The LOCKSS Approach: A Primer

34th International Conference on Massive Storage Systems and Technology
May 14, 2018
Santa Clara, California

Thib Guicherd-Callin <thib@cs.stanford.edu>
Technical Manager, LOCKSS Program
Digital Library Systems and Services, Stanford University Libraries

Overview

1. Key Publications
2. Basic Polling, Voting and Repair
3. LCAP Concepts Illustrated
4. LCAP In Depth

Key Publications

Key Publications

- David S.H. Rosenthal, Vicky Reich. "*Permanent Web Publishing*." Proceedings of the 2000 USENIX Annual Technical Conference FREENIX Track, pg. 129-140, 2000. URL: https://www.usenix.org/legacy/publications/library/proceedings/usenix2000/freenix/rosenthal.html

Key Publications

- Petros Maniatis, Mema Roussopoulos, TJ Giuli, David S.H. Rosenthal, Mary Baker, and Yanto Muliadi. "*Preserving Peer Replicas By Rate-Limited Sampled Voting*." Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles (SOSP '03), pg. 44-59, 2003. DOI: `10.1145/945445.945451`
- Petros Maniatis, Mema Roussopoulos, TJ Giuli, David S.H. Rosenthal, Mary Baker, and Yanto Muliadi. "*LOCKSS: A Peer-To-Peer Digital Preservation System*." Technical report `cs.CR/0303026`, Stanford University, 2003. URL: `http://www.eecs.harvard.edu/~mema/publications/SOSP2003-long.pdf`

Key Publications

- David S.H. Rosenthal, Thomas S. Robertson, Tom Lipkis, Vicky Reich, Seth Morabito. "*Requirements for Digital Preservation Systems: A Bottom-Up Approach*." D-Lib Magazine, vol. 11, iss. 11, November 2005. DOI: `10.1045/november2005-rosenthal`

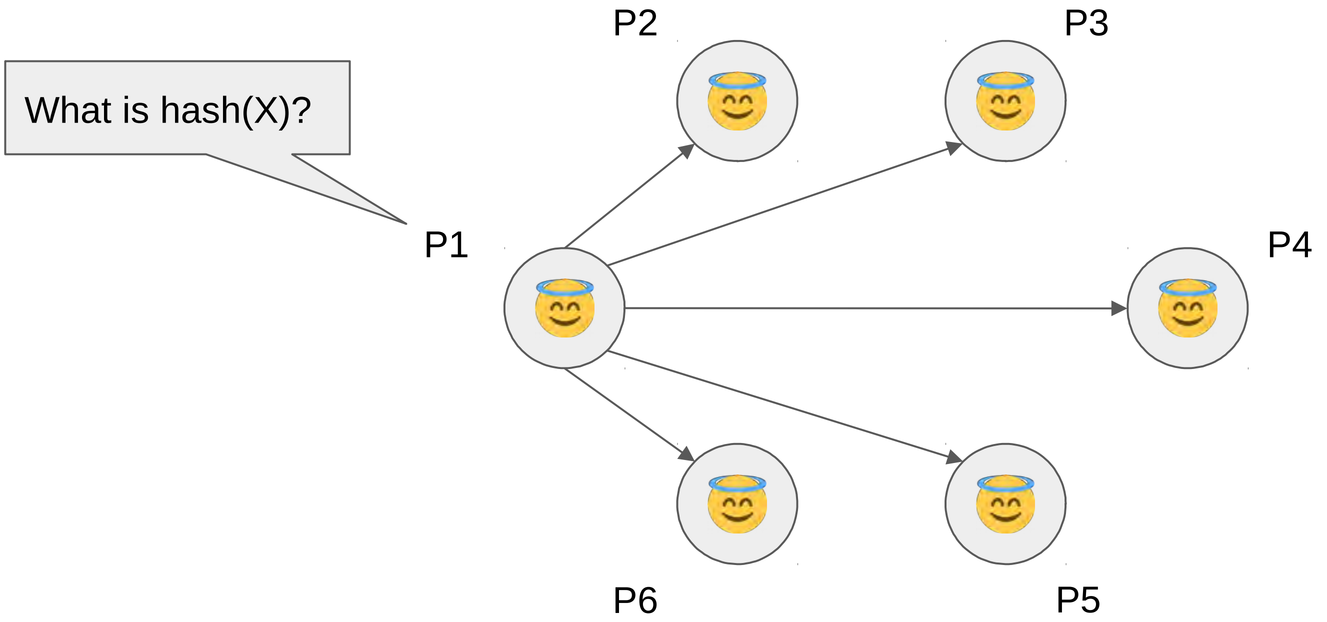
Key Publications

- David S.H. Rosenthal, Daniel Vargas, Tom Lipkis and Claire Griffin. "Enhancing the LOCKSS Digital Preservation Technology." D-Lib Magazine, vol. 21, iss. 9/10, September/October 2015. DOI: `10.1045/september2015-rosenthal`

Basic Polling, Voting and Repair

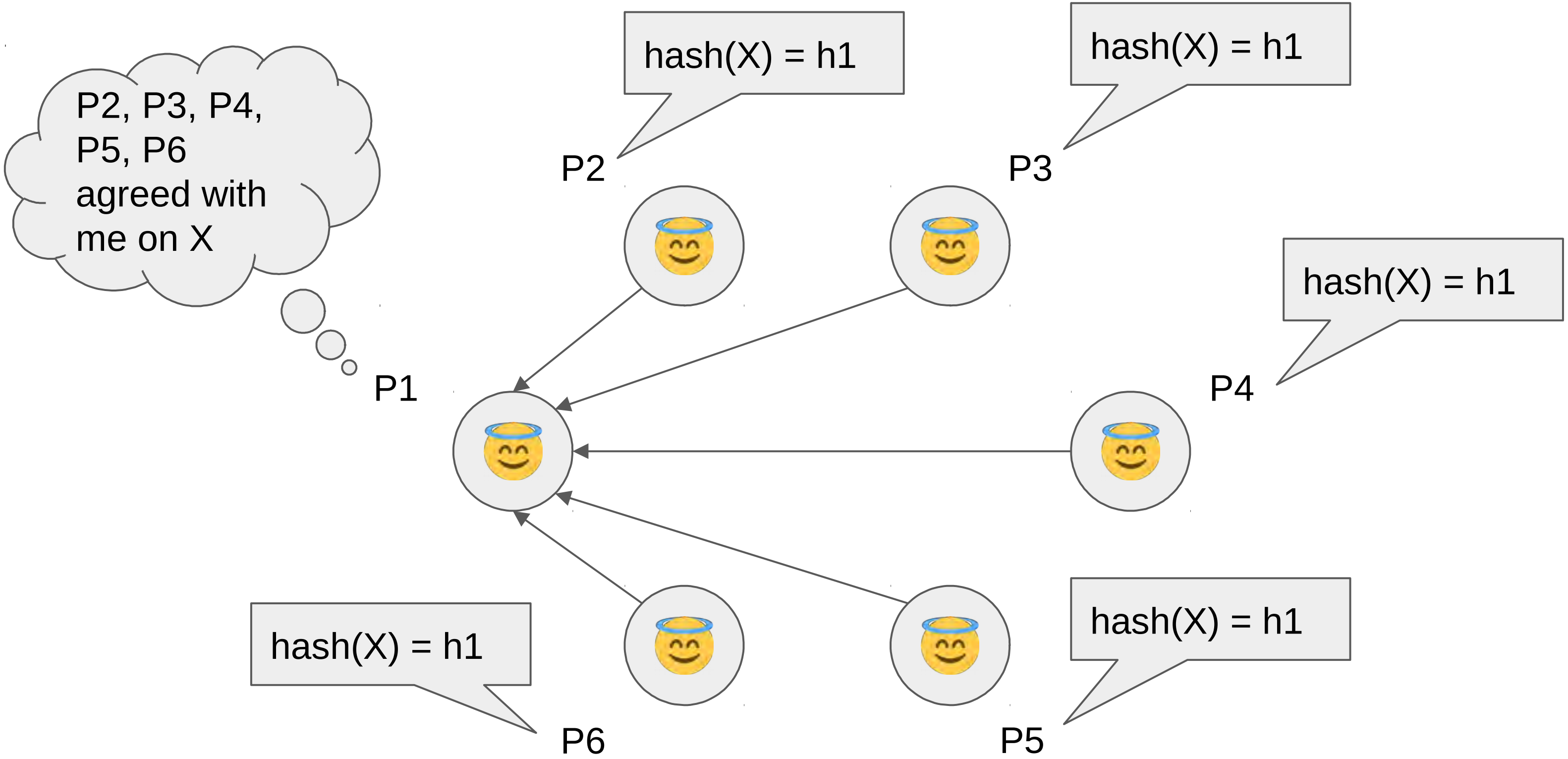
X

The peers hold identical replicas of X
Peer P1 calls a poll on content X



Landslide agreement

X



P2, P3, P4, P5, P6 agreed with me on X

P1

P2

hash(X) = h1

P3

hash(X) = h1

P4

hash(X) = h1

P5

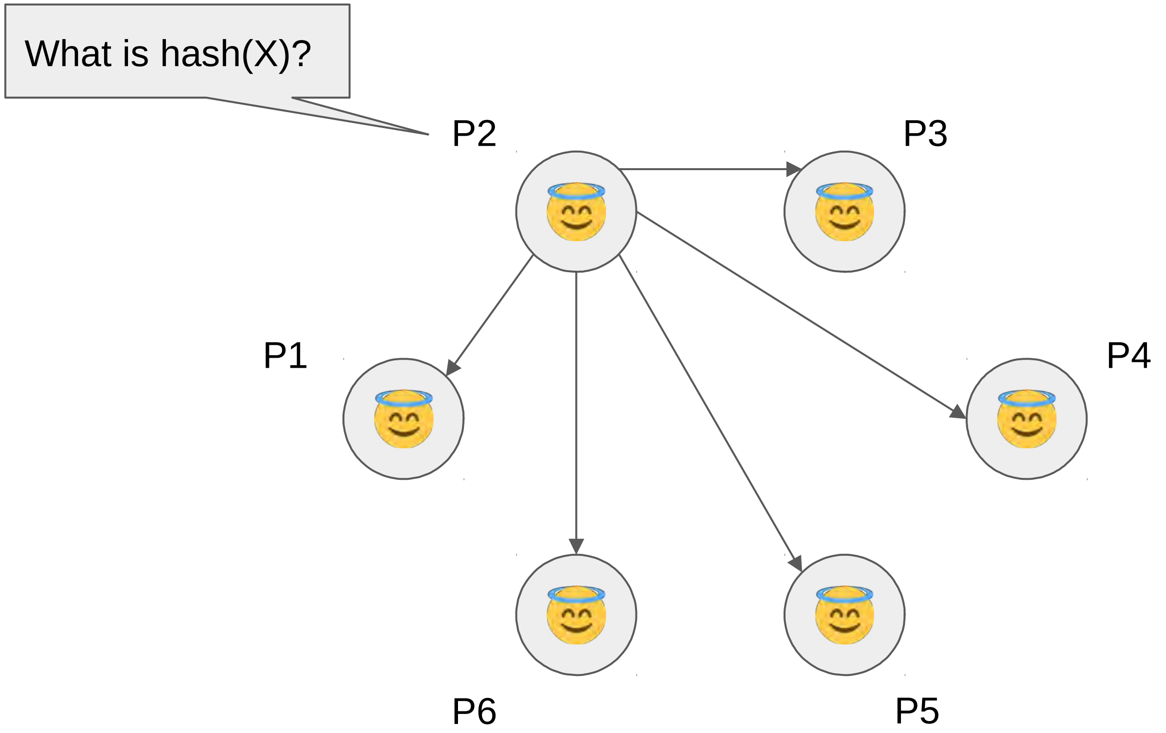
hash(X) = h1

P6

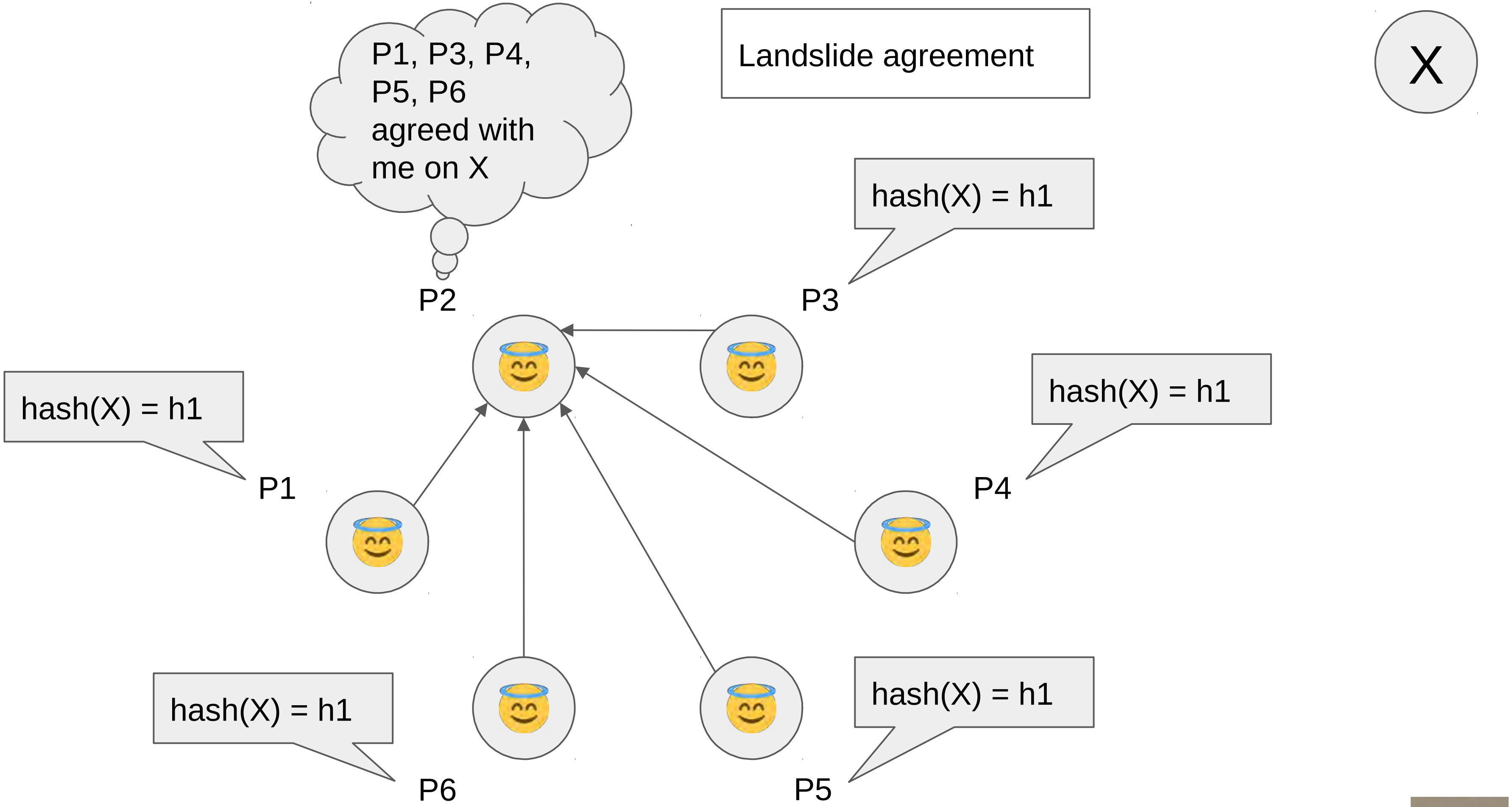
hash(X) = h1

X

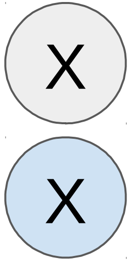
Peer P2 calls a poll on content X

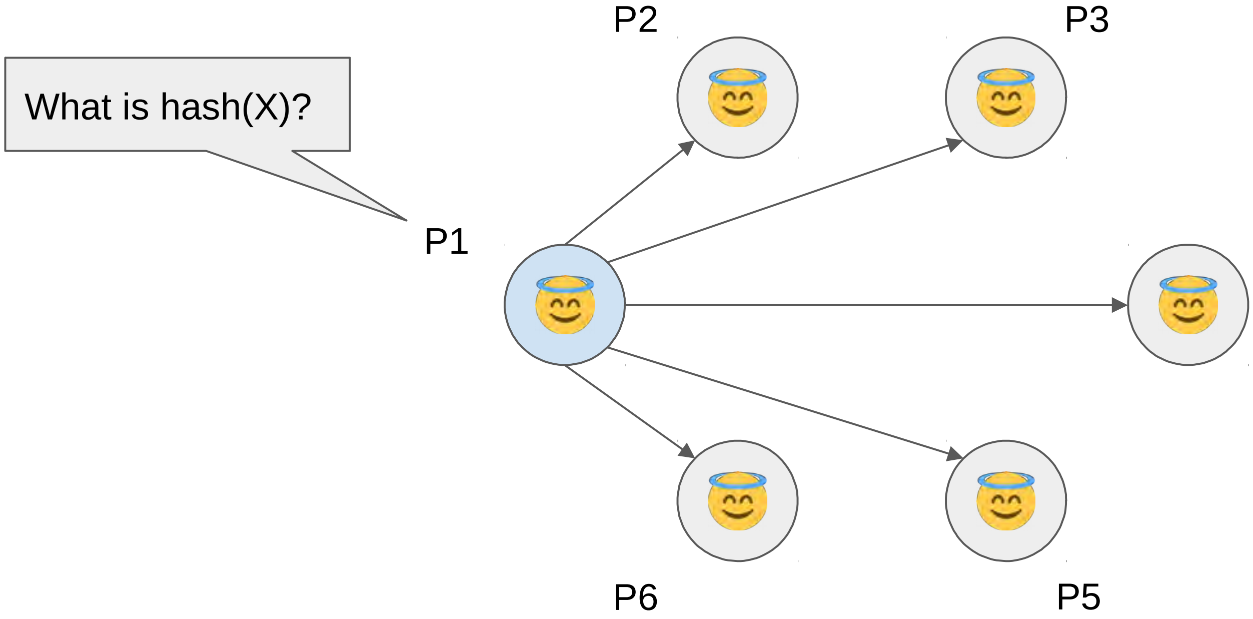


Landslide agreement

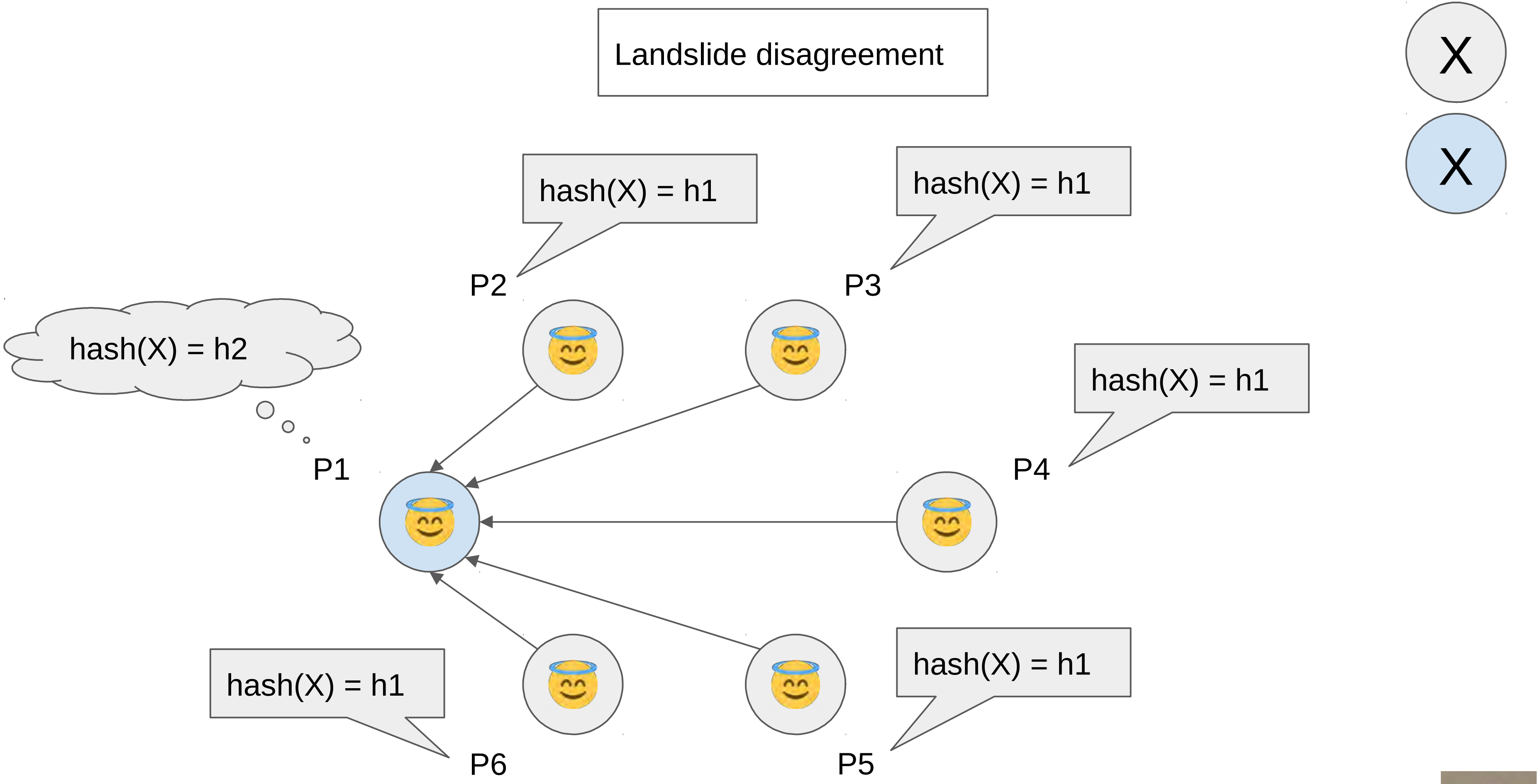


Peer P1 incurs damage on content X
Peer P1 later calls a poll on content X





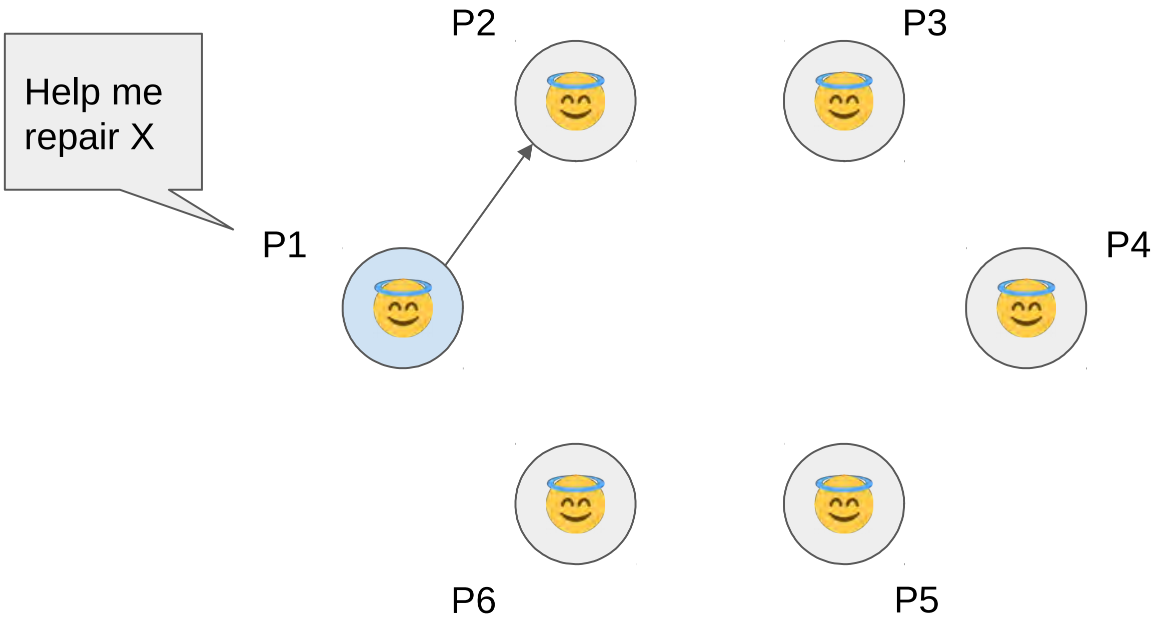
Landslide disagreement



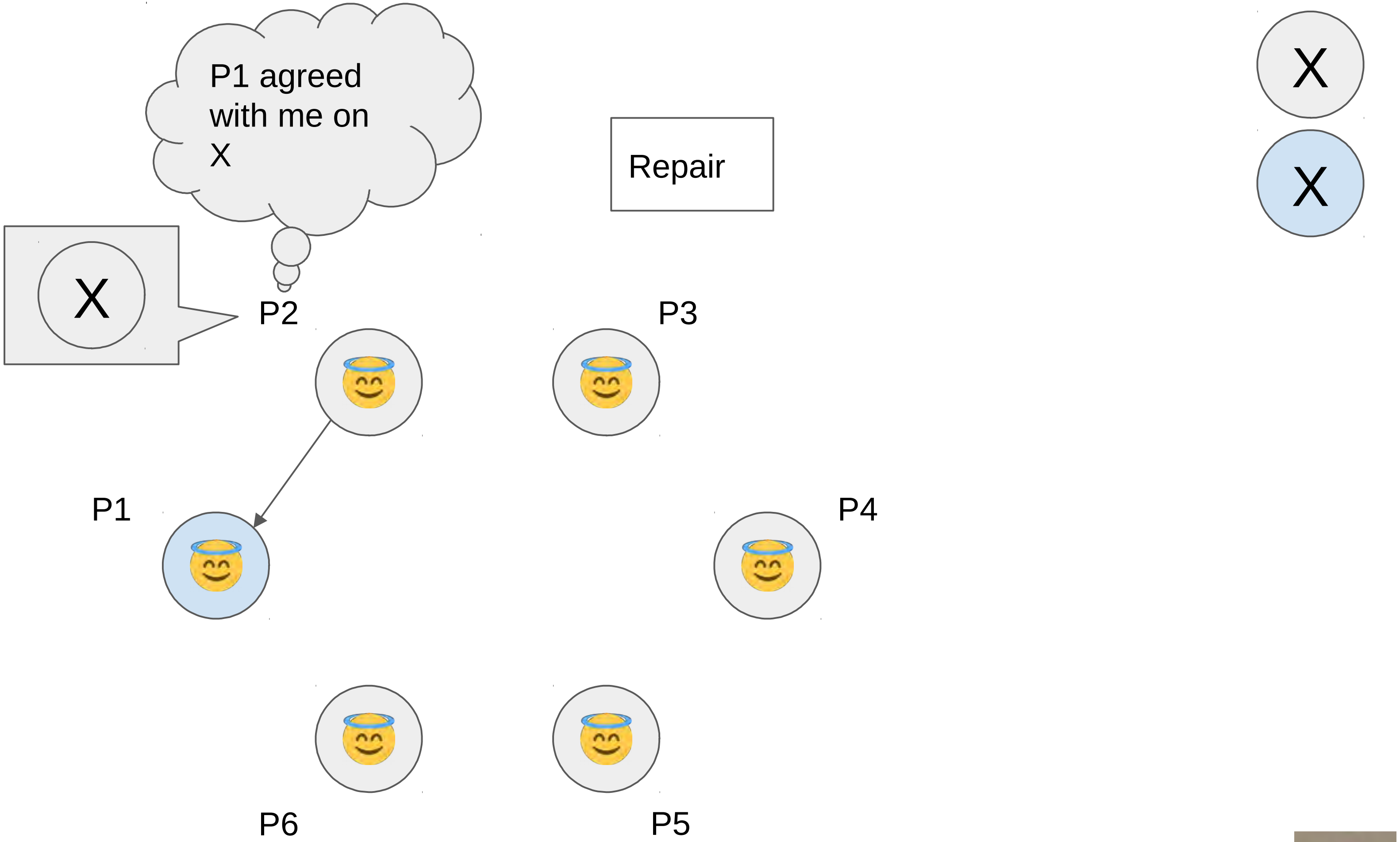
Repair request

X

X



Help me repair X



P1 agreed with me on X

Repair

X

X

X

P2

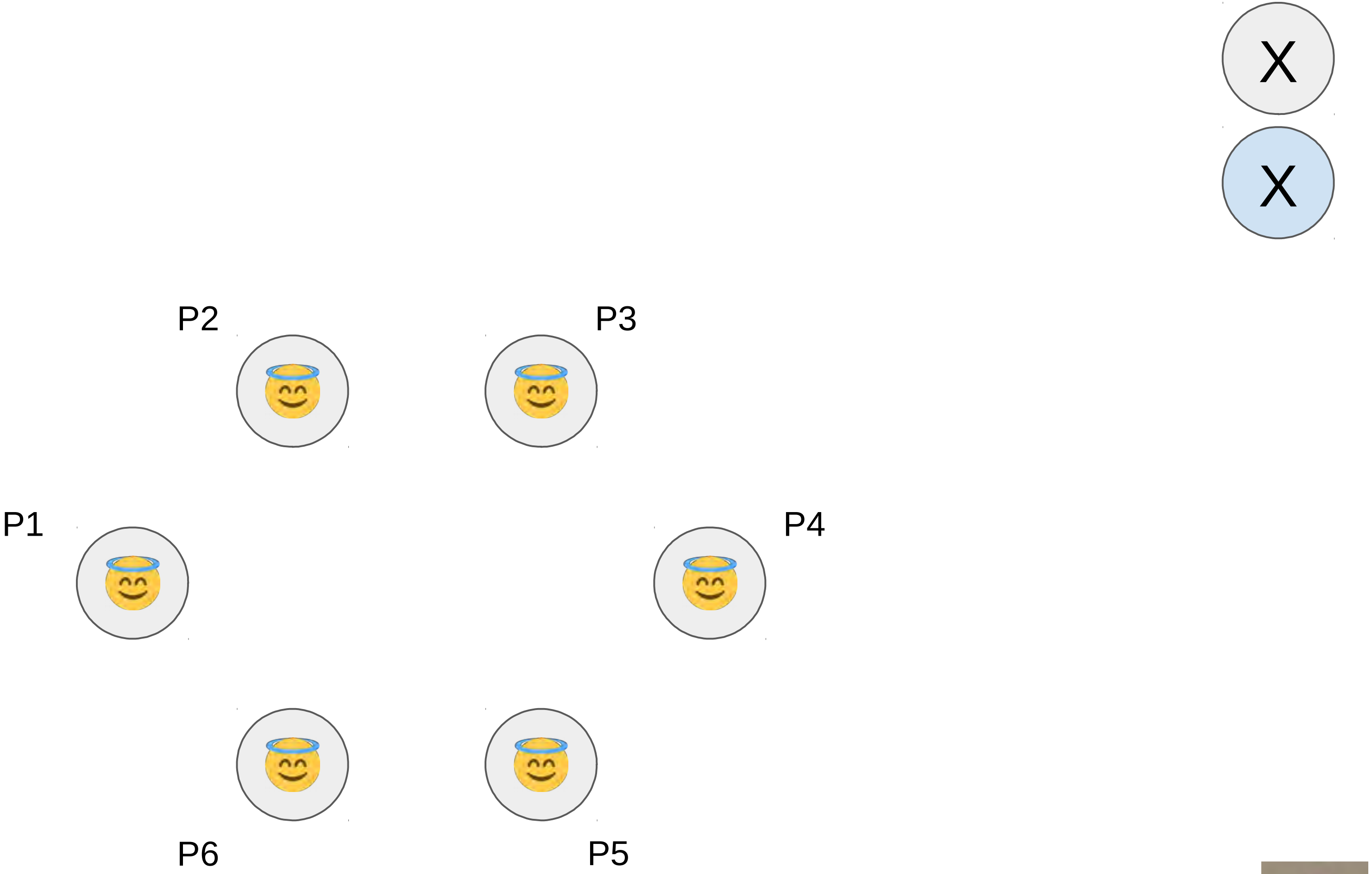
P3

P1

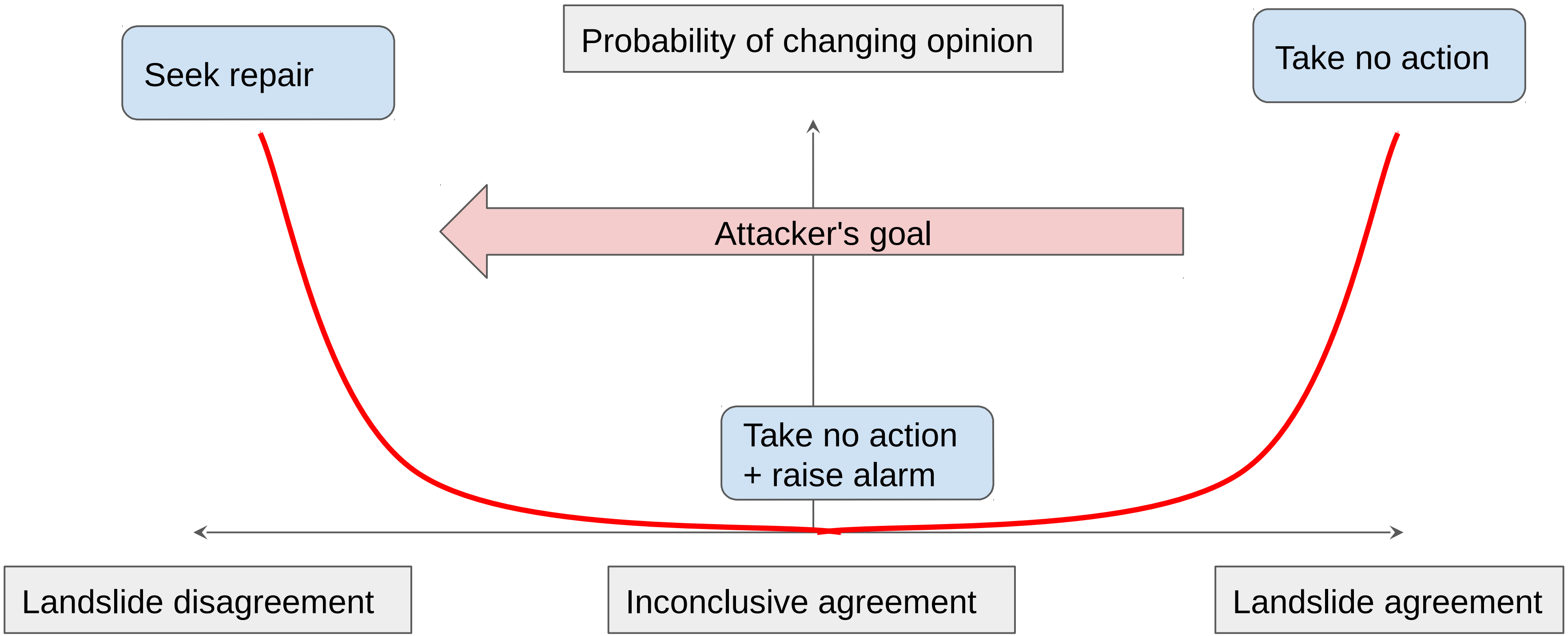
P4

P6

P5


X
X
P2
P3
P1
P4
P6
P5

Stealth Modification Gap



Probability of changing opinion

Seek repair

Take no action

Attacker's goal

Take no action
+ raise alarm

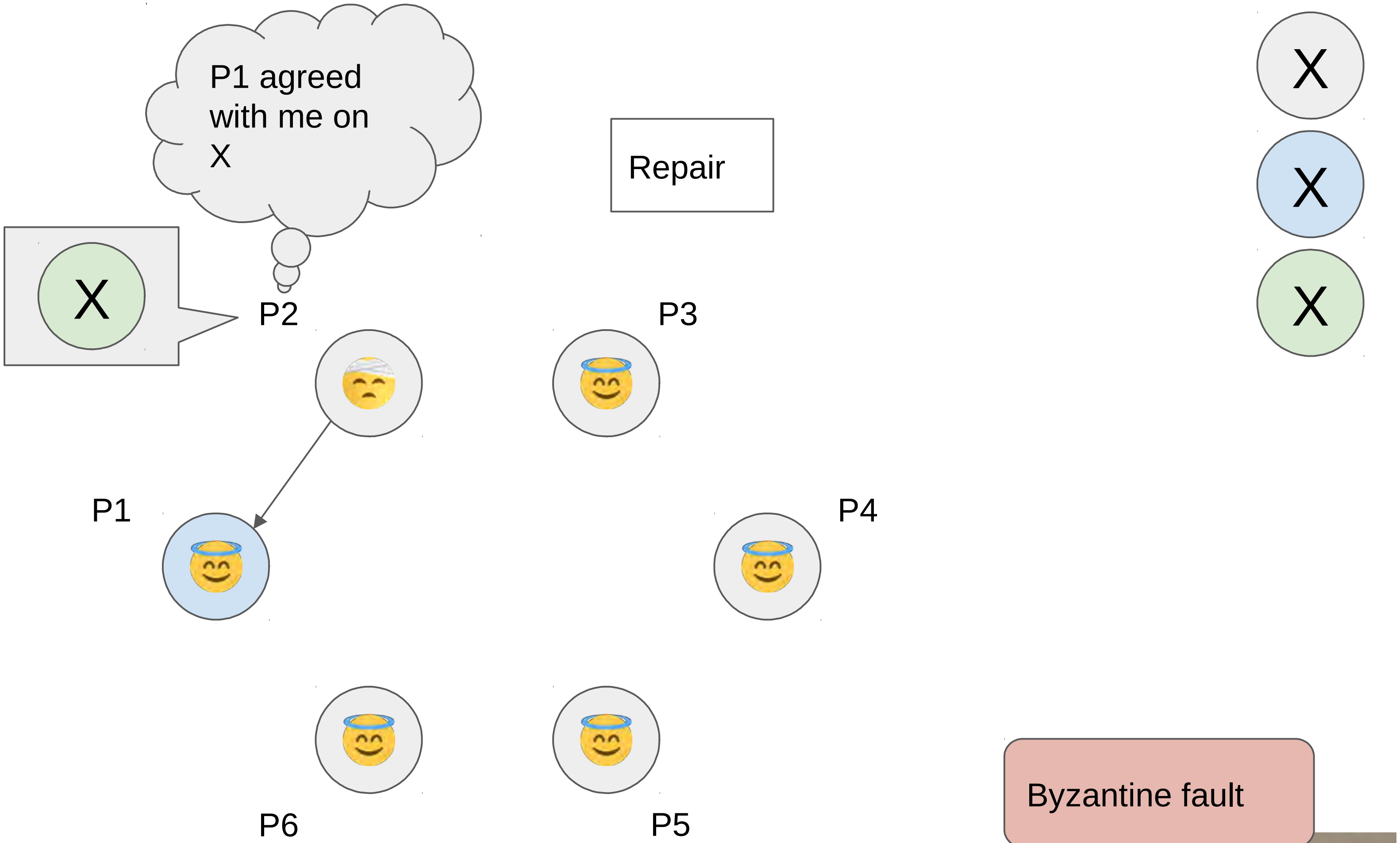
Landslide disagreement

Inconclusive agreement

Landslide agreement

LCAP Concepts Illustrated

Byzantine Fault
Bait and Switch



P1 agreed with me on X

Repair

X

P2

P3

P1

P4

P6

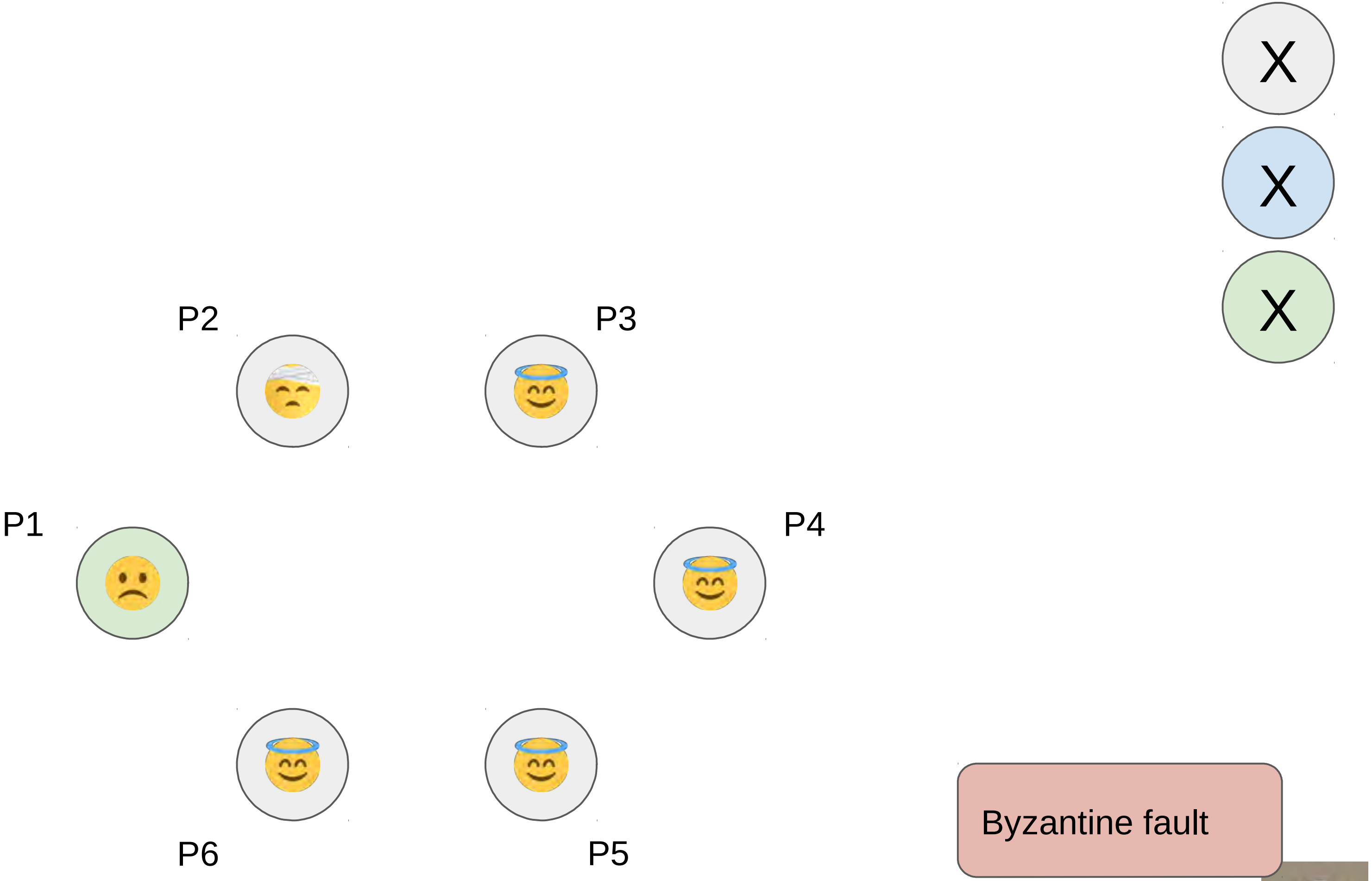
P5

X

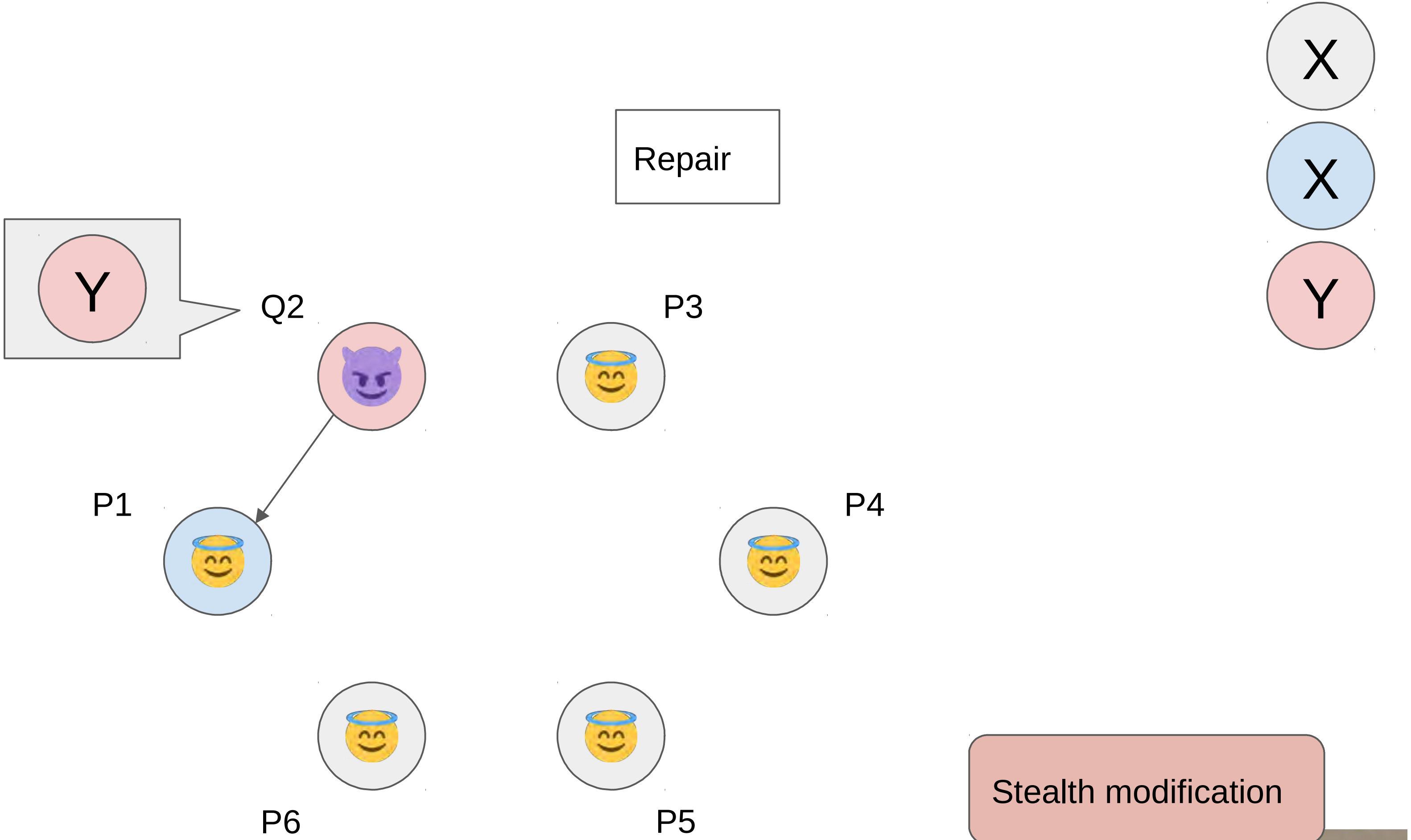
X

X

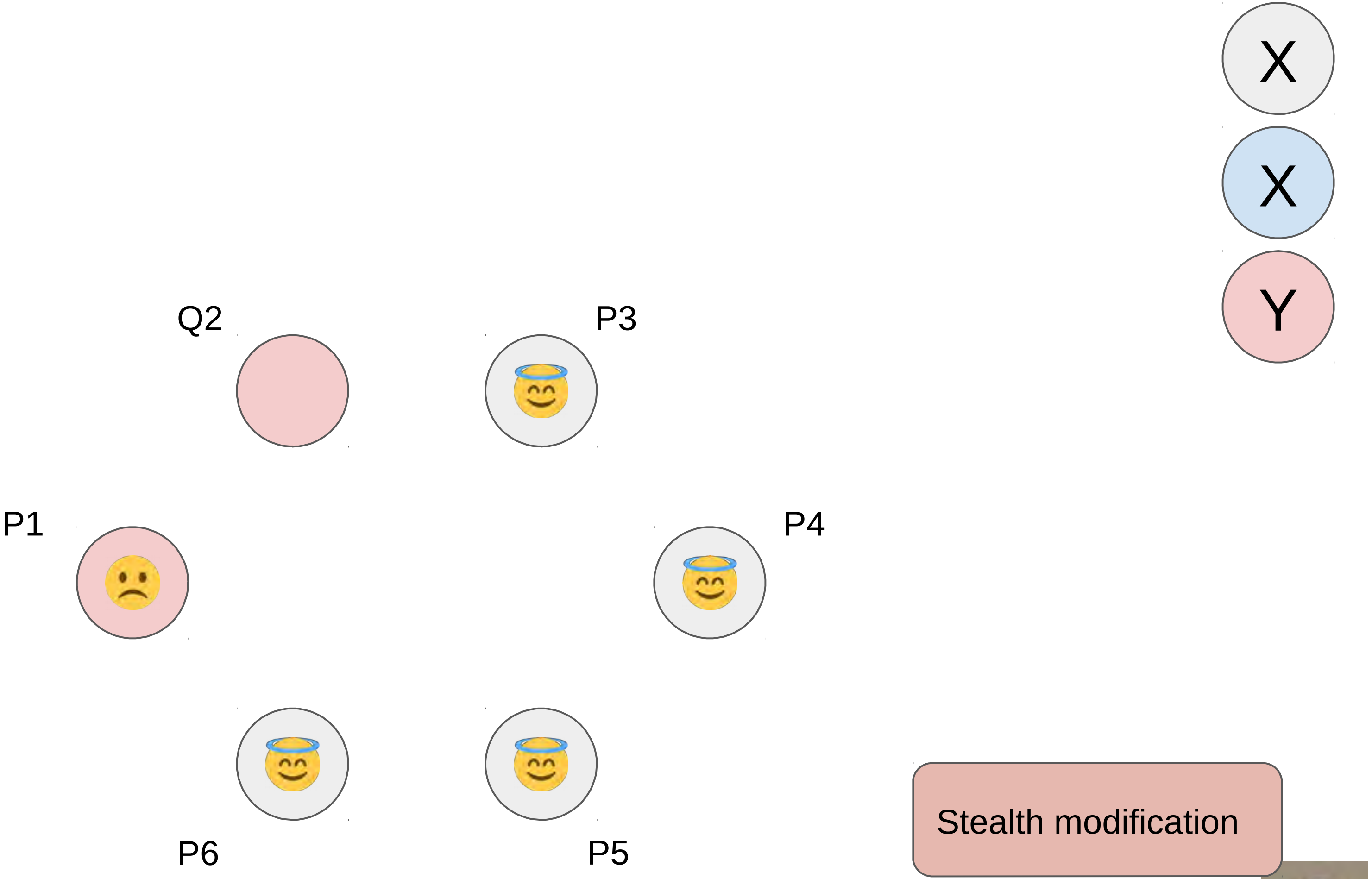
Byzantine fault



Byzantine fault

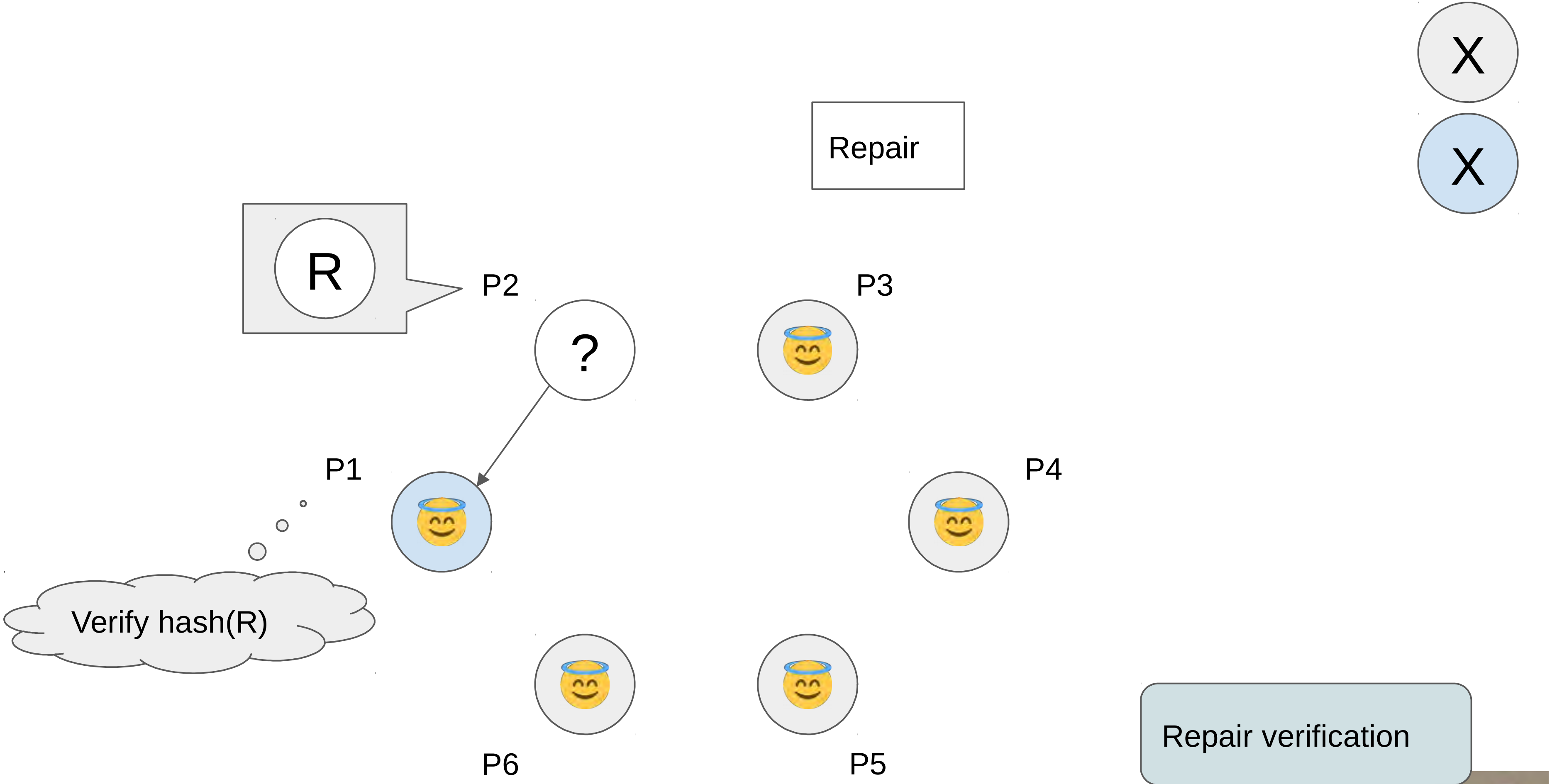


Stealth modification

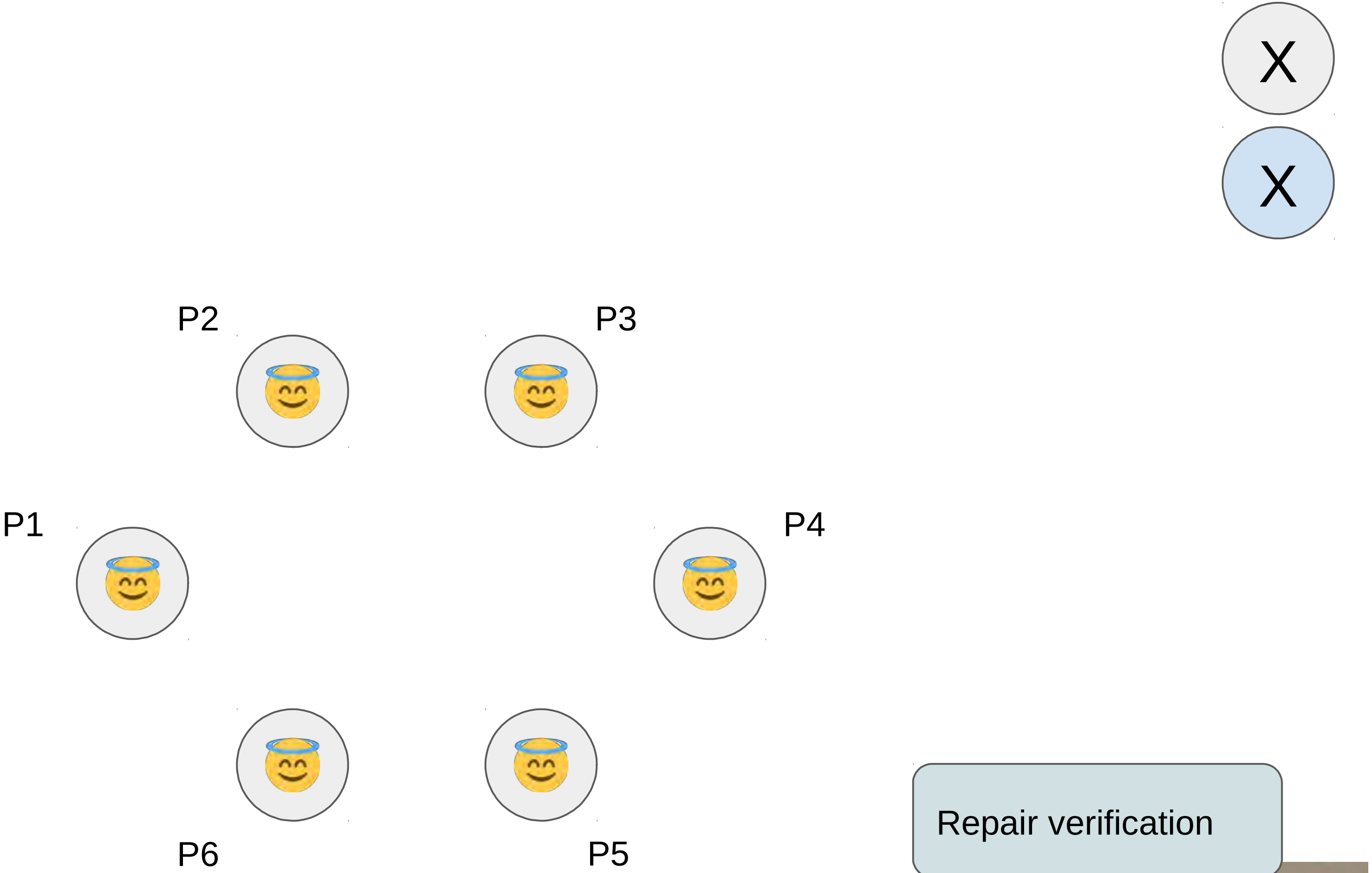


Stealth modification

Repair Verification

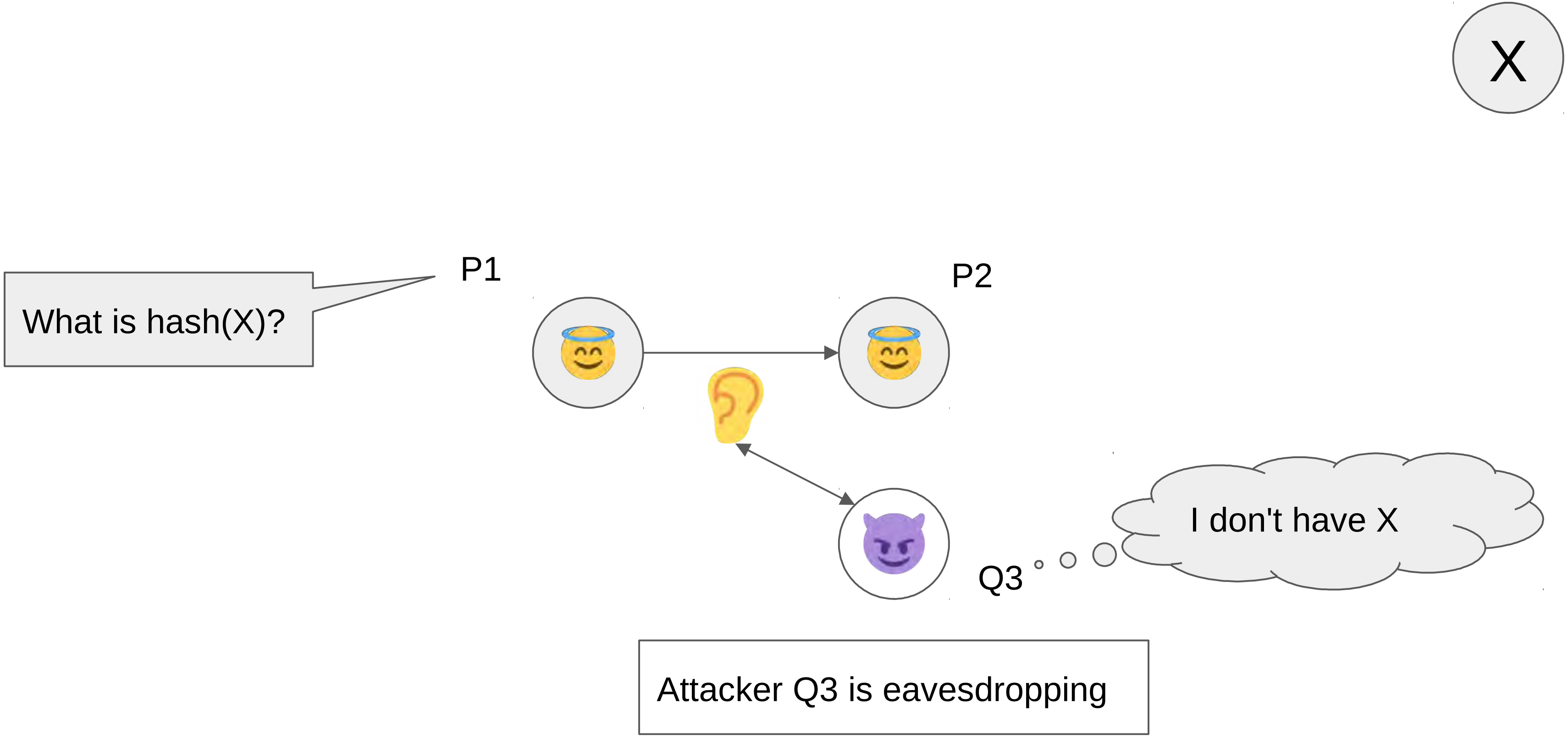


Repair verification



Repair verification

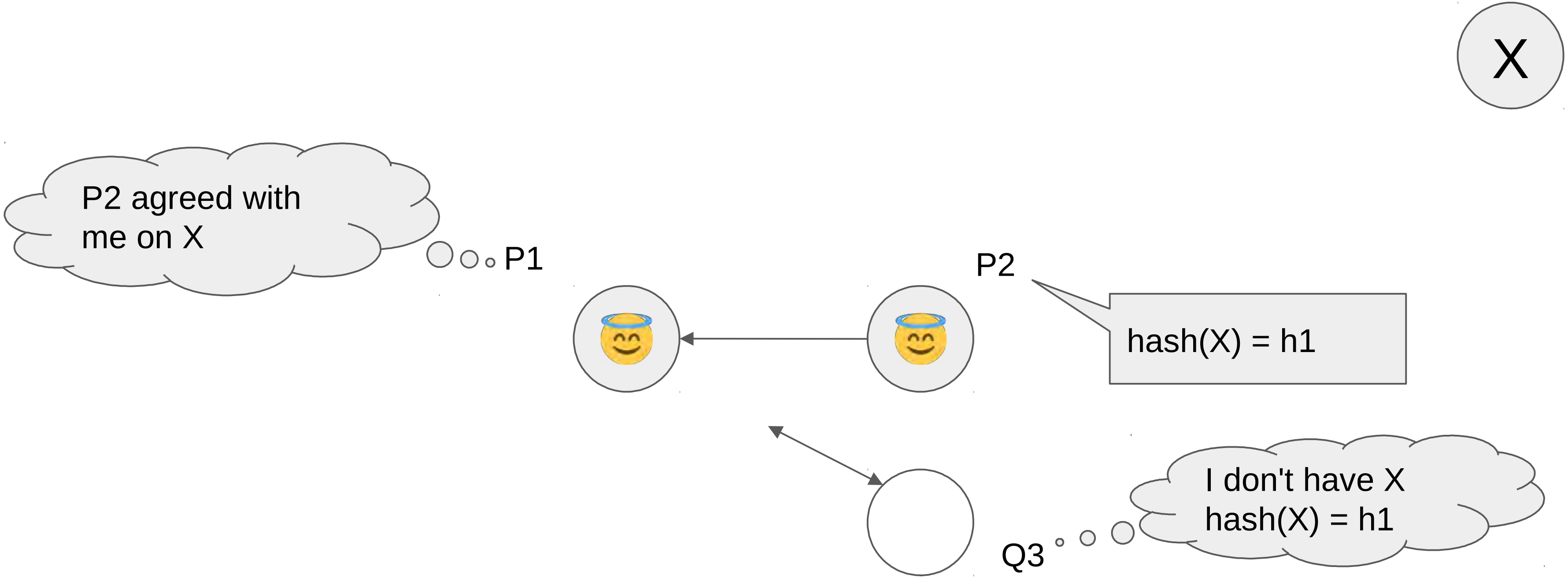
Replay Attack



What is hash(X)?

I don't have X

Attacker Q3 is eavesdropping



X

P2 agreed with me on X

P1

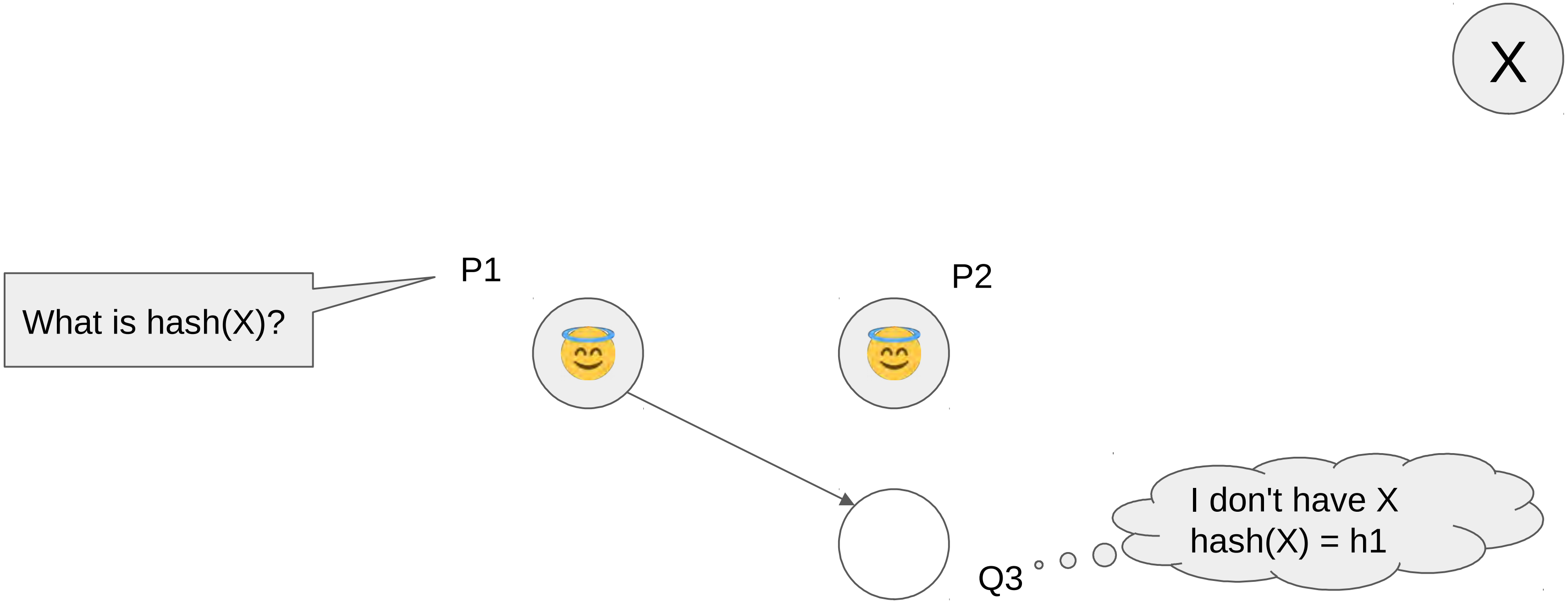
P2

hash(X) = h1

Q3

I don't have X
hash(X) = h1

Attacker Q3 is eavesdropping



X

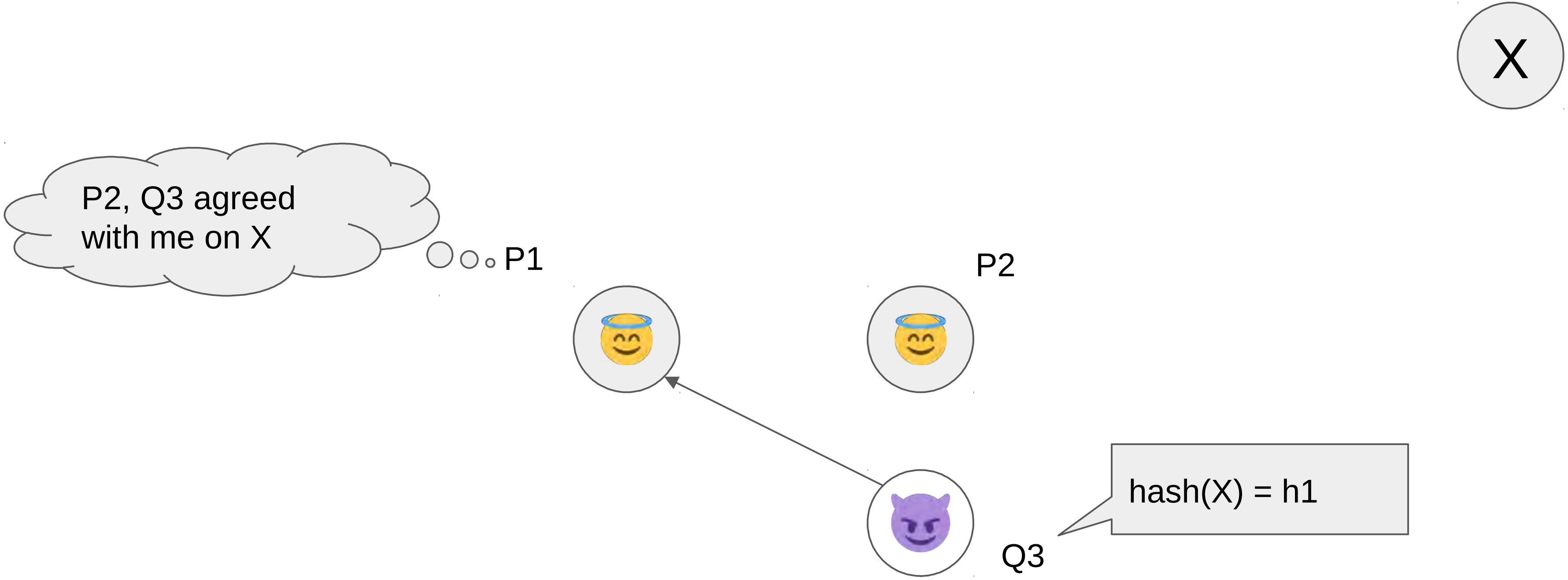
What is hash(X)?

P1

P2

Q3

I don't have X
hash(X) = h1



X

P2, Q3 agreed
with me on X

P1

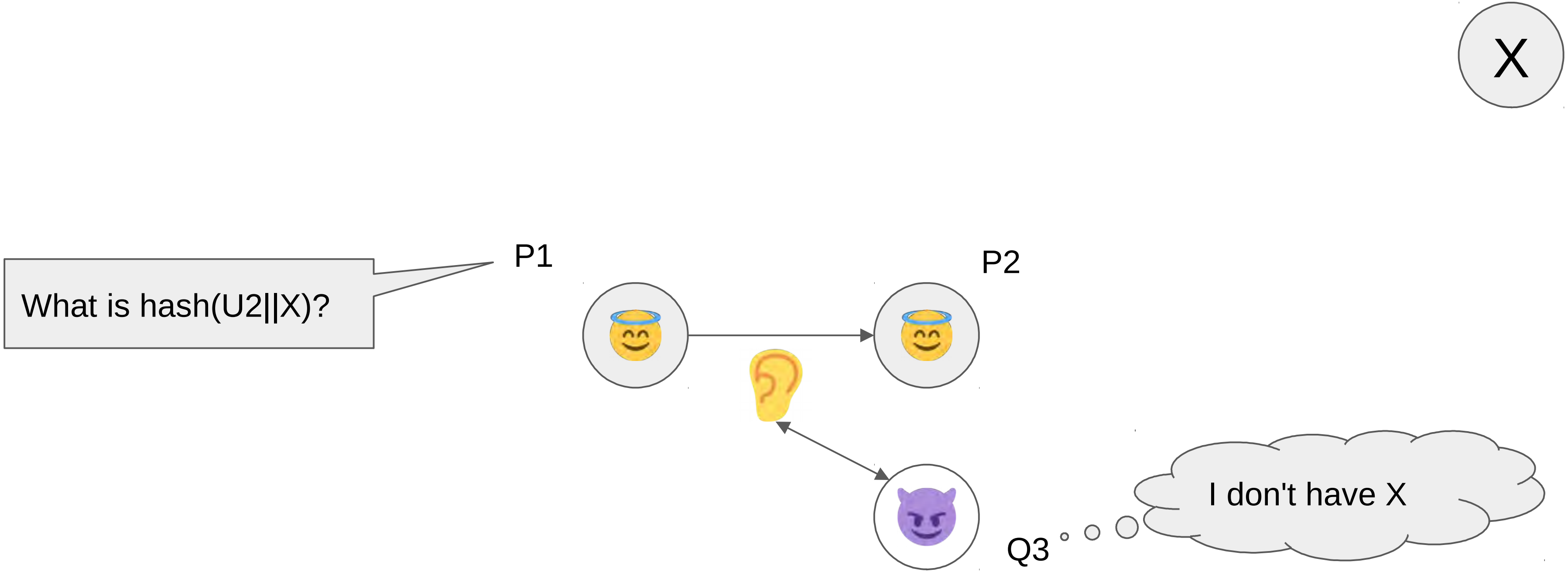
P2

Q3

hash(X) = h1

Poller Nonce

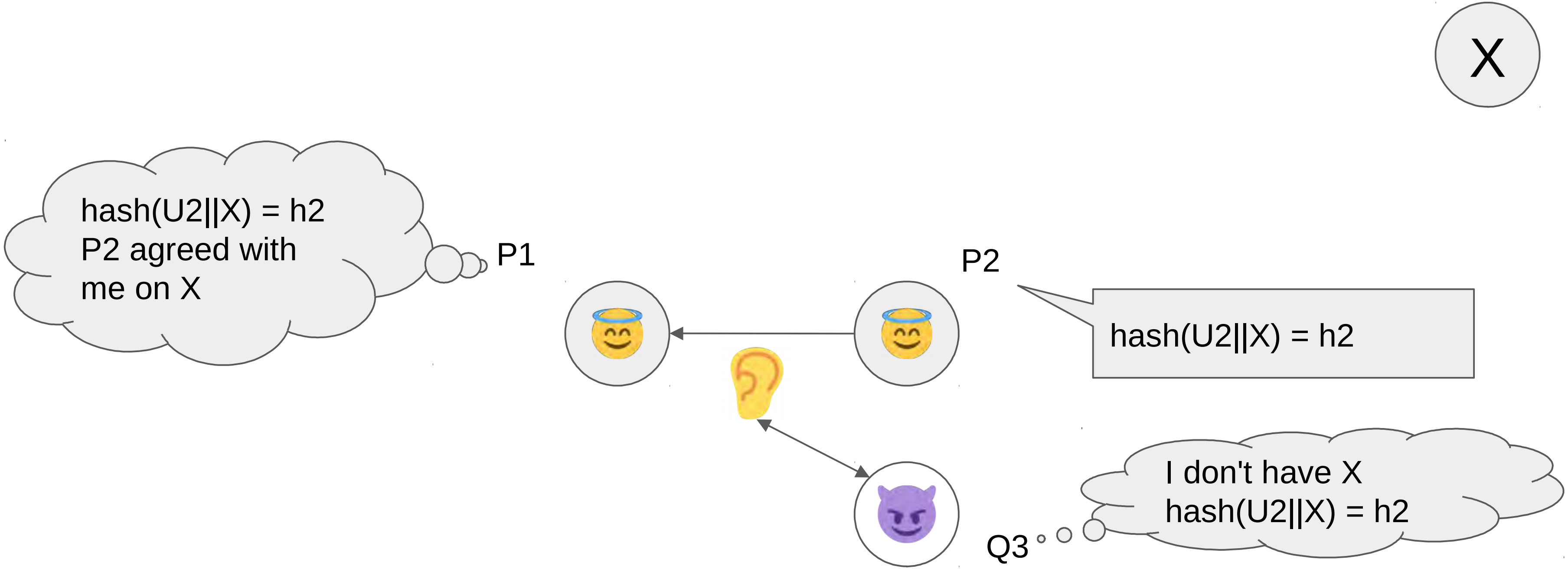
- Nonce: single-use string of random bits
- For each poll over content X, the poller sends a fresh poller nonce U
- Instead of asking for hash(X), the poller asks for hash(U||X)



What is hash(U2||X)?

I don't have X

Attacker Q3 is eavesdropping



X

hash(U2||X) = h2
P2 agreed with
me on X

P1

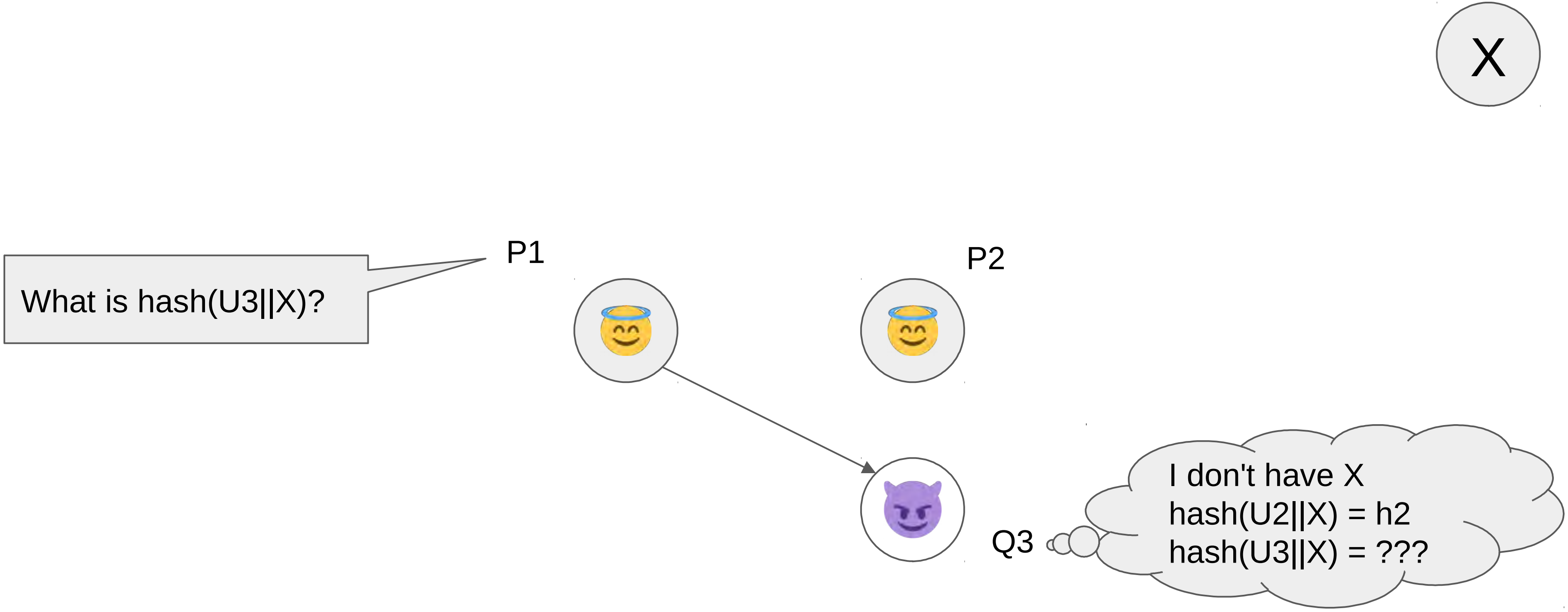
P2

hash(U2||X) = h2

I don't have X
hash(U2||X) = h2

Q3

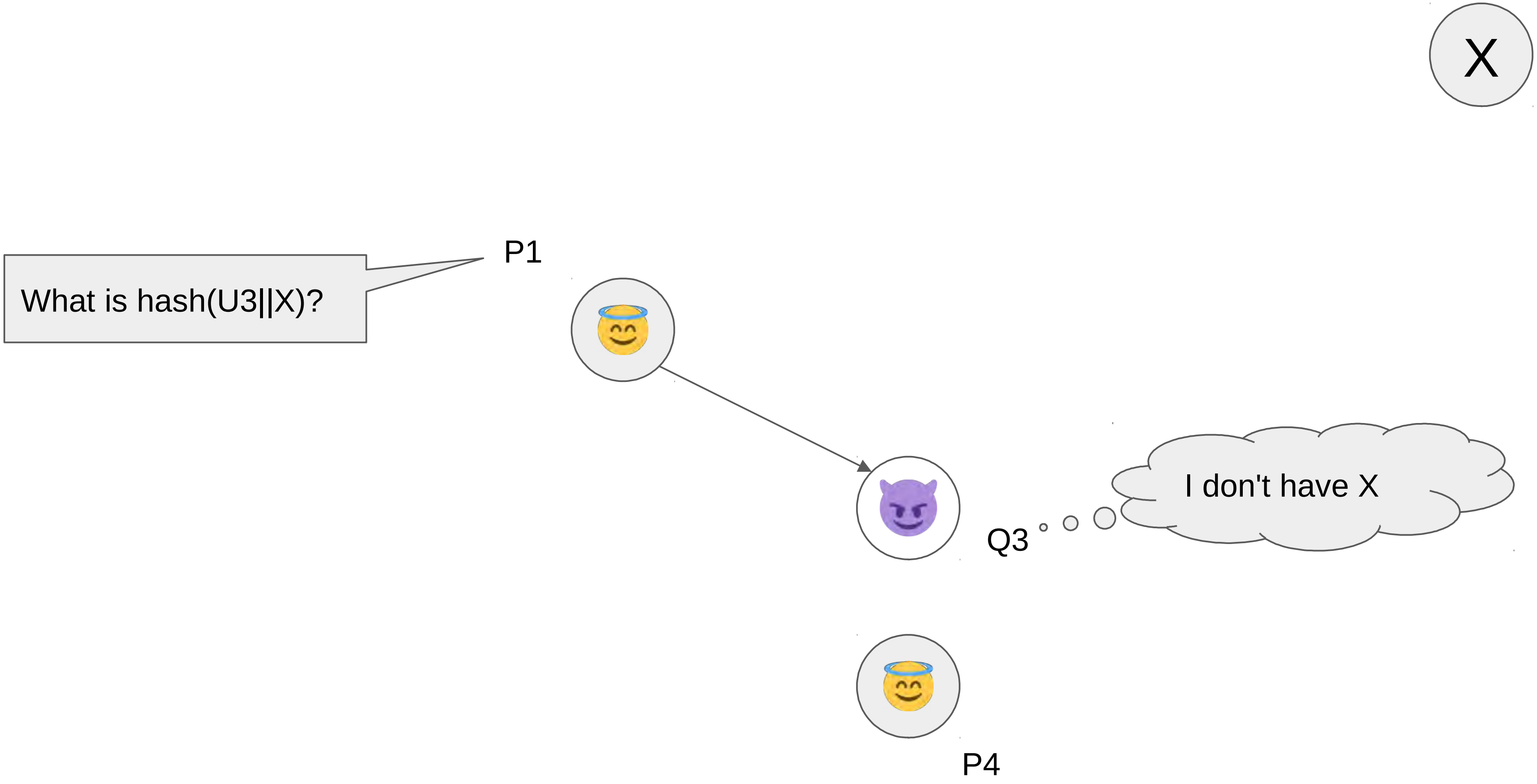
Attacker Q3 is eavesdropping



What is hash(U3||X)?

I don't have X
hash(U2||X) = h2
hash(U3||X) = ???

Peer-in-the-Middle Attack?



X

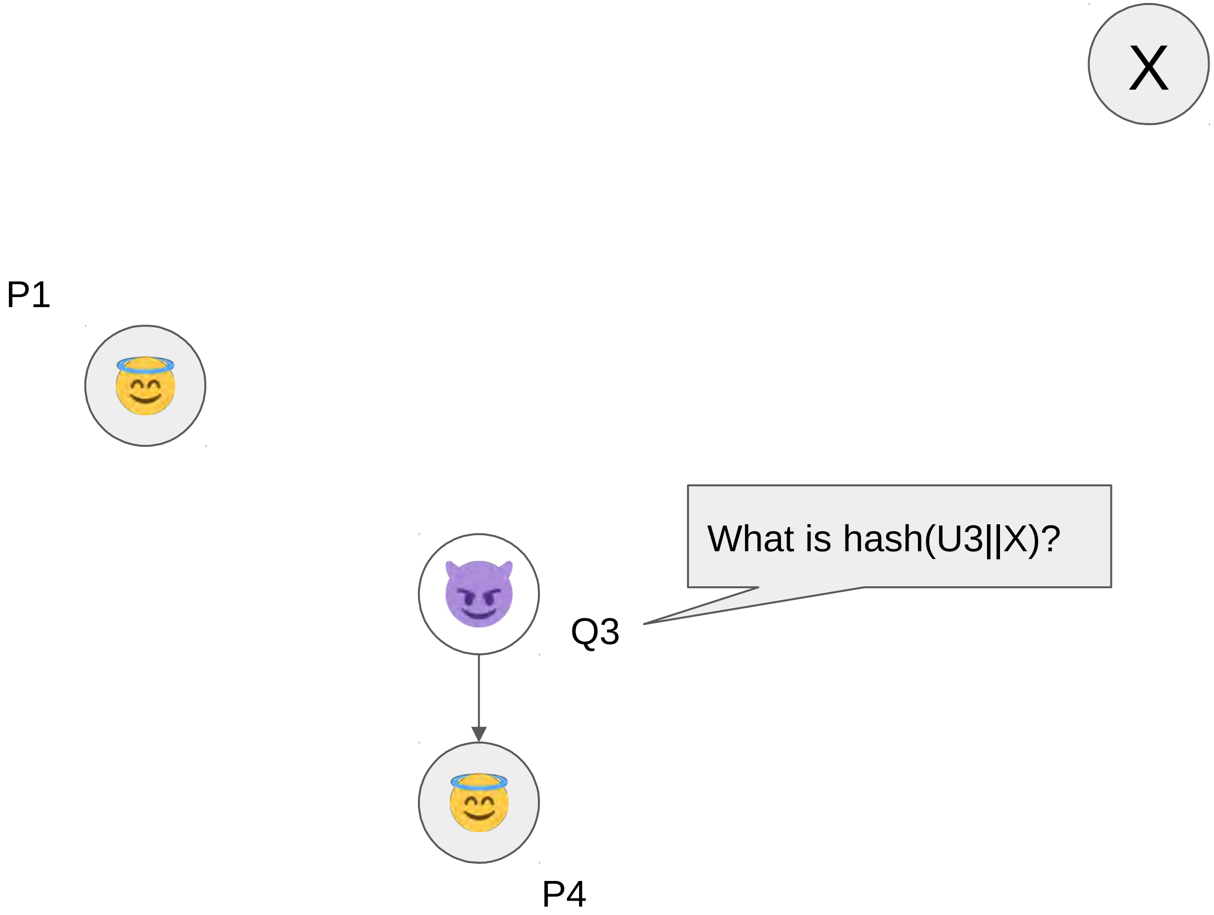
What is hash(U3||X)?

P1

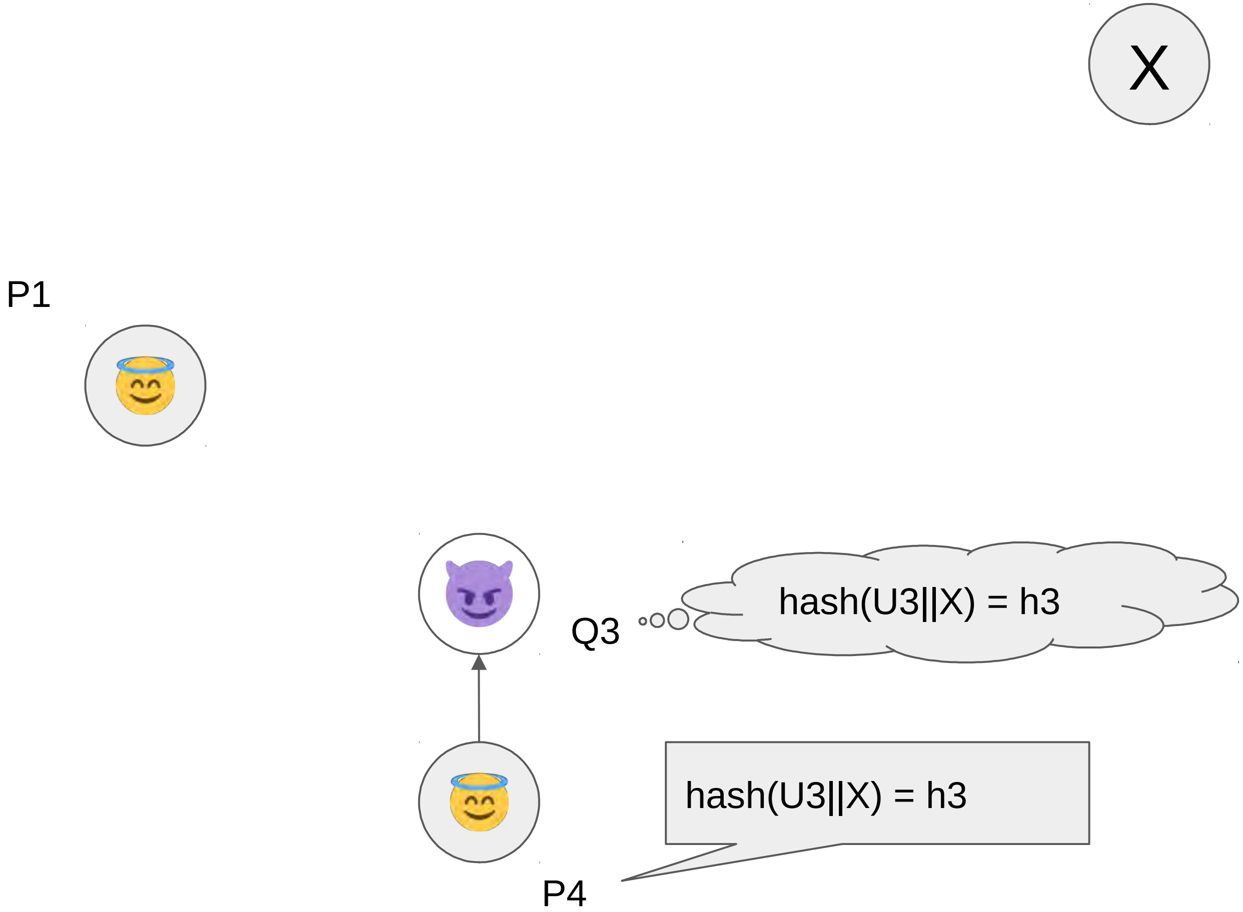
Q3

I don't have X

P4

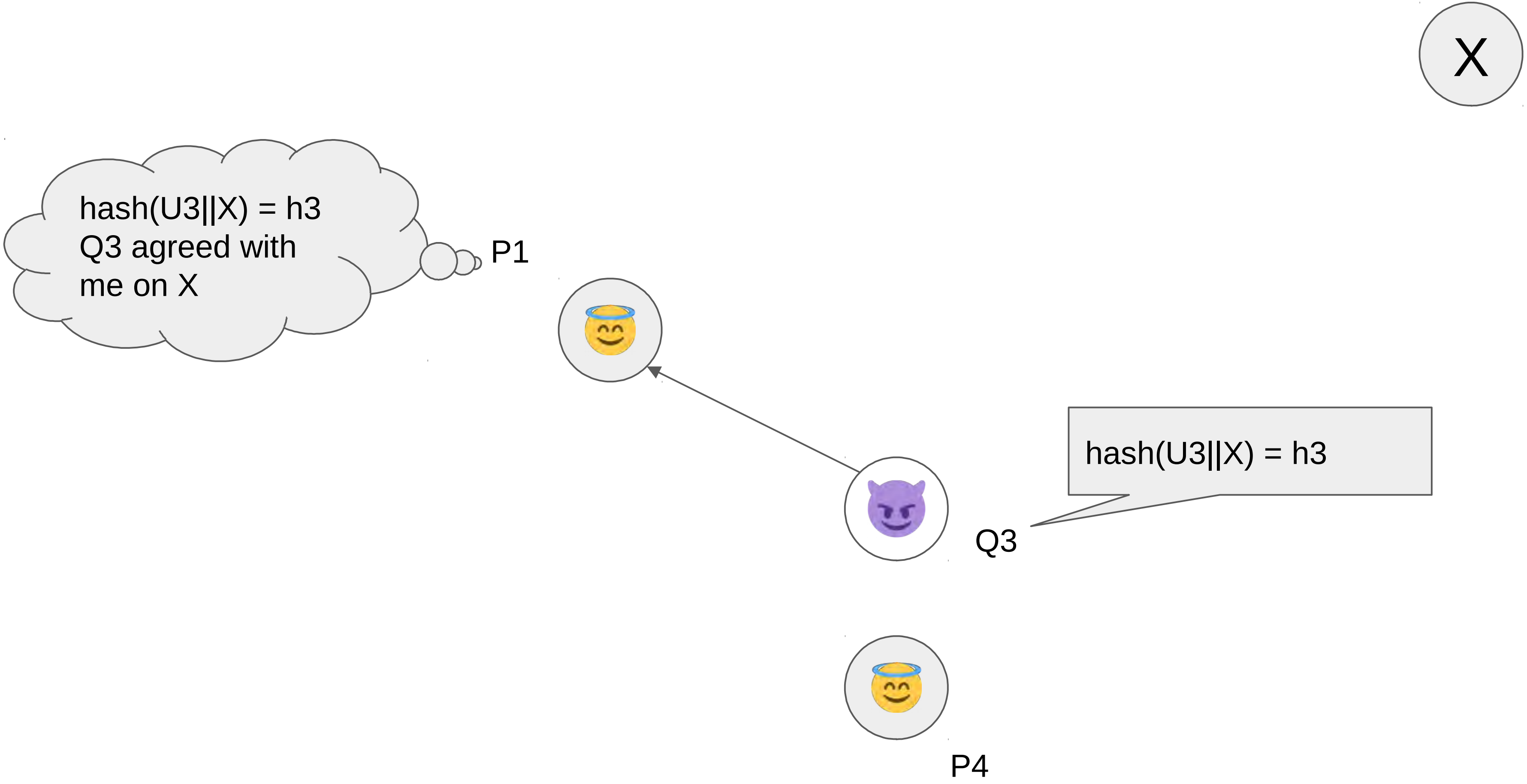


What is hash(U3||X)?



hash(U3||X) = h3

hash(U3||X) = h3

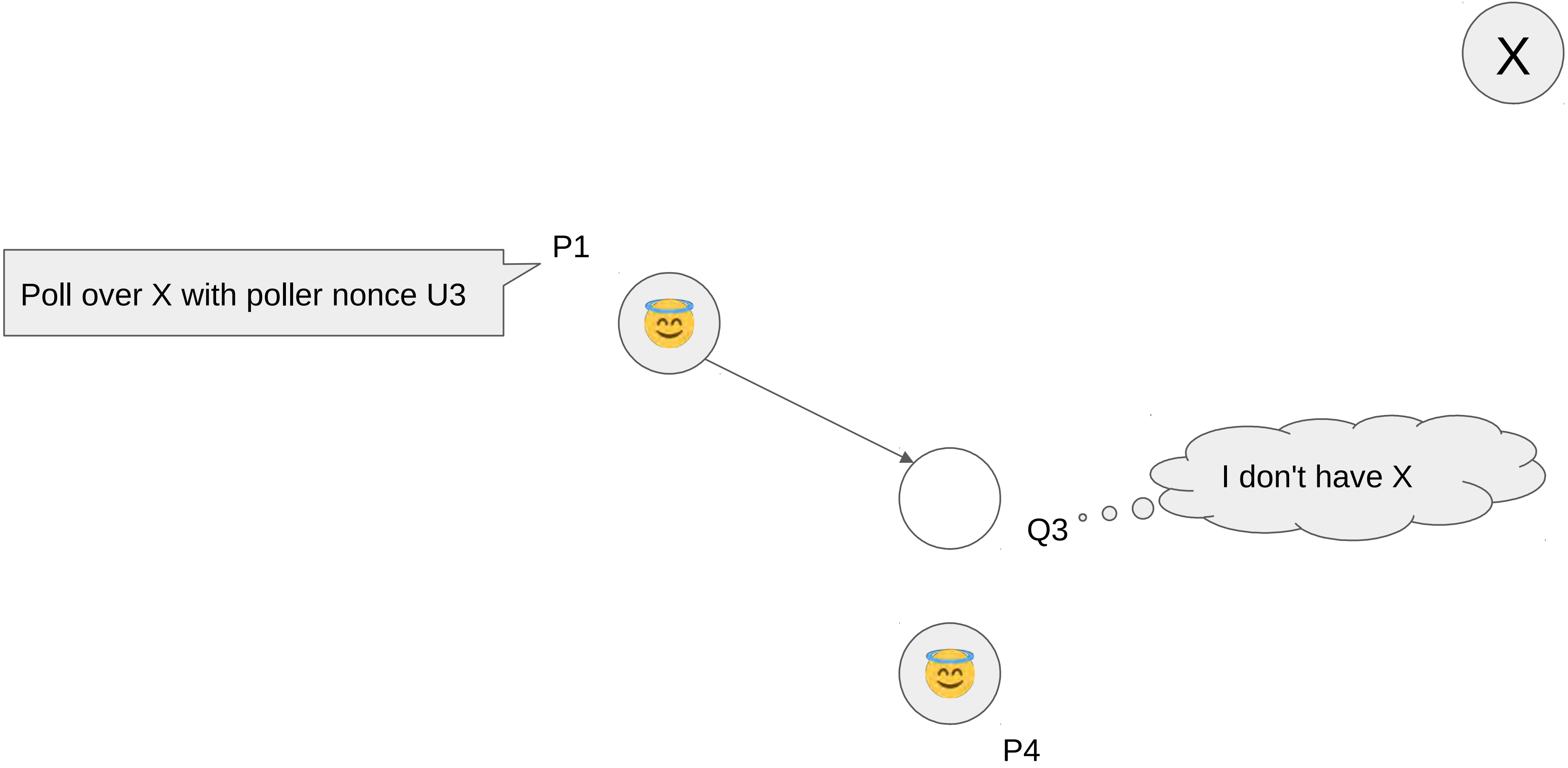


hash(U3||X) = h3
Q3 agreed with
me on X

hash(U3||X) = h3

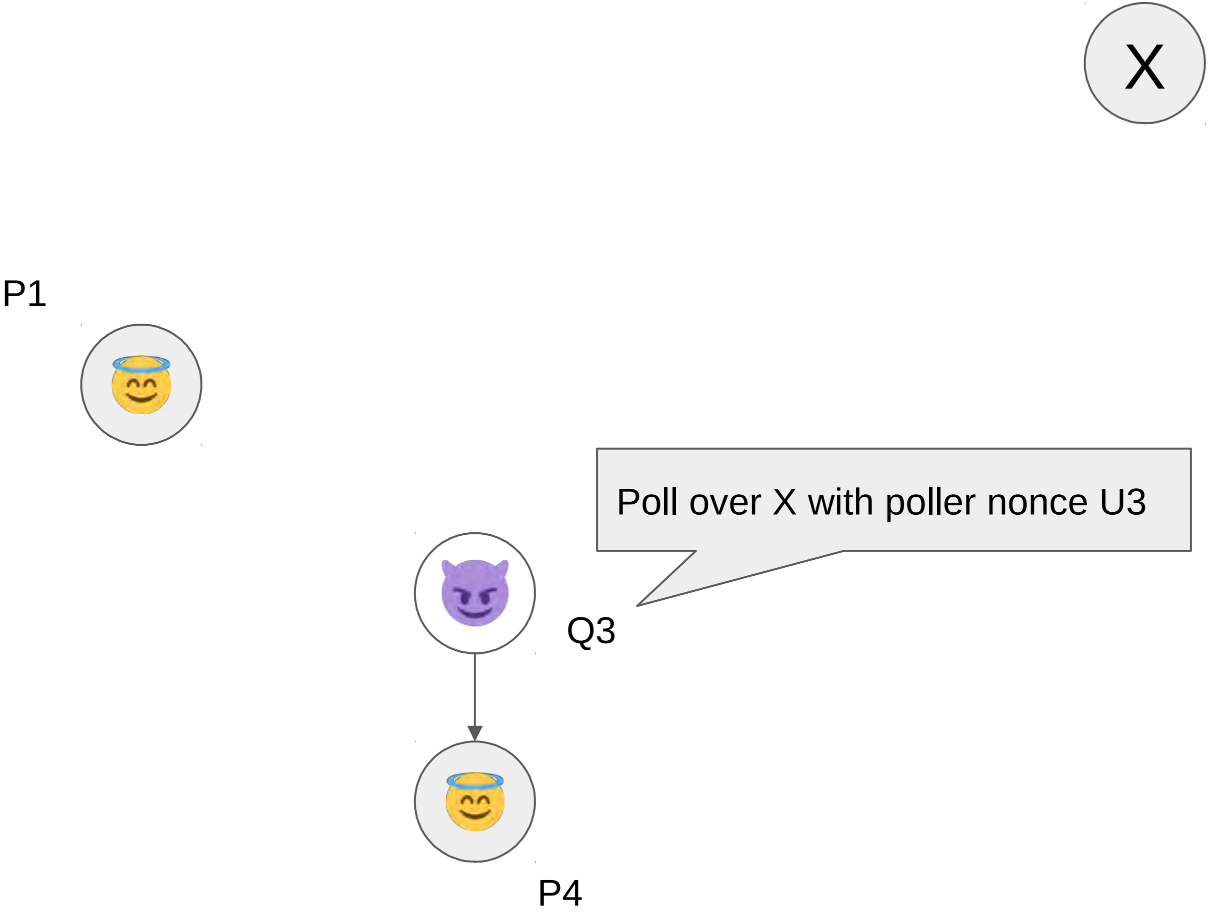
Voter Nonce

- For each poll request over content X with poller nonce U, the voter sends a fresh voter nonce V
- Does it help mitigate peer-in-the-middle attacks?

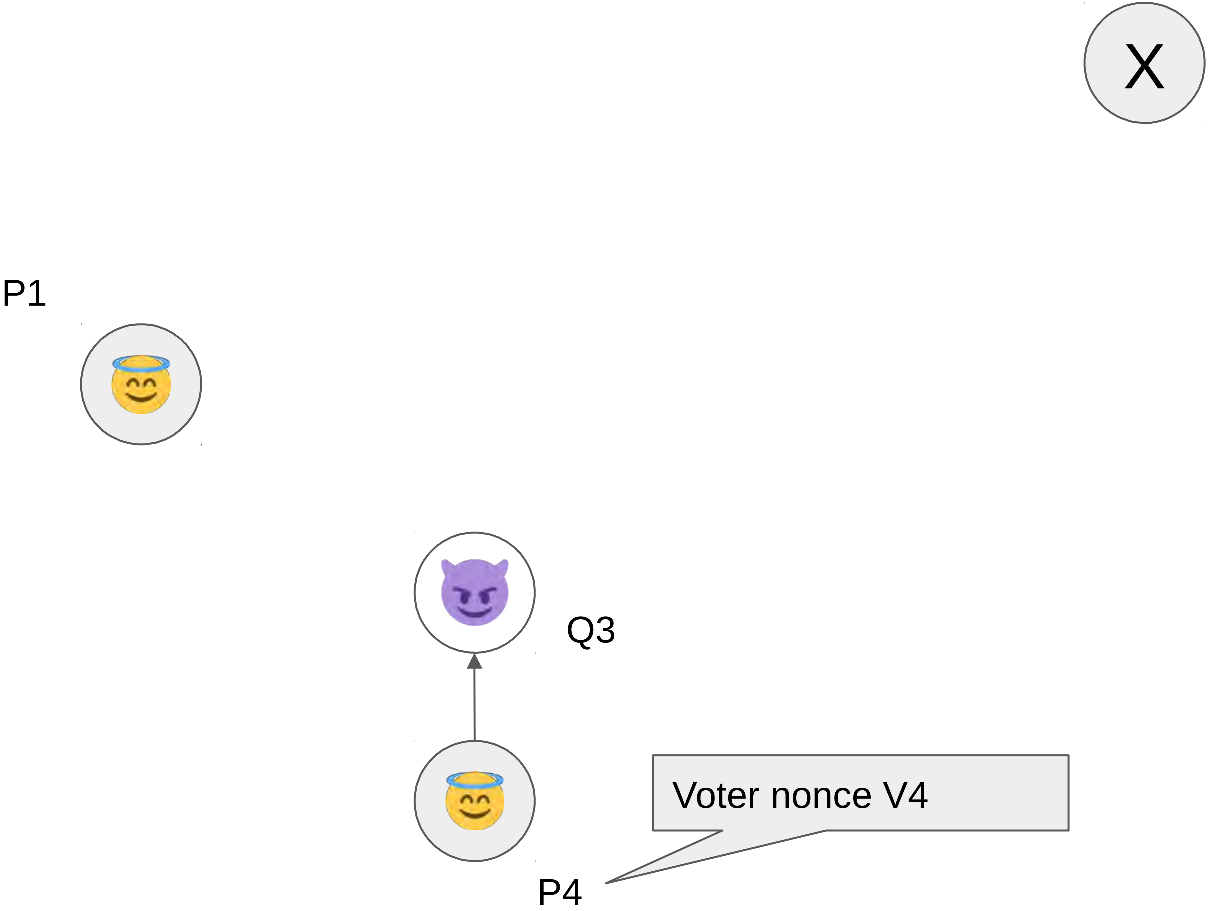


Poll over X with poller nonce U3

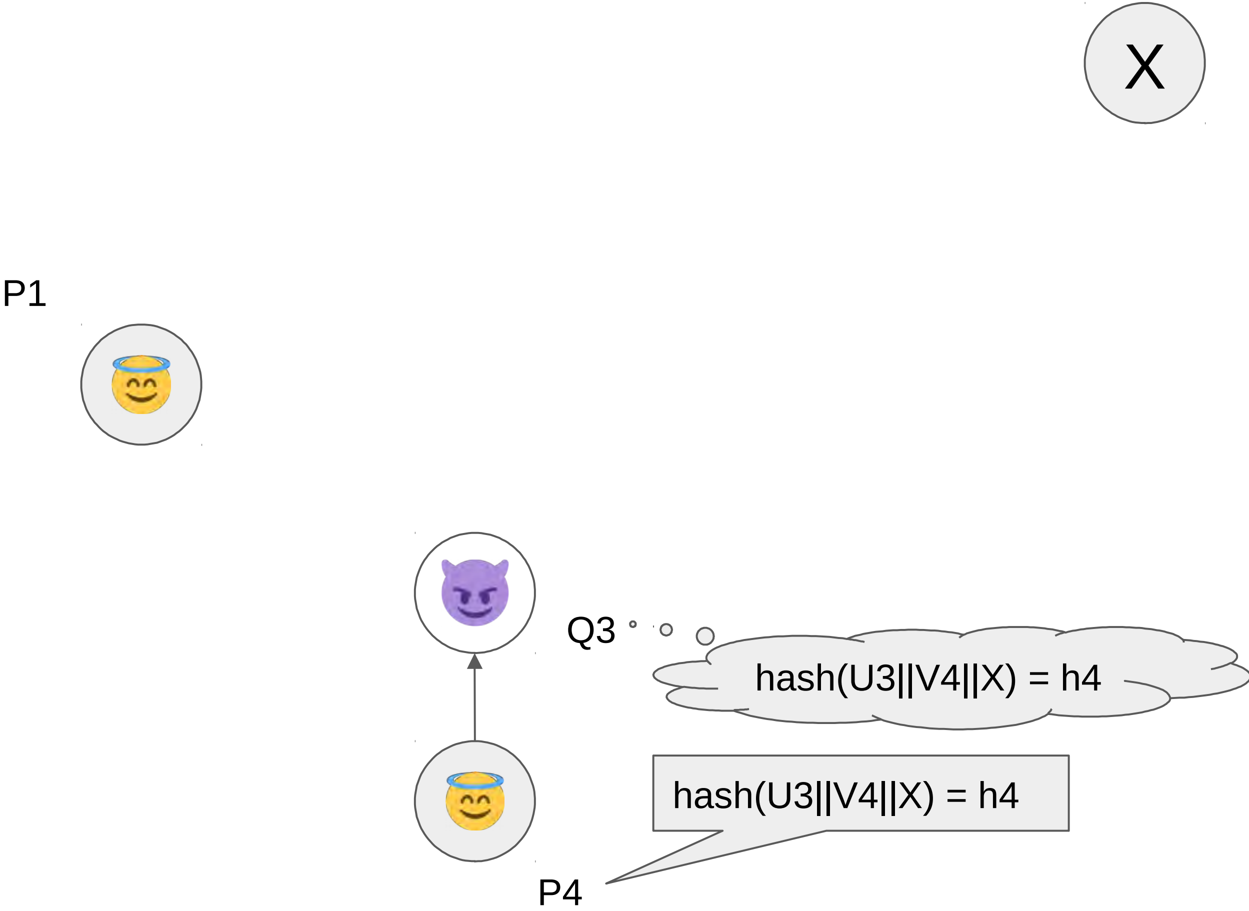
I don't have X



Poll over X with poller nonce U3



Voter nonce V4



X

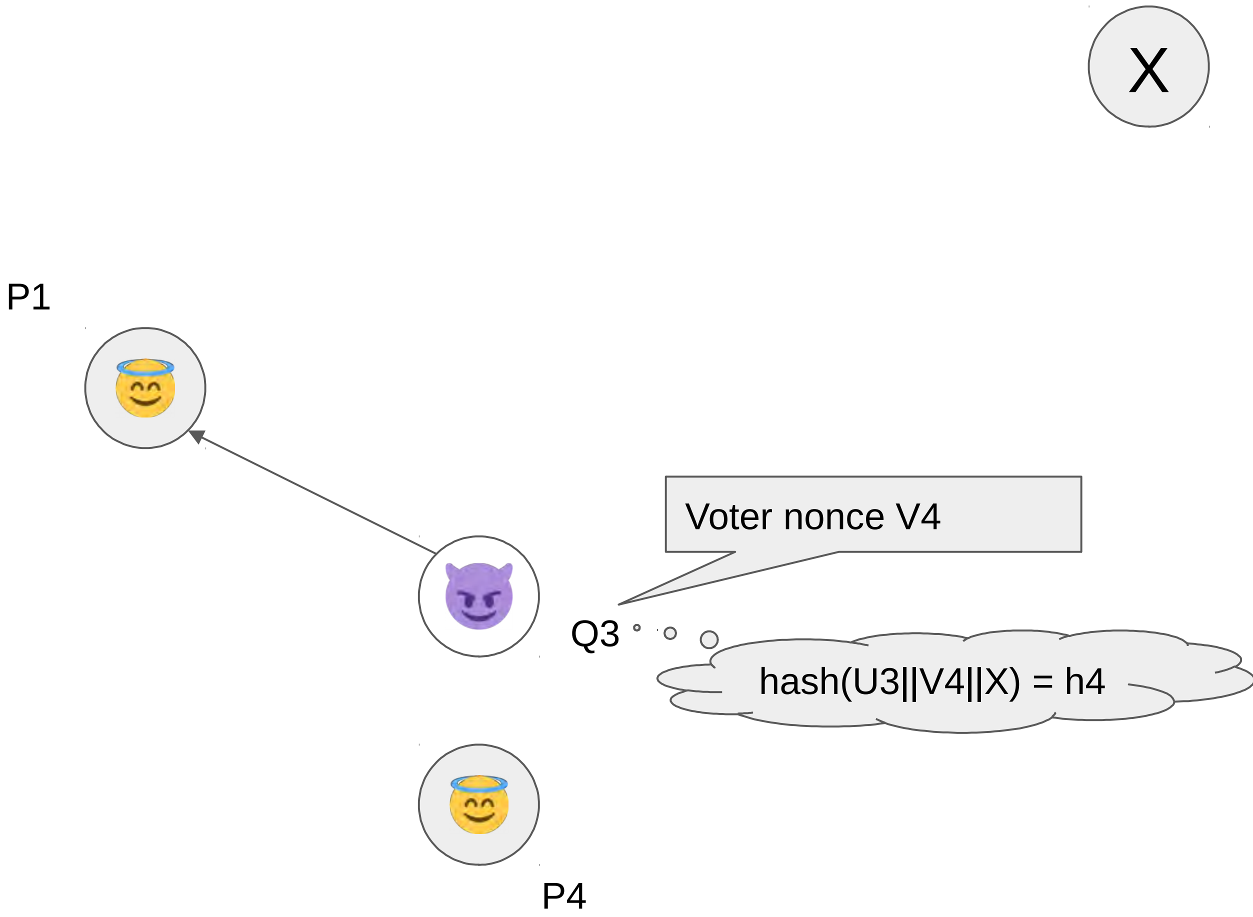
P1

Q3

hash(U3||V4||X) = h4

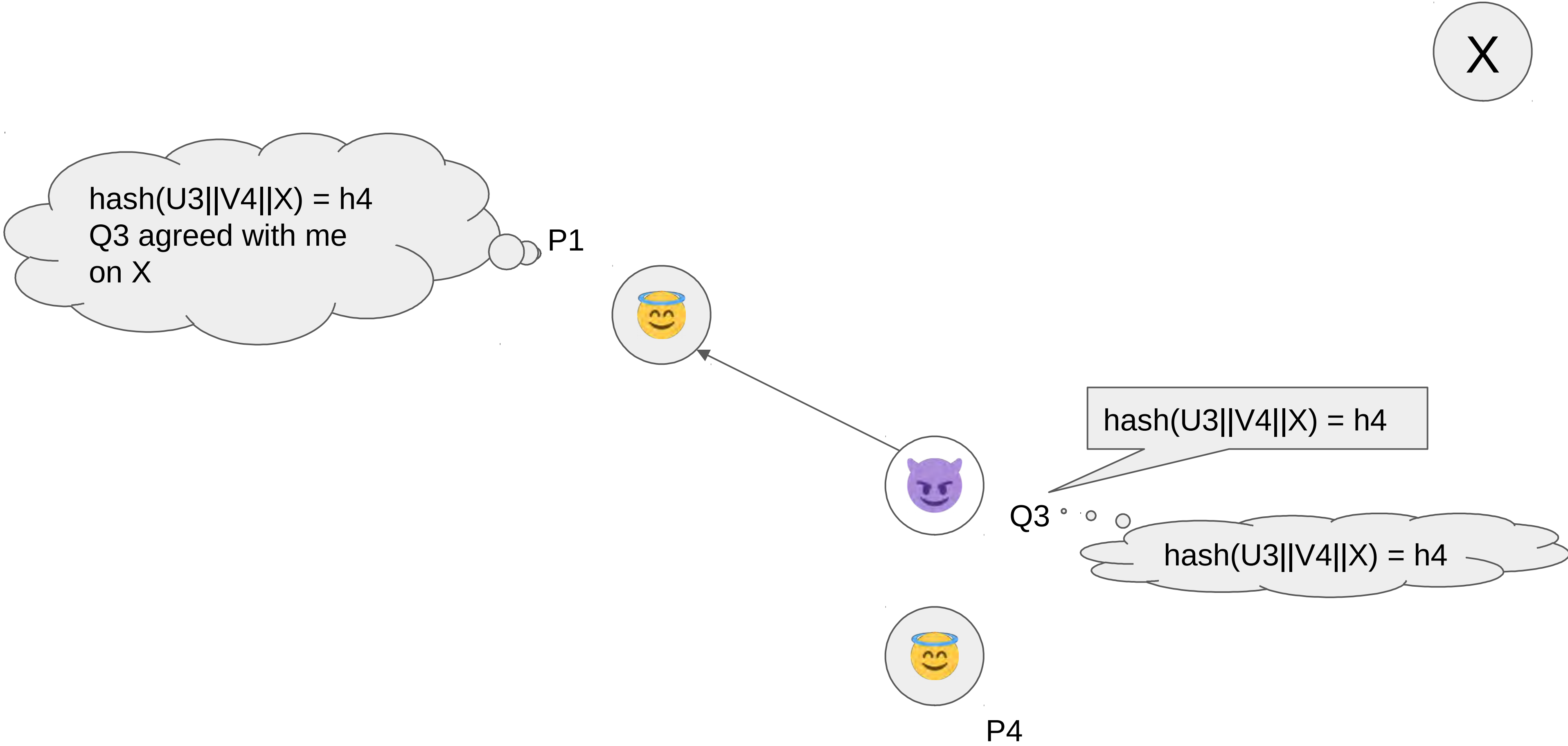
P4

hash(U3||V4||X) = h4



Voter nonce V4

hash(U3||V4||X) = h4



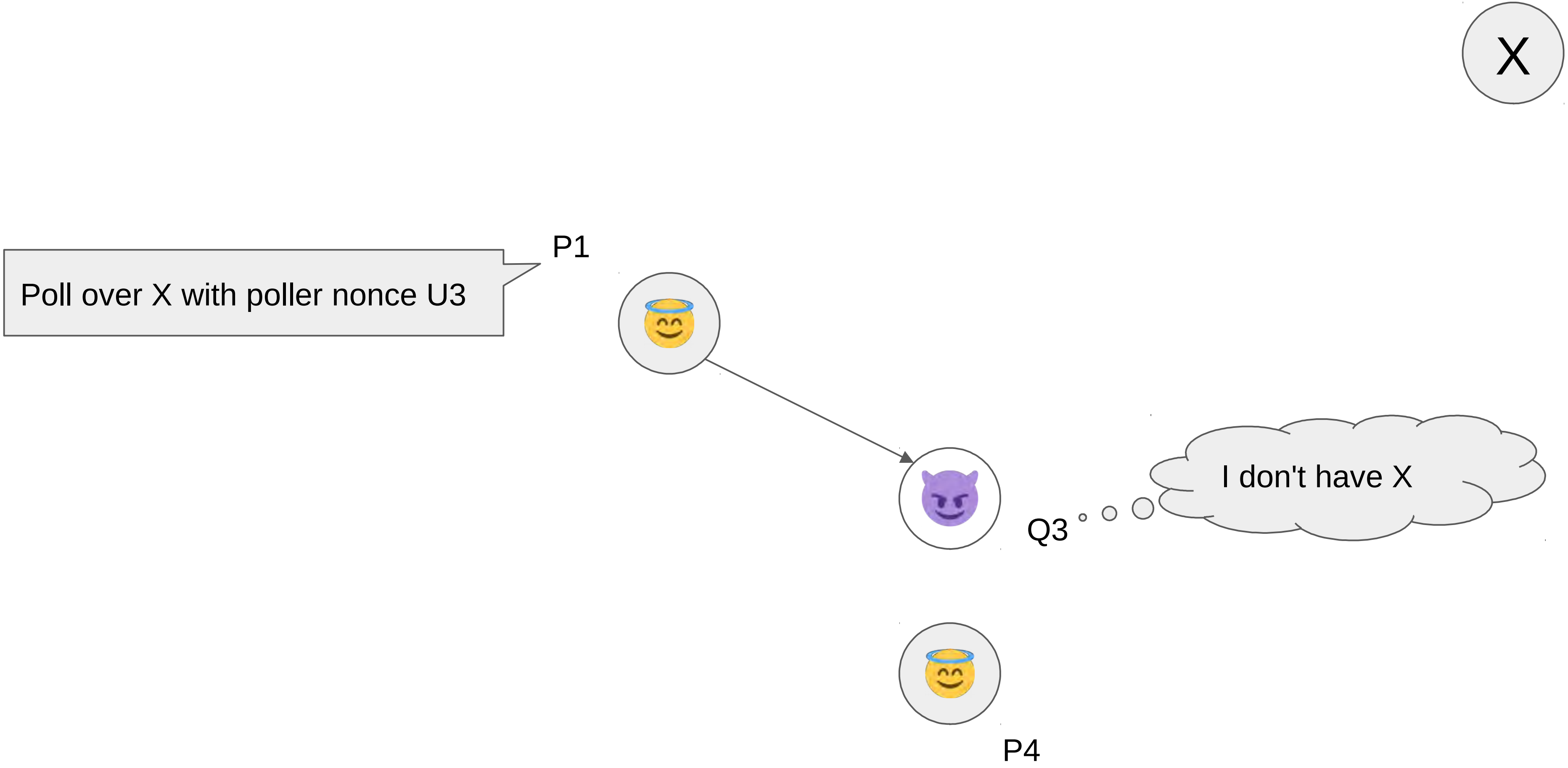
hash(U3||V4||X) = h4
Q3 agreed with me
on X

hash(U3||V4||X) = h4

hash(U3||V4||X) = h4

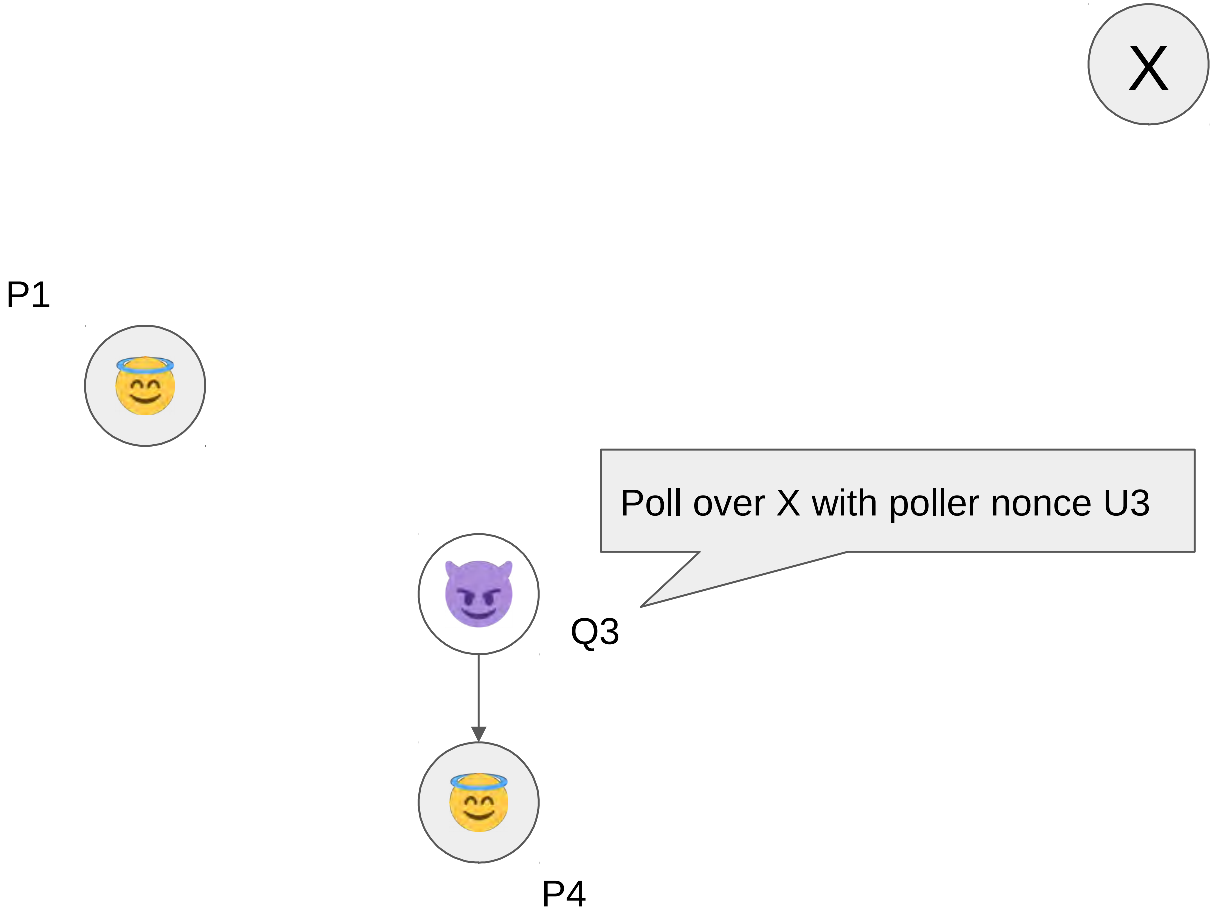
Poll Effort Verification

- Before providing hash(U||V||X), the voter challenges the poller to a computation involving the content X, the poller nonce U and the voter nonce V

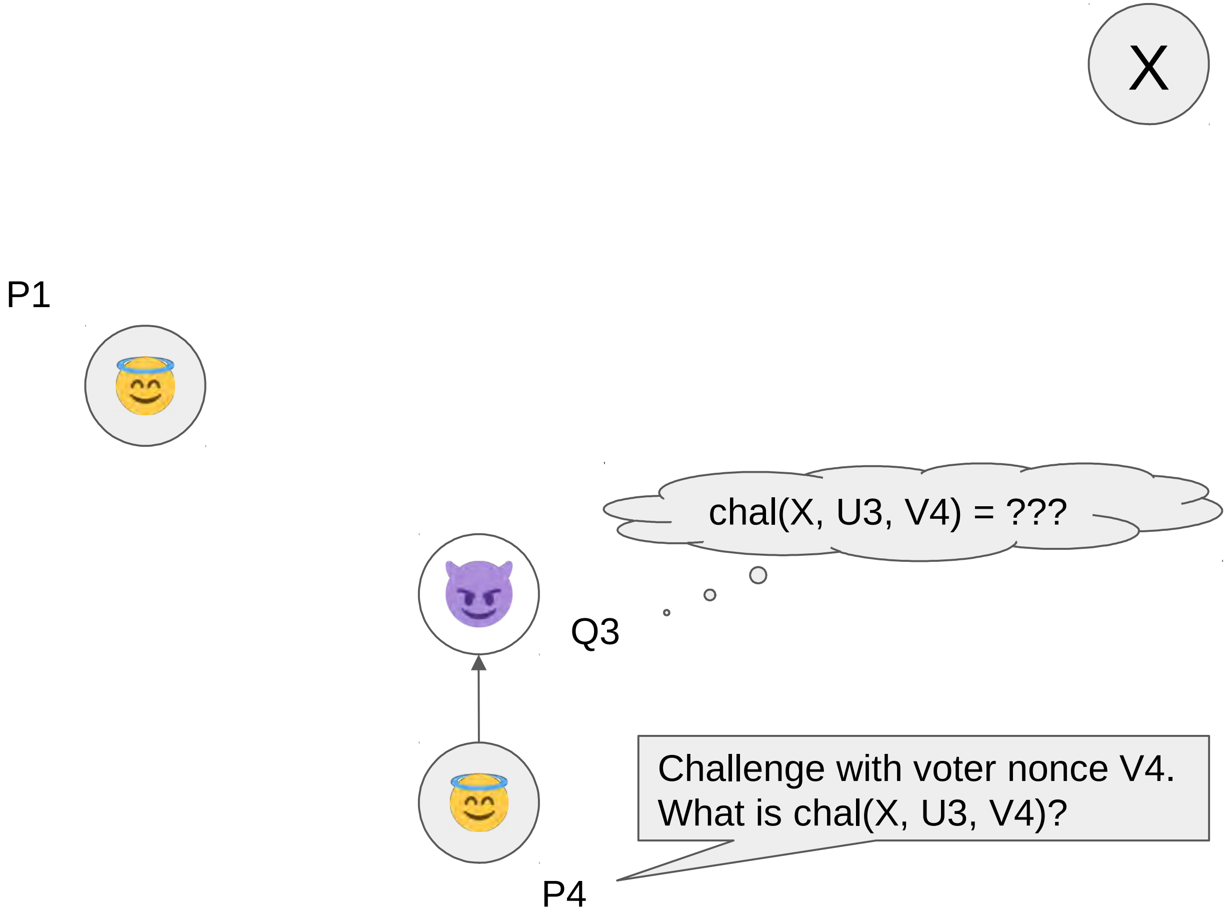


Poll over X with poller nonce U3

I don't have X

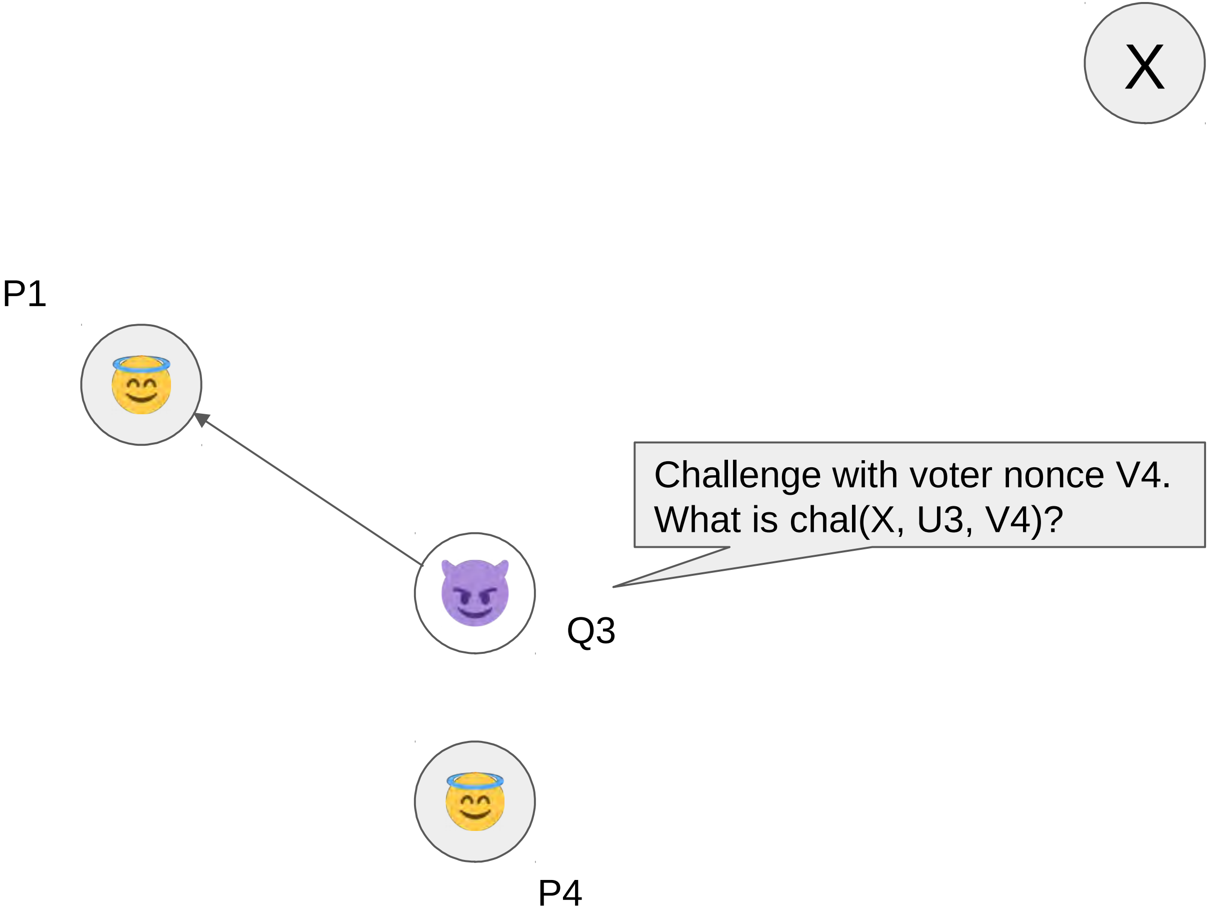


Poll over X with poller nonce U3

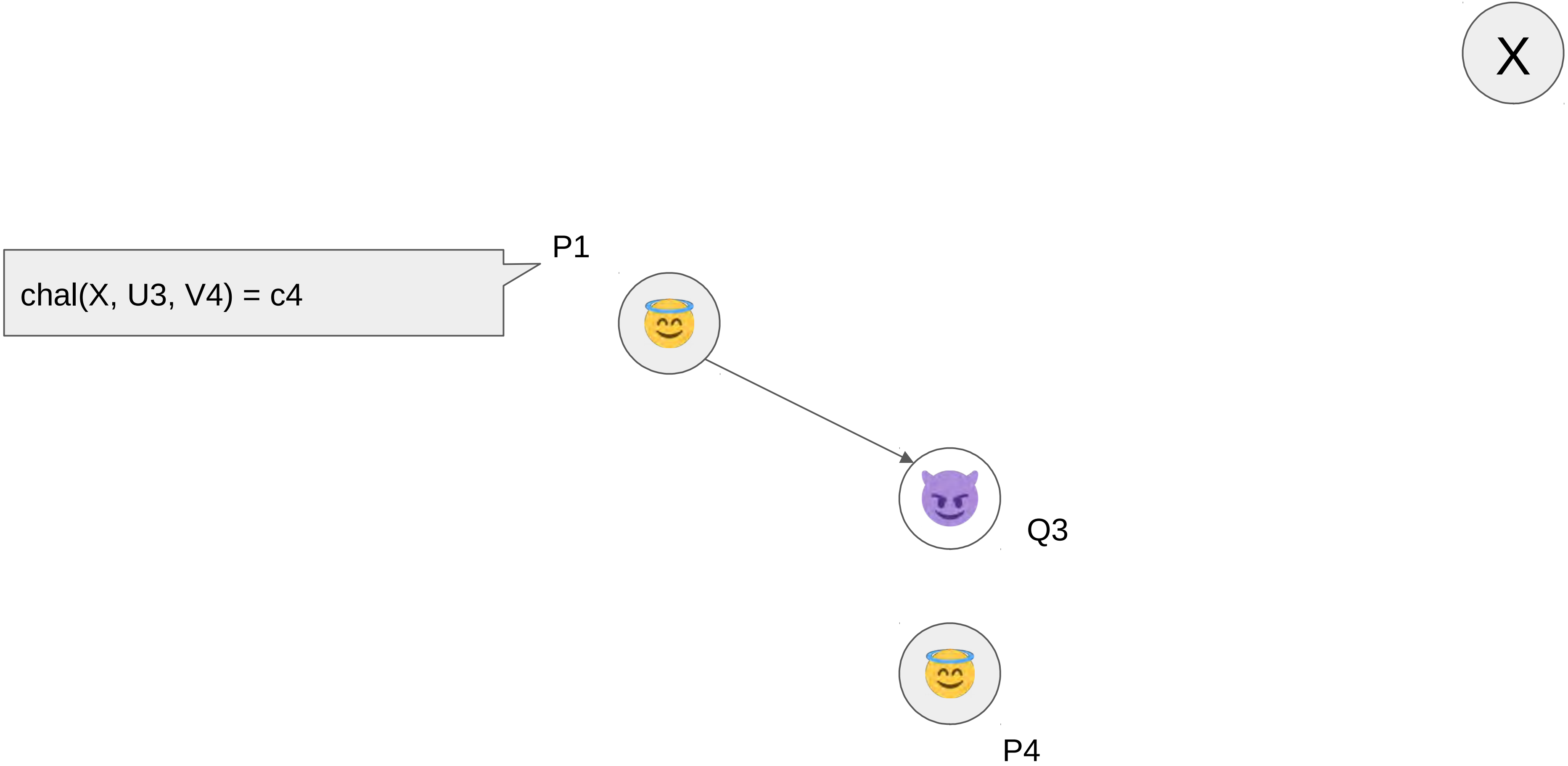


chal(X, U3, V4) = ???

Challenge with voter nonce V4.
What is chal(X, U3, V4)?



Challenge with voter nonce V4. What is chal(X, U3, V4)?



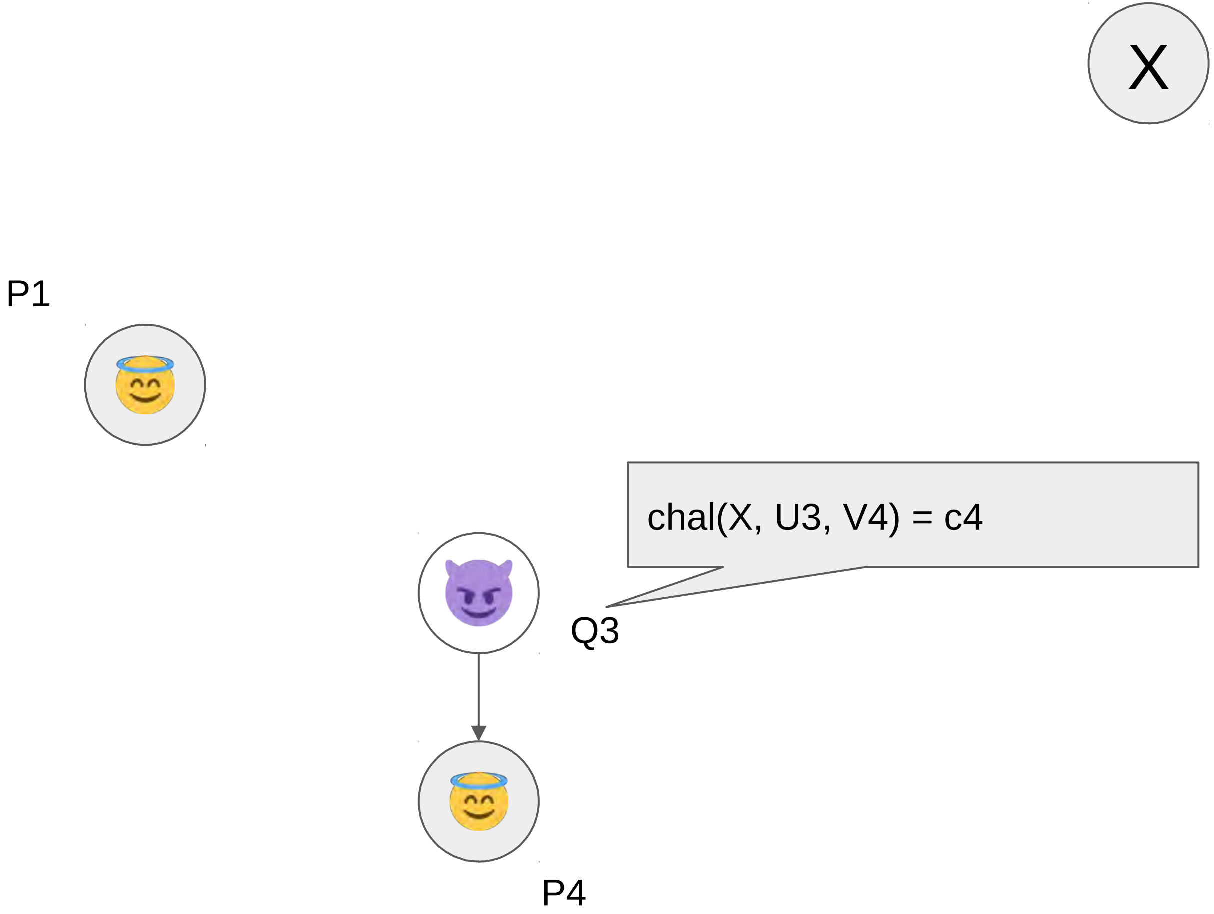
chal(X, U3, V4) = c4

X

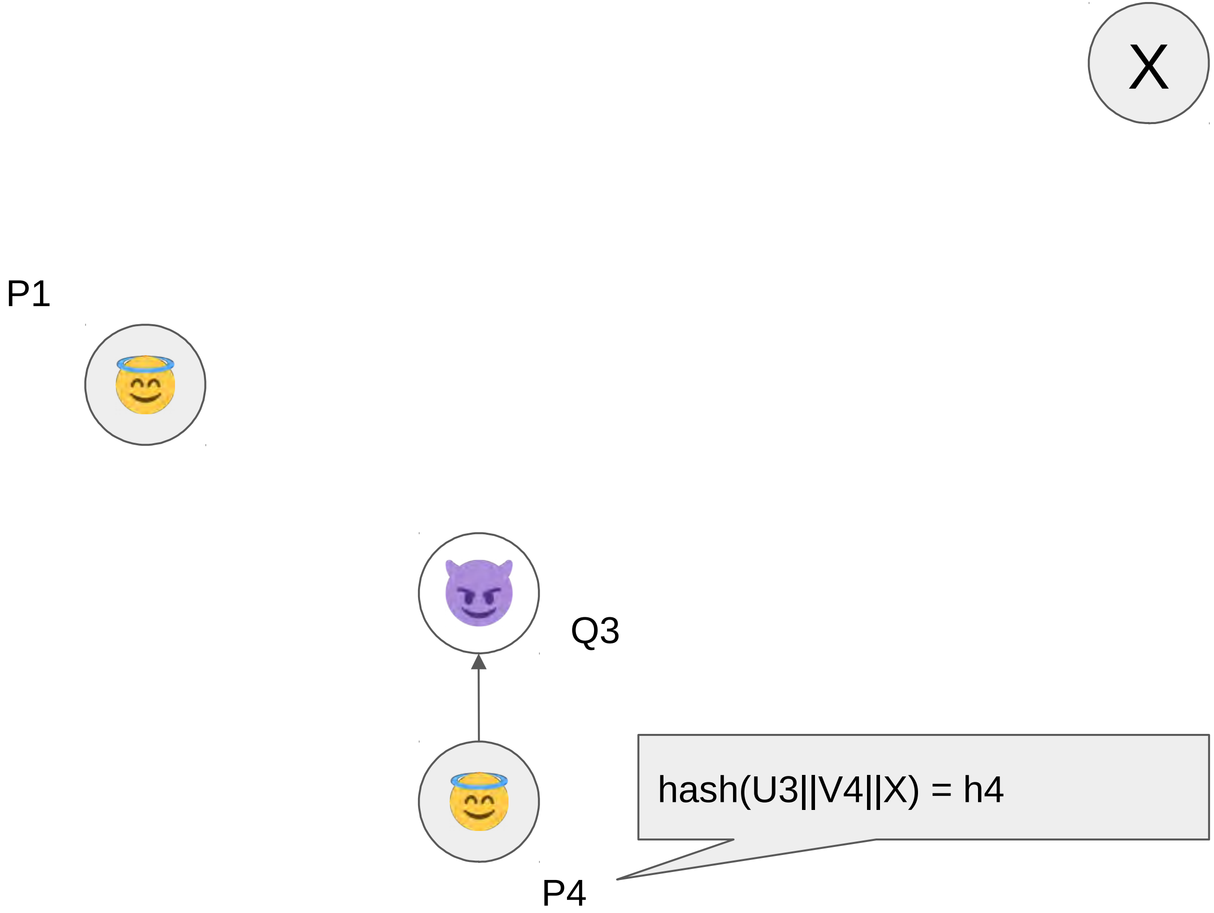
P1

Q3

P4



chal(X, U3, V4) = c4



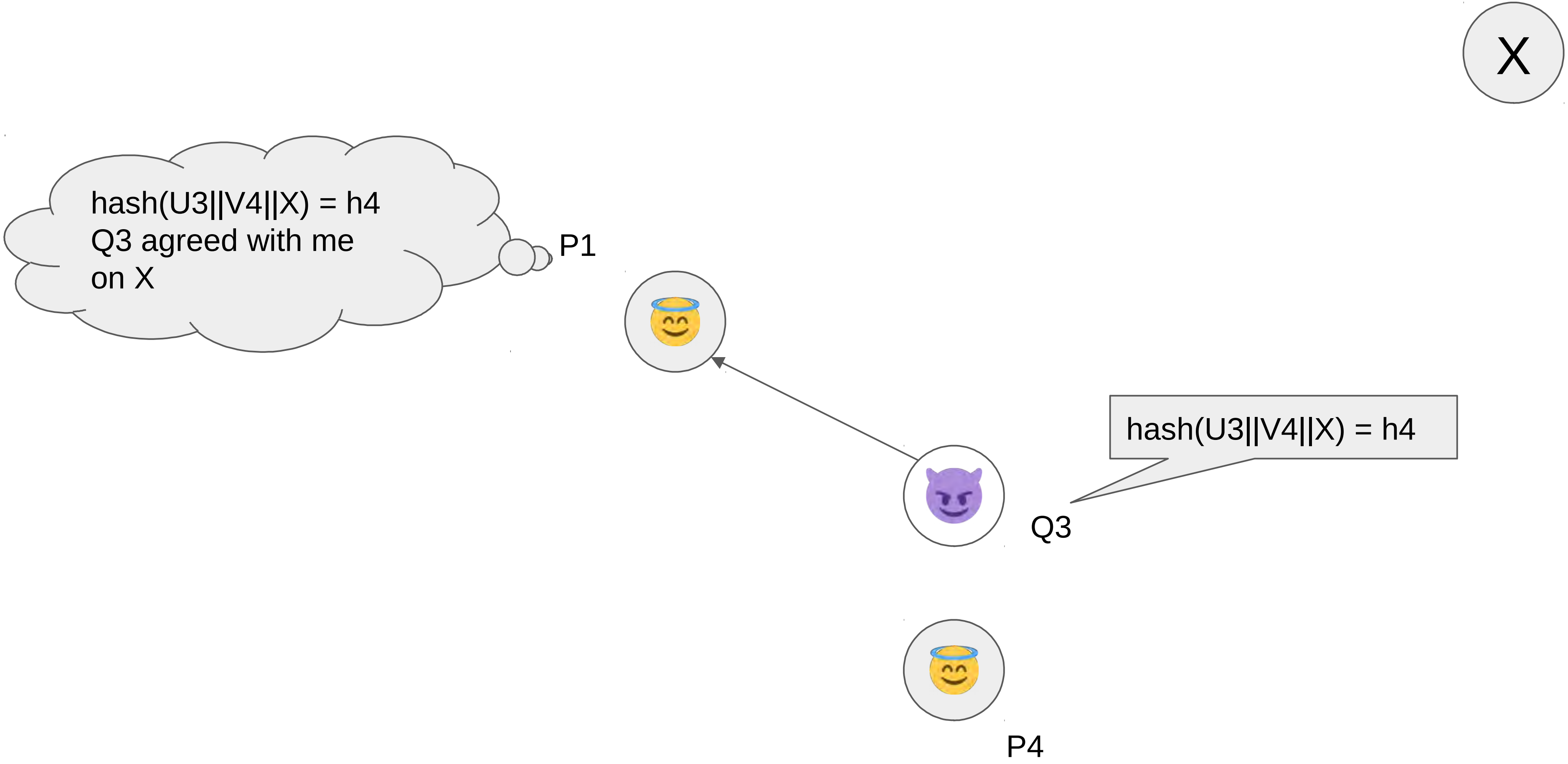
X

P1

Q3

P4

hash(U3||V4||X) = h4



hash(U3||V4||X) = h4
Q3 agreed with me
on X

P1

hash(U3||V4||X) = h4

Q3

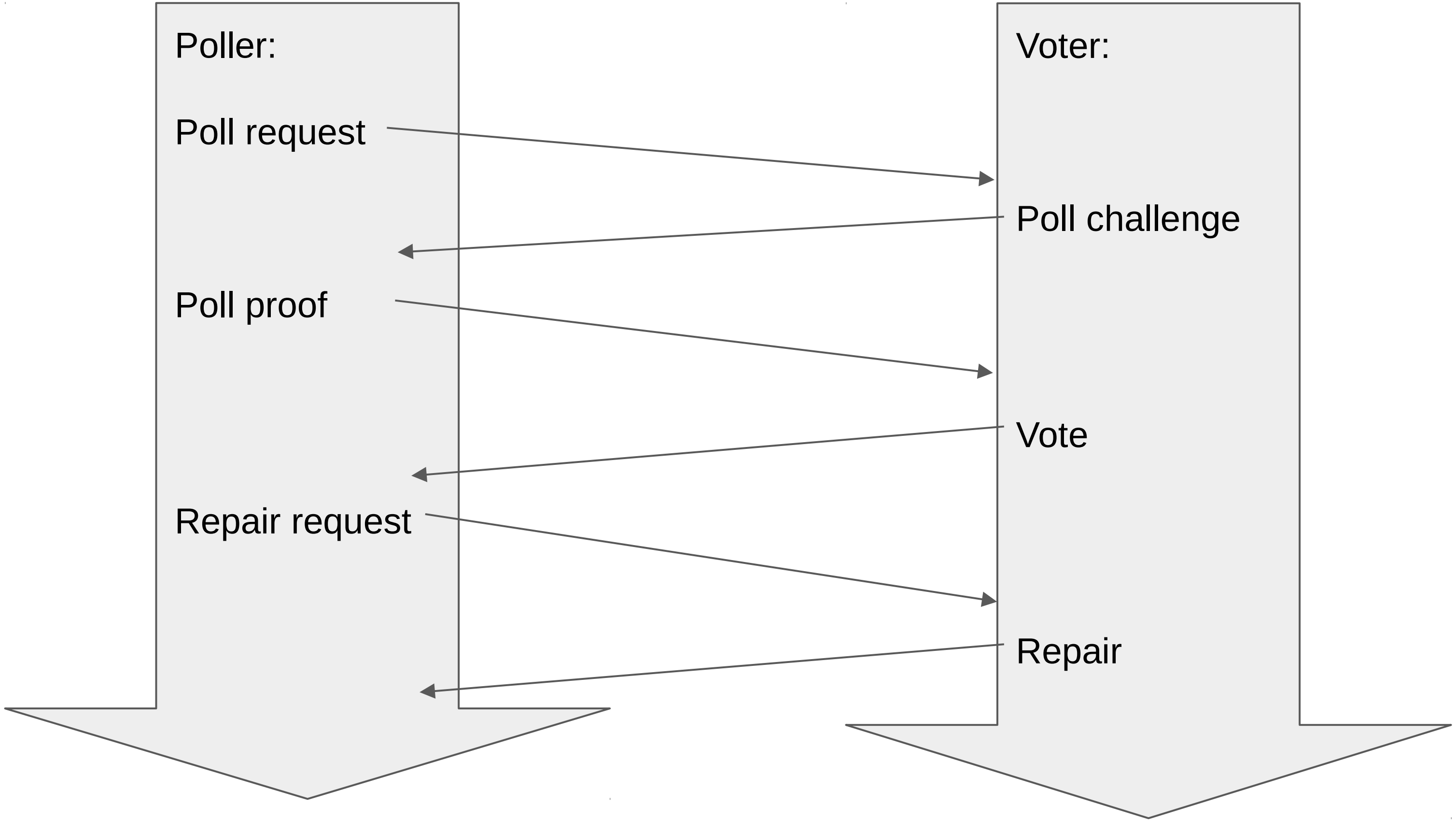
P4

X

Has the attacker gained anything?

- Malign peer Q3 led loyal peer P1 to think (for now) that they have a good copy of X
- Number of good copies of X in the system has not changed
- Isolated success in one poll sitting between two loyal peers will not survive repeated attempts over time due to randomization

Diagram



Poller:

Poll request

Poll proof

Repair request

Voter:

Poll challenge

Vote

Repair

Physical Fixity vs. Logical Fixity

- Roots of the LOCKSS Program in Web Preservation
- Domain-specific knowledge in LOCKSS plugins
- Normalize byte streams before hashing
- Paradox: preservation of replicas even when none are identical

LCAP In Depth

Peer Discovery

- Network with open participation
- List of peers currently under consideration ("reference list") bootstrapped with list of initially trusted peers ("friends list")
- Two rounds of poll invitations: "inner circle" and "outer circle"
    - Poller invites peers selected randomly from reference list: "inner circle"
    - When voter verifies poll proof from poller, voter sends nominations of other peers to poller
    - Poller invites previously unknown peers selected randomly from nominations: "outer circle"
    - Only inner circle votes influence poll results; outer circle votes help identify agreeing peers

Timeliness and Rate Limiting

- Only proof of recent effort can affect system decisions
- Peers must continually be sustained by minimum effort expenditure
- Adversary can damage loyal peer only when that peer calls a poll
- Attack progress limited by smaller of adversary and victims' efforts

Reference List Churning

- Increase difficulty and reduce predictability of attacker effort to populate loyal peer's reference list with malign peers
- Churning after poll conclusion:
  - Remove disagreeing inner circle peers
  - Remove randomly selected agreeing inner circle peers
  - Insert agreeing outer circle peers
  - Insert randomly selected peers from friends list

Symmetric Polls

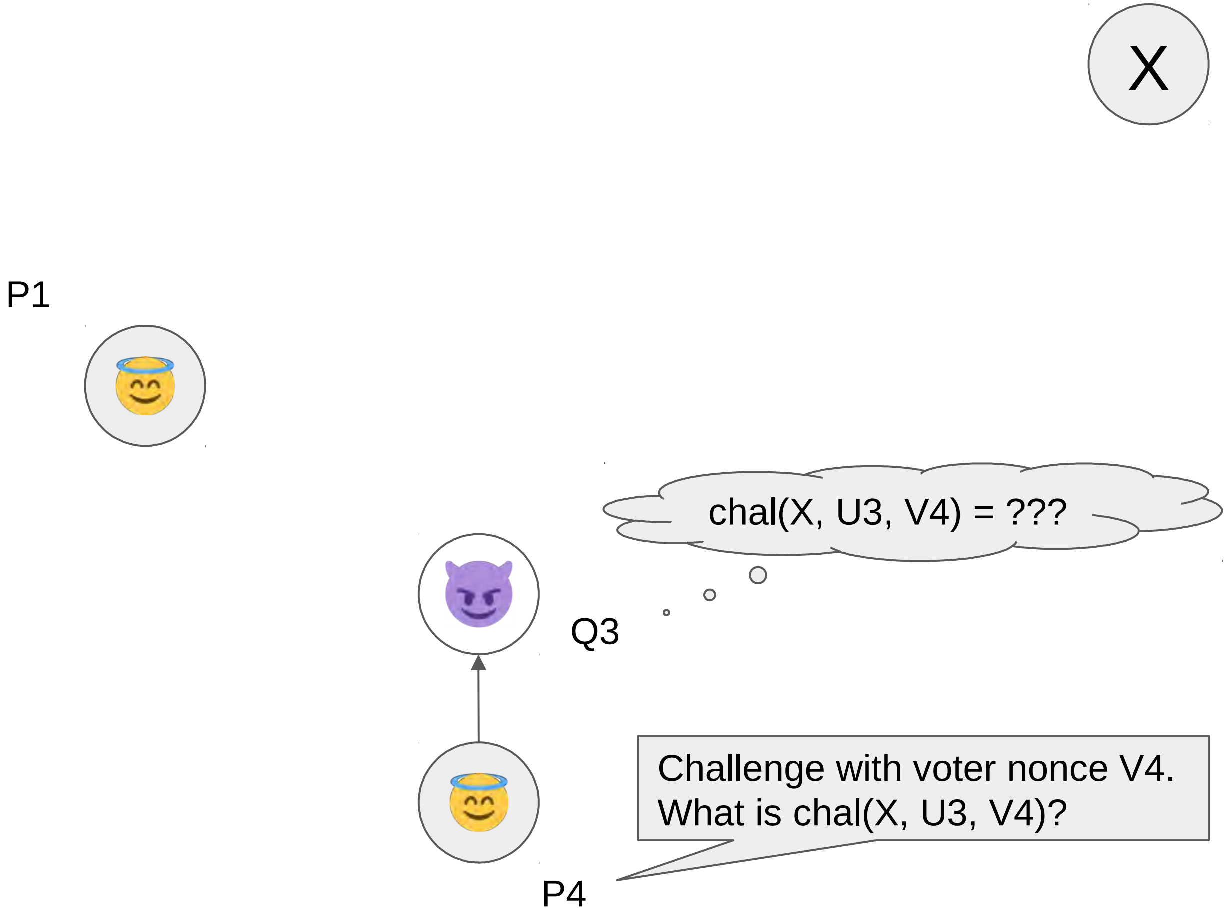
- In asymmetric protocol:
  - Poller generates poller nonce U
  - Voter generates voter nonce V
  - Voter computes hash(U||V||X): poll from poller to voter predicated on U and V
- In symmetric protocol:
  - Poller generates poller nonce U
  - Voter generates voter nonce V and secondary voter nonce W
  - Voter computes hash(U||V||X): poll from poller to voter predicated on U and V
  - While computing hash(U||V||X), poller computes hash(U||W||X): poll from voter to poller predicated on U and W
- Performance trade-off

Proof of Retrievability vs. Proof of Possession

- PoR over entirety of content: guarantee that prover had access to complete, intact copy of file
- PoP over sample of content: high confidence that prover had access to file (without proving that it is complete or intact)
- Adequacy of high confidence vs. guarantee in different contexts

Local Polls

- Local hashes as hints that damage or subversion has occurred
- Triggers polls only, does not cause repairs from other peers



X

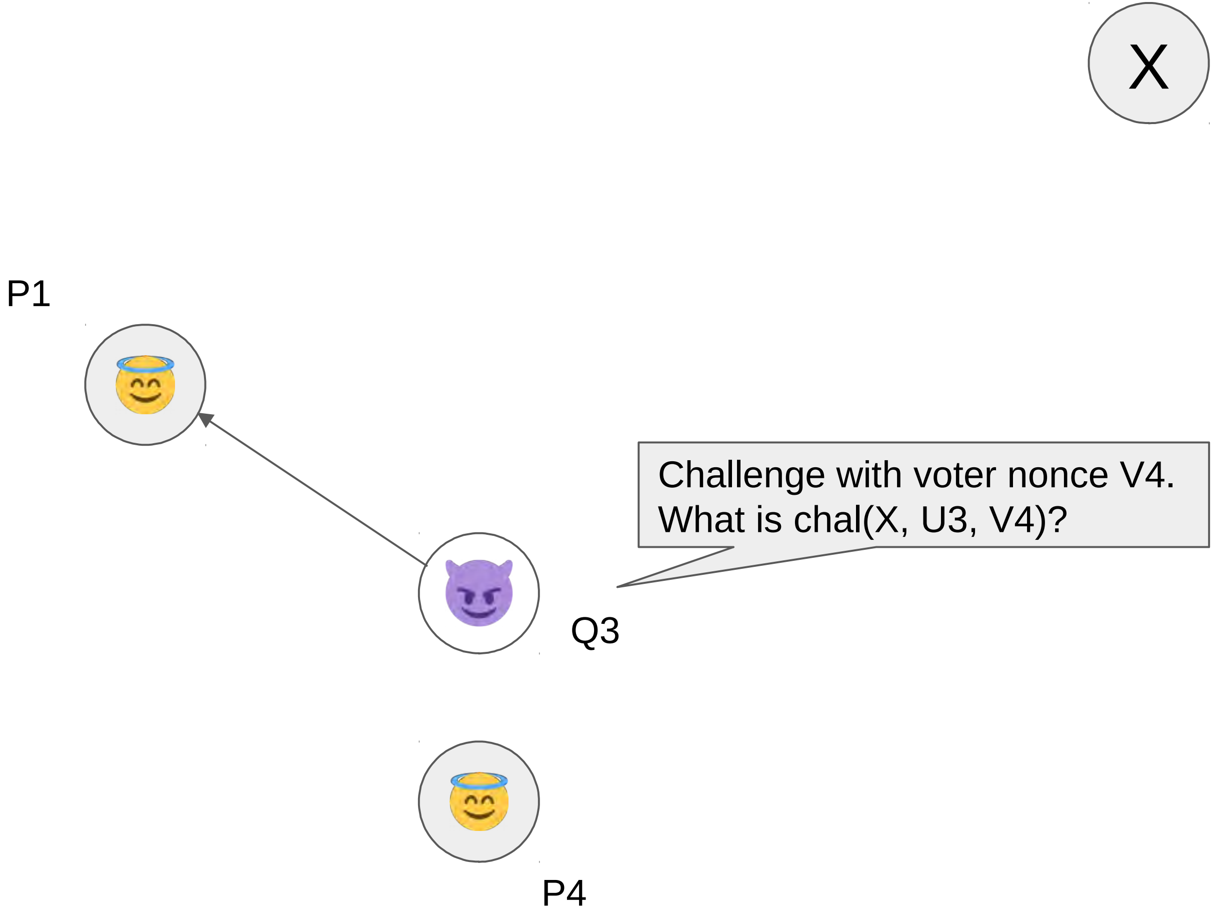
P1

Q3

chal(X, U3, V4) = ???

P4

Challenge with voter nonce V4.
What is chal(X, U3, V4)?



Challenge with voter nonce V4.
What is chal(X, U3, V4)?