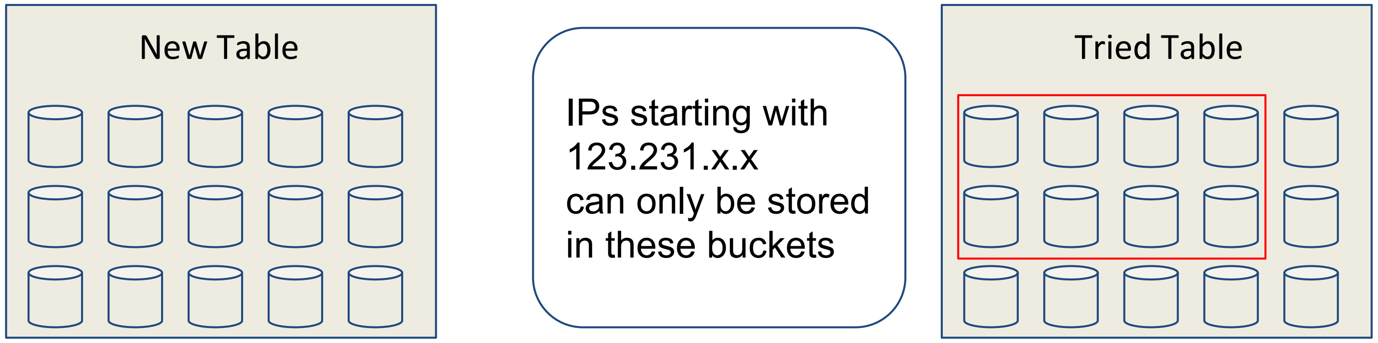
What can an attacker do if they can eclipse a node?

Defenses against peer table manipulation

Assumption: Attacker may only have access to contiguous IP addresses.

- Ensure all outgoing connections are made to different /16s
  this means the attacker needs to control at least 8 IPs in 8 different /16s

- In the tried/new tables limit the buckets a particular /16 can be stored in


New Table
IPs starting with 123.231.x.x can only be stored in these buckets
Tried Table

Why does this provide security?

Defenses against peer table manipulation cont...

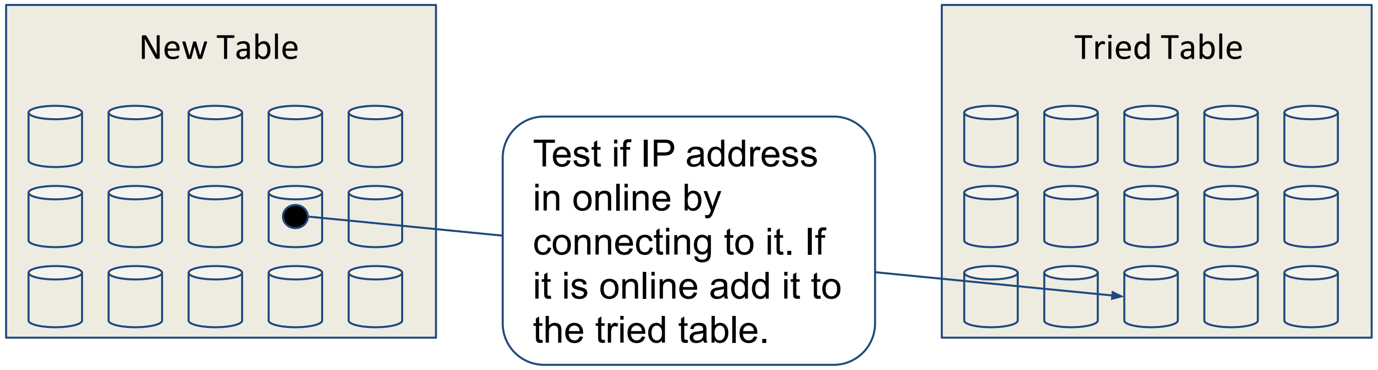
Assumption: IPs in table are honest, malicious IPs are added later

- Reasonable to assume since if attacker already controls the tables attacker has already won
- ...thus we don't evict a IP address from the tried table without first checking if that IP is online. If it is online we don't evict it.
  We call this defense test-before-evict

Assumption: The new table is easy to fill up with trash IP addresses. It is not a defensible position, but the tried table requires that the node actually be online. Thus it is more expensive for an attacker.

- The more honest IP addresses we have in the tried table the more malicious IP addresses the attacker must have as well.
- To increase the number of IP addresses in the tried table we use a defense feeler-connections.
  Feeler connections test IP addresses in the new table and add them to the tried table.

Feeler connections

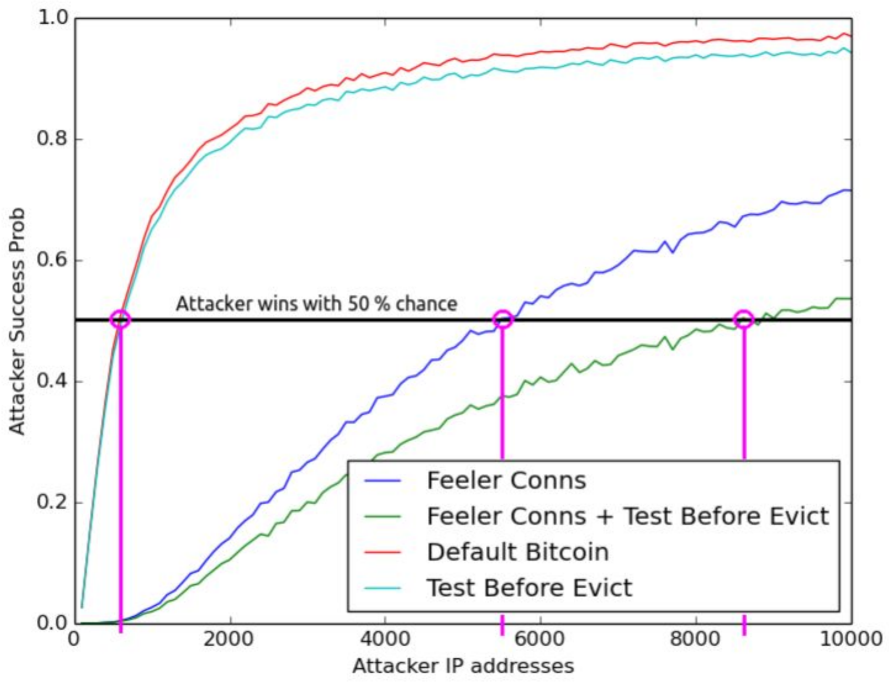

New Table
Tried Table
Test if IP address in online by connecting to it. If it is online add it to the tried table.

How many IPs?

Filling the new table is easy:
Fill it with made up "trash" IPs.

Filling the tried table depends on:
# of honest IPs in tried
# of attacker IPs in different \16s

Feeler connections: increase the number of honest IPs in tried by testing IPs in new and moving them to tried.
Currently Deployed!



Default node: 595 attacker IPs for ~50% attack success.
Default node + test-before-evict: 620 attacker IPs for ~50% attack success.
Feeler node: 5540 attacker IPs for ~50% attack success.
Feeler node + test-before-evict: 8600 attacker IPs for ~50% attack success.

Defenses against incoming connections cont...

To eclipse a node you need to control both incoming and outgoing connections

To defend against both these attacks and a DoS connection exhaustion attack Bitcoin allows new incoming connections to evict old incoming connections

Evicting incoming connections is dangerous because they can be used to partition manipulate the p2p network for this reason Bitcoin only makes some of the incoming connections evictable

Open questions:

1. How many IP addresses would you need to do a connection exhaustion attack?

2. How easy is it to monopolize all the incoming connections to a node? Can you just blast it with lots of connections?

Rules for evicting an incoming connection

Create a list of all incoming connections

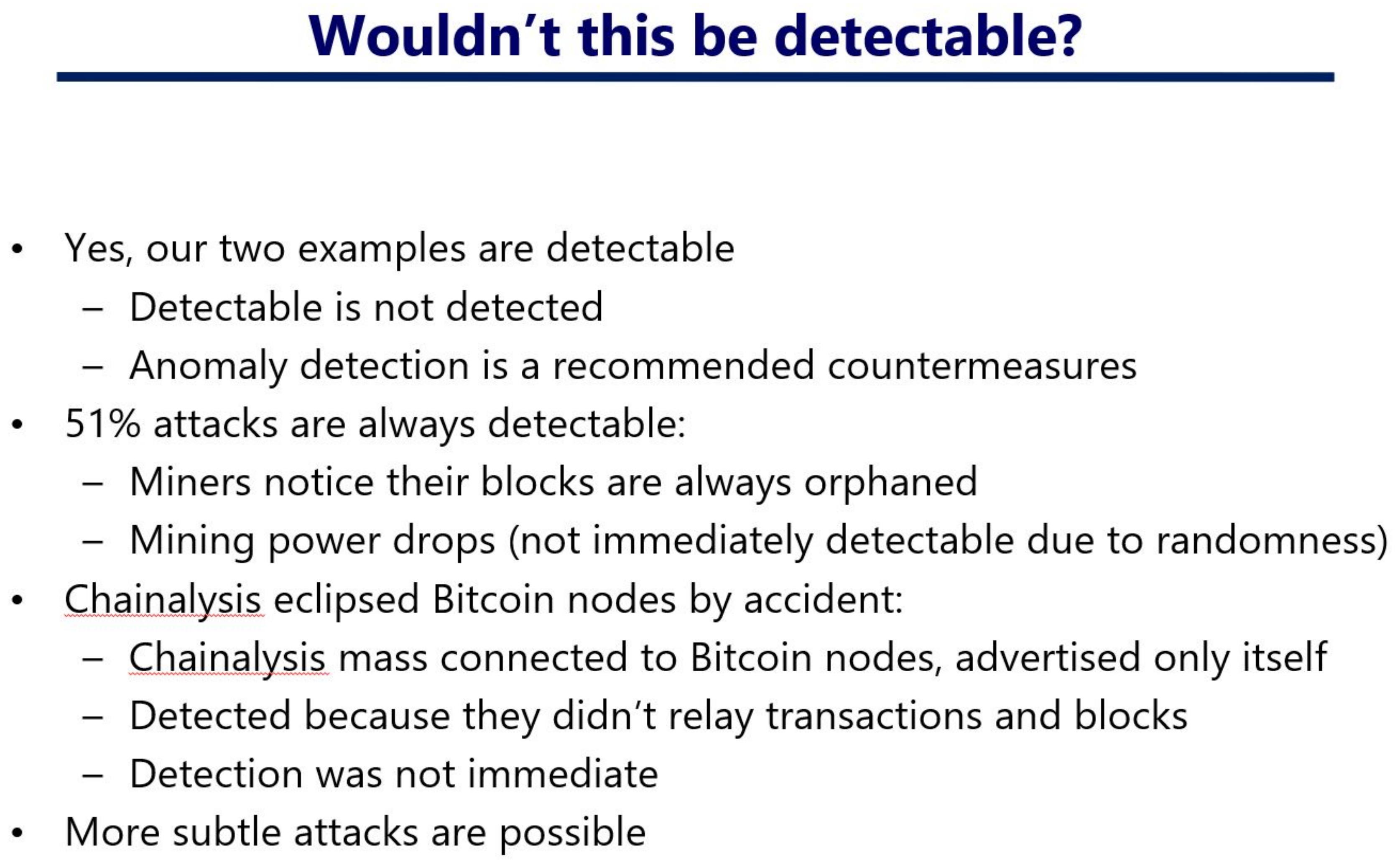
1. Remove up to 4 IPs with a particular \16 (a.k.a netgroup)
2. Remove the 8 IPs with the lowest ping time
3. Remove the 4 IPs that most recently sent us transactions
4. Remove the 4 IPs that most recently sent us blocks
5. Remove oldest connections (50% of the list)

If any members on the list are have the prefer evict set, return that IP to be evicted.

From what remains on the list select the \16 (a.k.a.) net group with the most connections and evict the youngest connection from that net group.

From https://github.com/bitcoin/bitcoin/blob/df7addc4c6e990141869c41decaf3ef98c4e45d2/src/net.cpp#L857

Detection as a defense



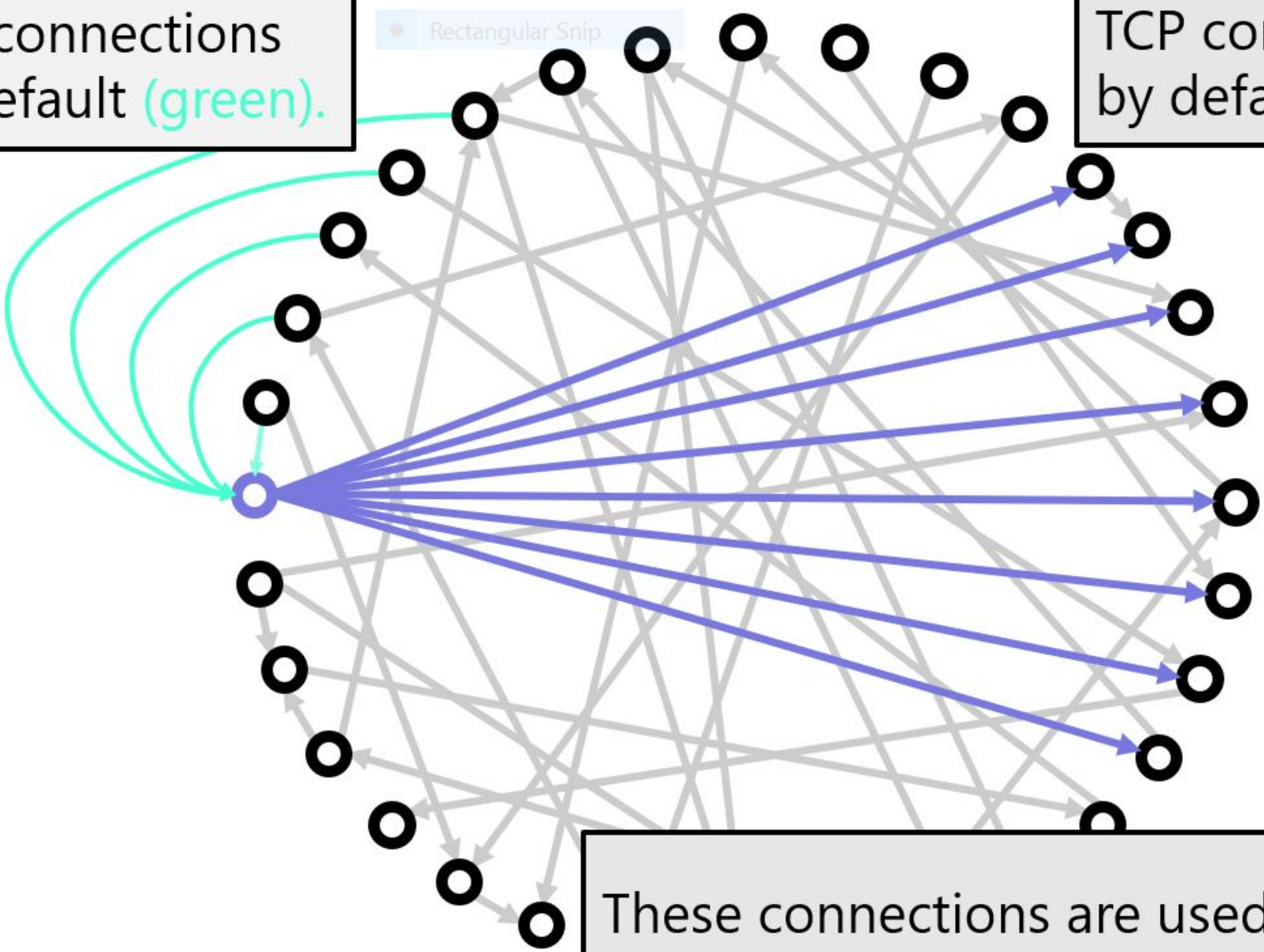
Wouldn’t this be detectable?

- Yes, our two examples are detectable
  - Detectable is not detected
  - Anomaly detection is a recommended countermeasures
- 51% attacks are always detectable:
  - Miners notice their blocks are always orphaned
  - Mining power drops (not immediately detectable due to randomness)
- Chainalysis eclipsed Bitcoin nodes by accident:
  - Chainalysis mass connected to Bitcoin nodes, advertised only itself
  - Detected because they didn’t relay transactions and blocks
  - Detection was not immediate
- More subtle attacks are possible

Bitcoin's Peer-to-Peer Network

Max 117 incoming TCP connections by default (green).

Max 8 outgoing TCP connections by default (purple).



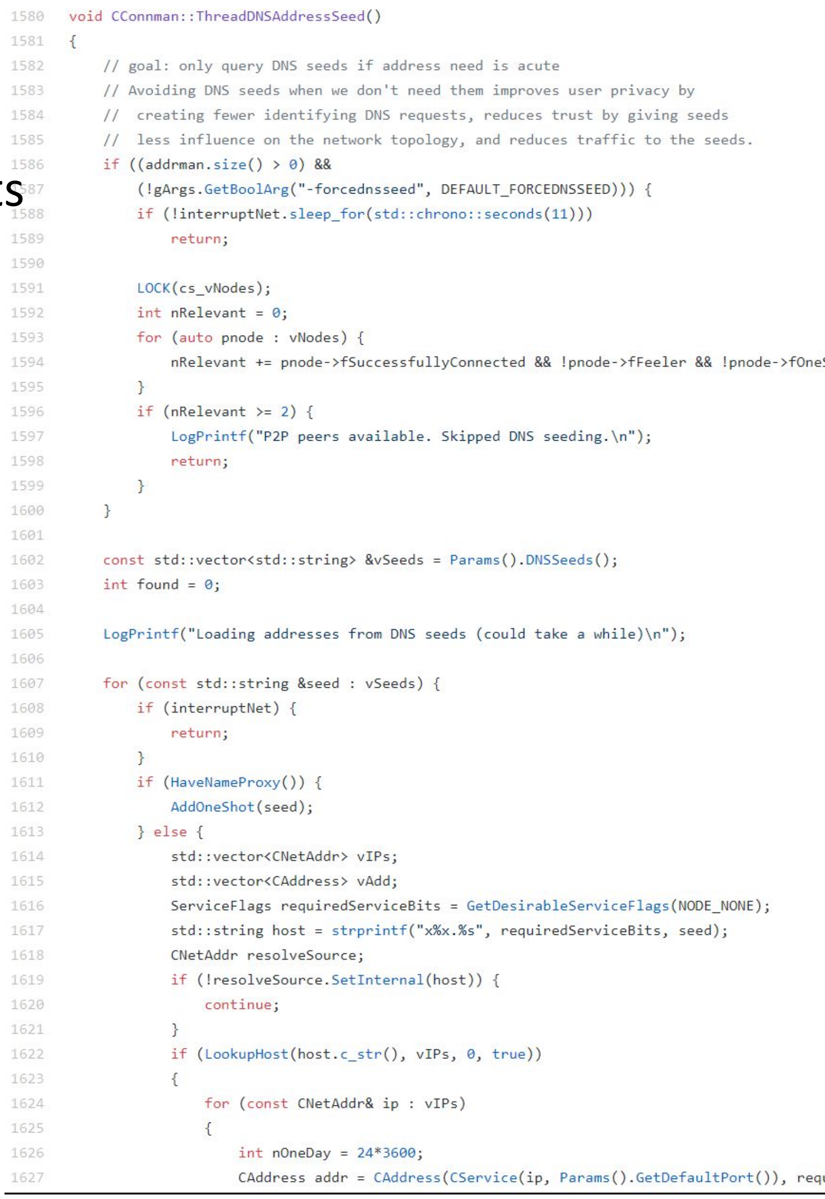
These connections are used to form a P2P gossip network to propagate bitcoin transactions and blocks (←).

Off-path attacks: DNS

How to Eclipse a node: DNS

When node is started:
If new and tried tables are empty
then: populate new table via DNS requests
The seed domains are:

```
seed.bitcoin.sipa.be
dnsseed.bluematt.me
dnsseed.bitcoin.dashjr.org
seed.bitcoinstats.com
seed.bitcoin.jonasschnelli.ch
seed.btc.petertodd.org
```



```
void CConnman::ThreadDNSAddressSeed()
{
    // goal: only query DNS seeds if address need is acute
    // Avoiding DNS seeds when we don't need them improves user privacy by
    //  creating fewer identifying DNS requests, reduces trust by giving seeds
    //  less influence on the network topology, and reduces traffic to the seeds.
    if ((addrman.size() > 0) &&
        (!gArgs.GetBoolArg("-forcednsseed", DEFAULT_FORCEDNSSEED))) {
        if (!interruptNet.sleep_for(std::chrono::seconds(11)))
            return;

        LOCK(cs_vNodes);
        int nRelevant = 0;
        for (auto pnode : vNodes) {
            nRelevant += pnode->fSuccessfullyConnected && !pnode->fFeeler && !pnode->fOne
        }
        if (nRelevant >= 2) {
            LogPrintf("P2P peers available. Skipped DNS seeding.\n");
            return;
        }
    }

    const std::vector<std::string> &vSeeds = Params().DNSSeeds();
    int found = 0;

    LogPrintf("Loading addresses from DNS seeds (could take a while)\n");

    for (const std::string &seed : vSeeds) {
        if (interruptNet) {
            return;
        }
        if (HaveNameProxy()) {
            AddOneShot(seed);
        } else {
            std::vector<CNetAddr> vIPs;
            std::vector<CAddress> vAdd;
            ServiceFlags requiredServiceBits = GetDesirableServiceFlags(NODE_NONE);
            std::string host = strprintf("x%x.%s", requiredServiceBits, seed);
            CNetAddr resolveSource;
            if (!resolveSource.SetInternal(host)) {
                continue;
            }
            if (LookupHost(host.c_str(), vIPs, 0, true))
            {
                for (const CNetAddr& ip : vIPs)
                {
                    int nOneDay = 24*3600;
                    CAddress addr = CAddress(CService(ip, Params().GetDefaultPort()), req
```

Eclipse new nodes to the network via DNS:

Attack 1: Control some of the seeders

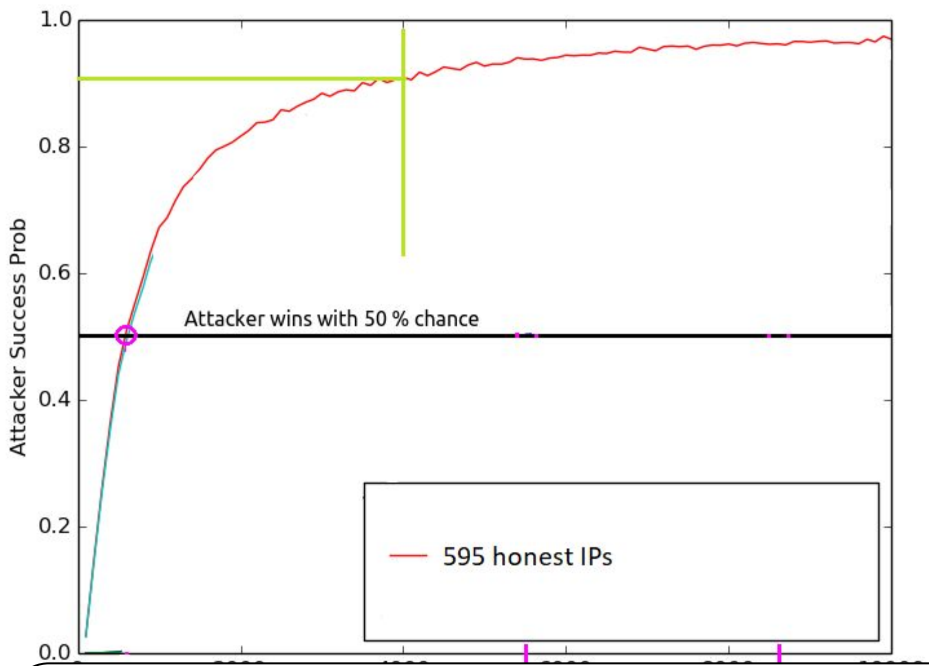
Attack 2: Control the local DNS server

Attack 3: DNS cache poisoning? Has anyone tried this?

Attack 4: Stuffing seeders with attacker IPs?

Attack 1: Controlling some seeders

If an attacker controls `seed.bitcoin.sipa.be` it can return ~4,000 IPs
I tested dnsseed.bluematt.me it returned ~32 records
5*32 = 160 Honest IP addresses
1*5,956 = 4000 Attacker IP addresses

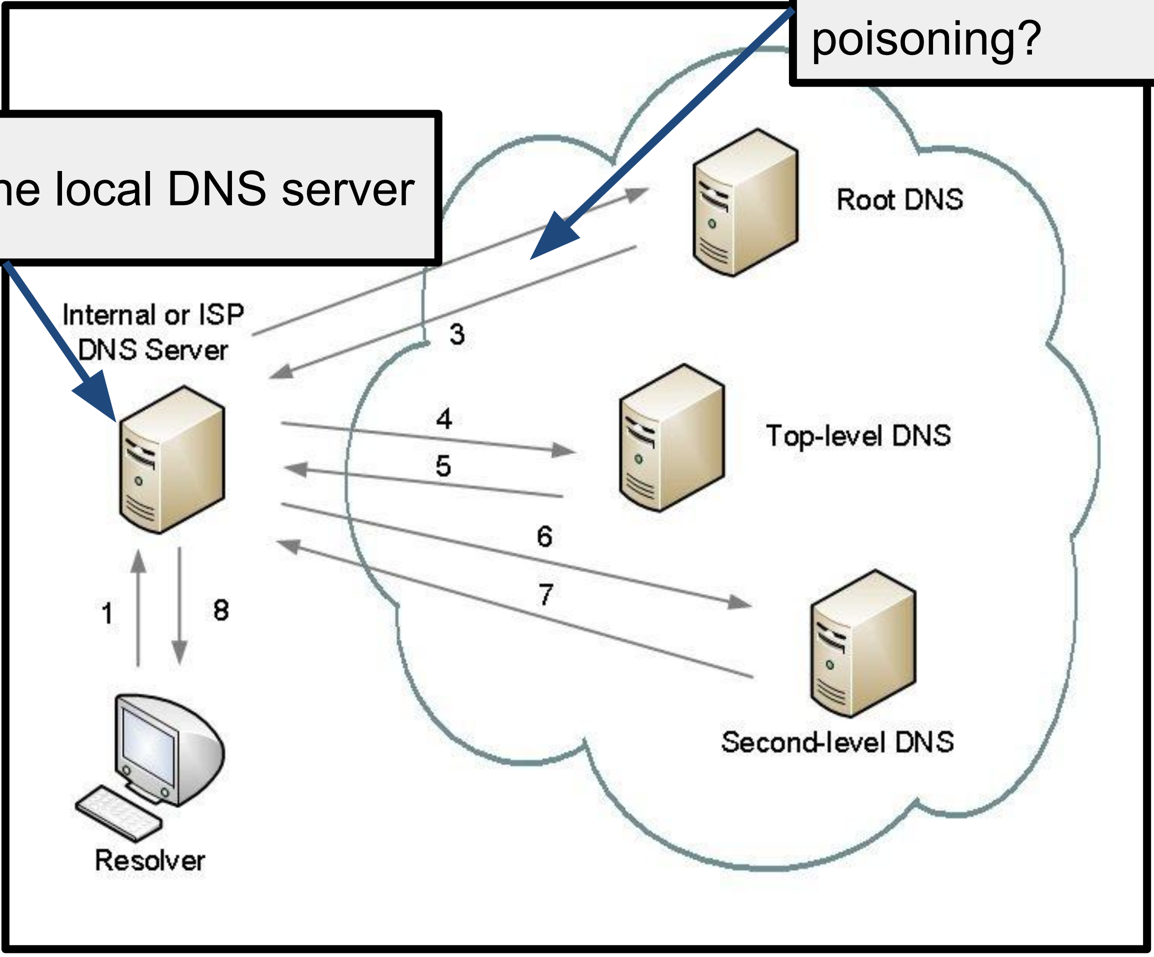


This fixed! Bitcoin now limits 256 IPs per DNS seeder

Attack 2 & 3

Attack 3: DNS cache poisoning?

Attack 2: Control the local DNS server



How to fix? Authenticate seeders? Turn off DNS caching?

Discussion

- How important is the Peer-to-Peer network? Should we be investing energy in hardening it further against eclipse attacks.
- Do we need to worry about DNS attacks?
- What about the threat of On-path attackers? Can we leverage the diff. between In-path and On-path to prevent MiTM attacks? If an On-path attacker can set up an Eclipse attack they gain abilities similar to an In-path attacker.
- How does the Fibre network impact this?
- What are the privacy implications of an eclipse attack? Does Tor make eclipse attacks easier?

Project ideas

- Harden Seeders/DNS against attack
- Look at how crypto might make on-path attacks harder
- Look at how UDP might make on-path attacks harder
- Can you use TLS/HTTPS to block explorers mitigate an eclipse attacks?
- Is the incoming connection logic optimal/vulnerable?
  Any way to game it for an attack?