Artifact Evaluated
NDSS SYMPOSIUM
Available
Functional
Reproduced

Content Censorship in the InterPlanetary File System

Srivatsan Sridhar
(Stanford University)

Onur Ascigil
(Lancaster Univ.)

Navin Keizer
(Univ. College London)

François Genon
(UCLouvain)

Sébastien Pierre
(UCLouvain)

Yiannis Psaras
(Protocol Labs)

Etienne Rivière
(UCLouvain)

Michał Król
(City, Univ. of London)

Network and Distributed System Security (NDSS) Symposium 2024

27th February 2024

InterPlanetary File System

IPFS

https://ipfs.tech/

InterPlanetary File System

IPFS

https://ipfs.tech/

- Largest decentralized peer-to-peer filesystem

InterPlanetary File System

https://ipfs.tech/

- Largest decentralized peer-to-peer filesystem
- 25,000 peers active every day

InterPlanetary File System

https://ipfs.tech/

- Largest decentralized peer-to-peer filesystem
- 25,000 peers active every day
- >1 billion files stored

InterPlanetary File System

https://ipfs.tech/

- Largest decentralized peer-to-peer filesystem
- 25,000 peers active every day
- >1 billion files stored
- Key storage platform for decentralized apps (dApps) and NFTs

InterPlanetary File System

https://ipfs.tech/

- Largest decentralized peer-to-peer filesystem
- 25,000 peers active every day
- >1 billion files stored
- Key storage platform for decentralized apps (dApps) and NFTs
- Open participation

Contributions in a Nutshell

Contributions in a Nutshell

😱 Censorship attack exploiting the Kademlia DHT (AWS cost: **16¢/hour**)

Contributions in a Nutshell

😱 Censorship attack exploiting the Kademlia DHT (AWS cost: **16¢/hour**)

🔍 Statistical detection of **>99%** attacks

Contributions in a Nutshell

- 😱 Censorship attack exploiting the Kademlia DHT (AWS cost: **16¢/hour**)

- 🔍 Statistical detection of **>99%** attacks

- 🛡️ Mitigation of **100%** attacks; no overhead when no attack

Contributions in a Nutshell

- 😱 Censorship attack exploiting the Kademlia DHT (AWS cost: **16¢/hour**)
- 🔍 Statistical detection of **>99%** attacks
- 🛡️ Mitigation of **100%** attacks; no overhead when no attack
- 🌐 Experiments on the live IPFS network

Contributions in a Nutshell

😱 Censorship attack exploiting the Kademlia DHT (AWS cost: **16¢/hour**)

🔍 Statistical detection of **>99%** attacks

🛡️ Mitigation of **100%** attacks; no overhead when no attack

🌐 Experiments on the live IPFS network

🧑‍💻 Implementation to be integrated in IPFS

Content Resolution in IPFS

P. Maymounkov, D. Mazières, “Kademlia: A Peer-to-Peer Information System Based on the XOR Metric”

Content Resolution in IPFS

Distributed Hash Table (DHT)
Kademlia

P. Maymounkov, D. Mazières, “Kademlia: A Peer-to-Peer Information System Based on the XOR Metric”

Content Resolution in IPFS

Distributed Hash Table (DHT)
Kademlia

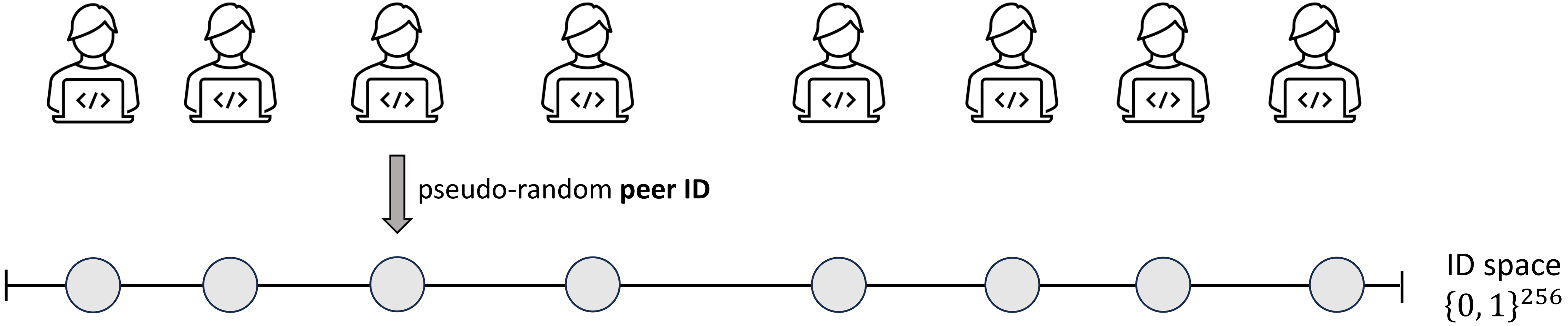
Distributed key-value store
Content ID → Content provider’s address

P. Maymounkov, D. Mazières, “Kademlia: A Peer-to-Peer Information System Based on the XOR Metric”

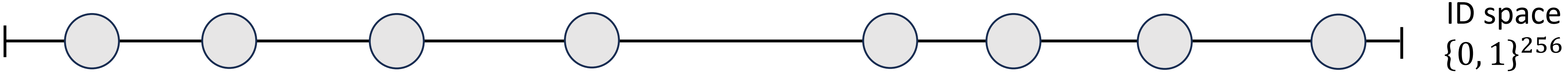
Content Resolution in IPFS

Distributed Hash Table (DHT)
Kademlia

Distributed key-value store
Content ID → Content provider’s address

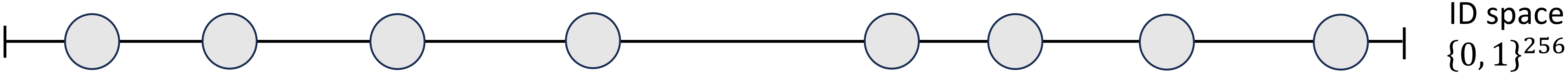


Kademlia in IPFS



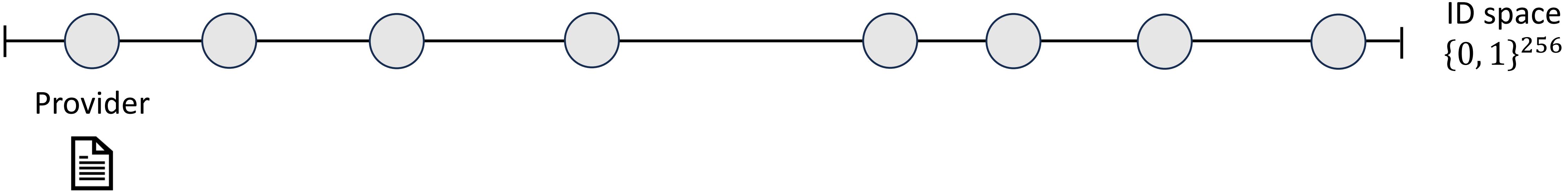
Kademlia in IPFS

Provide content:



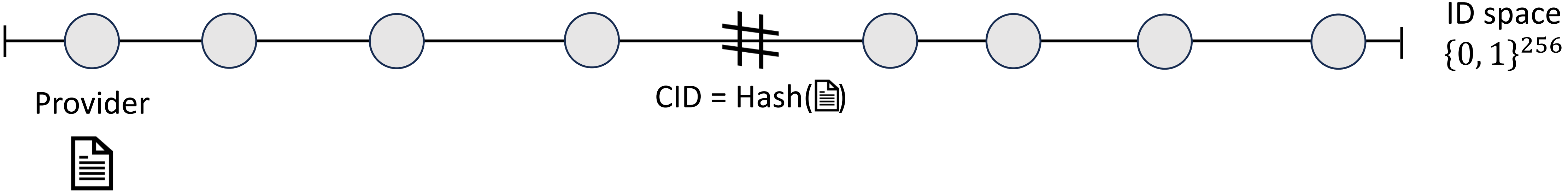
Kademlia in IPFS

Provide content:



Kademlia in IPFS

Provide content:

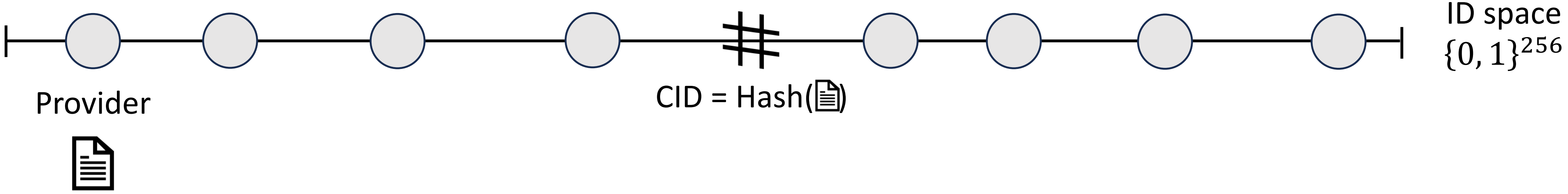


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

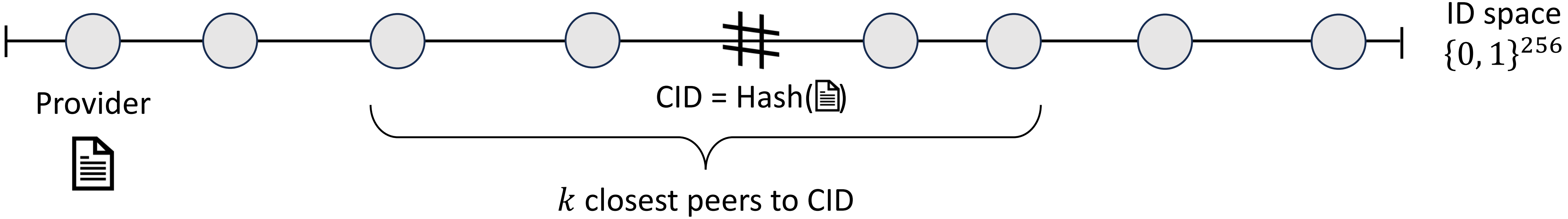


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

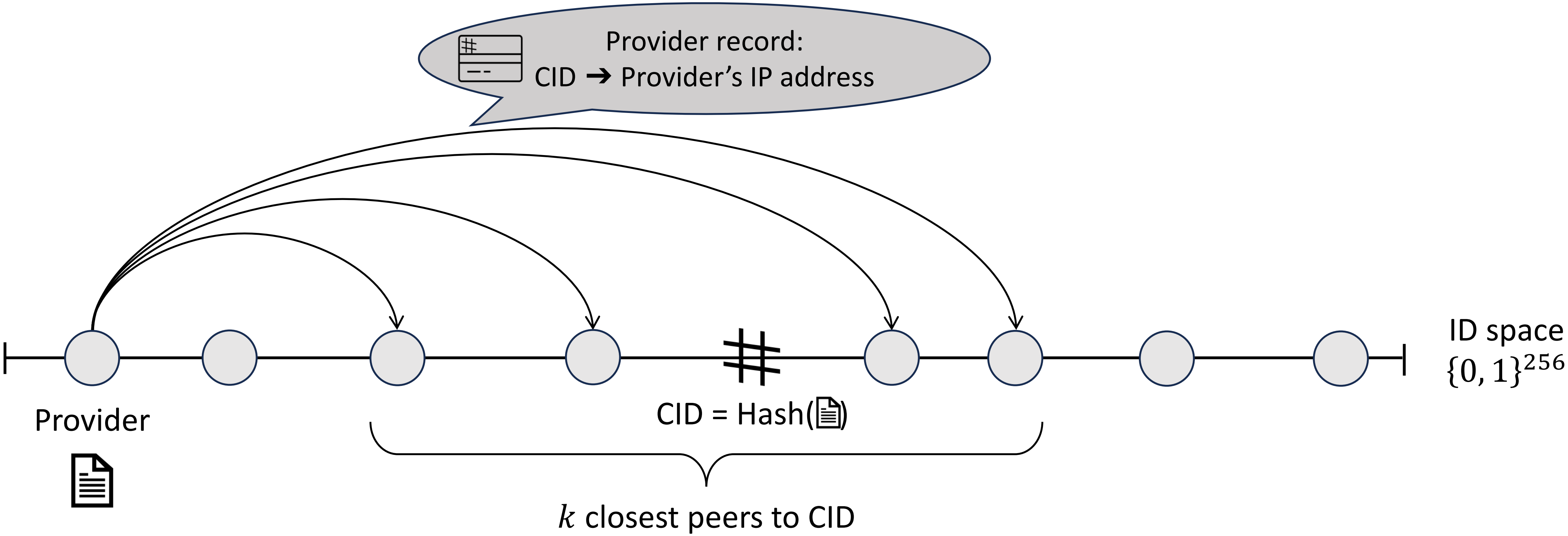
```
dist(id1,id2)
= id1 XOR id2
```



Kademlia in IPFS

Provide content:

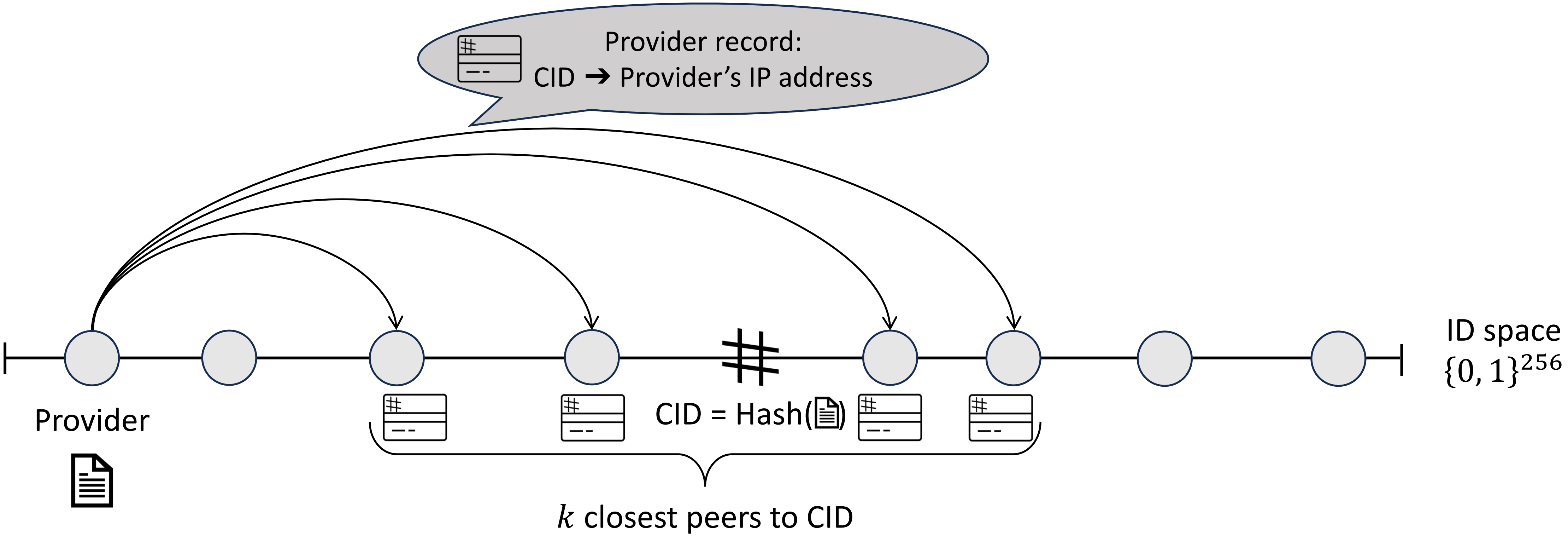
1. Find k (=20) closest peers to CID.
2. Send provider record to them.



Kademlia in IPFS

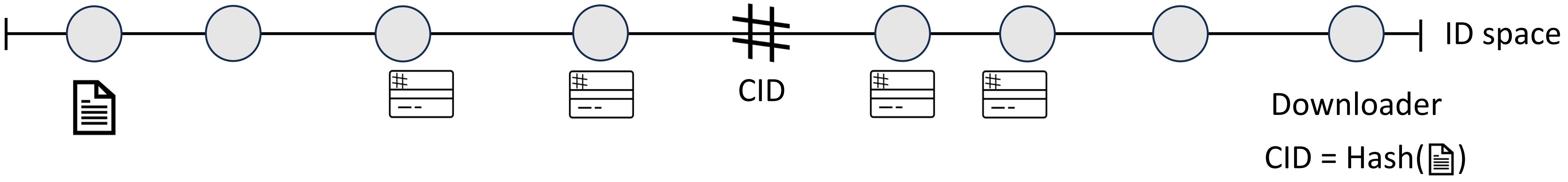
Provide content:

1. Find k (=20) closest peers to CID.
2. Send provider record to them.



Kademlia in IPFS

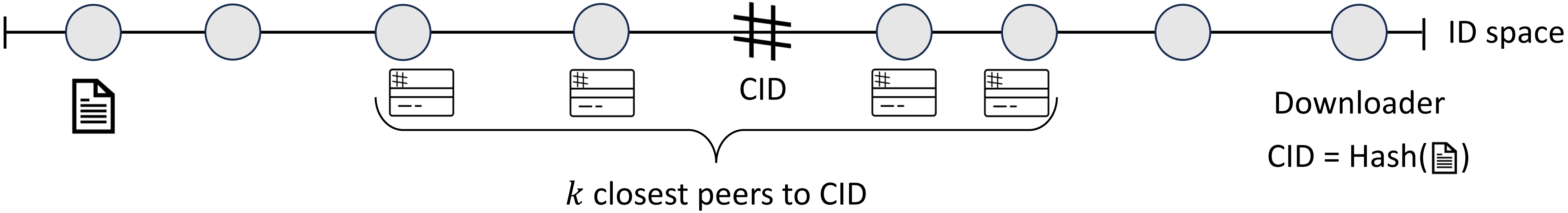
Download content:



Kademlia in IPFS

Download content:

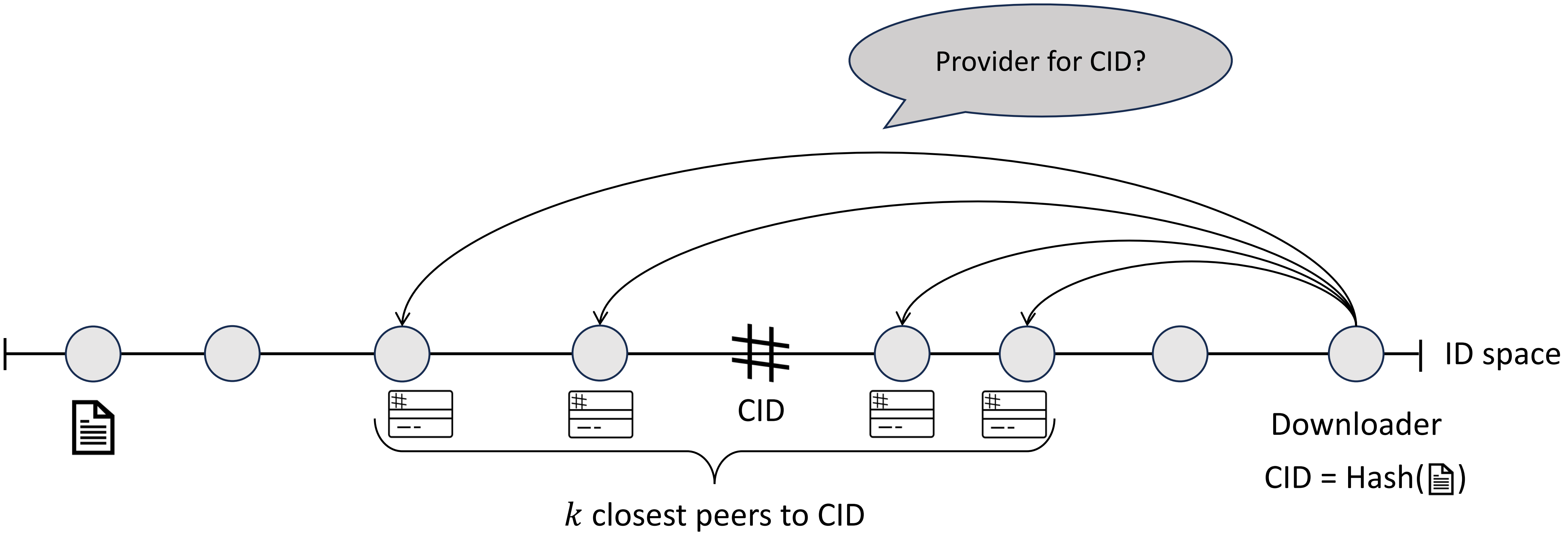
1. Find k (=20) closest peers to CID.



Kademlia in IPFS

Download content:

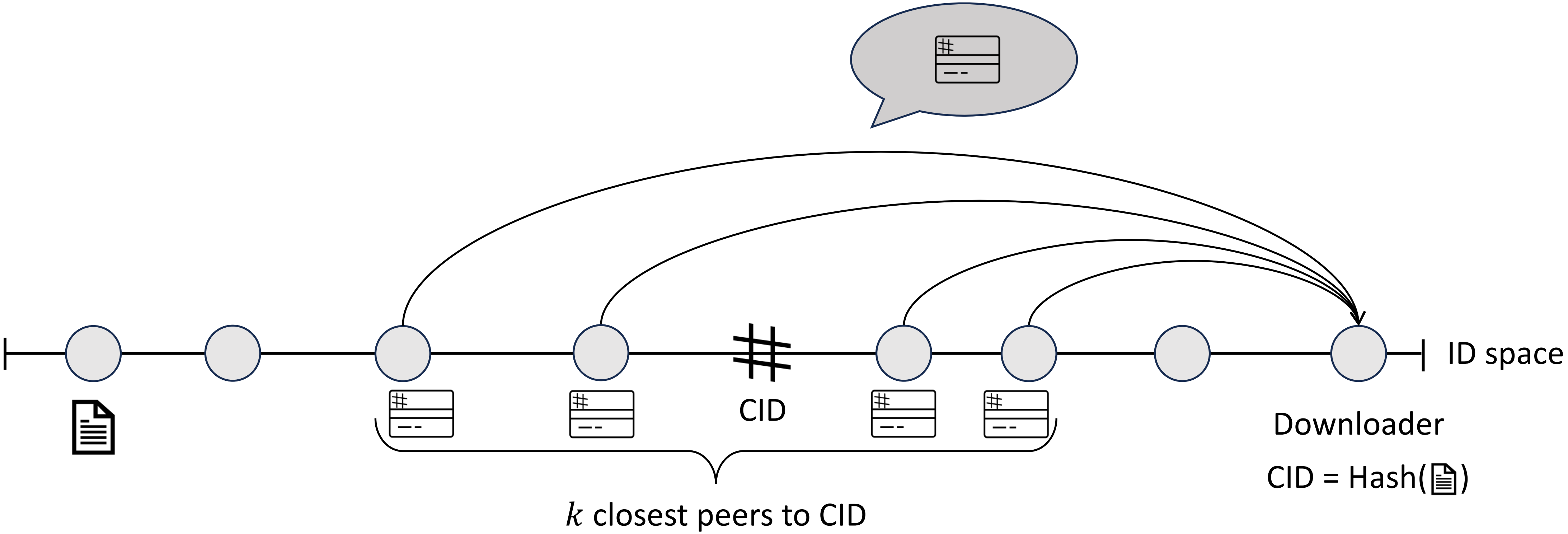
1. Find k (=20) closest peers to CID.
2. Request provider record from them.



Kademlia in IPFS

Download content:

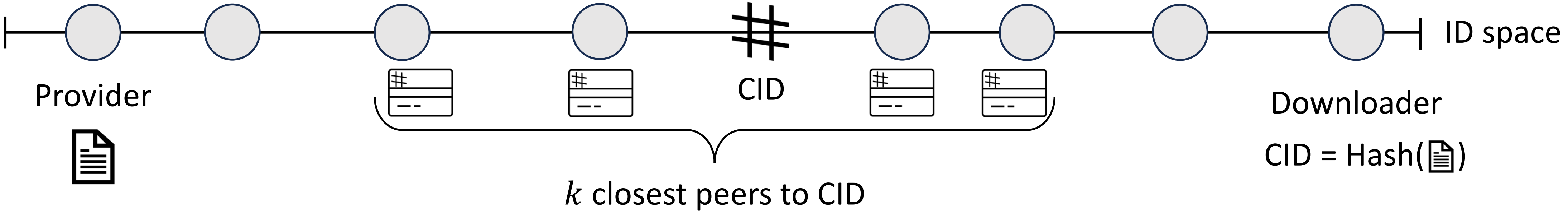
1. Find k (=20) closest peers to CID.
2. Request provider record from them.



Kademlia in IPFS

Download content:

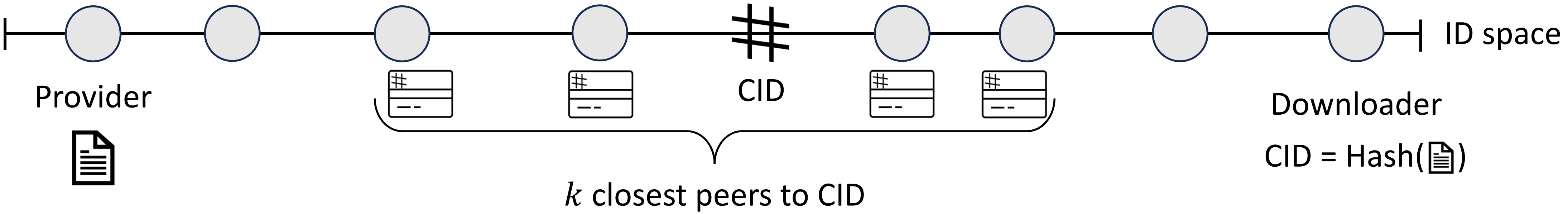
1. Find k (=20) closest peers to CID.
2. Request provider record from them.



Kademlia in IPFS

Download content:

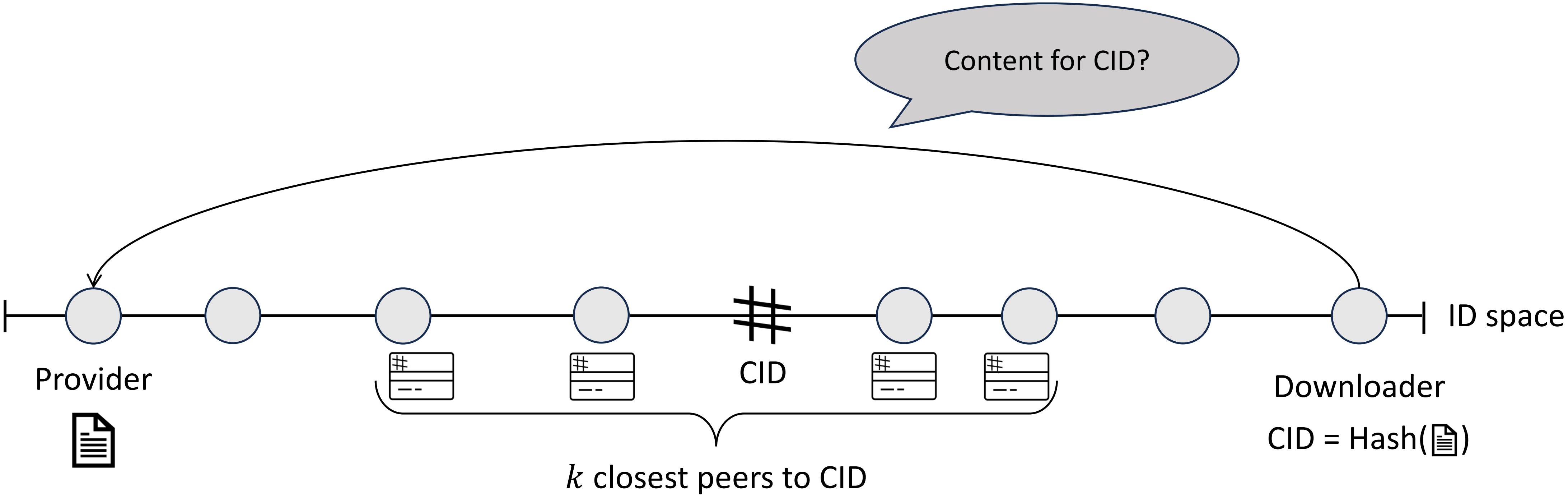
1. Find k (=20) closest peers to CID.
2. Request provider record from them.
3. Request content from provider.



Kademlia in IPFS

Download content:

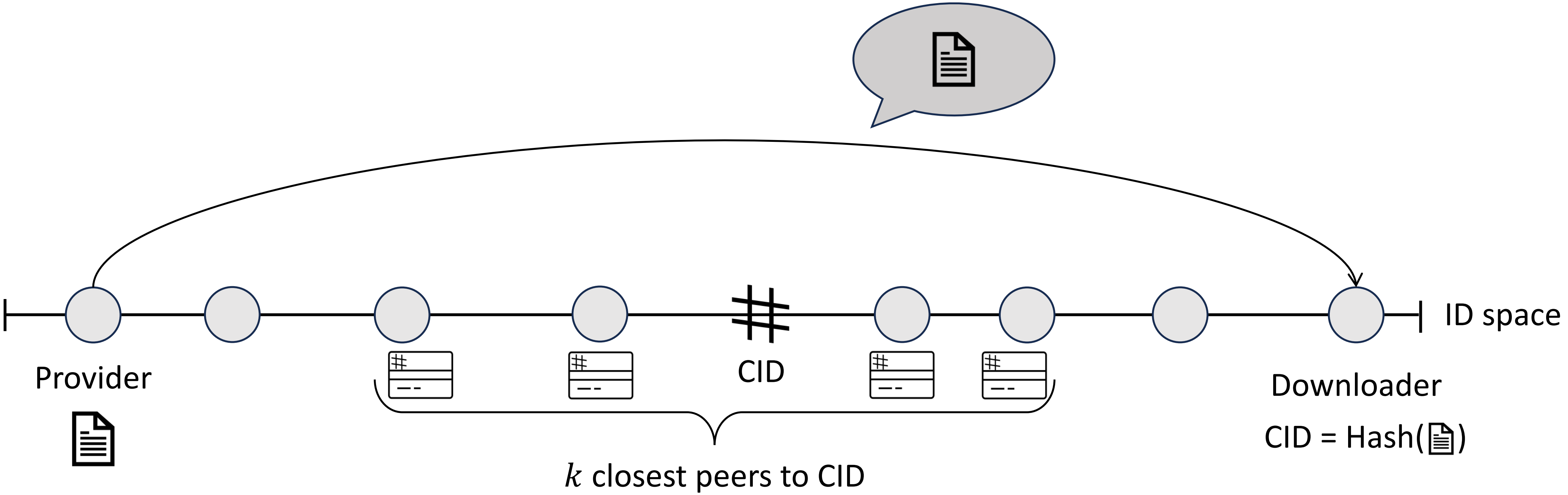
1. Find k (=20) closest peers to CID.
2. Request provider record from them.
3. Request content from provider.



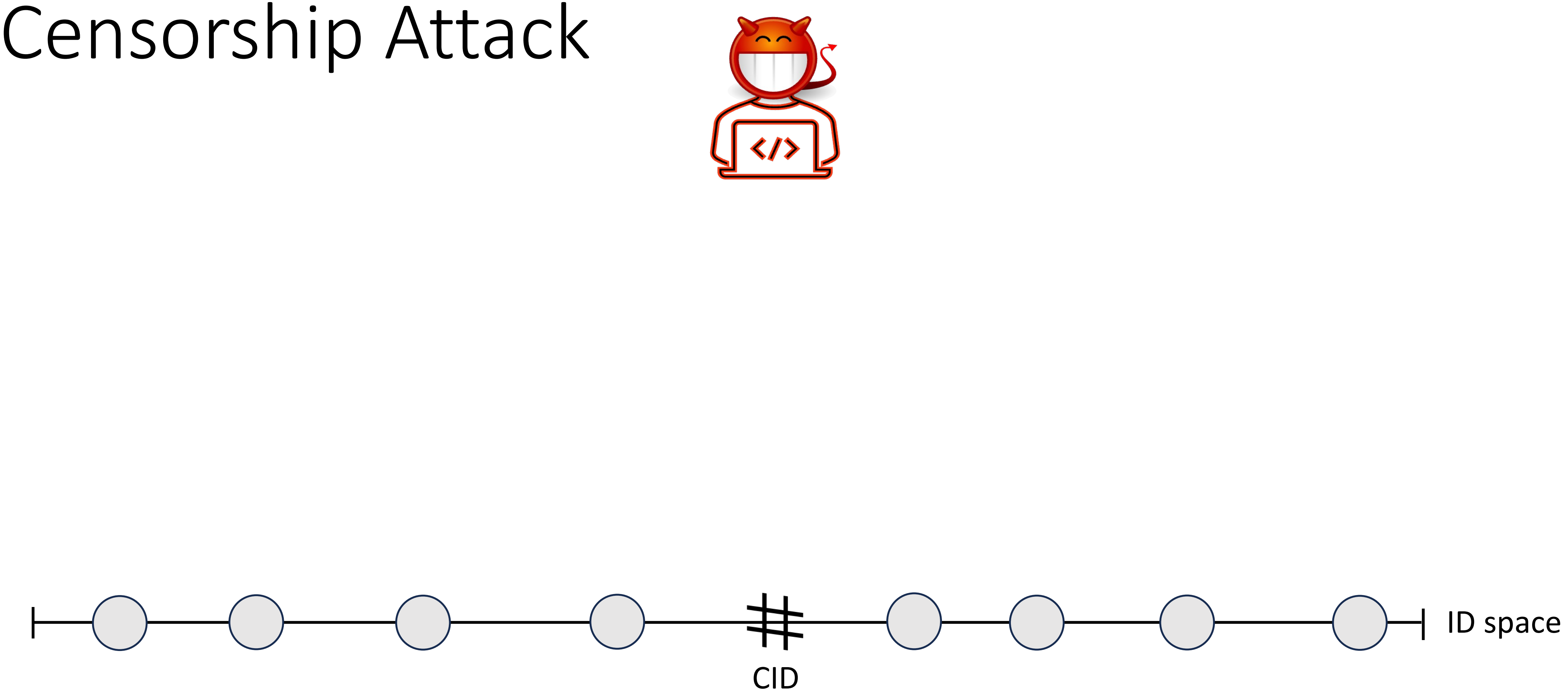
Kademlia in IPFS

Download content:

1. Find k (=20) closest peers to CID.
2. Request provider record from them.
3. Request content from provider.

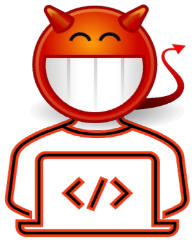


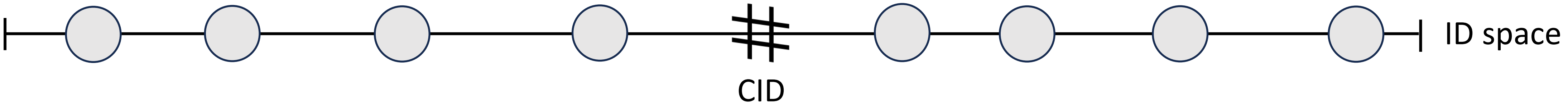
Censorship Attack



Censorship Attack

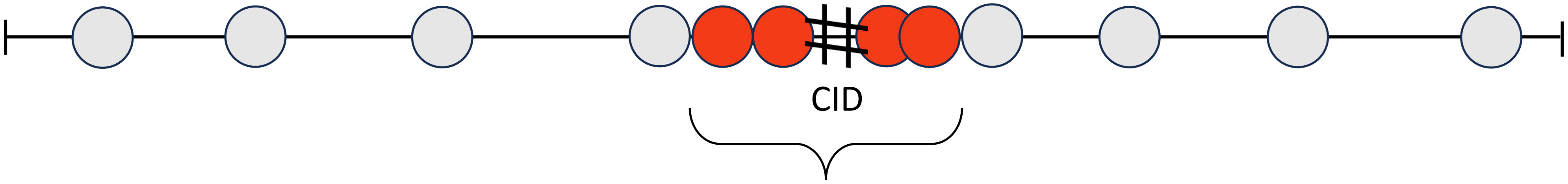
Attack content resolution





Censorship Attack

Attack content resolution



Sybil nodes

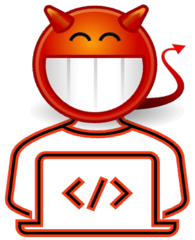
CID

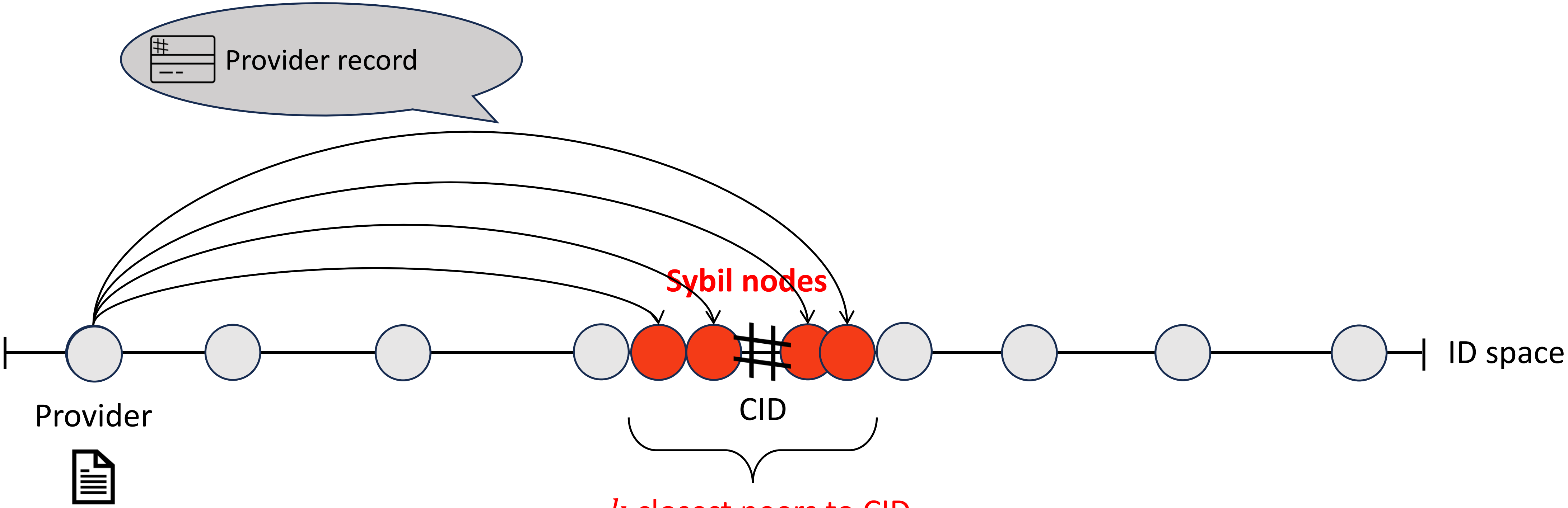
ID space

k closest peers to CID

Censorship Attack

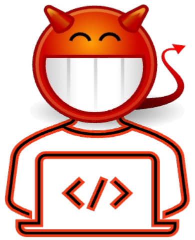
Attack content resolution

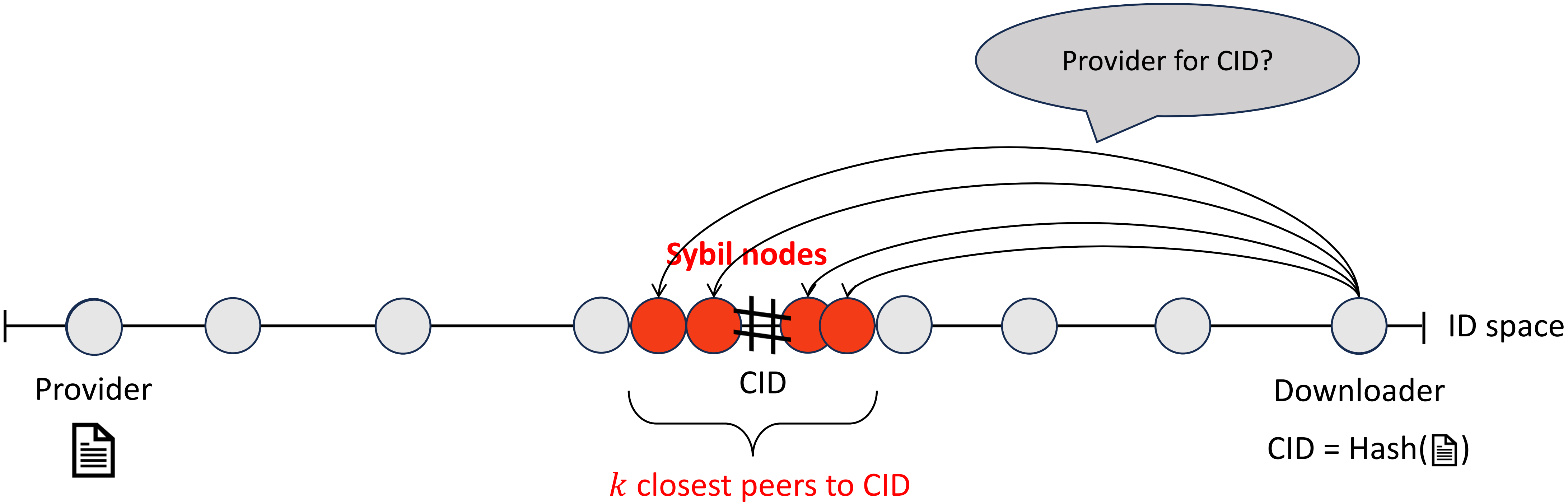




Censorship Attack

Attack content resolution





Provider for CID?

Sybil nodes

ID space

Provider

CID

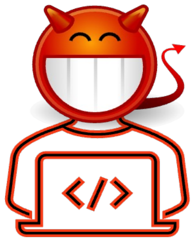
Downloader

CID = Hash(🗎)

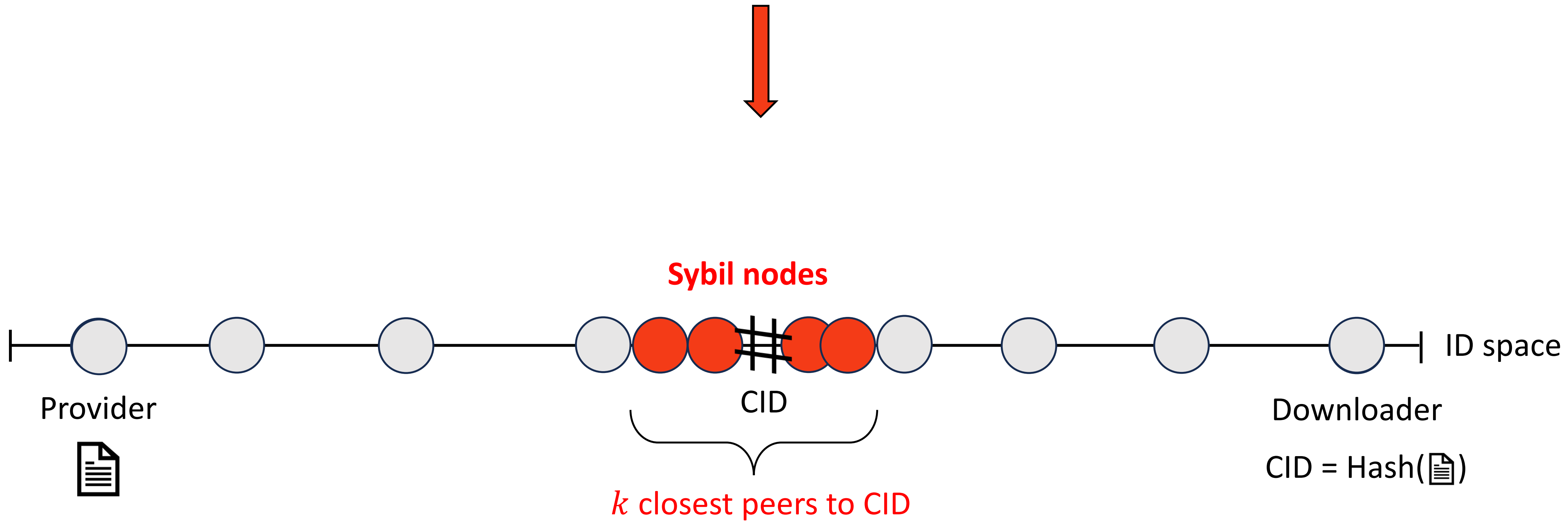
k closest peers to CID

Censorship Attack

Attack content resolution

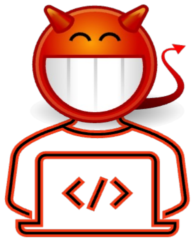


```
(pk,sk)← gen_key()
peerid ← Hash(pk)
```



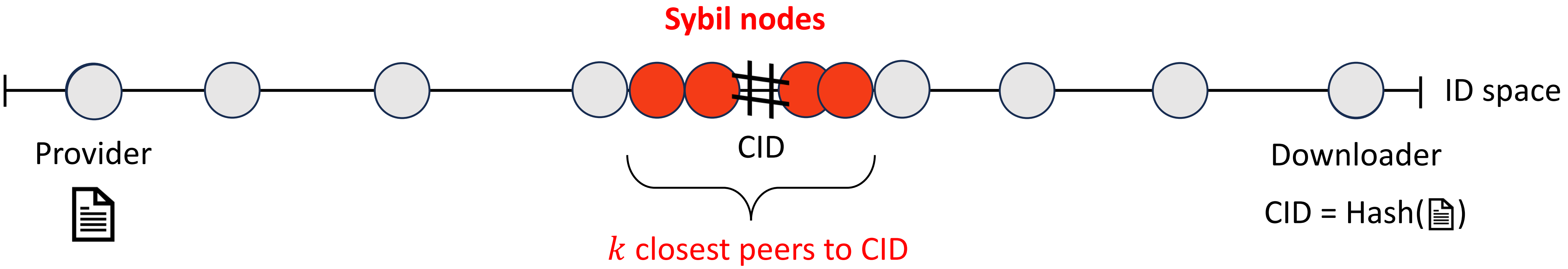
Censorship Attack

Attack content resolution



Generate peerids by brute-force search (takes <12 s)

```
(pk,sk)← gen_key()
peerid ← Hash(pk)
```

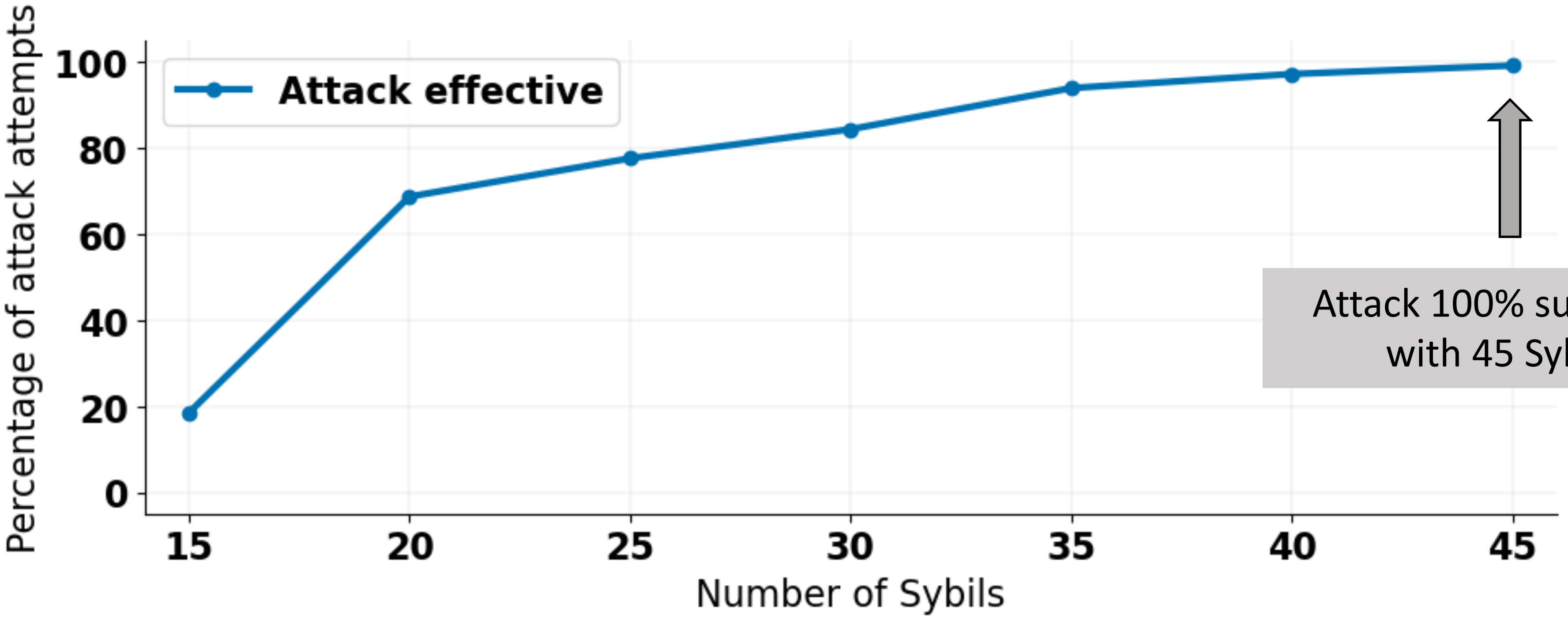


Attack Effectiveness

Experiments on the live IPFS network

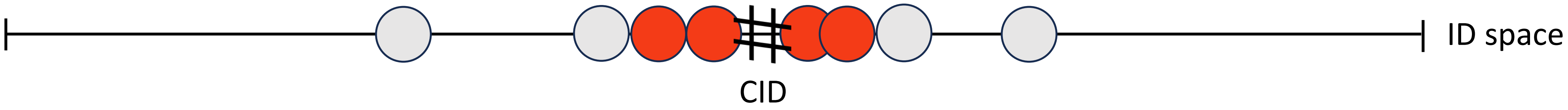
Attack Effectiveness

Experiments on the live IPFS network



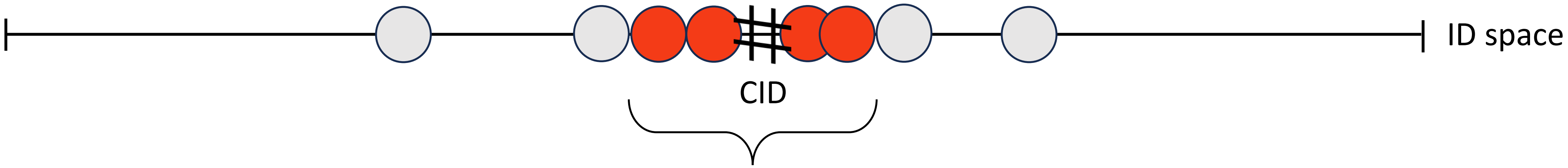
Attack 100% successful with 45 Sybils!

Attack Detection



T. Cholez, I. Chrisment, and O. Festor, “Efficient DHT attack mitigation through peers’ ID distribution”

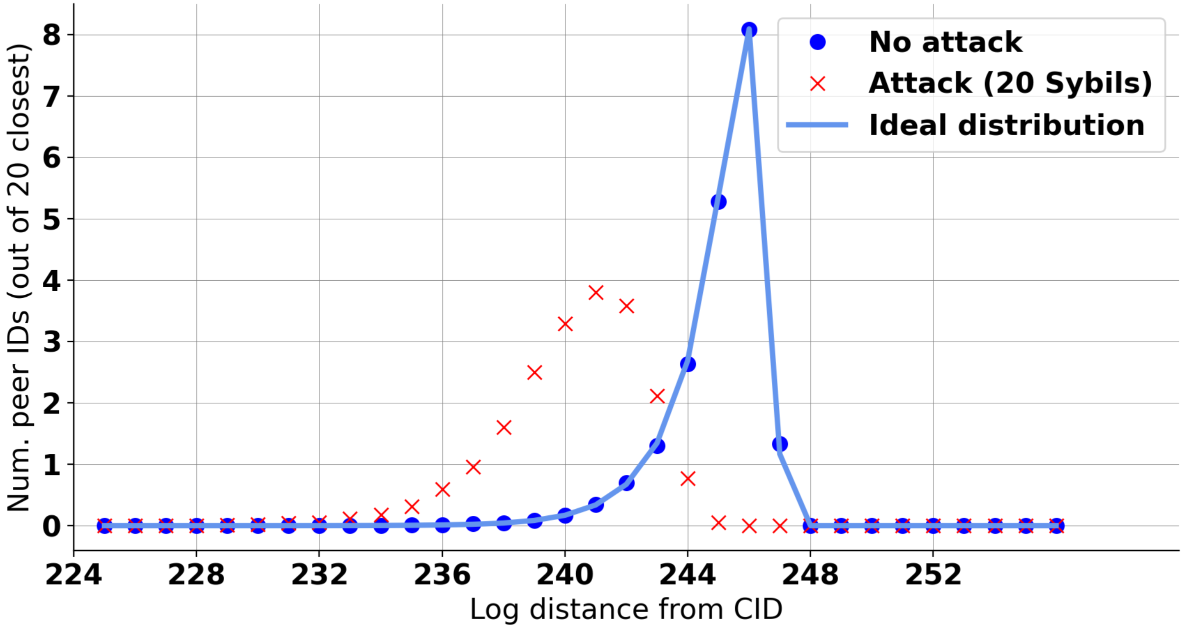
Attack Detection

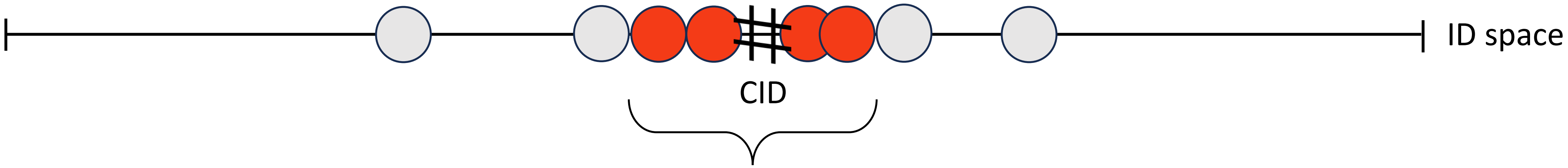


High density of peer IDs!

T. Cholez, I. Chrisment, and O. Festor, "Efficient DHT attack mitigation through peers' ID distribution"

Attack Detection

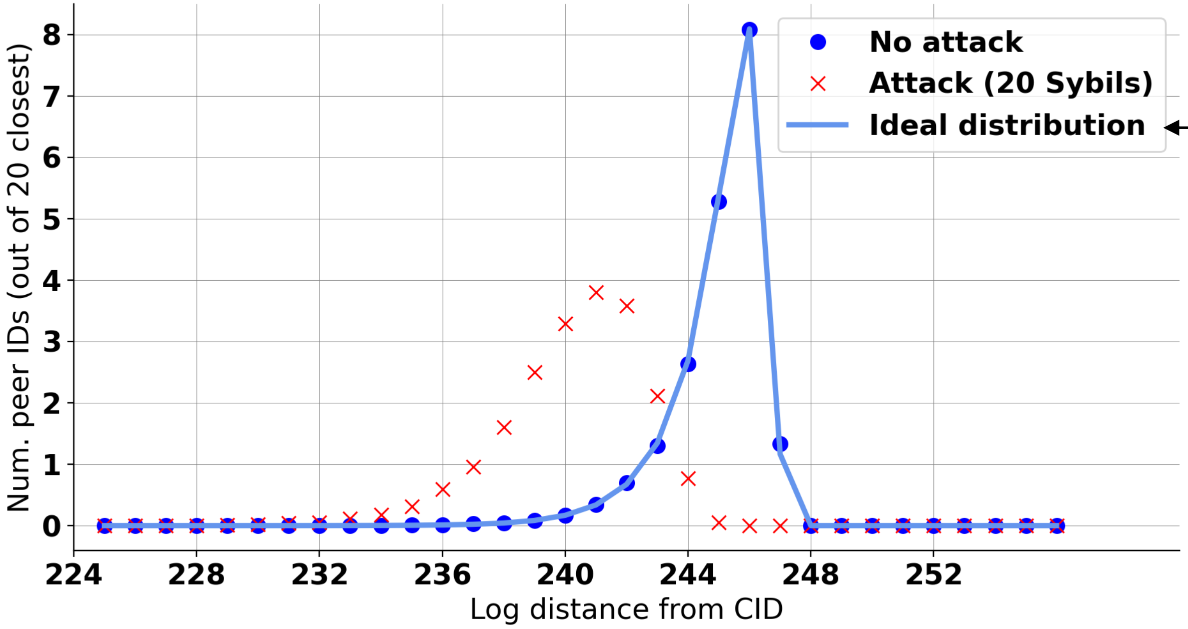




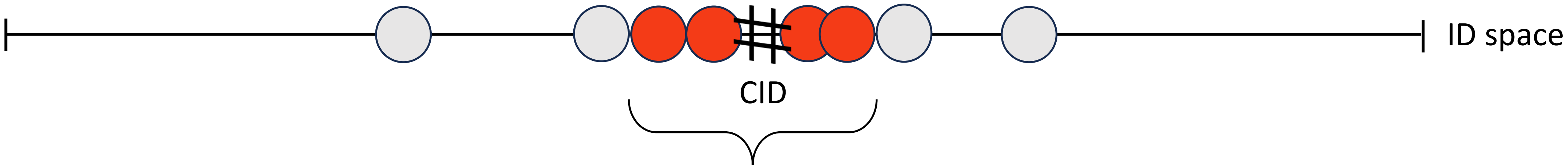
High density of peer IDs!

T. Cholez, I. Chrisment, and O. Festor, "Efficient DHT attack mitigation through peers' ID distribution"

Attack Detection



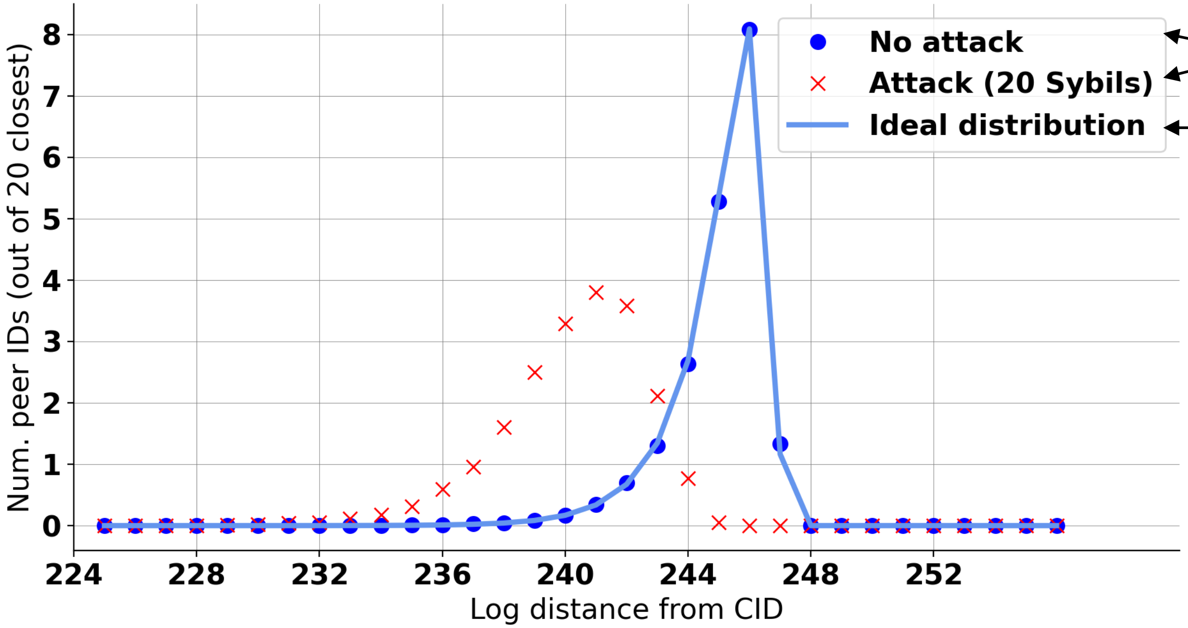
$p(x)$: calculated using estimate of network size



High density of peer IDs!

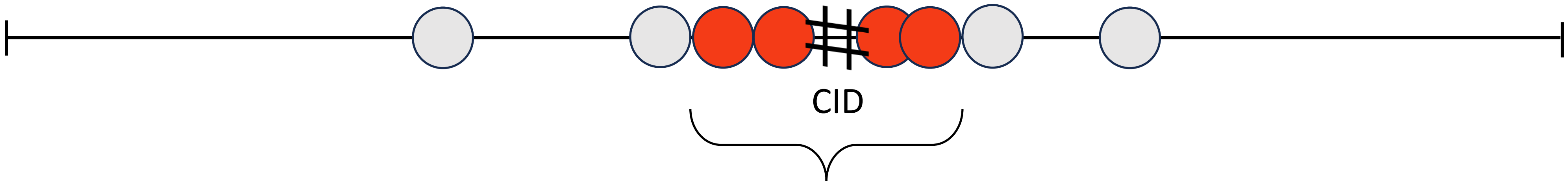
T. Cholez, I. Chrisment, and O. Festor, "Efficient DHT attack mitigation through peers' ID distribution"

Attack Detection



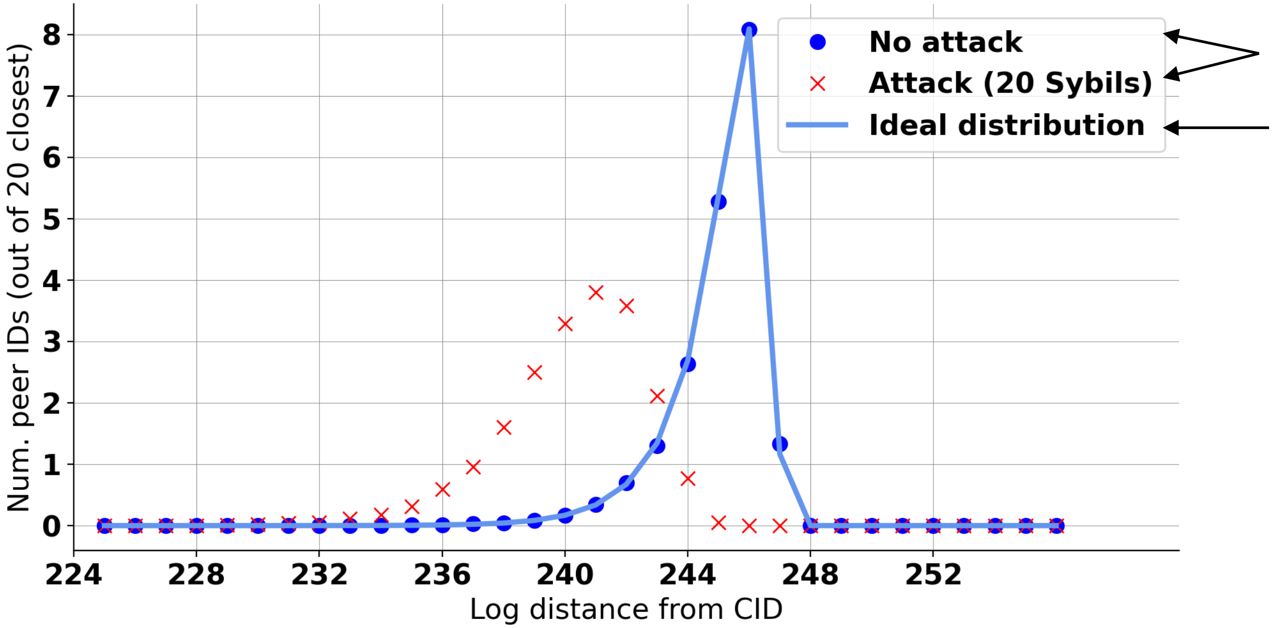
$q(x)$: observed distribution

$p(x)$: calculated using estimate of network size



High density of peer IDs!

Attack Detection

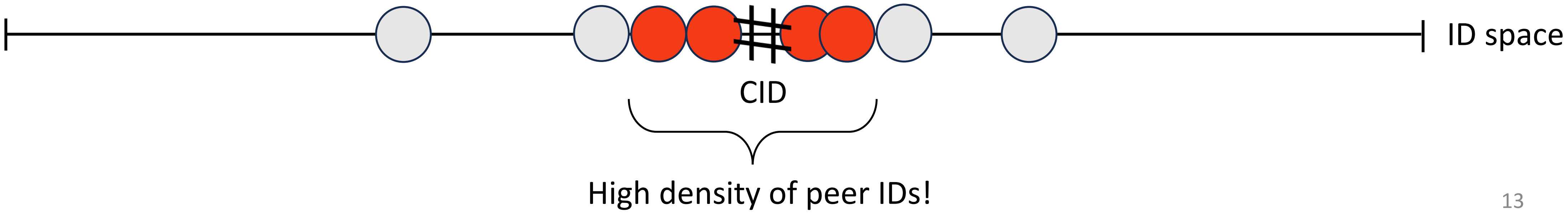


$q(x)$: observed distribution

$p(x)$: calculated using estimate of network size

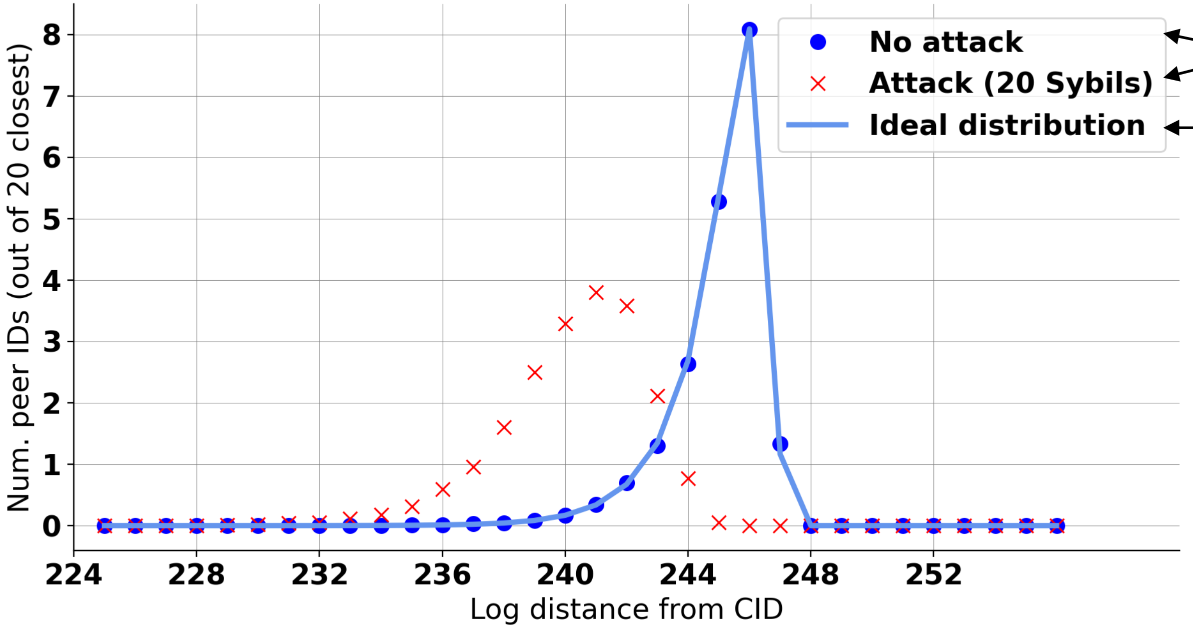
K-L Divergence

$$D(q||p) = \sum_{x} q(x) \ln\left(\frac{q(x)}{p(x)}\right)$$



T. Cholez, I. Chrisment, and O. Festor, "Efficient DHT attack mitigation through peers' ID distribution"

Attack Detection



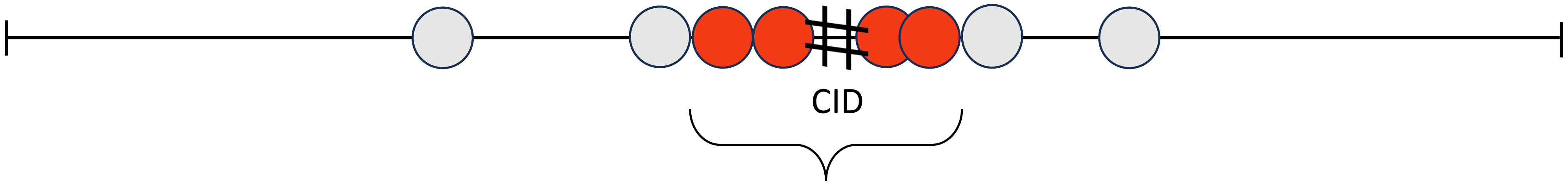
$q(x)$: observed distribution

$p(x)$: calculated using estimate of network size

K-L Divergence

$$D(q||p) = \sum_{x} q(x) \ln\left(\frac{q(x)}{p(x)}\right)$$

```
if KL > thr:
    detect "Attack"
```

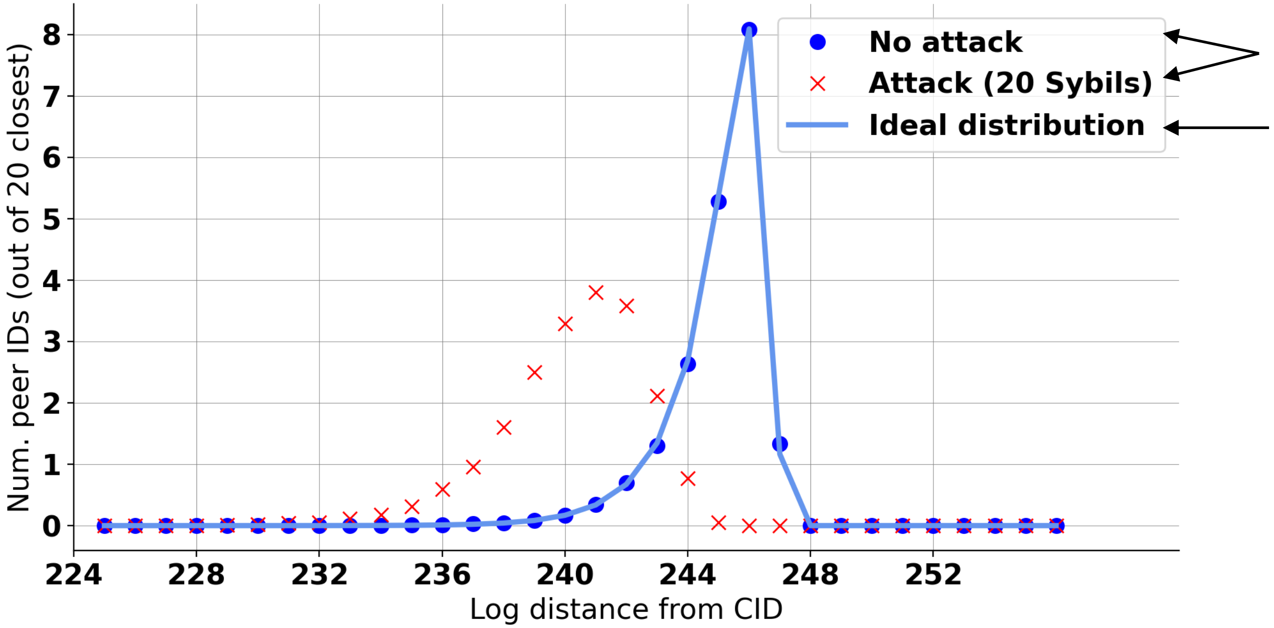


ID space

High density of peer IDs!

Attack Detection

Can be done by Provider or Downloader



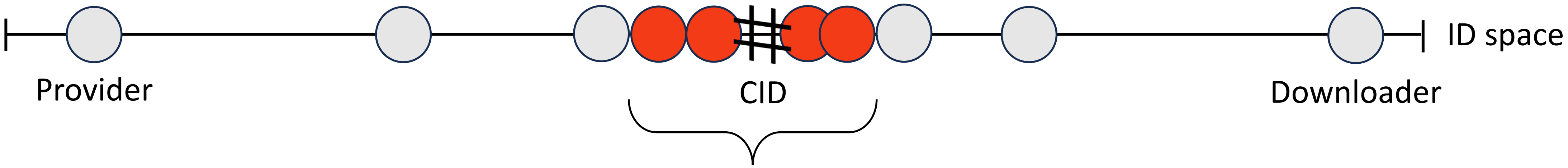
$q(x)$: observed distribution

$p(x)$: calculated using estimate of network size

K-L Divergence

$$D(q||p) = \sum_x q(x) \ln\left(\frac{q(x)}{p(x)}\right)$$

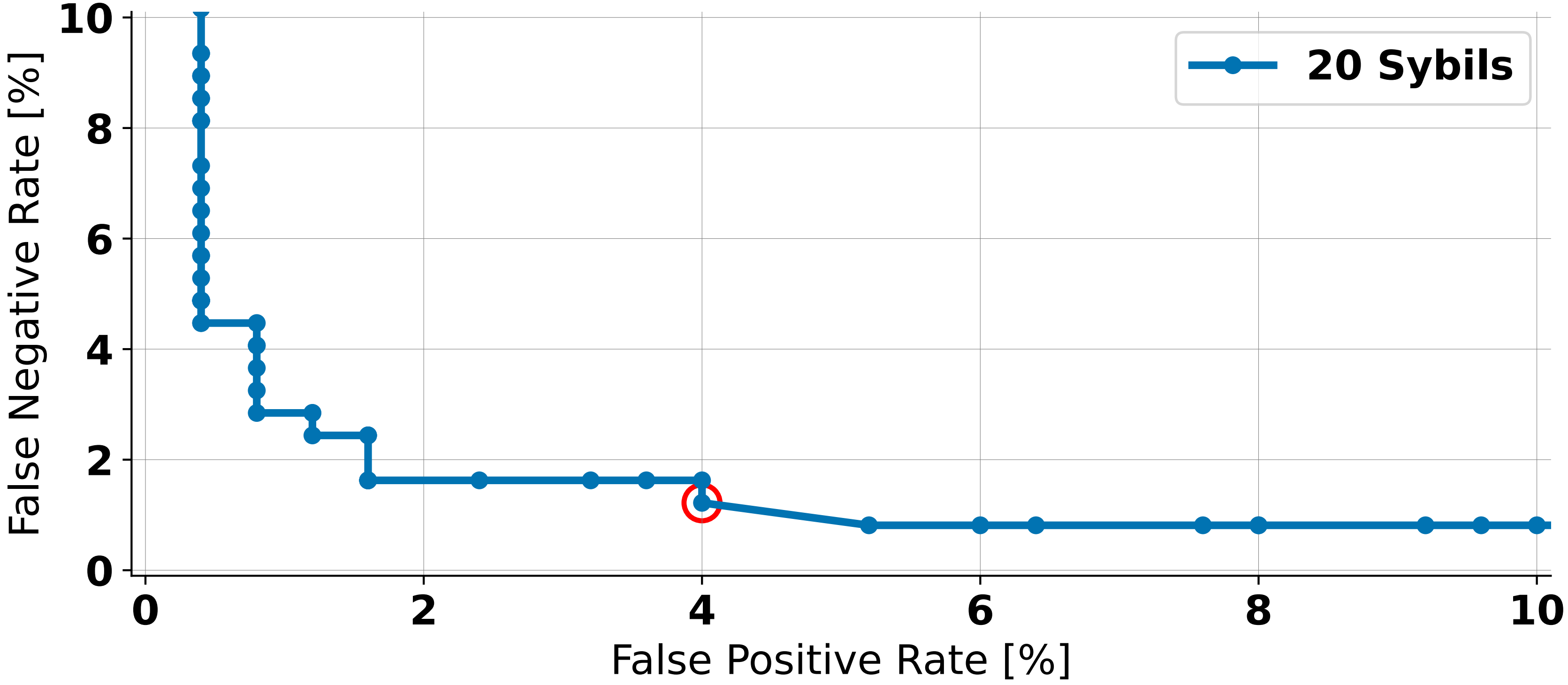
```
if KL > thr:
    detect “Attack”
```



High density of peer IDs!

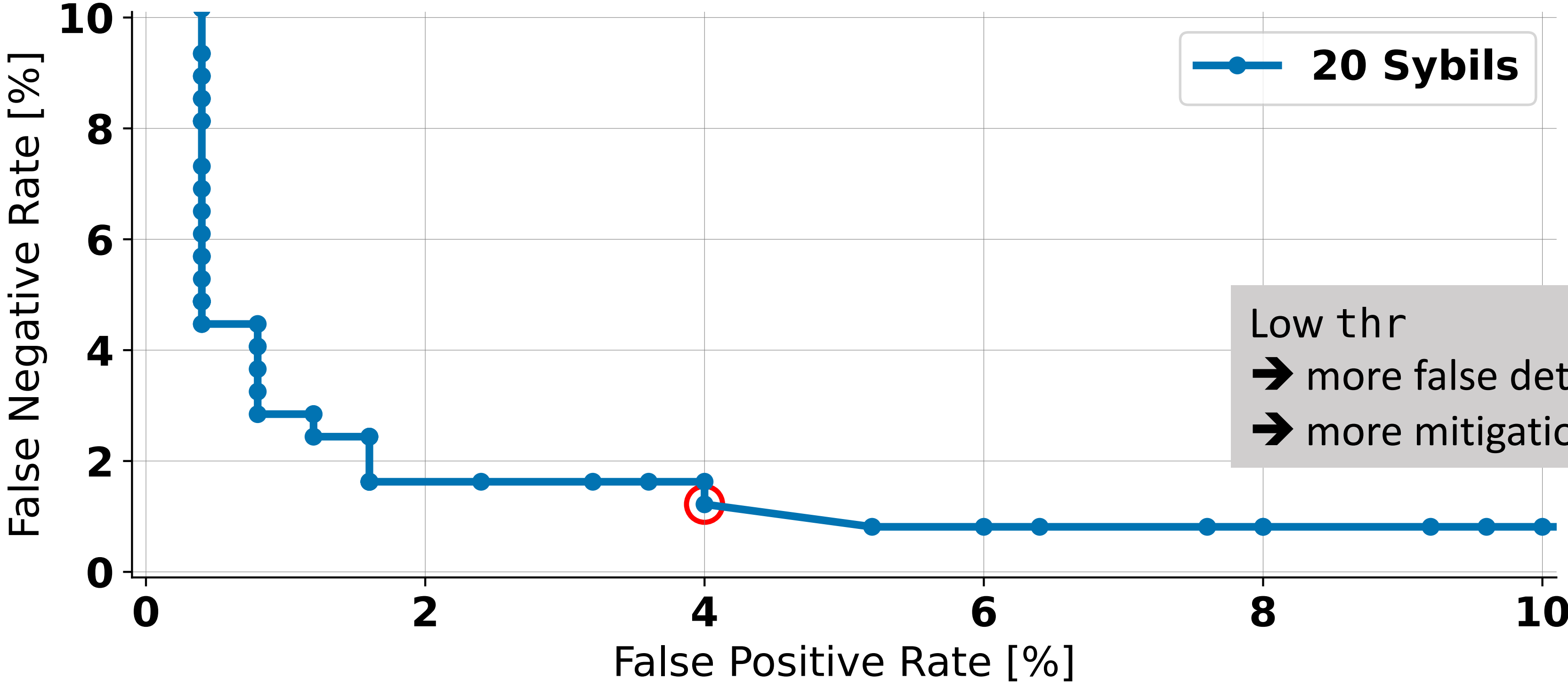
Detection Accuracy

```
if KL > thr:
    detect “Attack”
```



Detection Accuracy

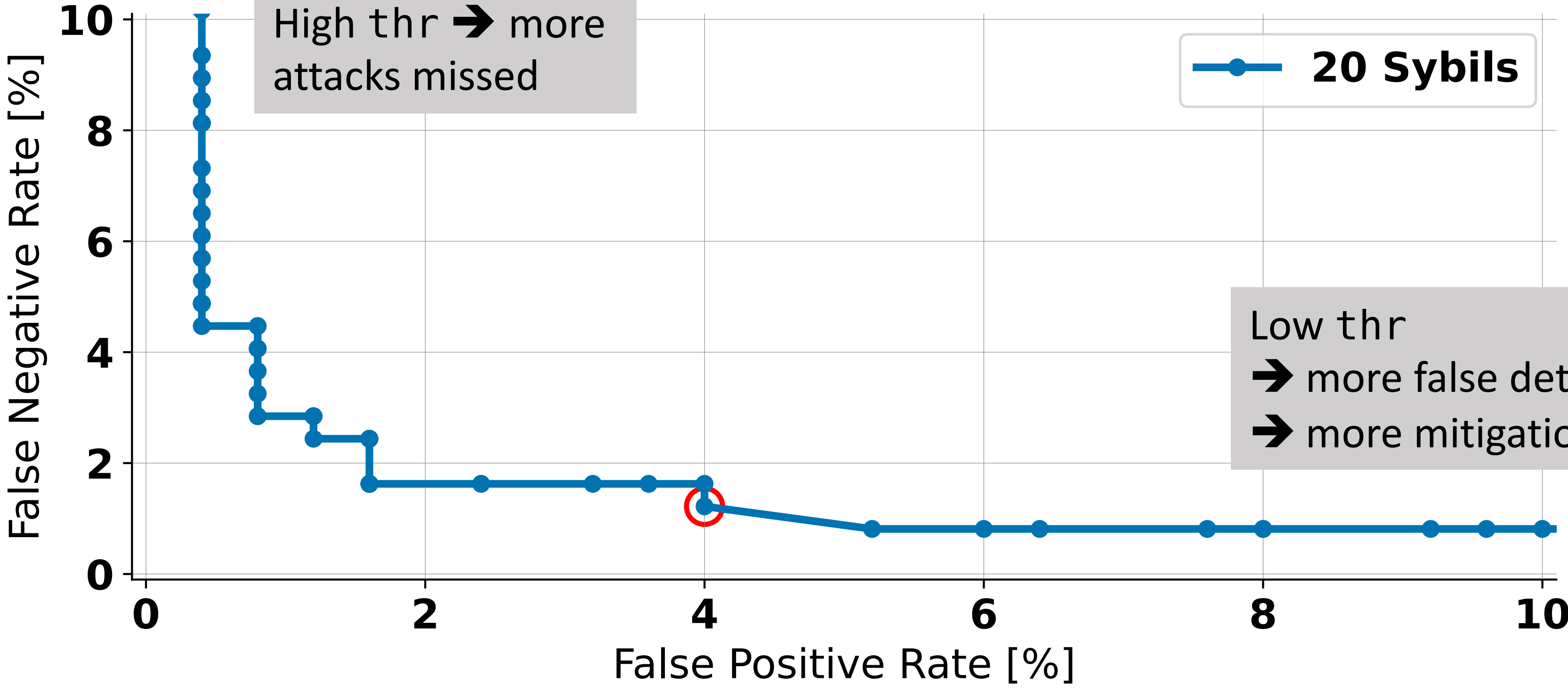
```
if KL > thr:
    detect “Attack”
```



Low thr
➔ more false detections
➔ more mitigation overhead

Detection Accuracy

```
if KL > thr:
    detect “Attack”
```

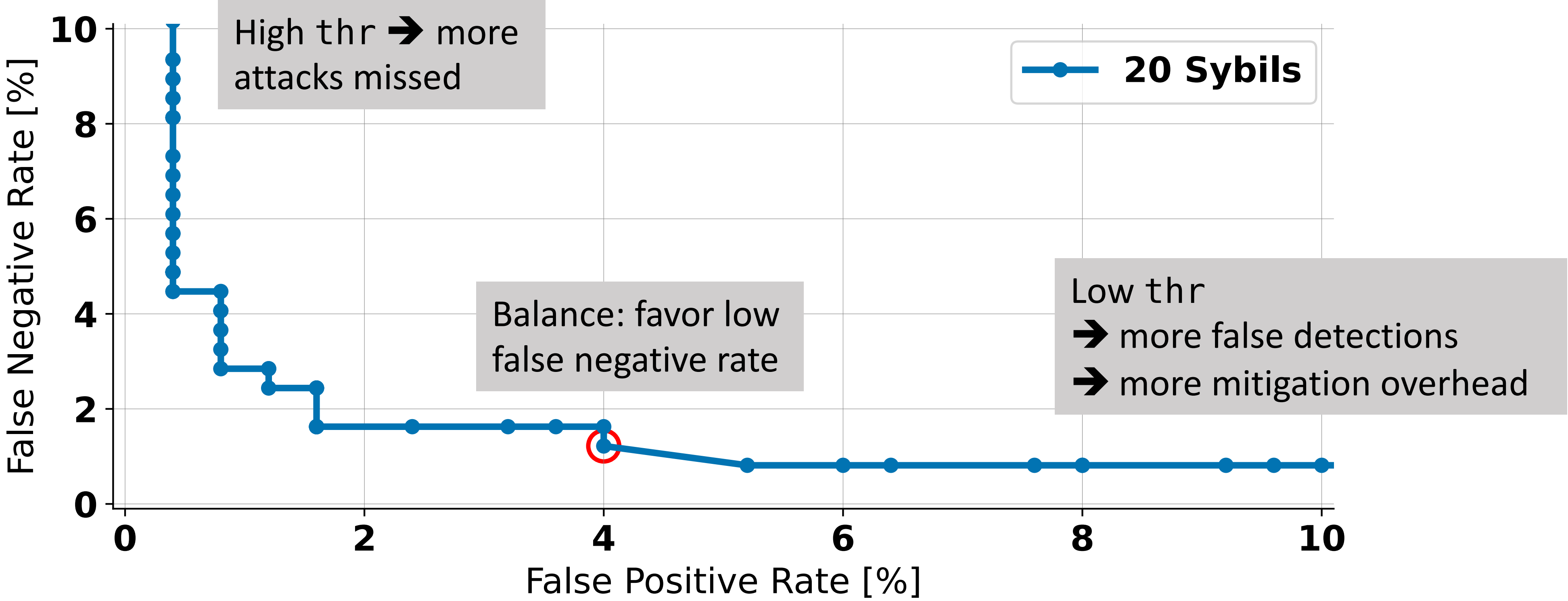


High thr ➔ more attacks missed

Low thr
➔ more false detections
➔ more mitigation overhead

Detection Accuracy

```
if KL > thr:
    detect “Attack”
```

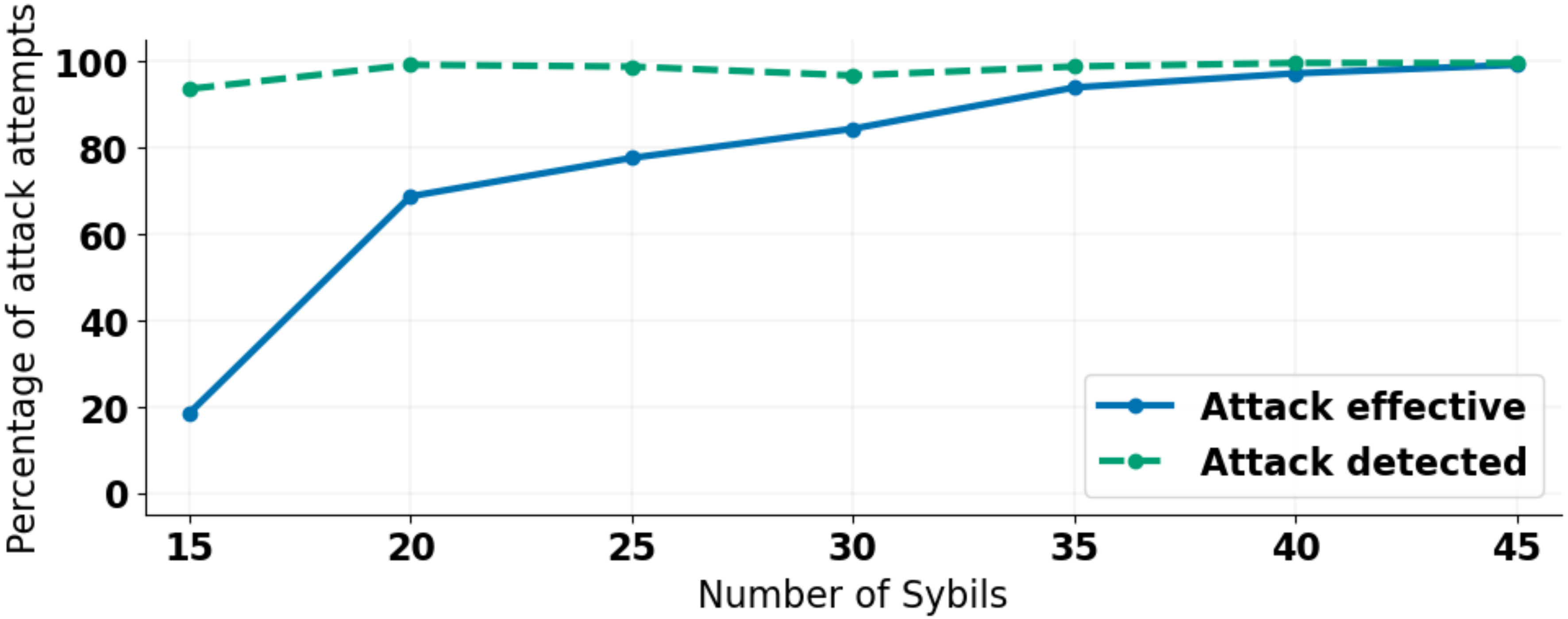


High thr ➔ more attacks missed

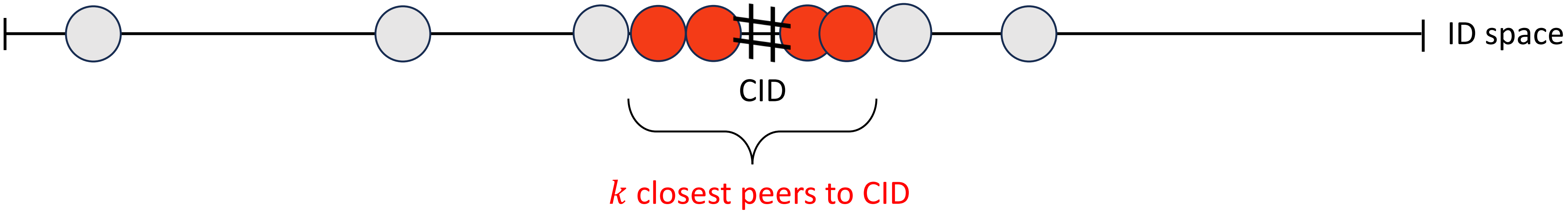
Balance: favor low false negative rate

Low thr
➔ more false detections
➔ more mitigation overhead

Detection Accuracy

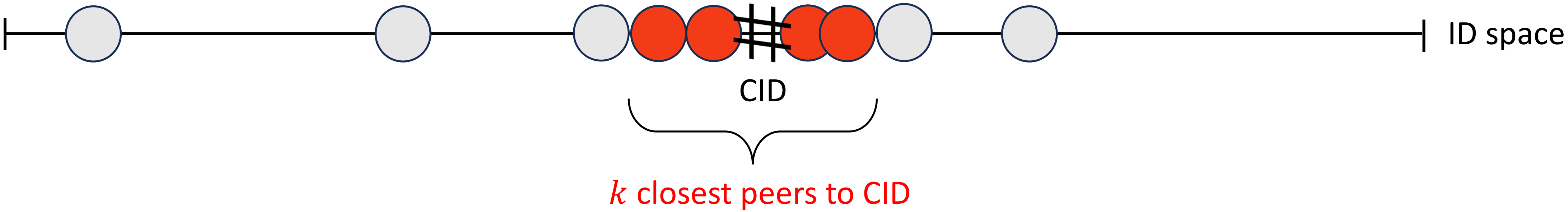


Attack Mitigation



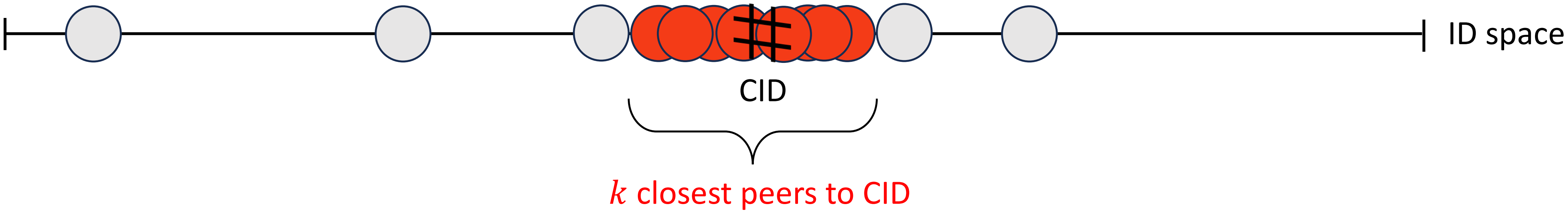
Attack Mitigation

Send provider record to 50 instead of 20 peers?



Attack Mitigation

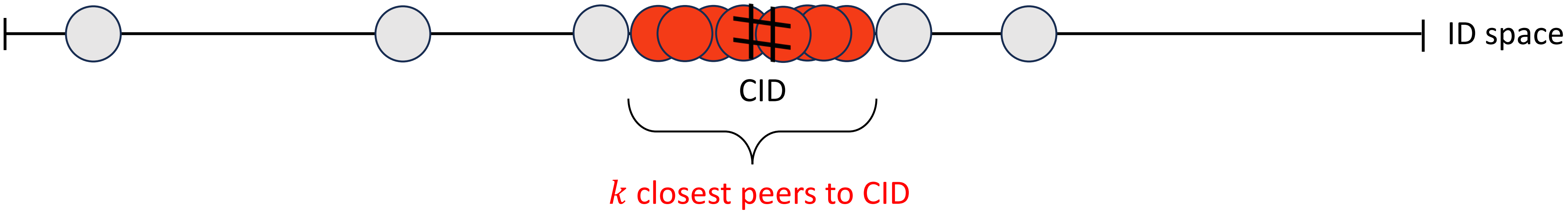
Send provider record to 50 instead of 20 peers?



Attack Mitigation

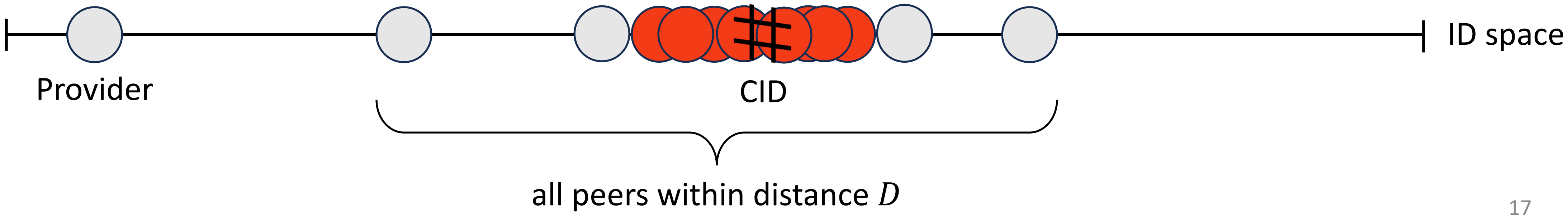
Send provider record to 50 instead of 20 peers?

Attacker launches more Sybils



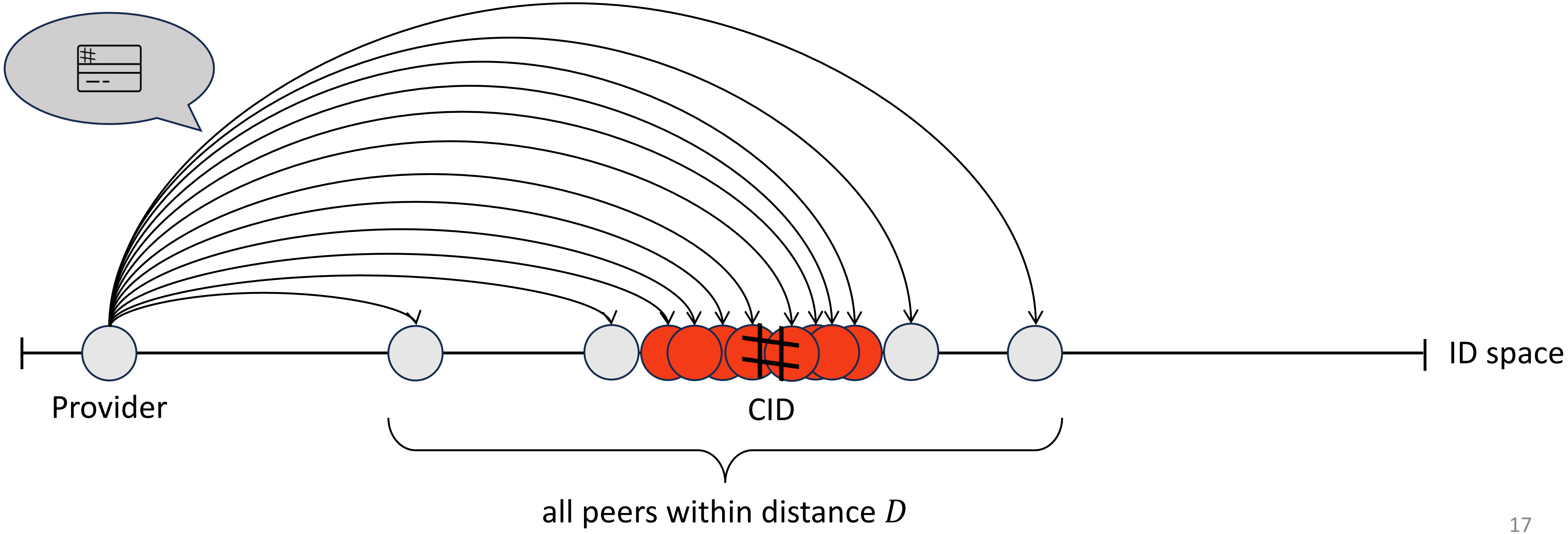
Attack Mitigation

Solution: Send to all peers within distance D expected to contain k honest peers



Attack Mitigation

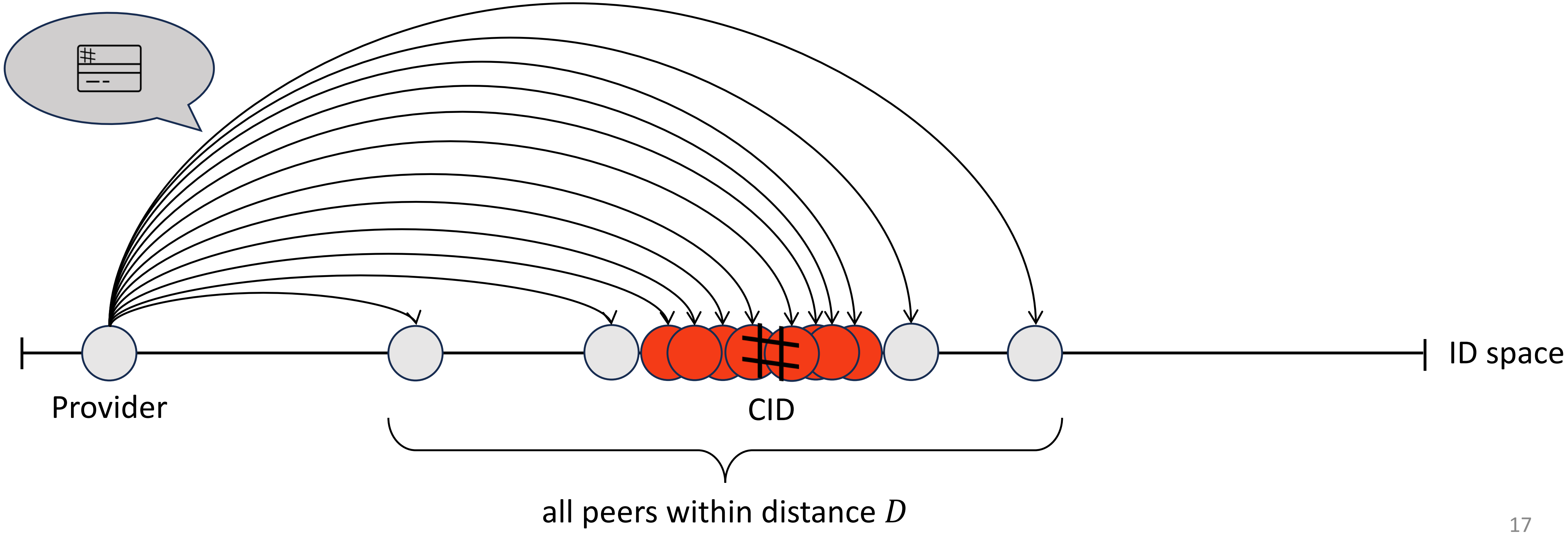
Solution: Send to all peers within distance D expected to contain k honest peers



Attack Mitigation

Solution: Send to all peers within distance D expected to contain k honest peers

How to calculate D? Use network size estimate!



Attack Mitigation

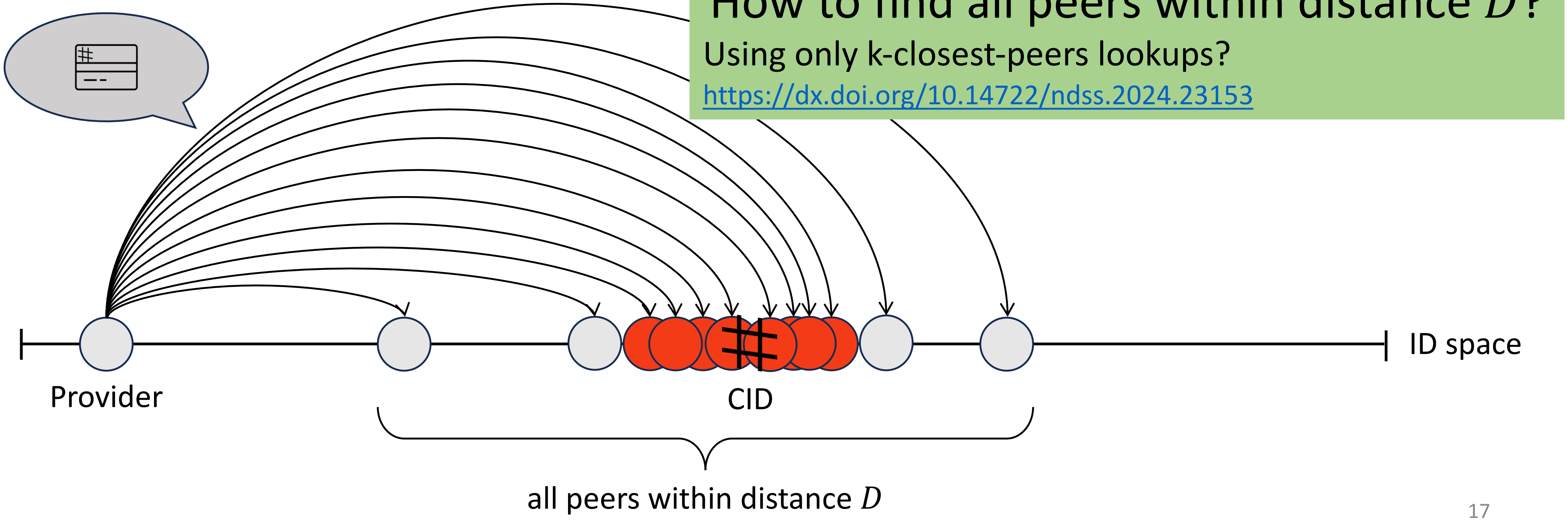
Solution: Send to all peers within distance D expected to contain k honest peers

How to calculate D? Use network size estimate!

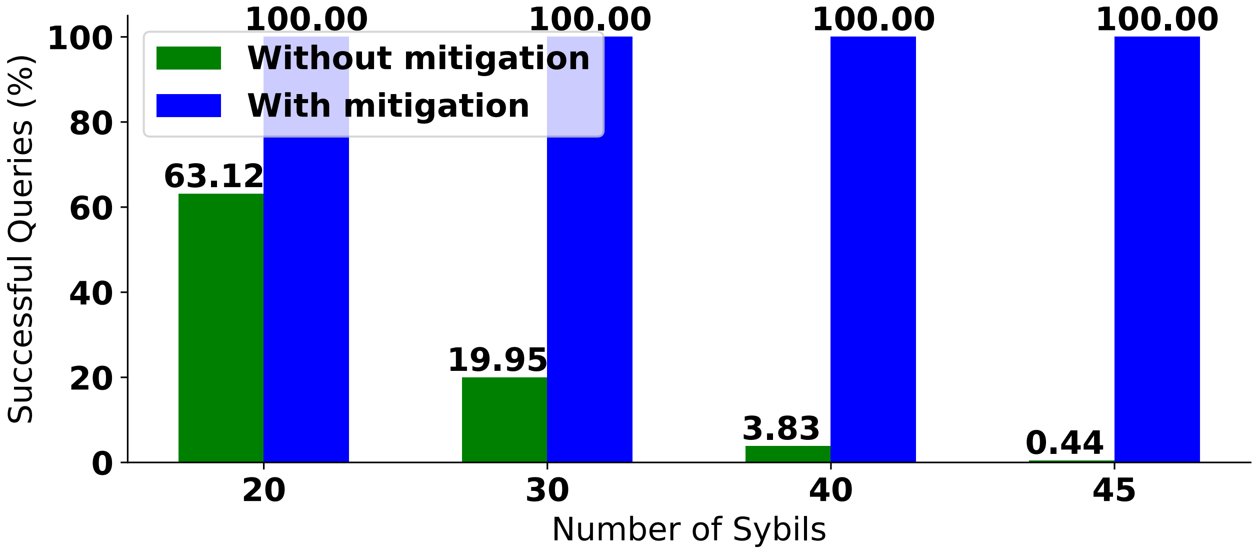
How to find all peers within distance D?

Using only k-closest-peers lookups?

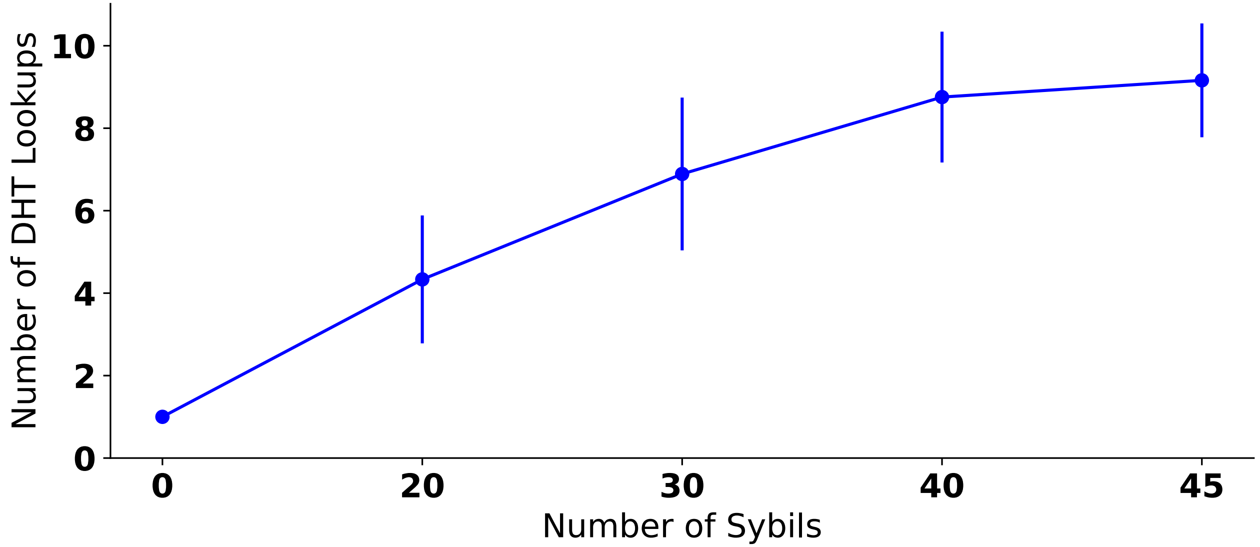
https://dx.doi.org/10.14722/ndss.2024.23153



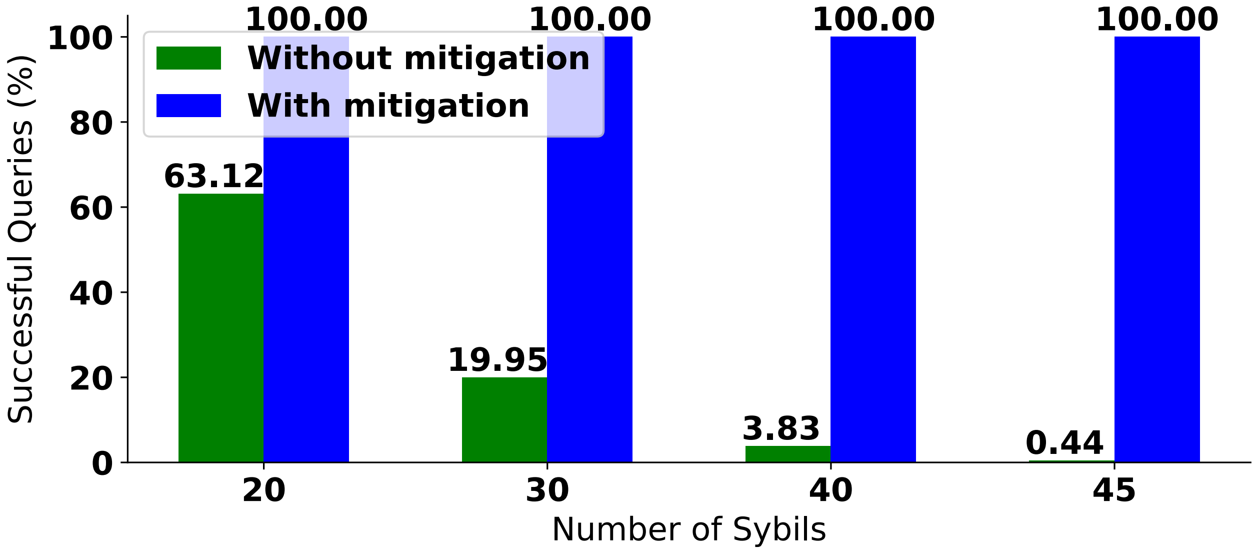
Mitigation Effectiveness



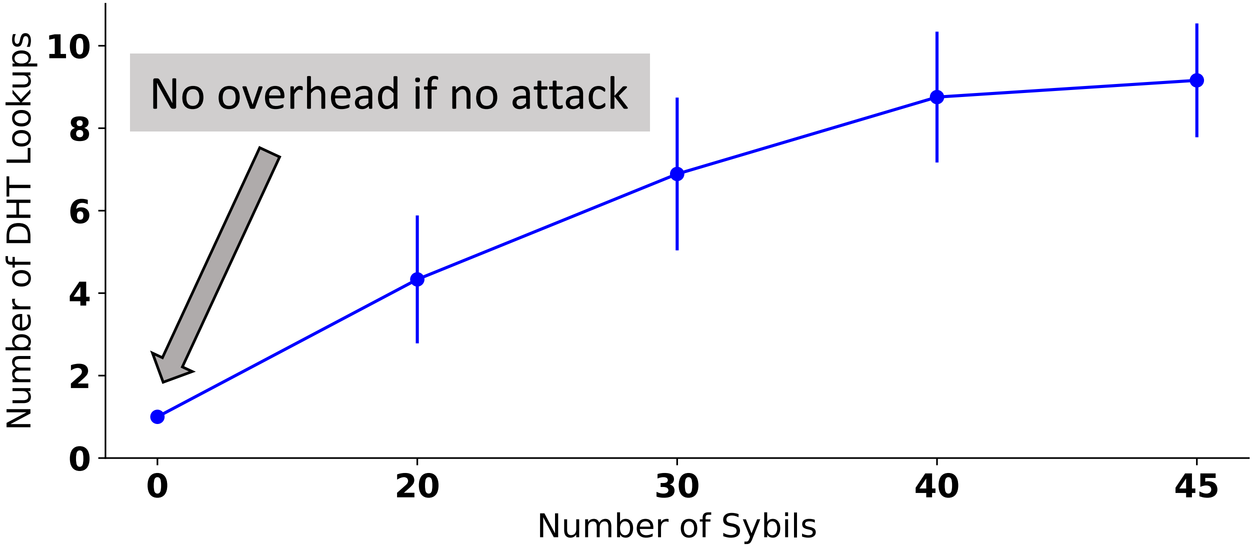
Mitigation Overhead



Mitigation Effectiveness



Mitigation Overhead



No overhead if no attack

Conclusion

Conclusion

- Effective mitigation against low-cost censorship attack

Conclusion

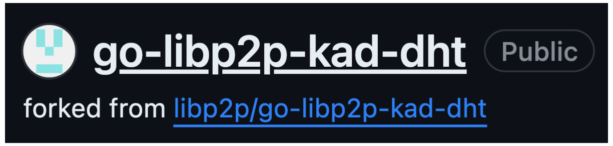
- Effective mitigation against low-cost censorship attack
- Attack detection to minimize mitigation overhead

Conclusion

- Effective mitigation against low-cost censorship attack
- Attack detection to minimize mitigation overhead
- Implementation to be deployed in IPFS

Conclusion

- Effective mitigation against low-cost censorship attack
- Attack detection to minimize mitigation overhead
- Implementation to be deployed in IPFS



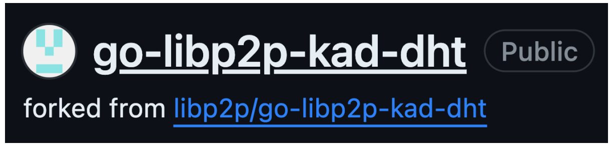
go-libp2p-kad-dht Public

forked from libp2p/go-libp2p-kad-dht

Conclusion

- Effective mitigation against low-cost censorship attack
- Attack detection to minimize mitigation overhead
- Implementation to be deployed in IPFS

CVE-2023-262481



go-libp2p-kad-dht Public

forked from libp2p/go-libp2p-kad-dht

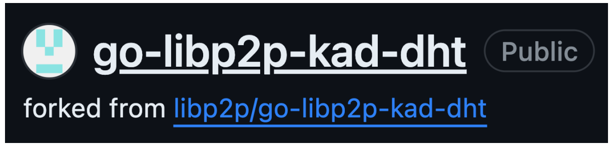
Conclusion

- Effective mitigation against low-cost censorship attack
- Attack detection to minimize mitigation overhead
- Implementation to be deployed in IPFS

Read more:

https://dx.doi.org/10.14722/ndss.2024.23153

CVE-2023-262481



go-libp2p-kad-dht Public

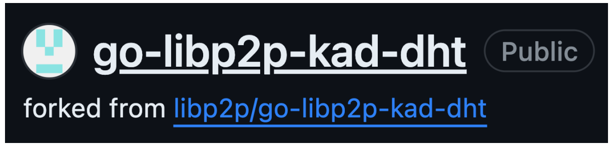
forked from libp2p/go-libp2p-kad-dht

Conclusion

- Effective mitigation against low-cost censorship attack
- Attack detection to minimize mitigation overhead
- Implementation to be deployed in IPFS

Read more:

https://dx.doi.org/10.14722/ndss.2024.23153



go-libp2p-kad-dht Public

forked from libp2p/go-libp2p-kad-dht

CVE-2023-262481

Artifact Evaluated

NDSS SYMPOSIUM

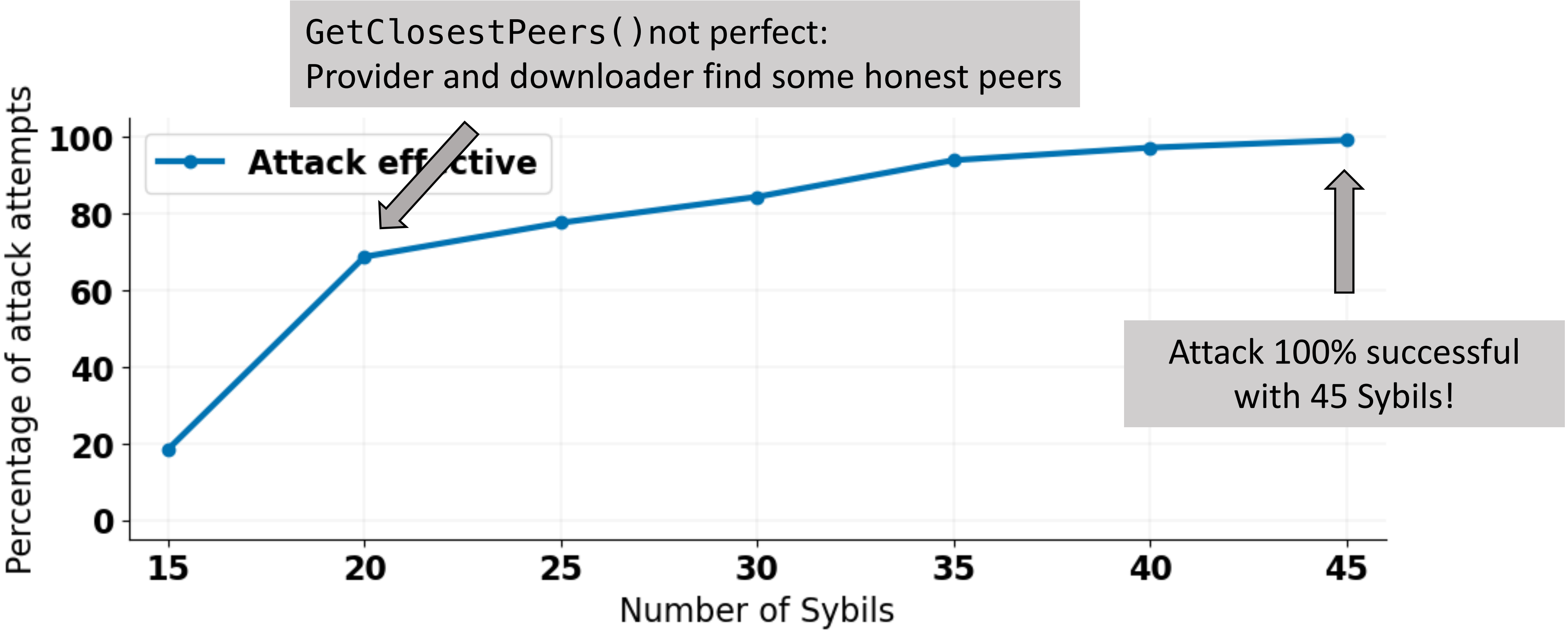
Available

Functional

Reproduced

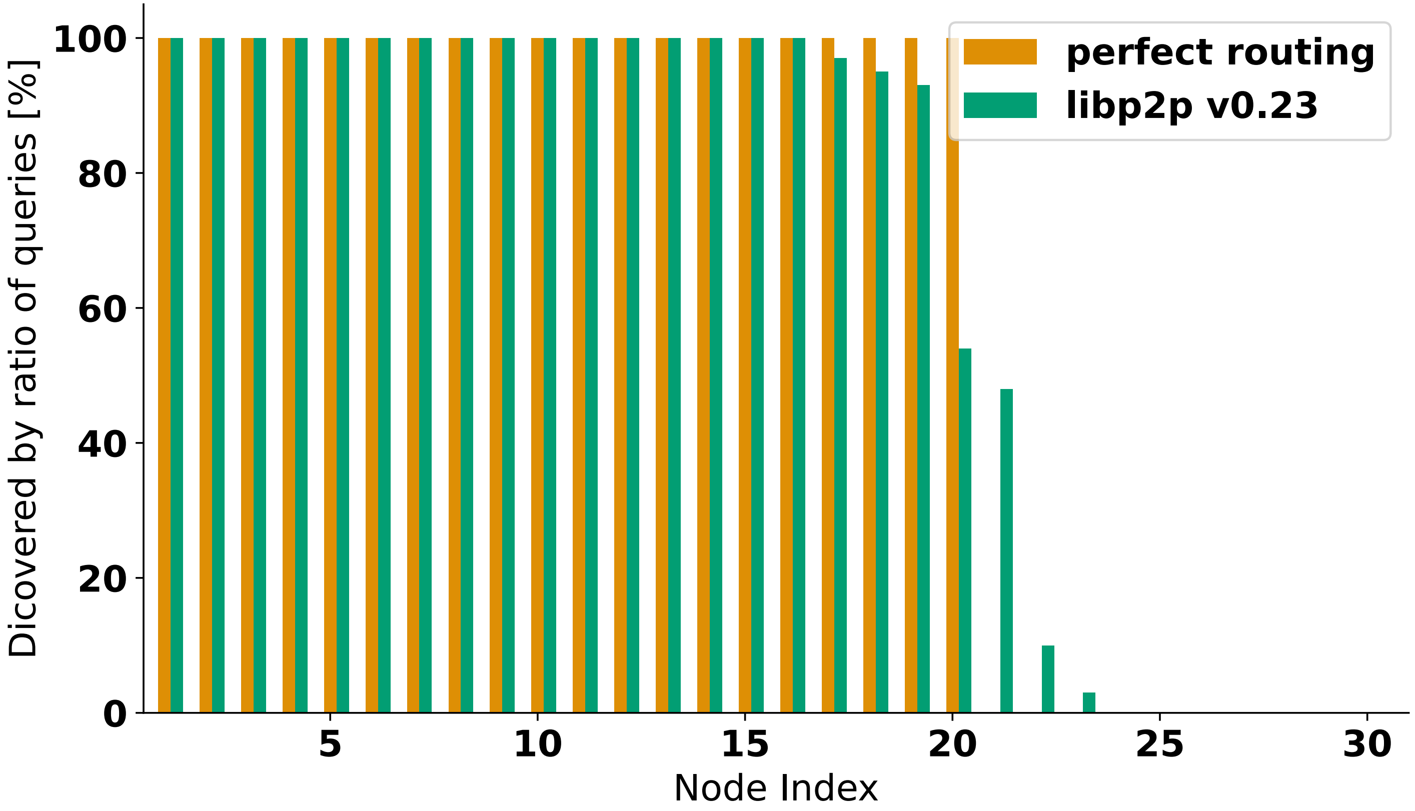
Attack Effectiveness

Experiments on the live IPFS network

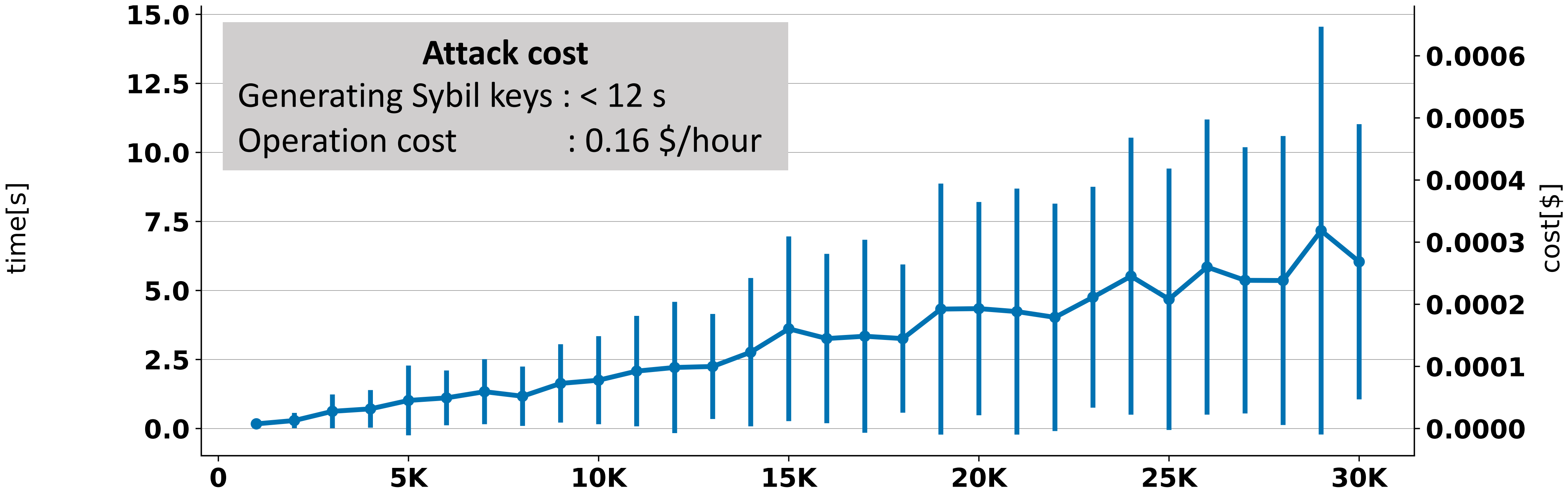


`GetClosestPeers()`not perfect:
Provider and downloader find some honest peers

Attack 100% successful
with 45 Sybils!


Dicovered by ratio of queries [%]
100
80
60
40
20
0
5
10
15
20
25
30
Node Index
perfect routing
libp2p v0.23

Attack Cost: Generating Sybil Keys

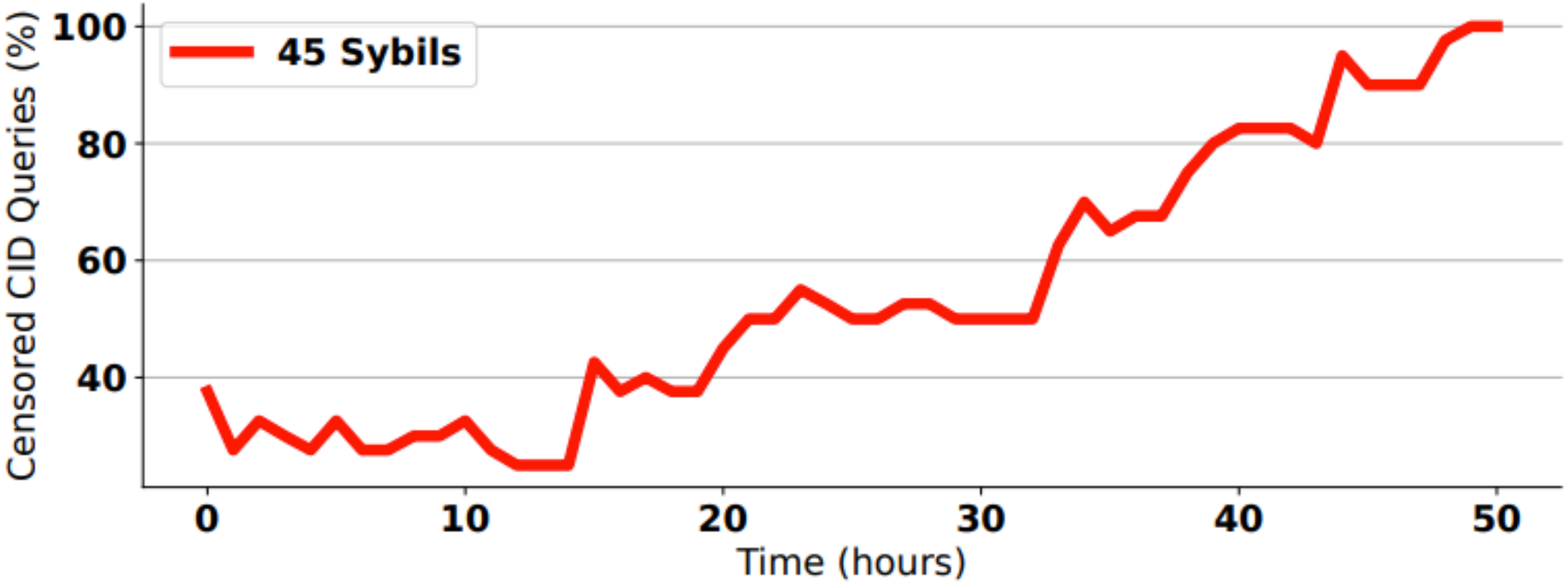


Attack cost

Generating Sybil keys : < 12 s

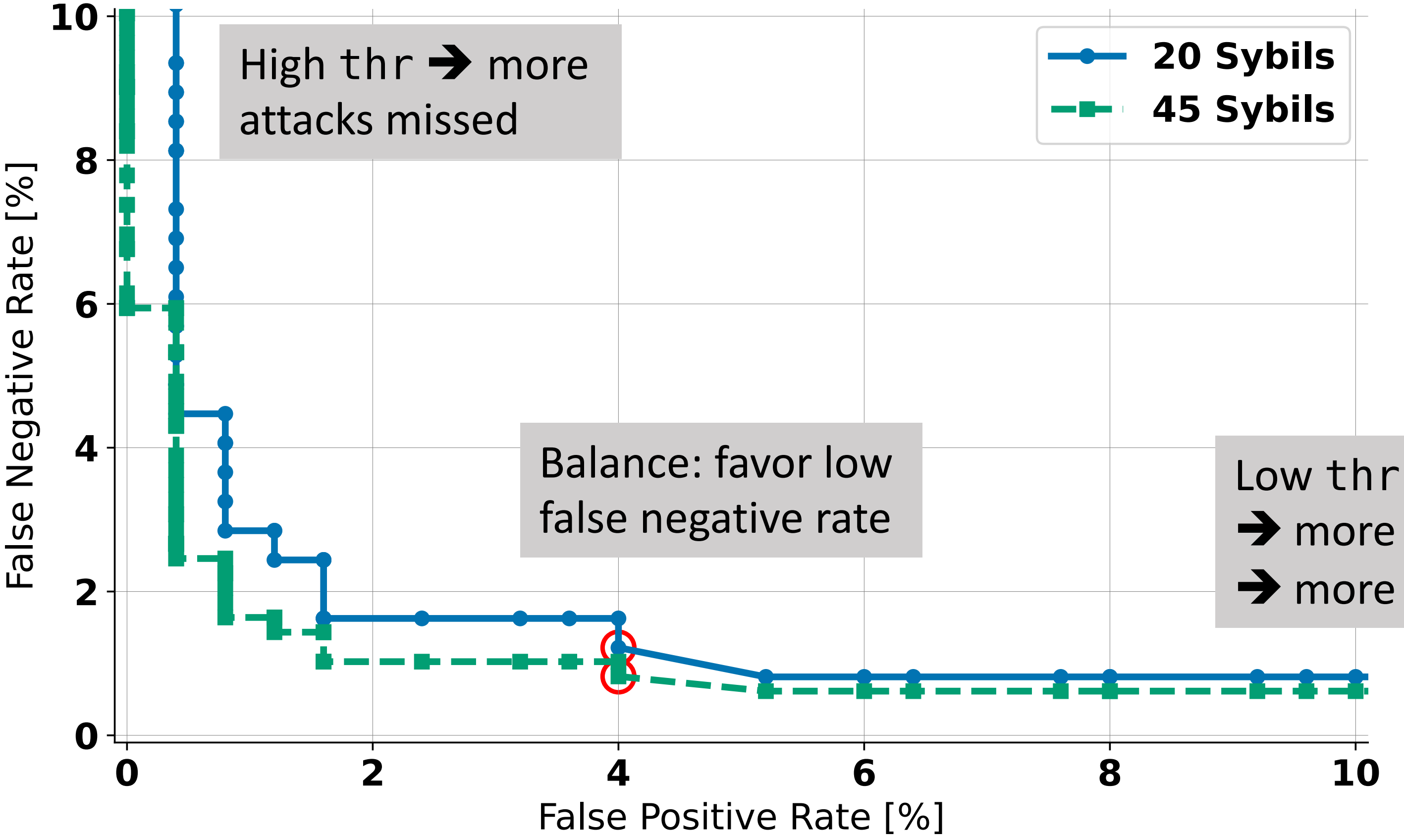
Operation cost : 0.16 $/hour

Censorship After Content is Provided



Detection Accuracy

```
if KL > thr:
    detect “Attack”
```



High thr → more attacks missed

Balance: favor low false negative rate

Low thr
→ more false detections
→ more mitigation overhead

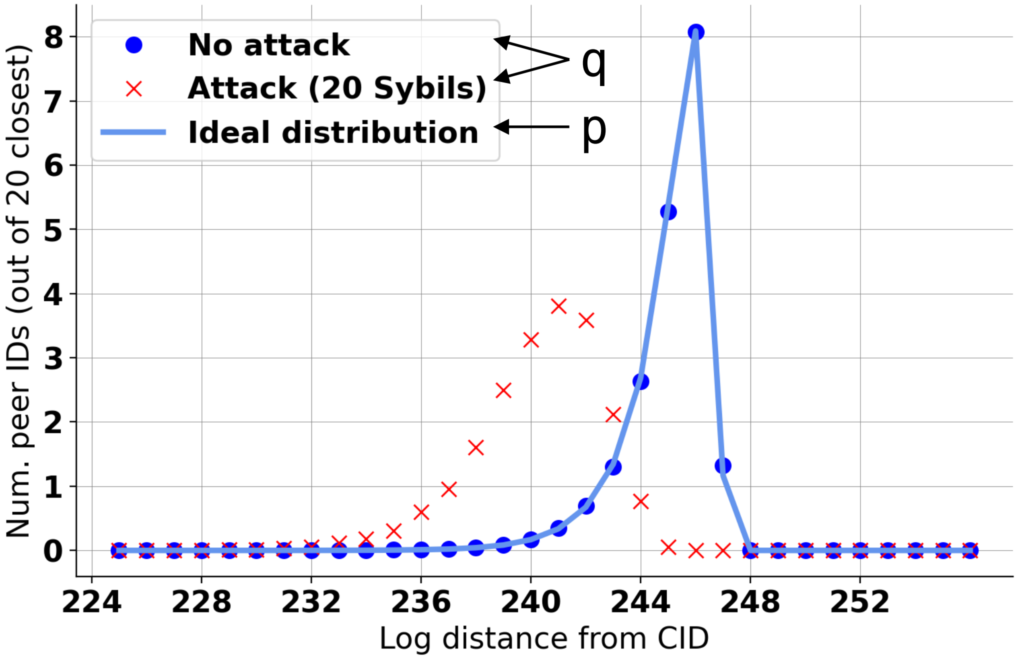
Attack Detection

Why?

- Detect before content resolution fails ⇒ Mitigate in advance
- Mitigate only when attack detected ⇒ Minimize overhead

T. Cholez, I. Chrisment, and O. Festor, “Efficient DHT attack mitigation through peers’ ID distribution”

Attack Detection

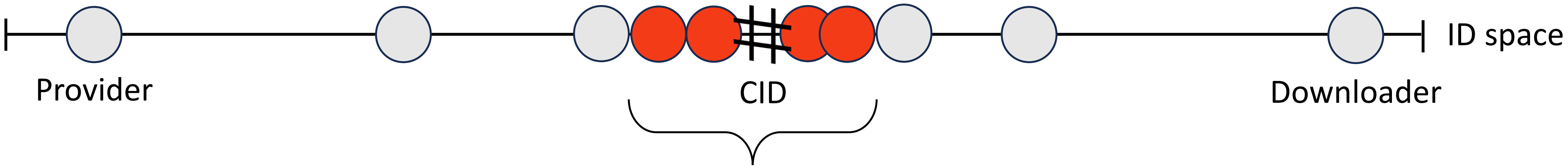


K-L Divergence

$$D(q||p) = \sum_{x} q(x) \ln\left(\frac{q(x)}{p(x)}\right)$$

```
ids ← GetClosestPeers(CID)
q ← empiricalDistribution(ids)
N ← getNetsizeEstimate()
p ← idealDistribution(N)
KL ← computeKL(p,q)
if KL > thr:
    detect “Attack”
```

Can be done by Provider or Downloader



ID space

Provider

CID

Downloader

High density of peer IDs!

Related Work

B. Prünster, A. Marsalek, T. Zefferer, “Total Eclipse of the Heart – Disrupting the InterPlanetary File System”

- Their attack eclipses one node. Our attack censors content for all nodes.
- Proposed countermeasure, IP address filters, does not defend against our attack

Other Countermeasures

Reference	Countermeasure	Problems
Prünster et al, 2022	Restrict peers with same IP address in routing table	Our attack overcomes this because Sybils may be in different peers' routing tables
S/Kademlia	Proof-of-work	Only slightly slows down attacker, makes system less sustainable
Awerbuch, Scheideler, 2009	Certificate authority	Not decentralized
CFS (Dabek et al, 2001)	Peerid = Hash(IP address)	Attacker can get many IPs, doesn't allow NAT

What Next?

- Other attacks on Kademlia: DoS, routing attacks?
- Resistance to massive-scale Sybil attacks?
- Impact on other systems using Kademlia (BitTorrent, eMule, Swarm, Storj)?

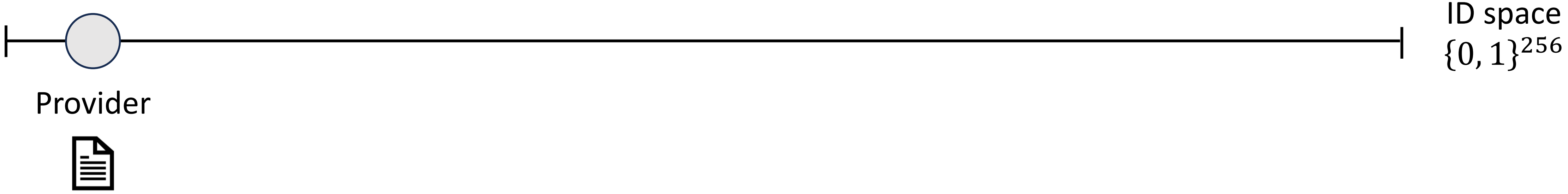
Kademlia in IPFS

Kademlia in IPFS

Provide content:

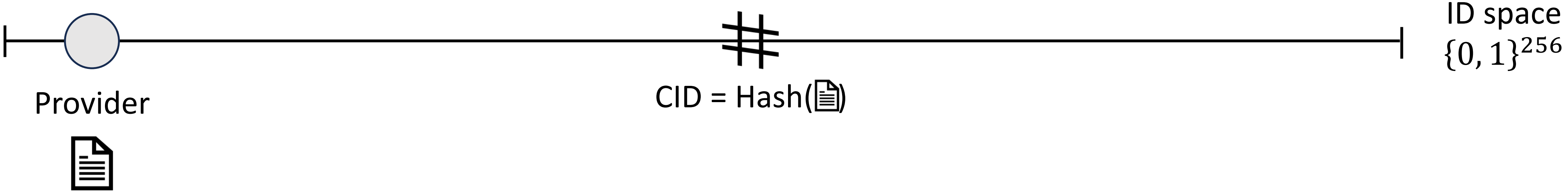
Kademlia in IPFS

Provide content:



Kademlia in IPFS

Provide content:

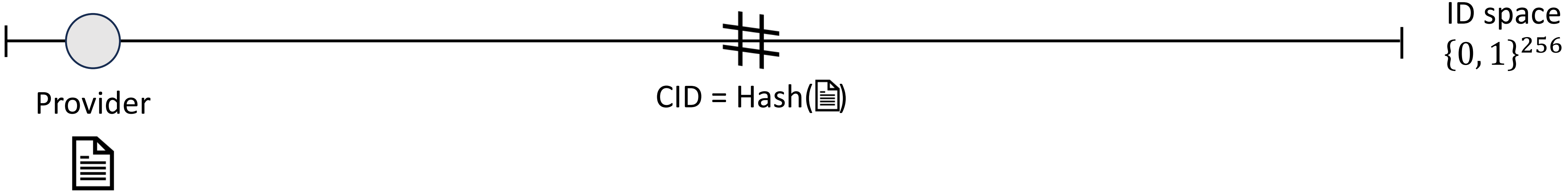


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

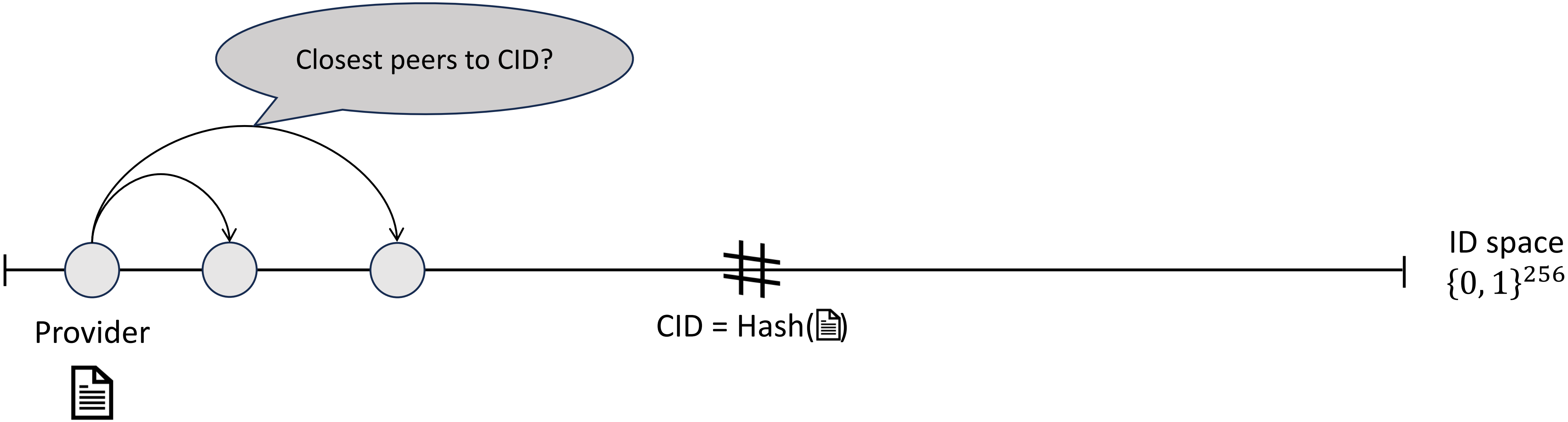


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

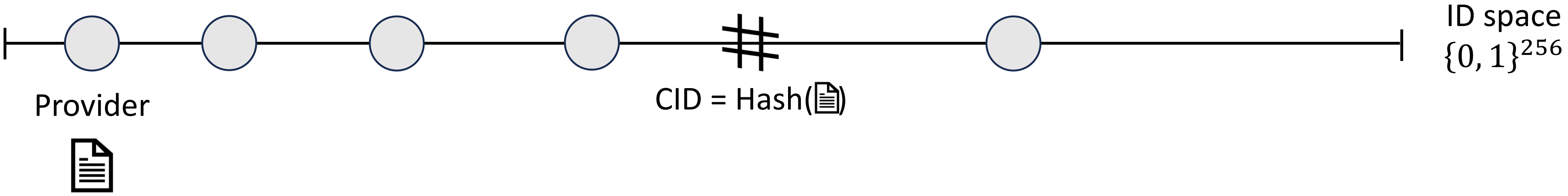


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

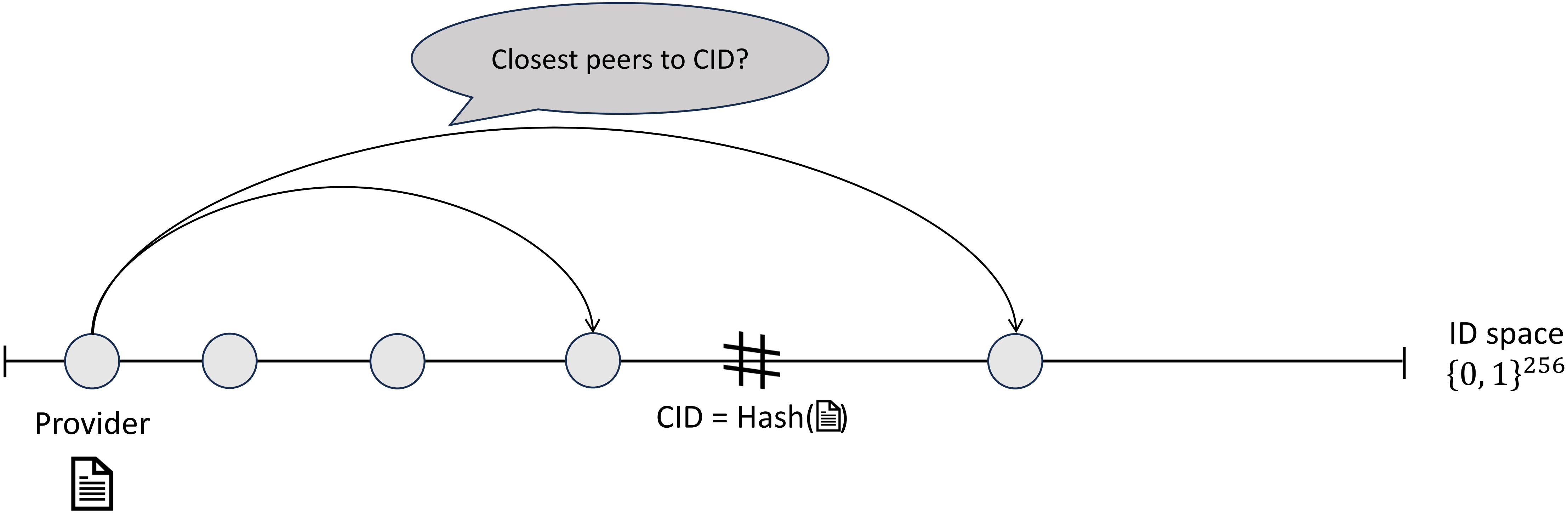


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

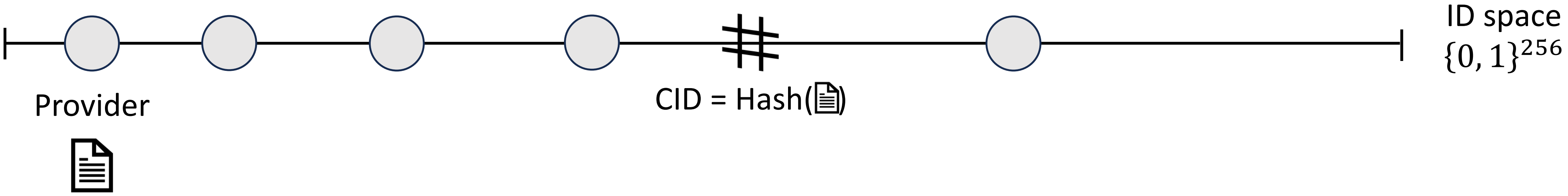


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

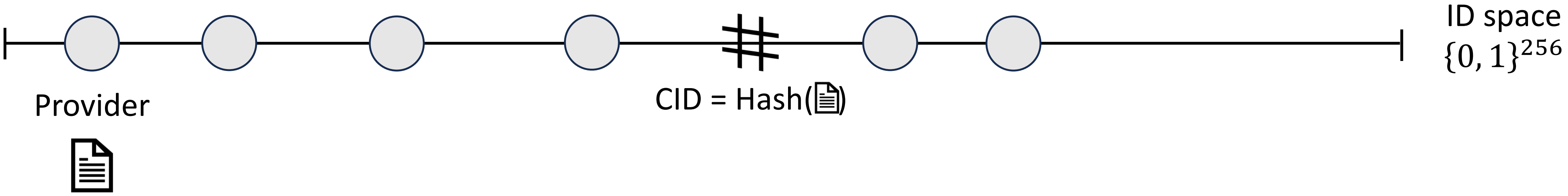


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

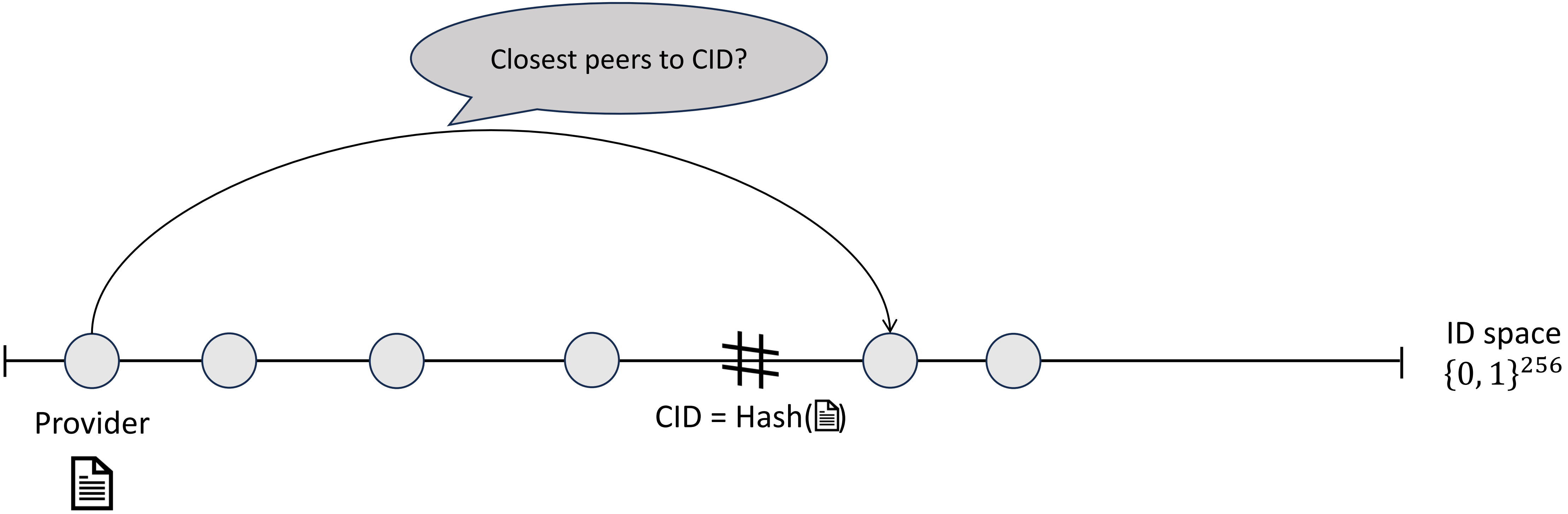


Kademlia in IPFS

Provide content:

1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

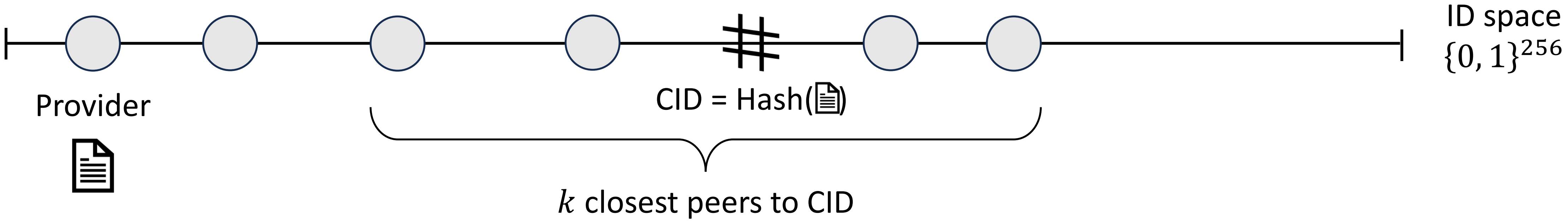


Kademlia in IPFS

Provide content:

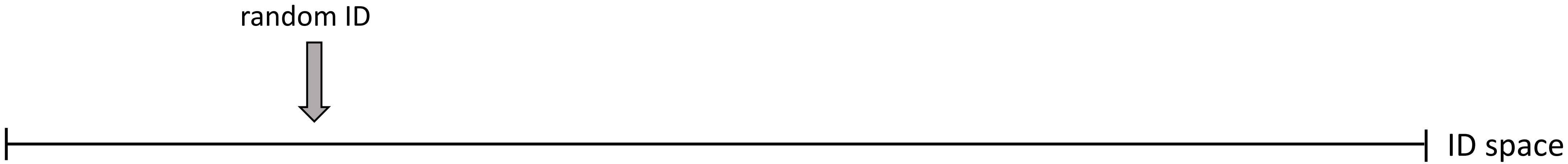
1. Find k (=20) closest peers to CID.

```
dist(id1,id2)
= id1 XOR id2
```

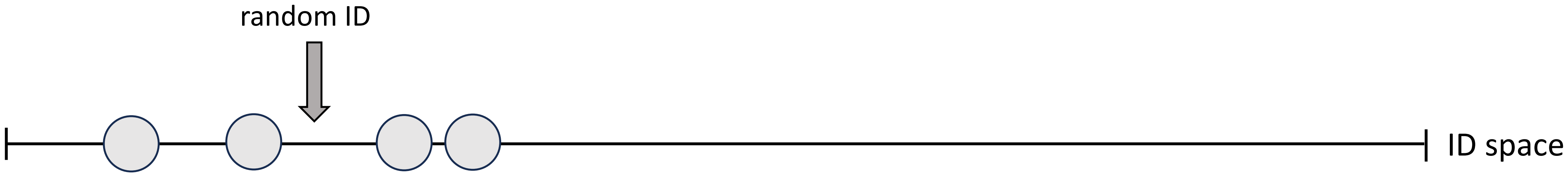


Network Size Estimation

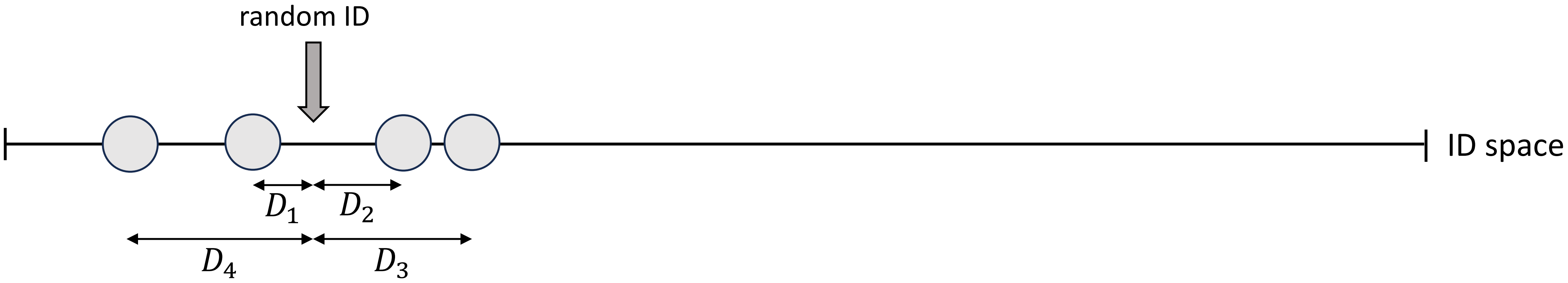
Network Size Estimation



Network Size Estimation



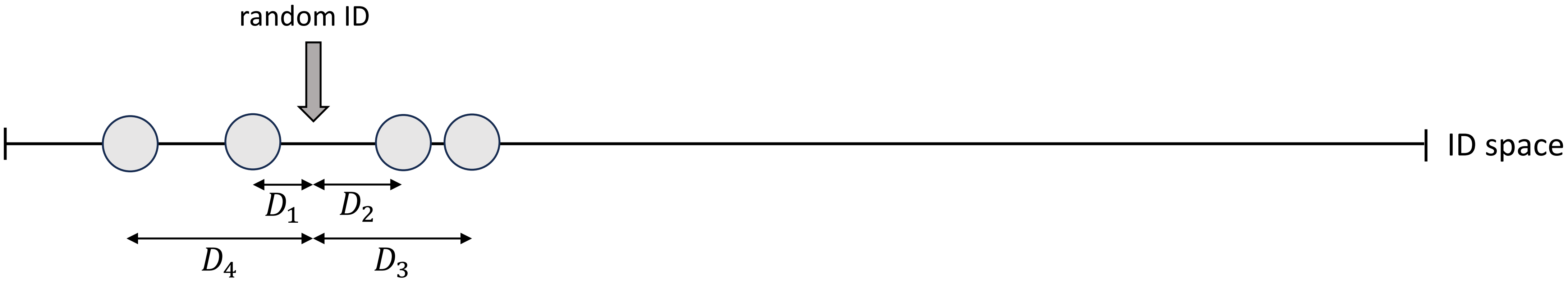
Network Size Estimation



Network Size Estimation

Estimate network size from distances

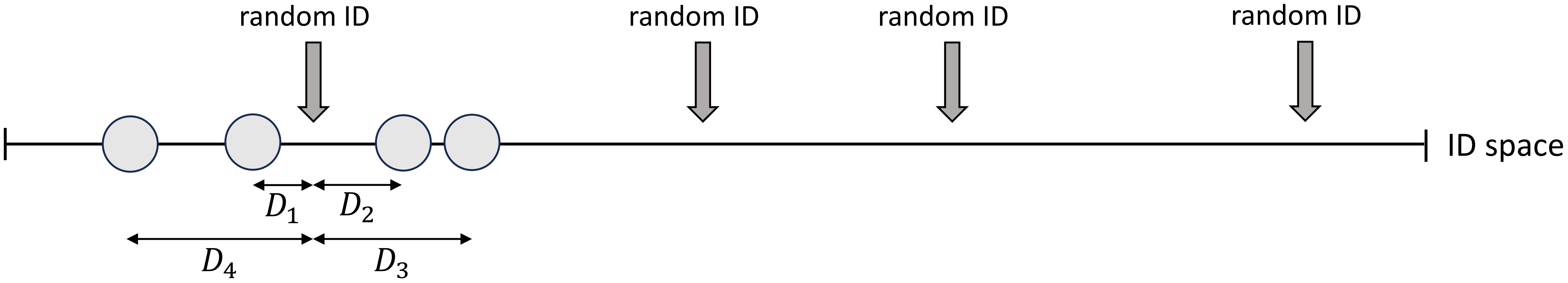
$$\hat{N} = \arg\min_{N} \sum_{i=1}^{k} \left(D_i - \frac{2^{256} i}{N+1} \right)^2$$



Network Size Estimation

Estimate network size from distances $\hat{N} = \arg\min_{N} \sum_{i=1}^{k} \left(D_i - \frac{2^{256} i}{N+1} \right)^2$

Average over many random ID



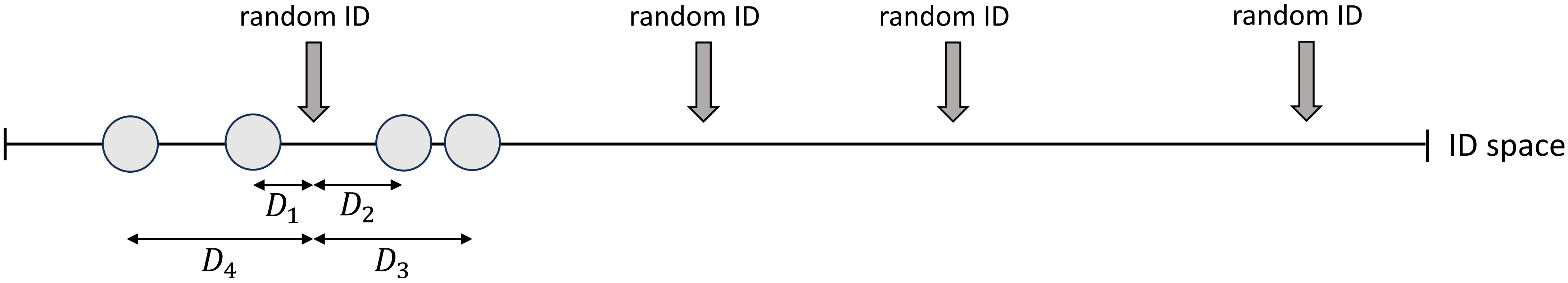
Network Size Estimation

Estimate network size from distances

$$\hat{N} = \arg\min_{N} \sum_{i=1}^{k} \left(D_i - \frac{2^{256} i}{N+1} \right)^2$$

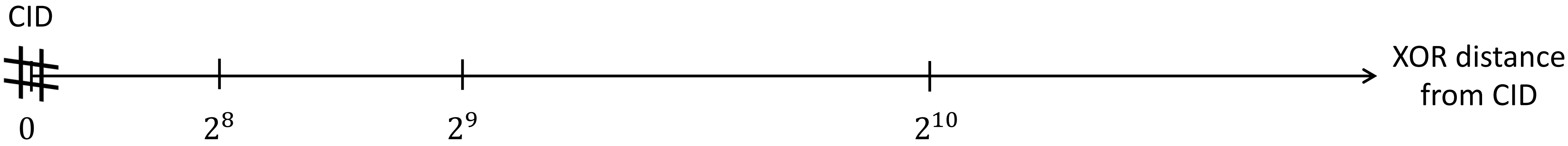
Average over many random ID

Random IDs hard to bias by attacker! ✅



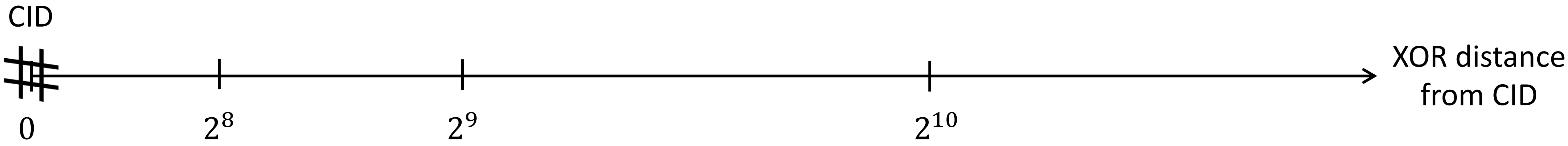
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
```



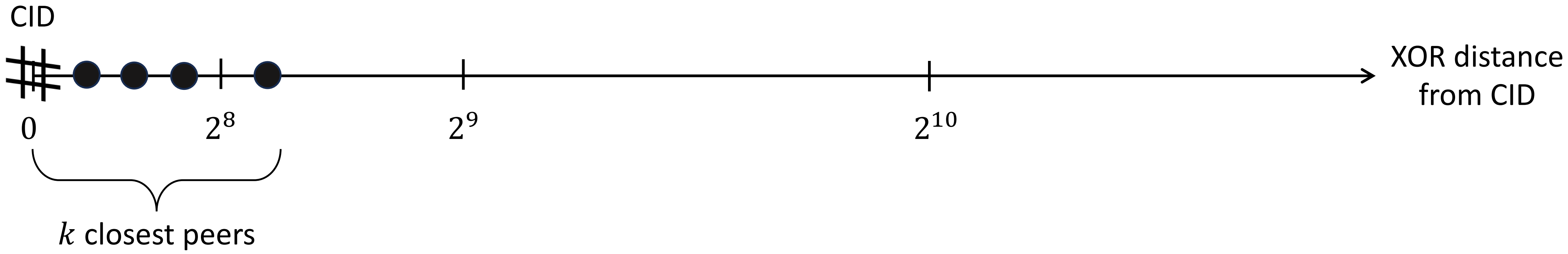
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
```



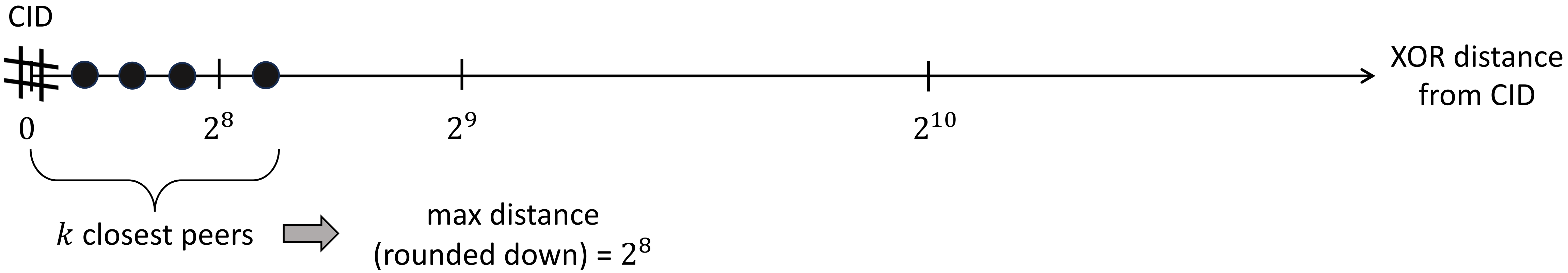
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
```



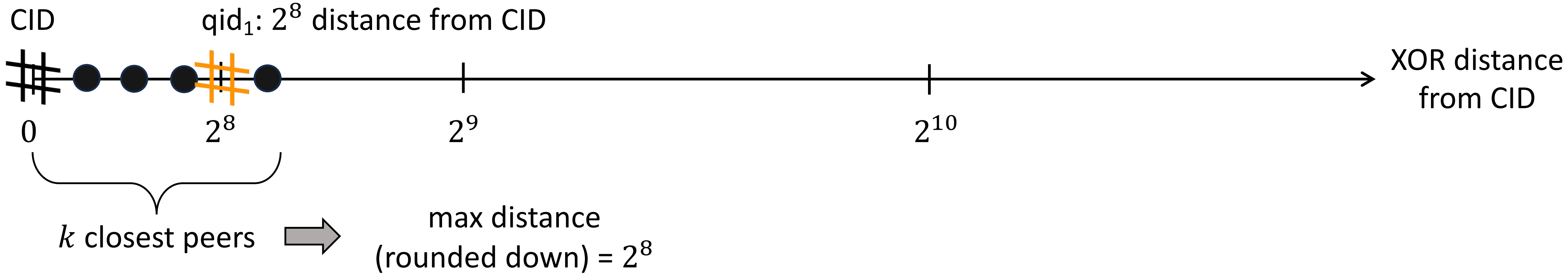
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
```



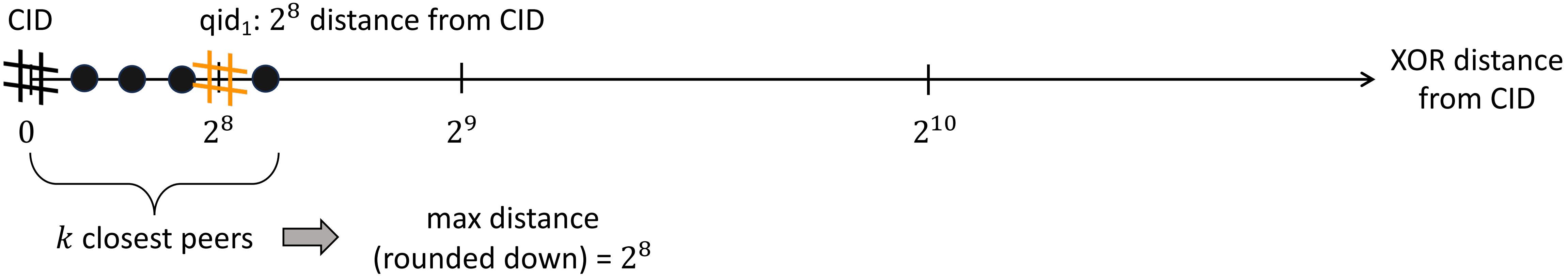
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
```


CID
qid$_1$: 2^8 distance from CID
XOR distance from CID
0
2^8
2^9
2^{10}
k closest peers
max distance (rounded down) = 2^8

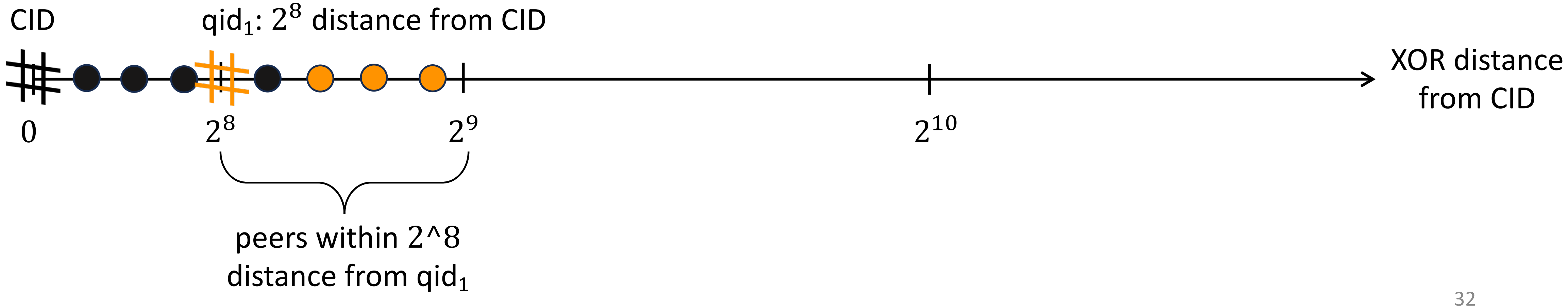
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
```


CID
qid$_1$: 2^8 distance from CID
XOR distance from CID
0
2^8
2^9
2^{10}
k closest peers
max distance (rounded down) = 2^8

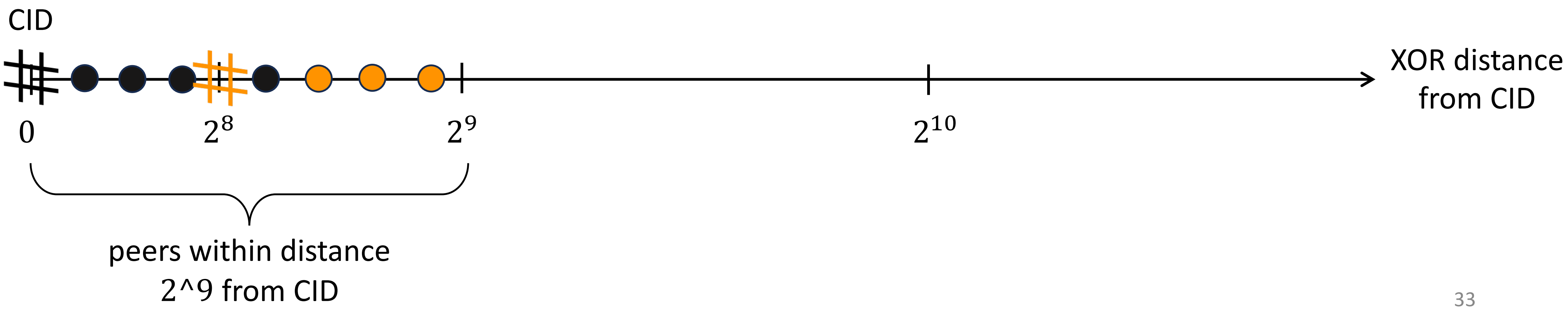
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
```


CID
qid_1: 2^8 distance from CID
XOR distance from CID
0
2^8
2^9
2^{10}
peers within 2^8 distance from qid_1

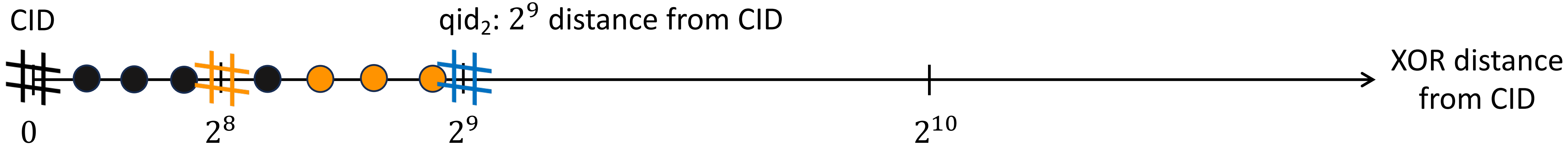
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
```



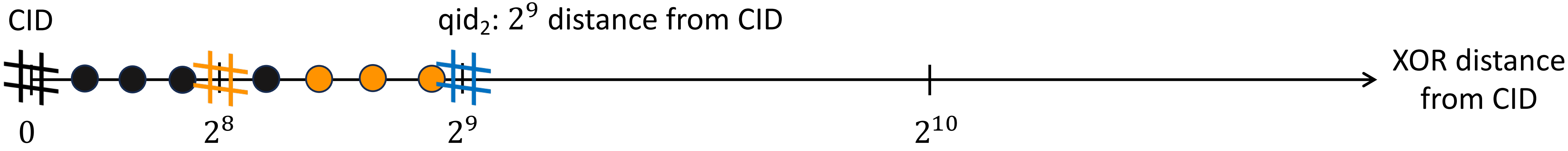
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
```



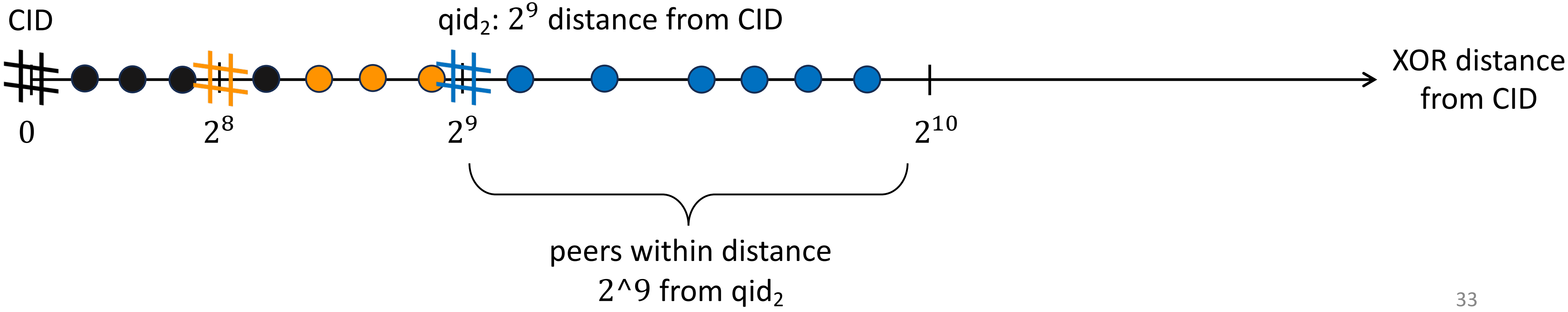
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
  GetPeersByDistance(qid₂, 2^9)
```



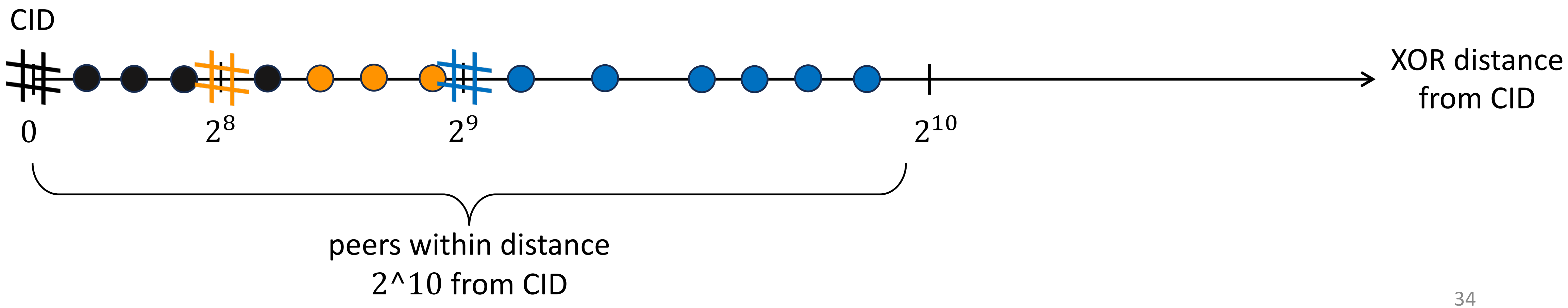
Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
  GetPeersByDistance(qid₂, 2^9)
```



Finding Peers Within Distance from CID

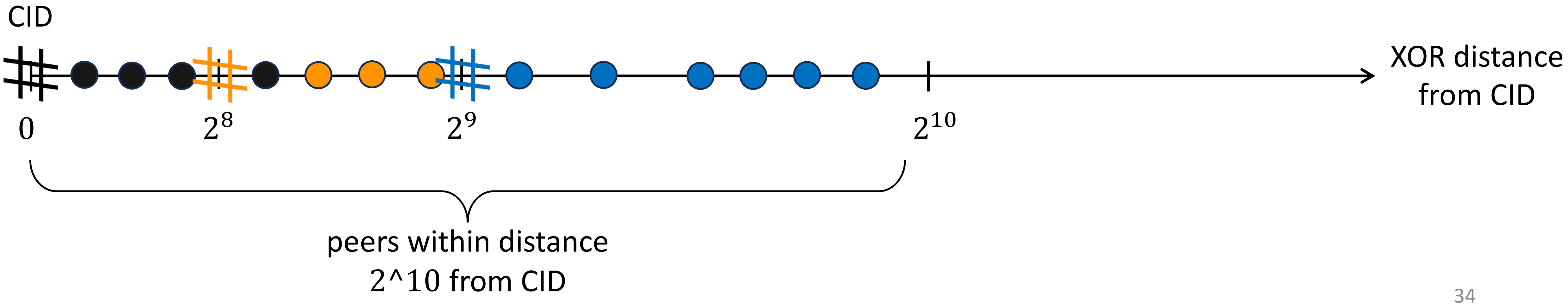
```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
  GetPeersByDistance(qid₂, 2^9)
```



Finding Peers Within Distance from CID

```
GetPeersByDistance(CID, 2^10):
  GetKClosestPeers(CID)
  GetPeersByDistance(qid₁, 2^8)
  GetPeersByDistance(qid₂, 2^9)
```

Recursive, using multiple GetKClosestPeers() lookups


CID
XOR distance from CID
0
2^8
2^9
2^{10}
peers within distance 2^10 from CID