You’ve Got the Wrong Use Case

Alan H. Karp
SitePassword

https://alanhkarp.com/UseCases.pdf

W3C CCG Meeting 27 January 2026

alanhkarp@gmail.com

What Took Me So Long - Part 1

- 1996: Why do all IAM systems have problems?
  - Vulnerability
  - Complexity
  - Usability

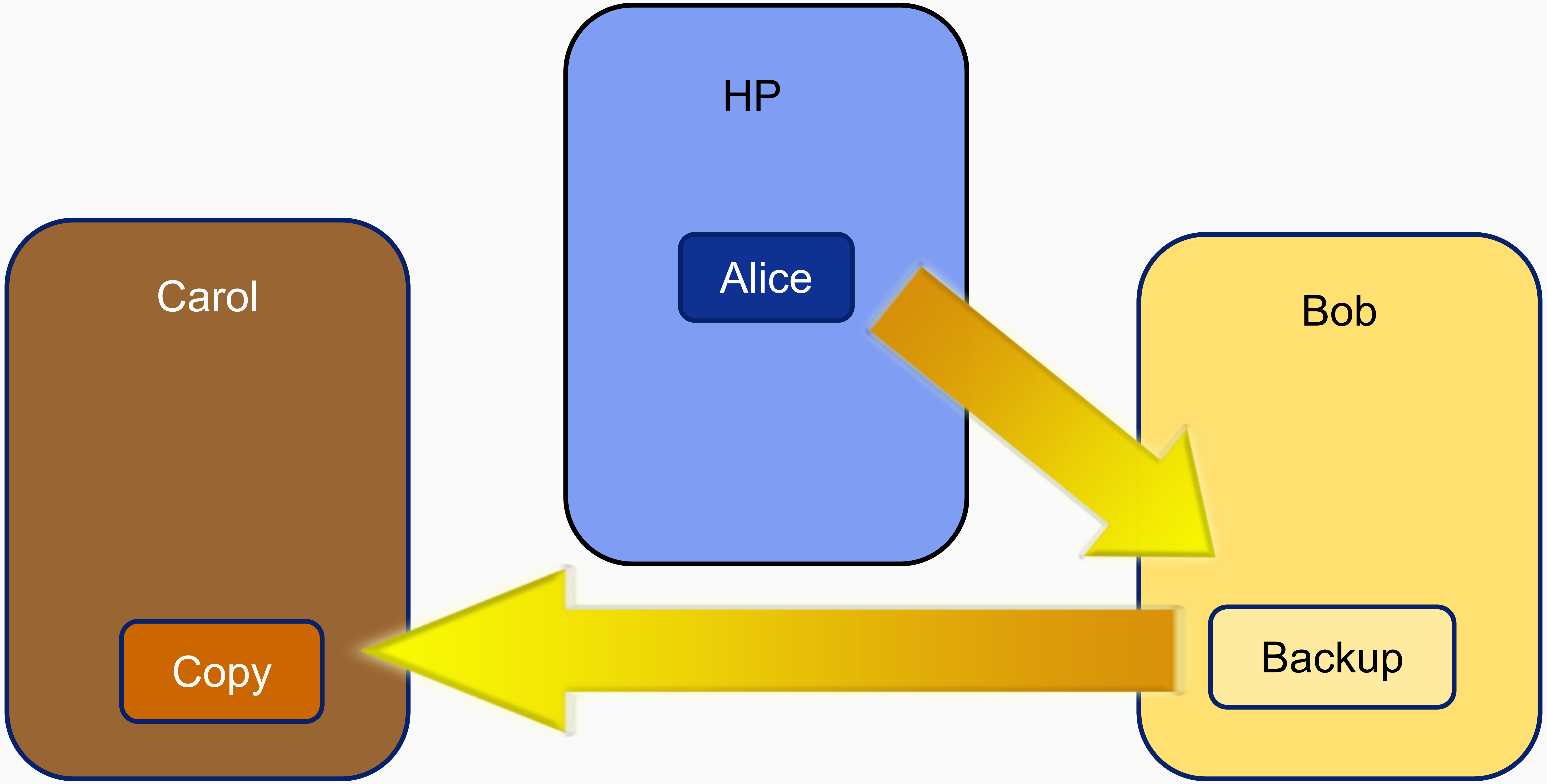
Some Projects with IAM

- Services Oriented Architecture (SOA)
    - XML and SOAP based
- AWS Cedar Policy System
    - Authorization policy management
- Linked Web Storage (Solid)
    - Decouple data storage, authentication, access control

What Took Me So Long - Part 2

- 2009: Transitive Access Paper
    - Managing permissions
    - Assigning responsibility

Service Composition



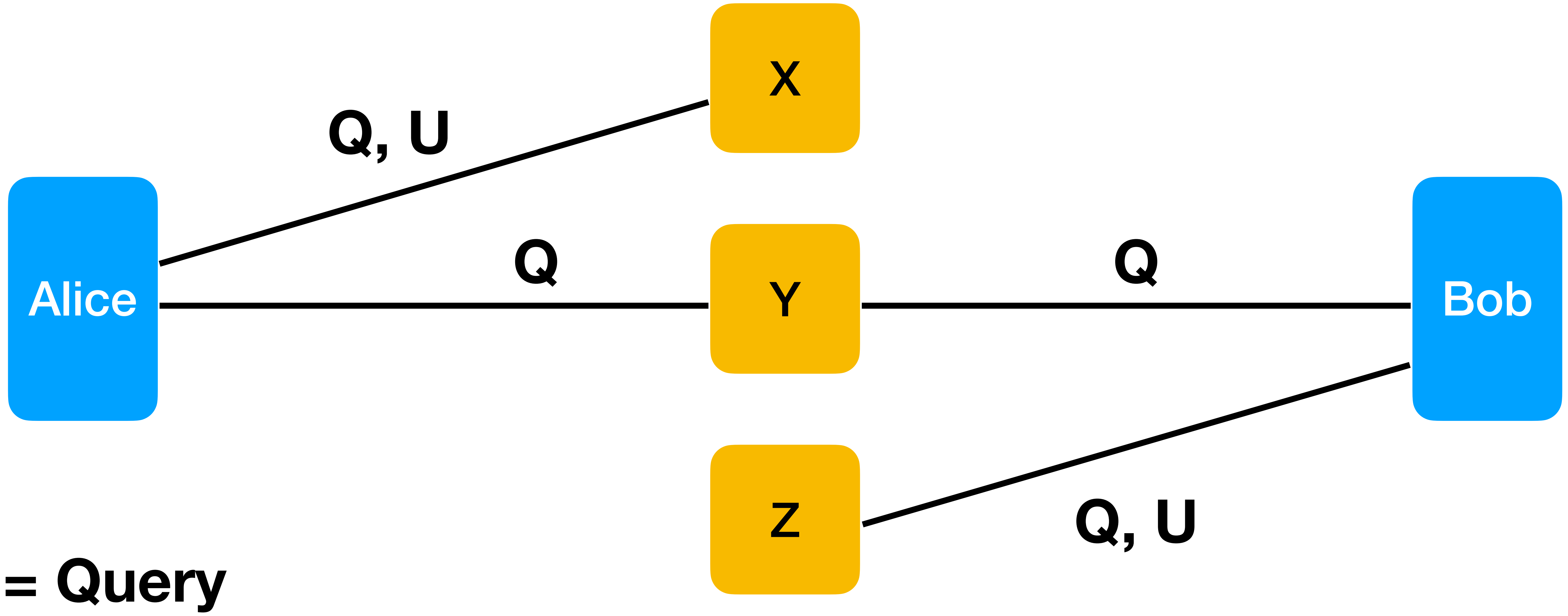
What Took Me So Long - Part 3

- 2024: The Aha Moment
  - My example answers my question from 30 years ago
  - It’s an important use case IAM designers didn’t consider

Use Case Hazards

Basic Access Control

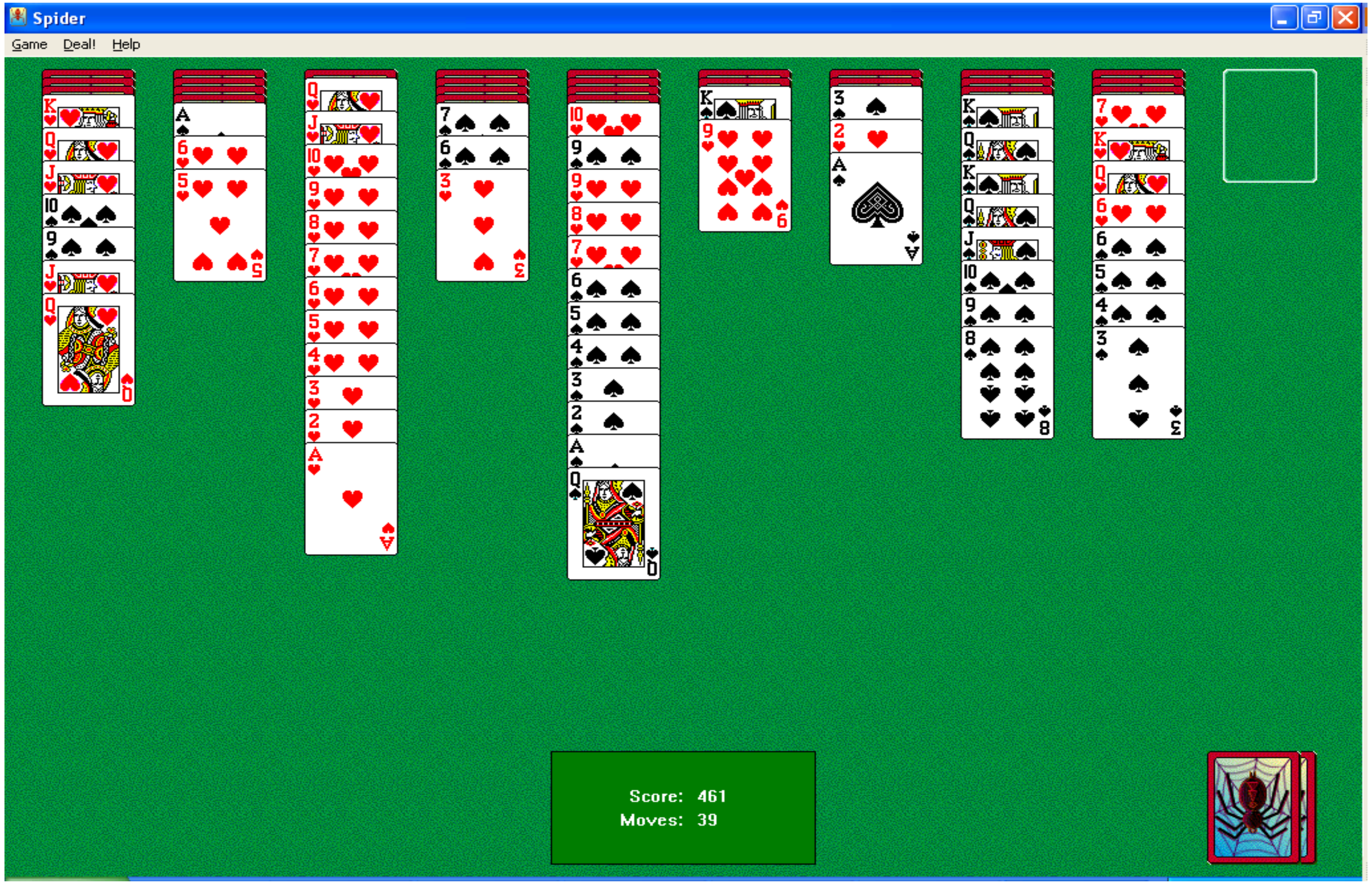
Different users have different permissions



Q = Query

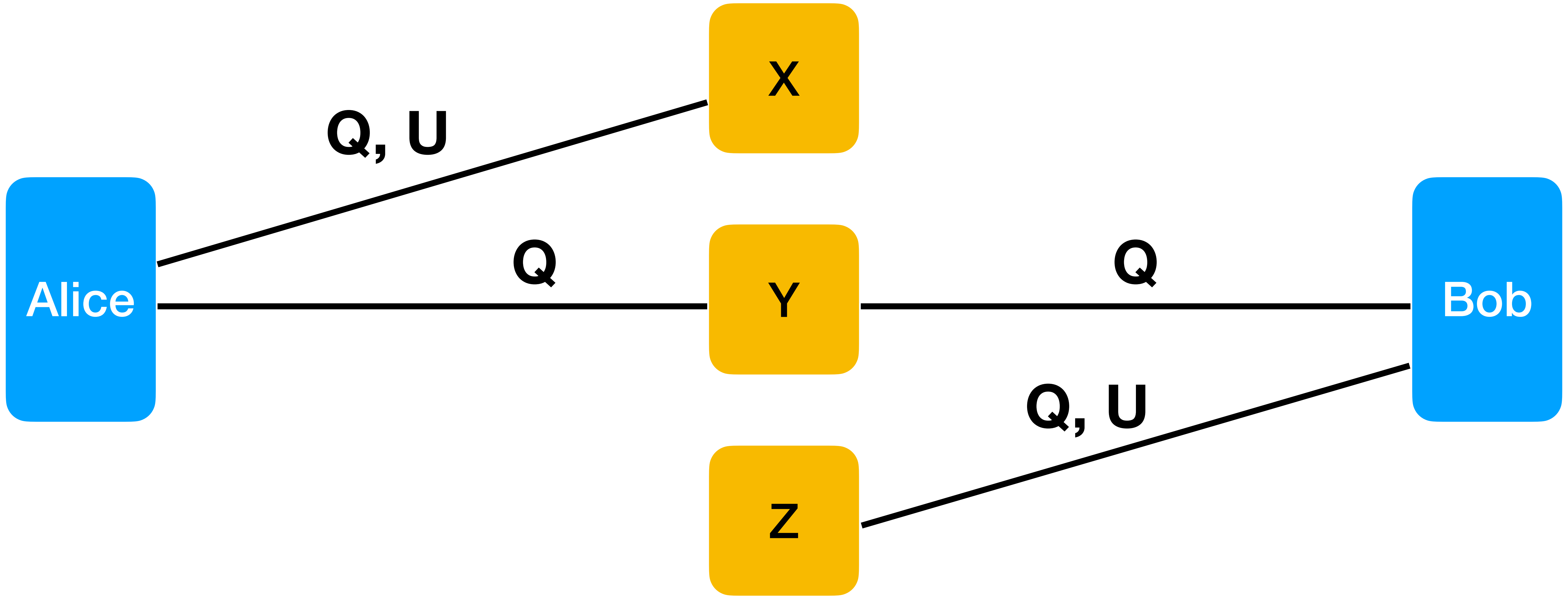
U = Update

Hazard - Excess Authority

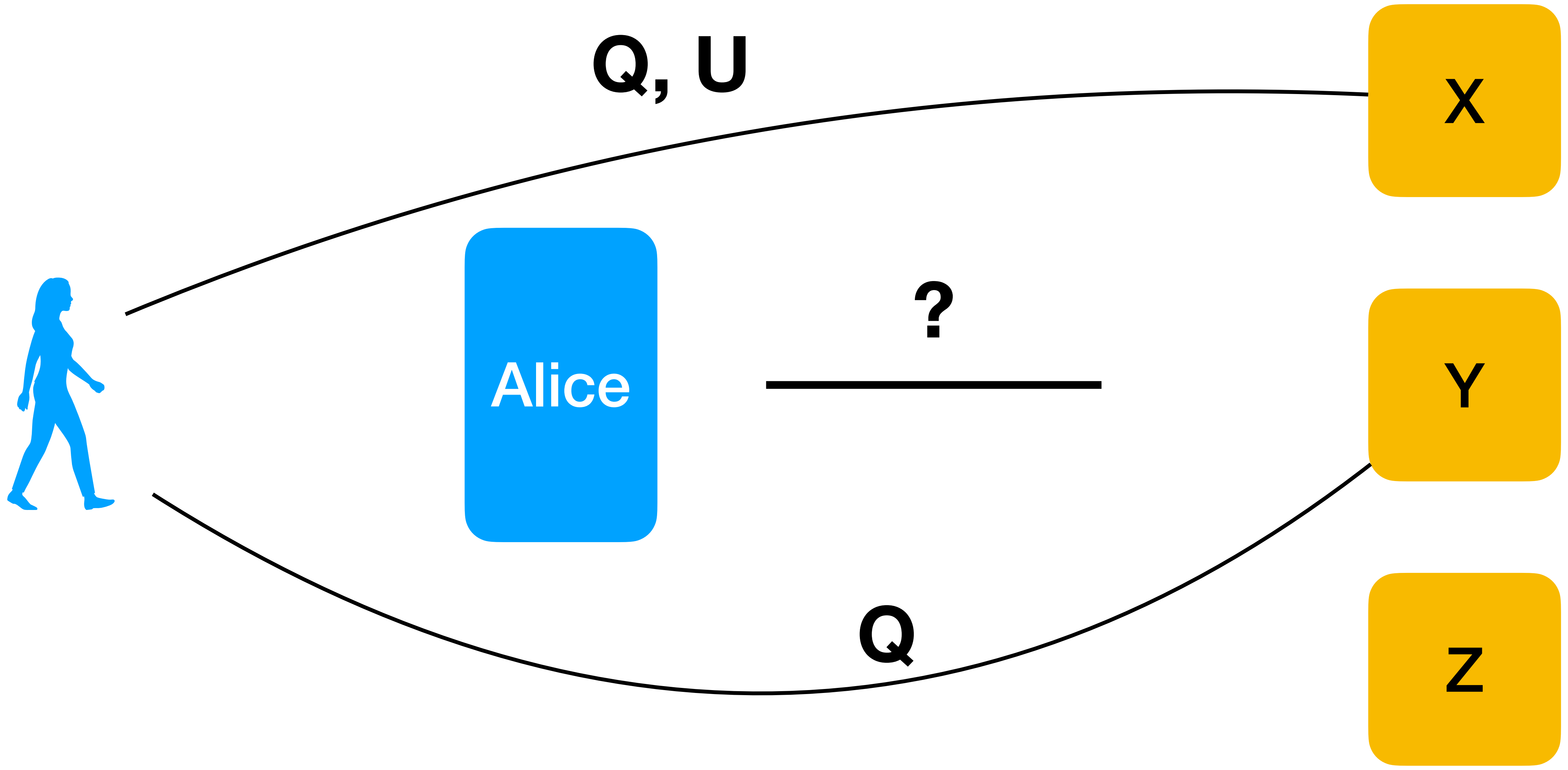


Basic Access Control

This picture is a lie!



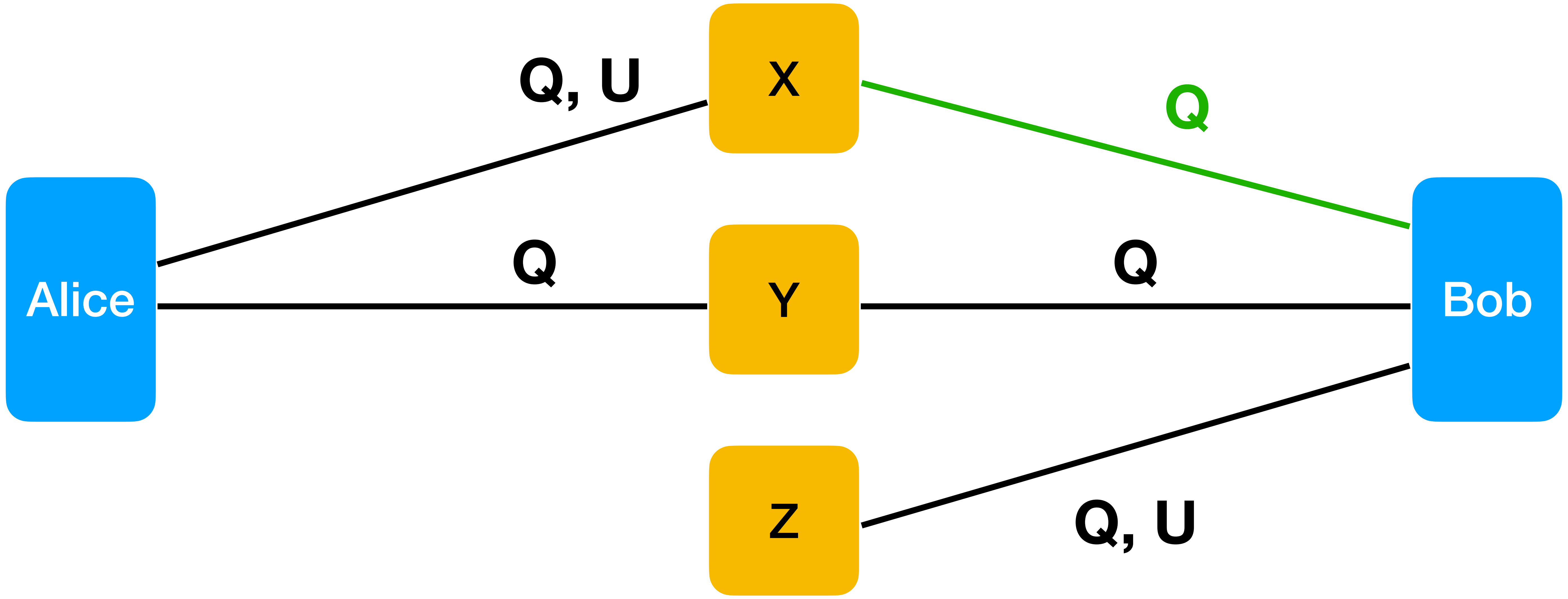
An Unasked Question



Does Alice need all of 🚶‍♀️'s permissions?

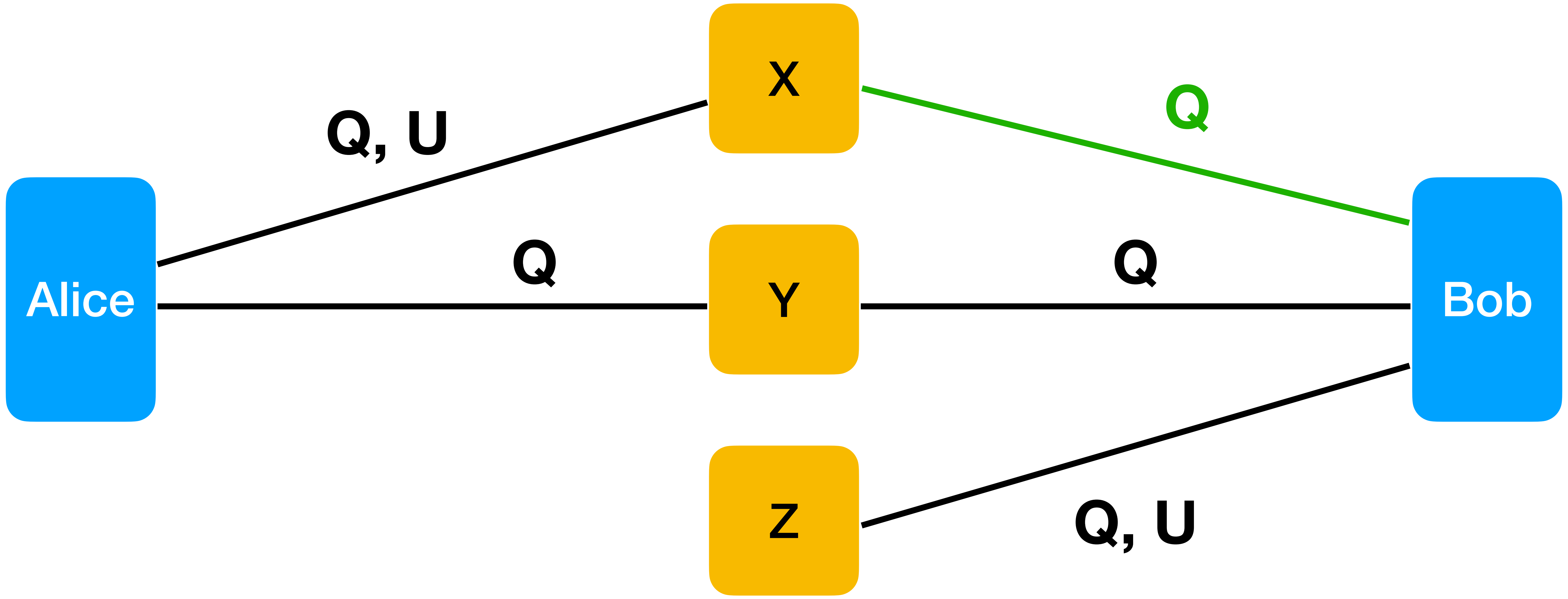
Delegation

Users Need to Delegate



Hazard - Availability

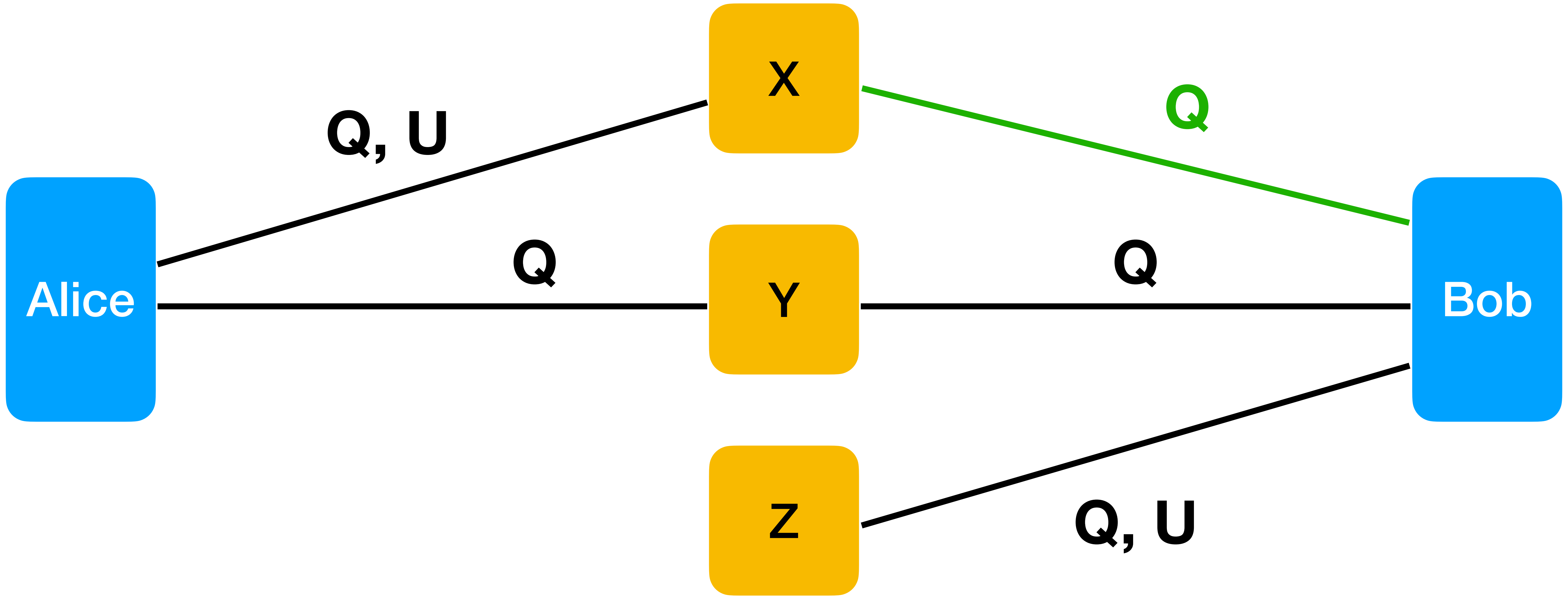
Is someone around to make the change?



Does your design allow Alice to update Bob's permissions?

Hazard - Responsibility

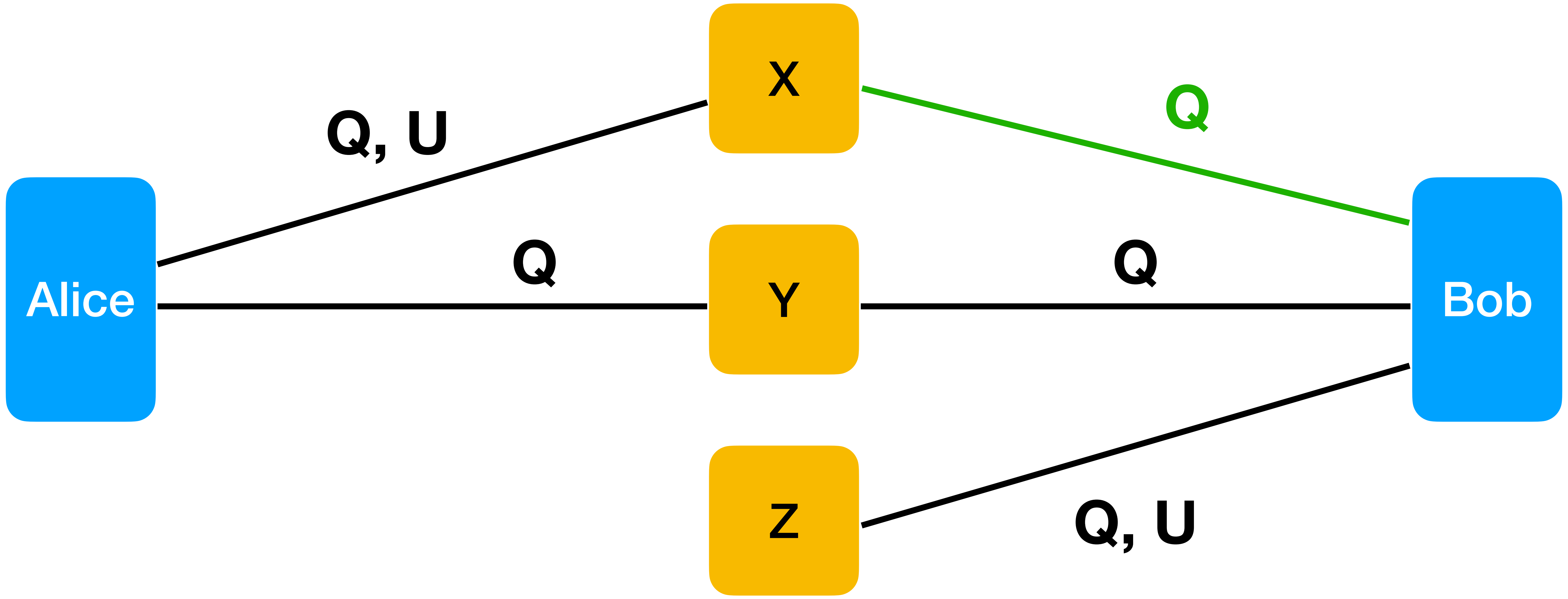
What if Bob does something bad?



Is there a way to know that Alice is
who asked for Bob to get permission?

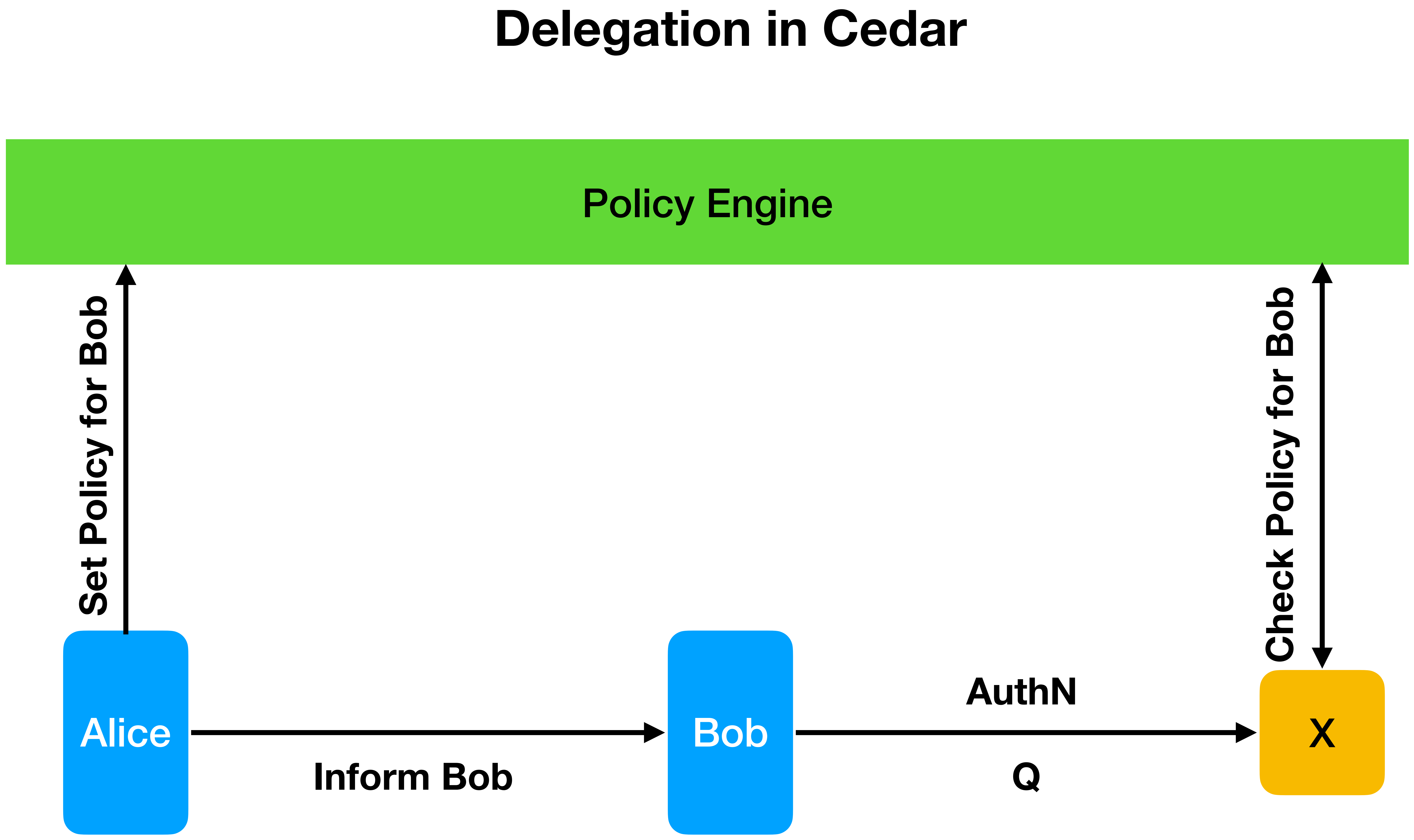
Hazard - Conditional Policy

Alice wants to limit Bob's access to working hours.



Does Alice have to ask someone to
revoke and re-delegate?

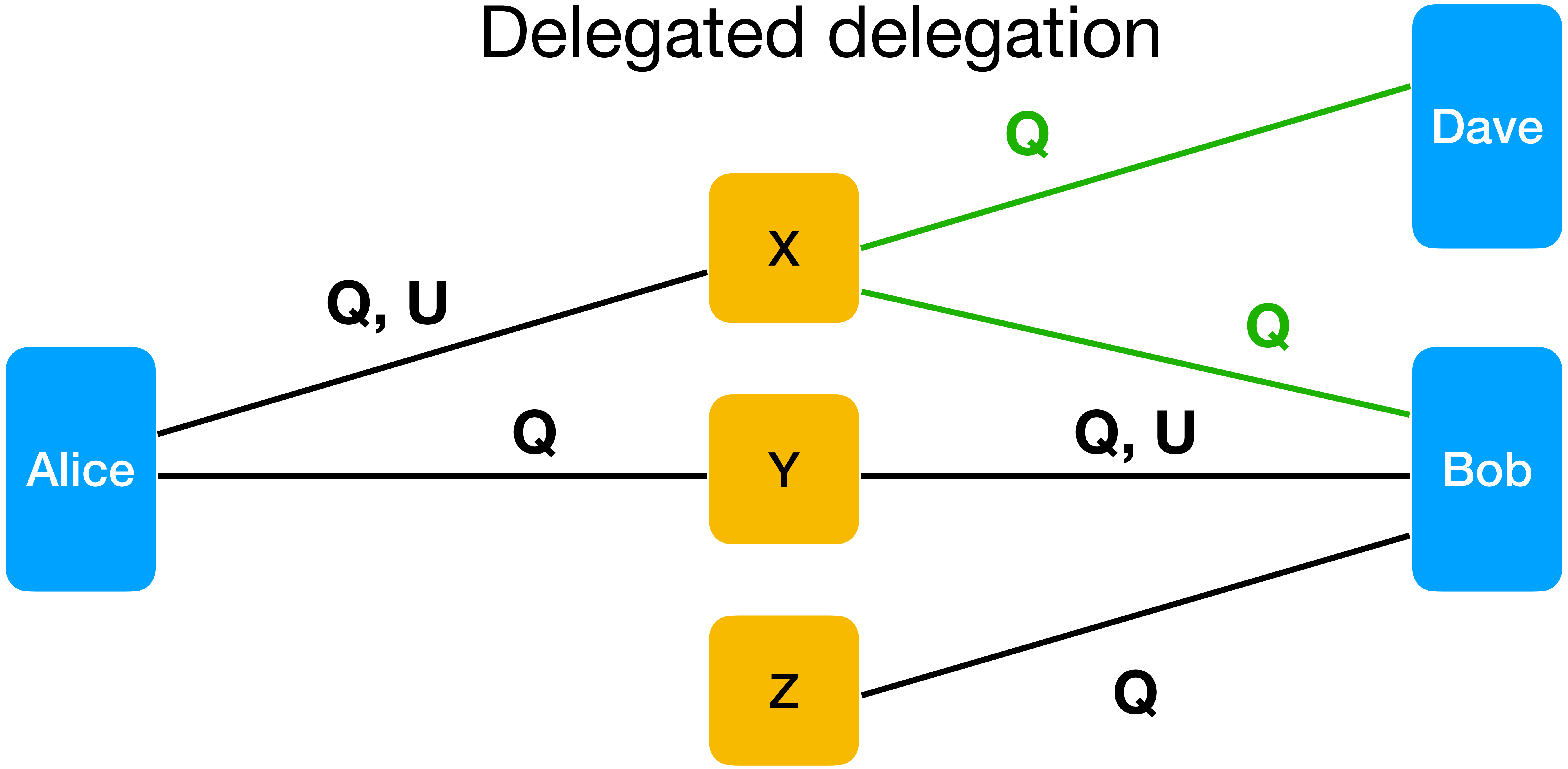
Hazard - Performance



Potentially complex policy calculation on the critical path

Chained Delegations

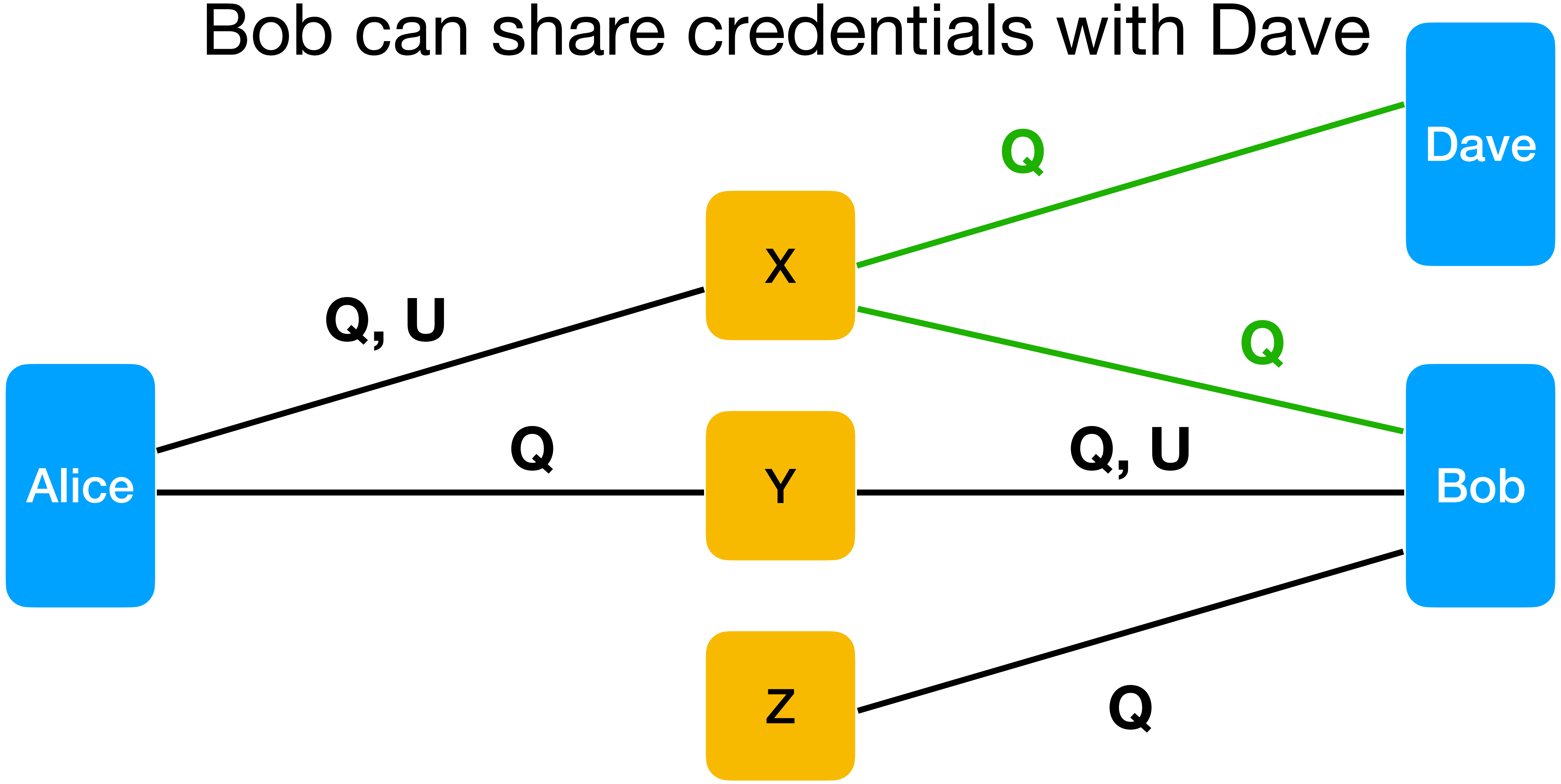
Delegated delegation



Bob delegates a delegated permission to Dave

Hazard -“Oh, no. You’ve lost control!”

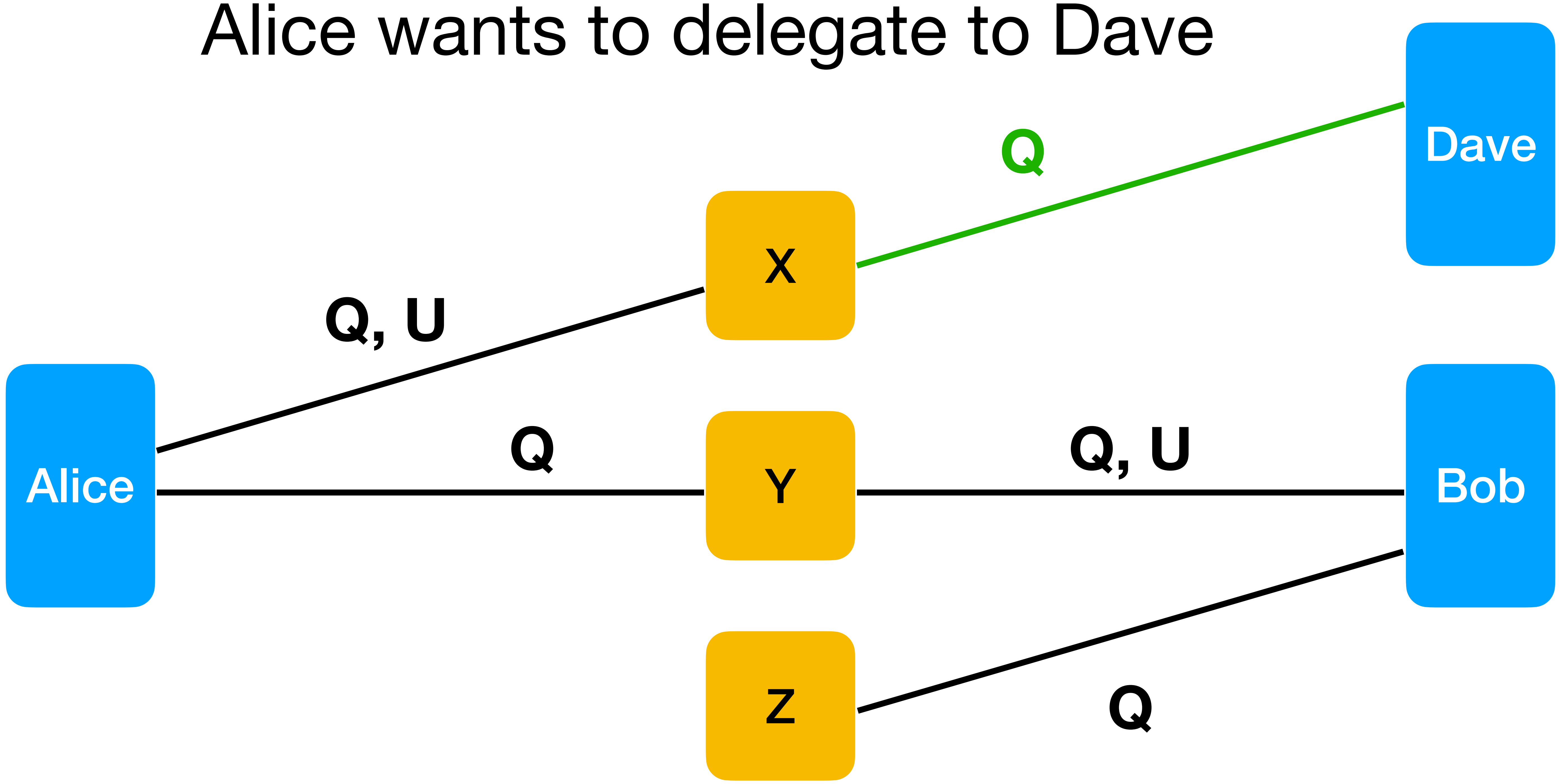
Bob can share credentials with Dave



Don’t prohibit what you can’t prevent

Oblivious Delegation

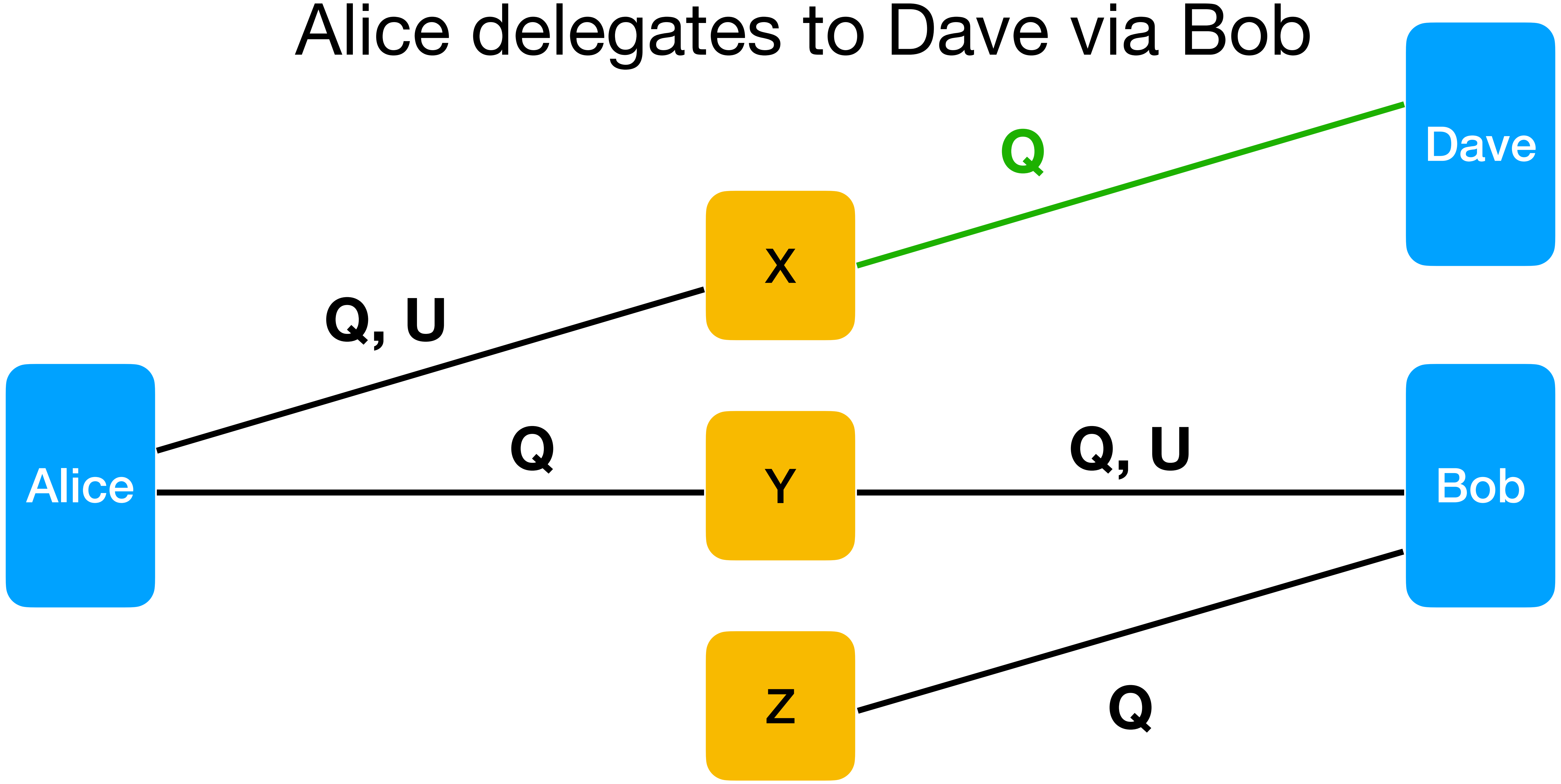
Alice wants to delegate to Dave



But must pass the request through Bob

Hazard - Oblivious Delegation

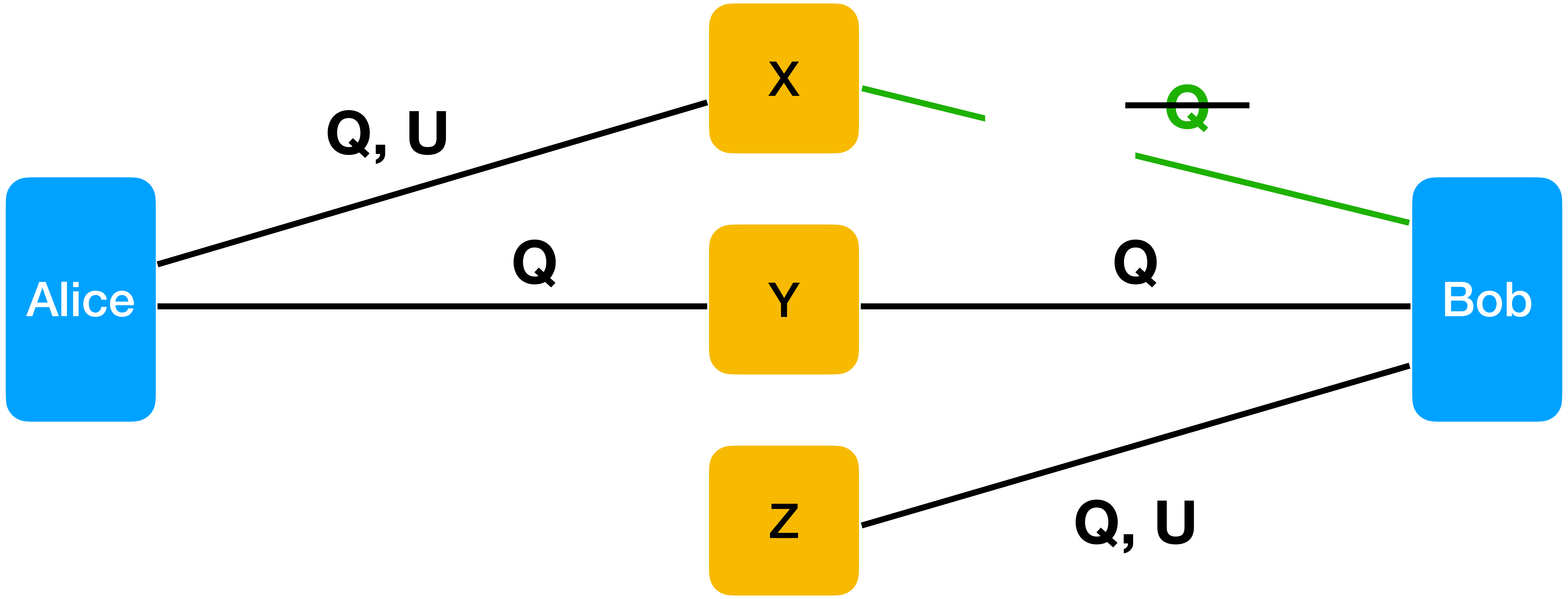
Alice delegates to Dave via Bob



Bob gets the permission

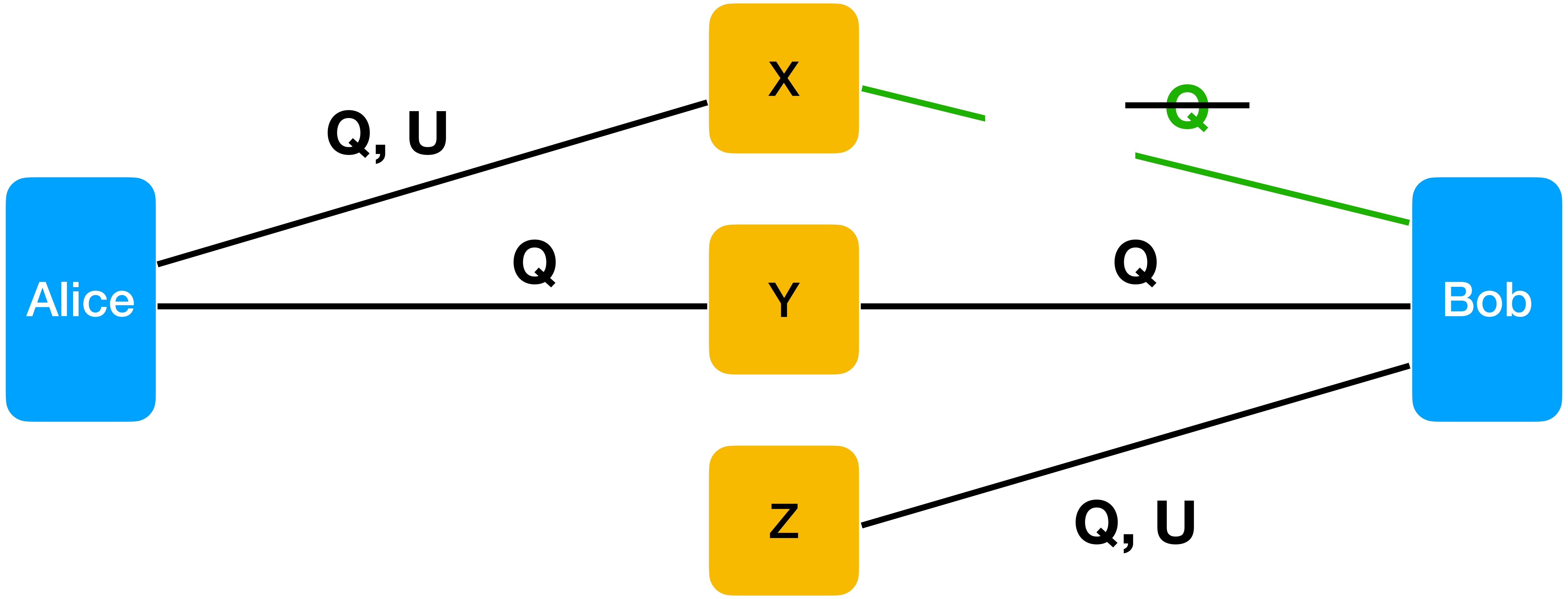
Revocation

There is a need to to revoke



Hazard - Permission to Revoke

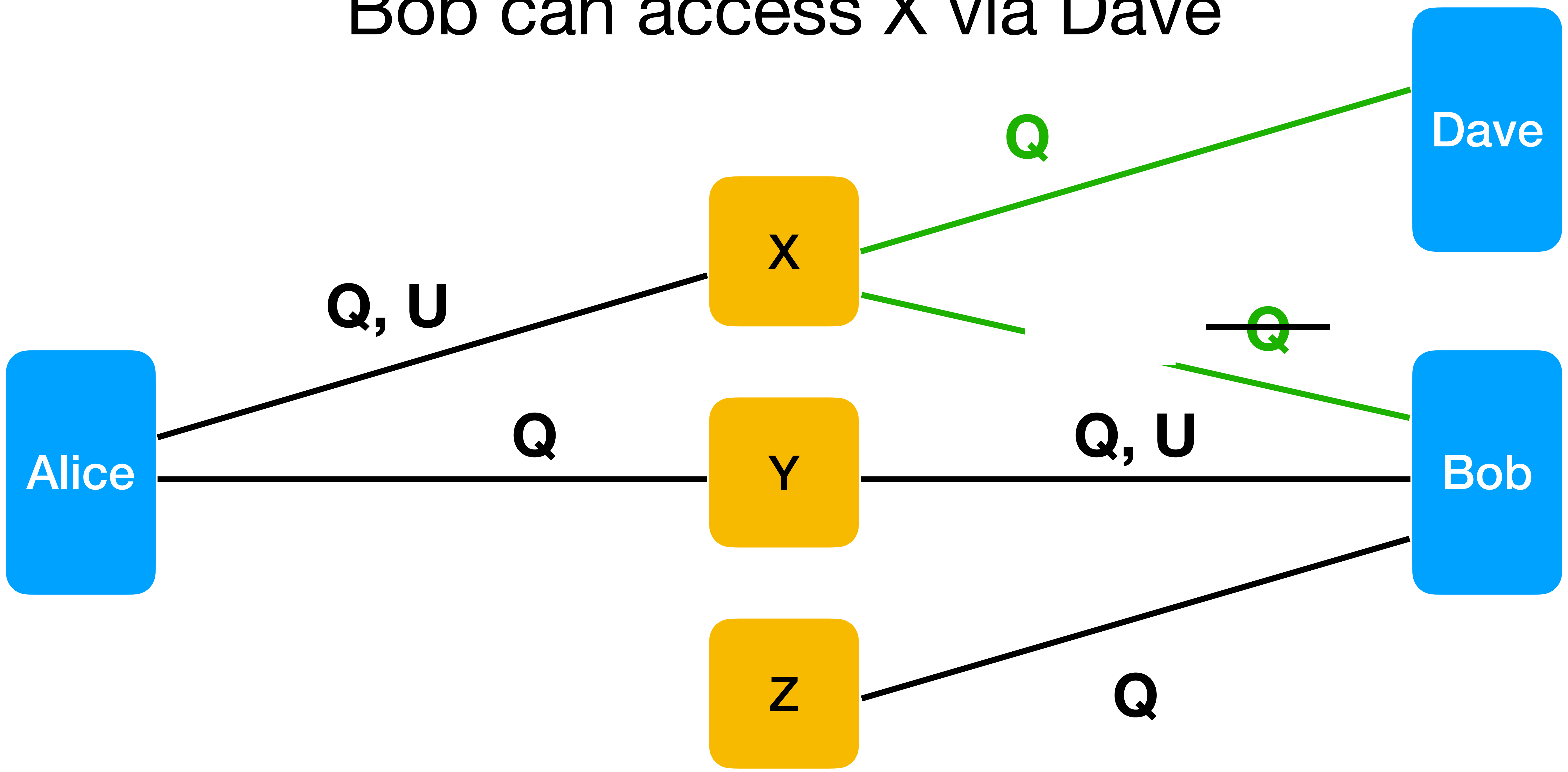
Who can revoke?



How do you know that Alice is allowed to revoke Bob’s access?

Hazard - Sock Puppet

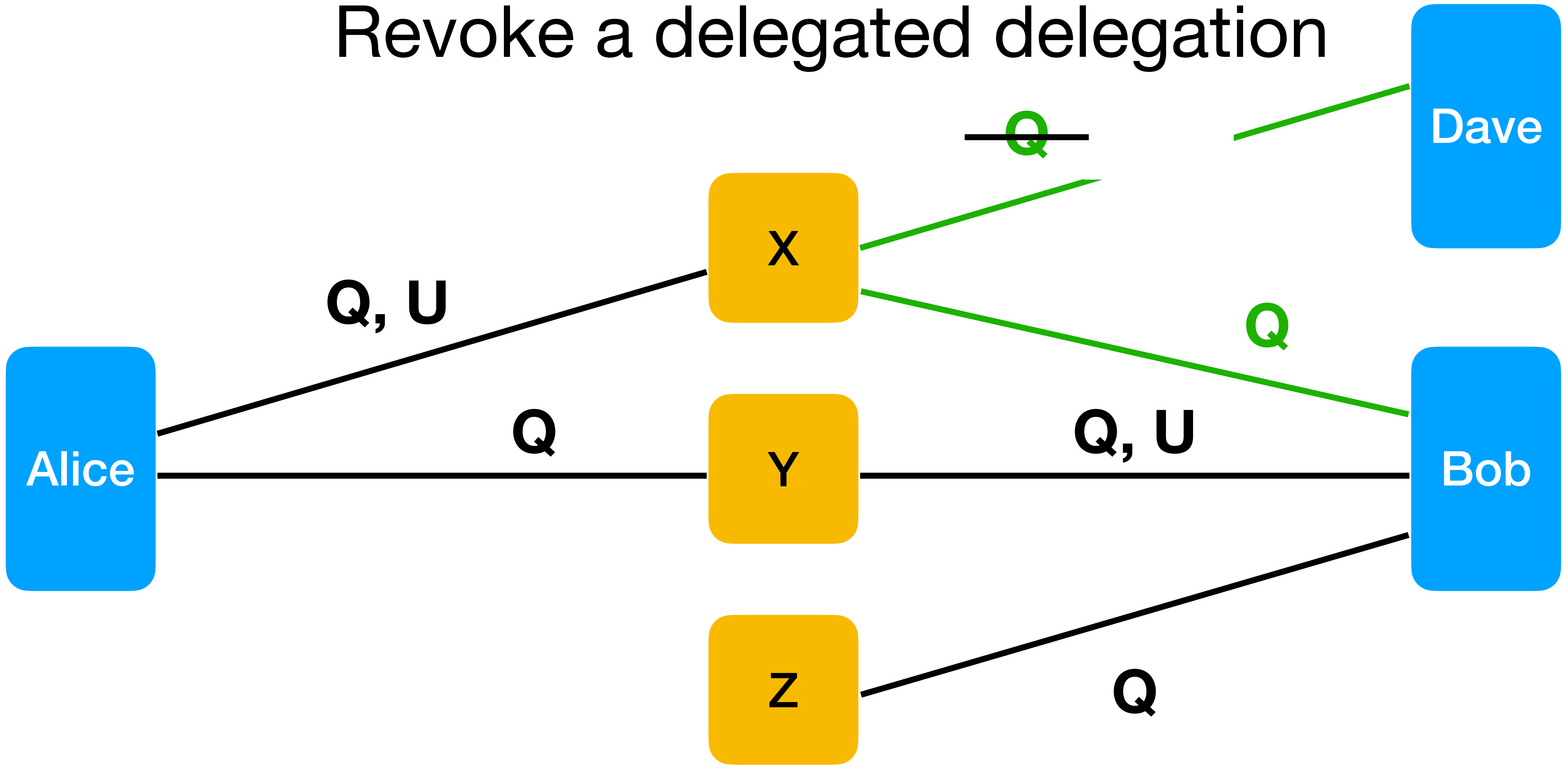
Bob can access X via Dave



Need to revoke all downstream delegations

Hazard - Skip Revocation

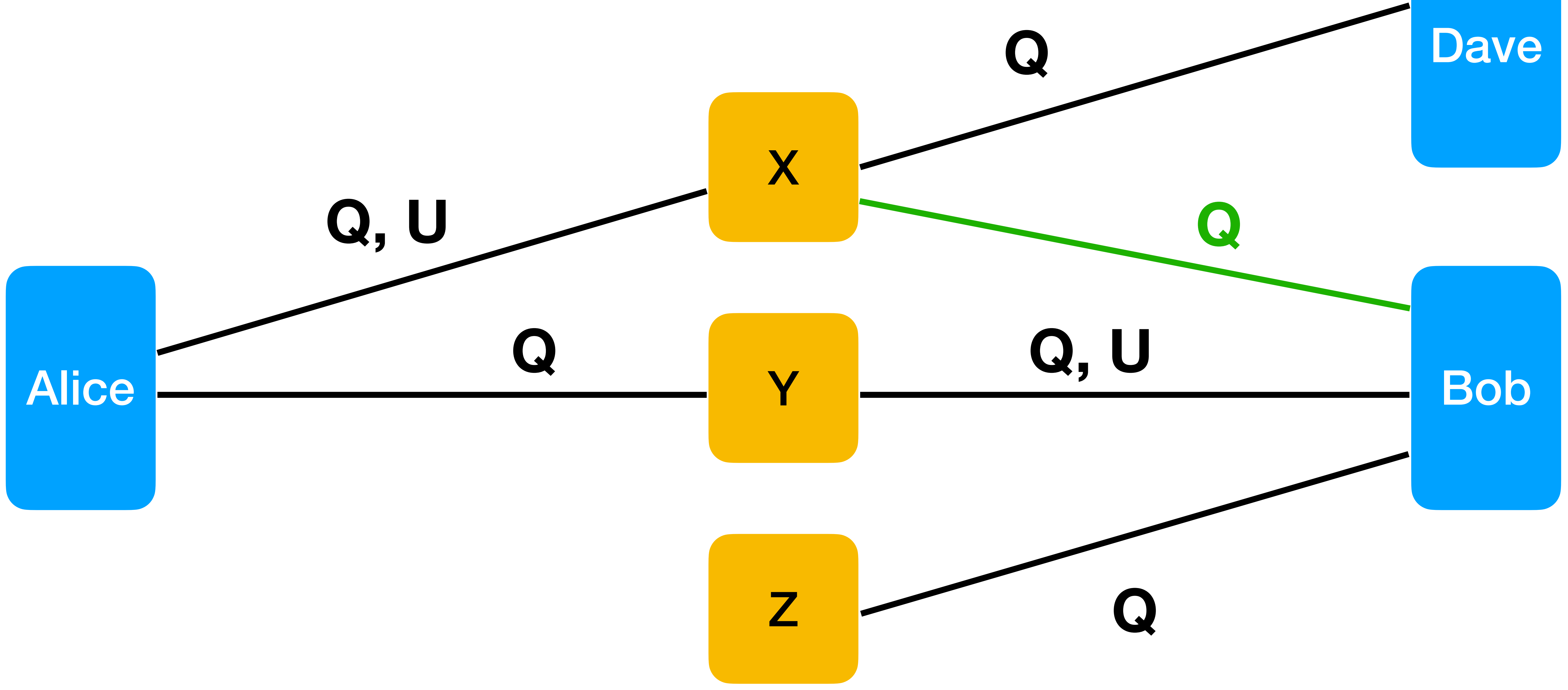
Revoke a delegated delegation



Can Alice revoke Dave’s access without revoking Bob’s?

Independent Delegations

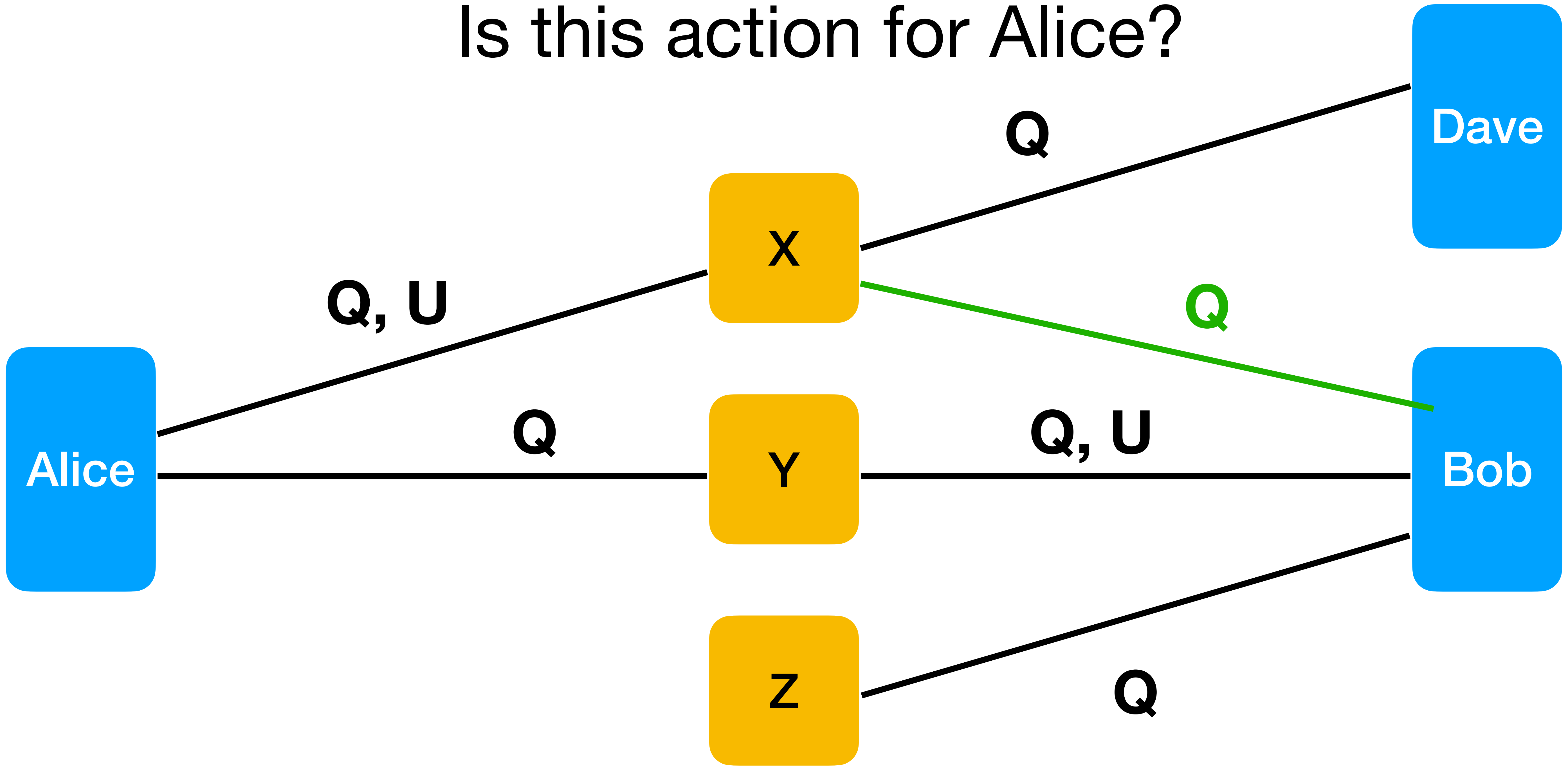
Overlapping delegations



Alice and Dave both delegate access to X to Bob

Hazard - Accounting

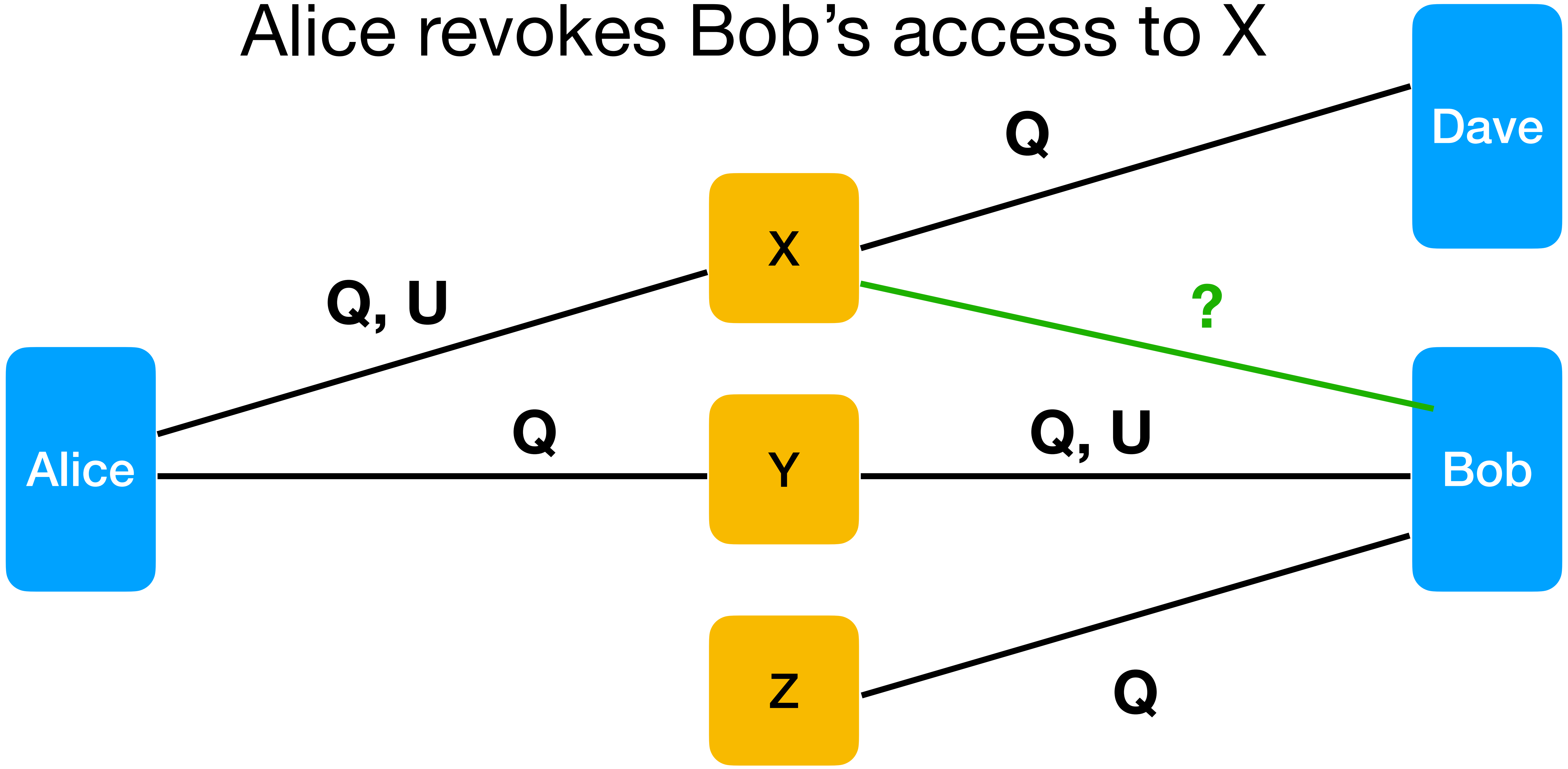
Is this action for Alice?



Or is it for Dave?

Hazard - Lost Delegation

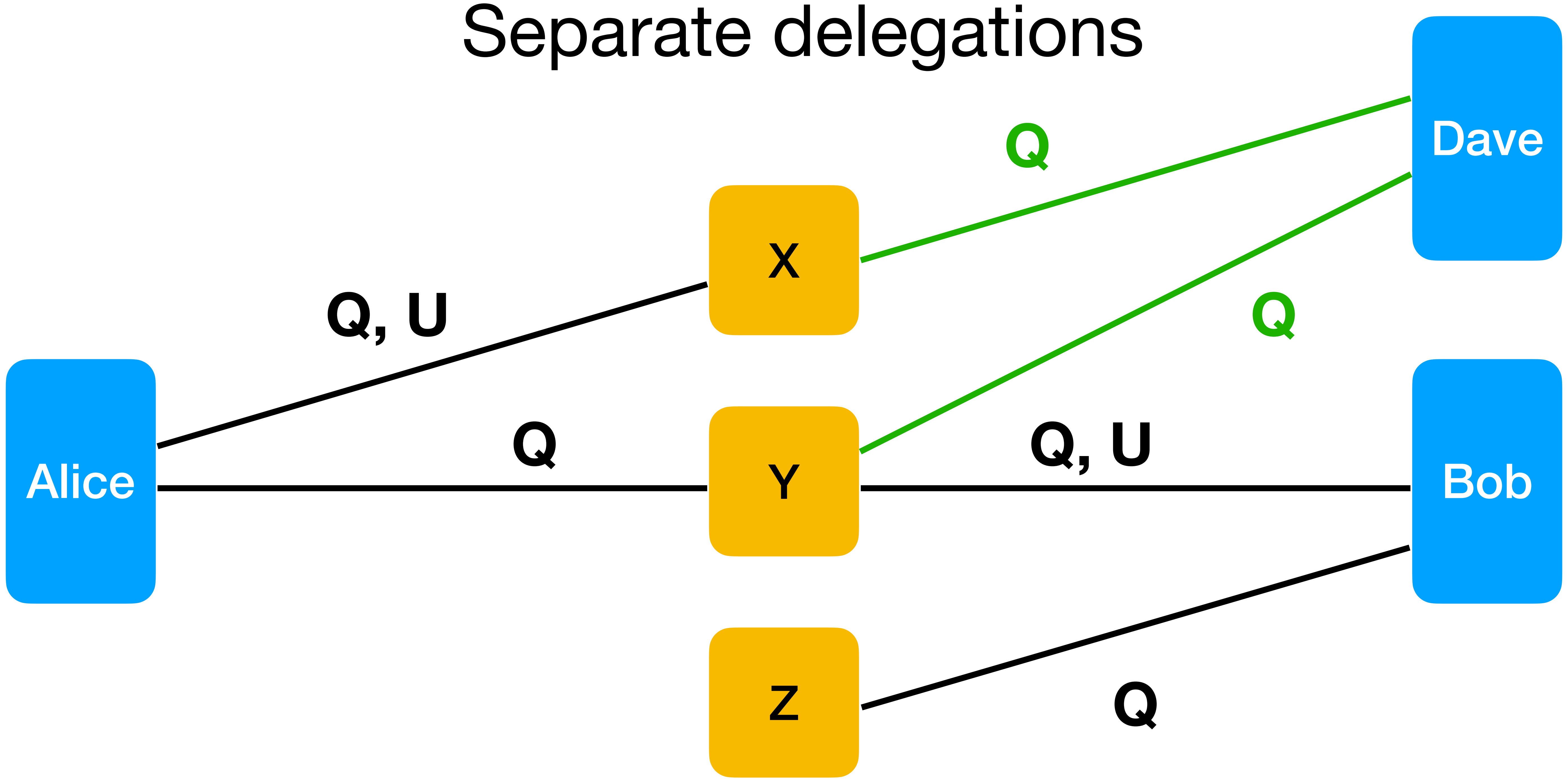
Alice revokes Bob’s access to X



Does Bob still have access to X?

Composed Delegations

Separate delegations

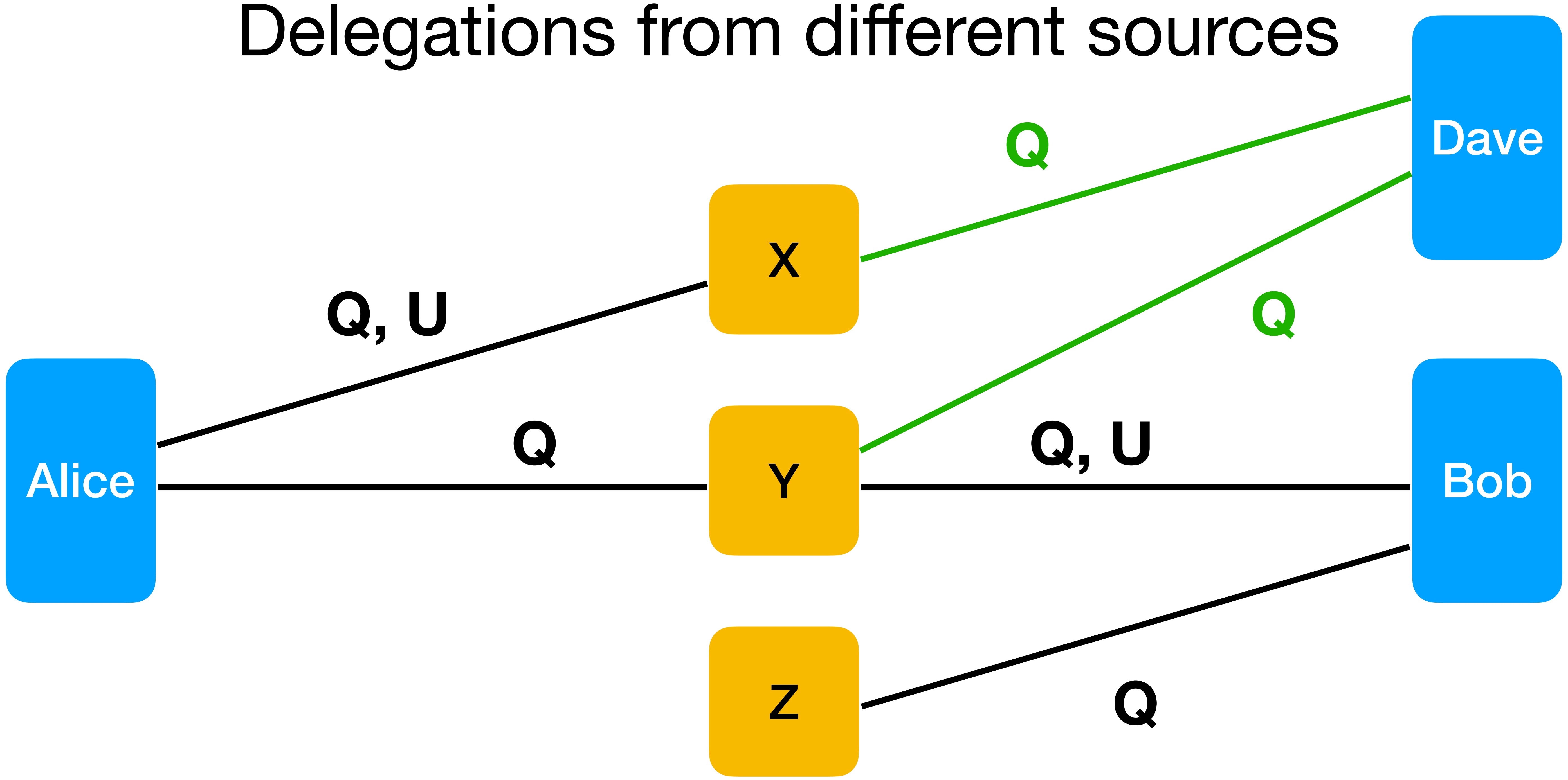


Alice delegates X to Dave

Bob delegates Y to Dave

Hazard - Composition

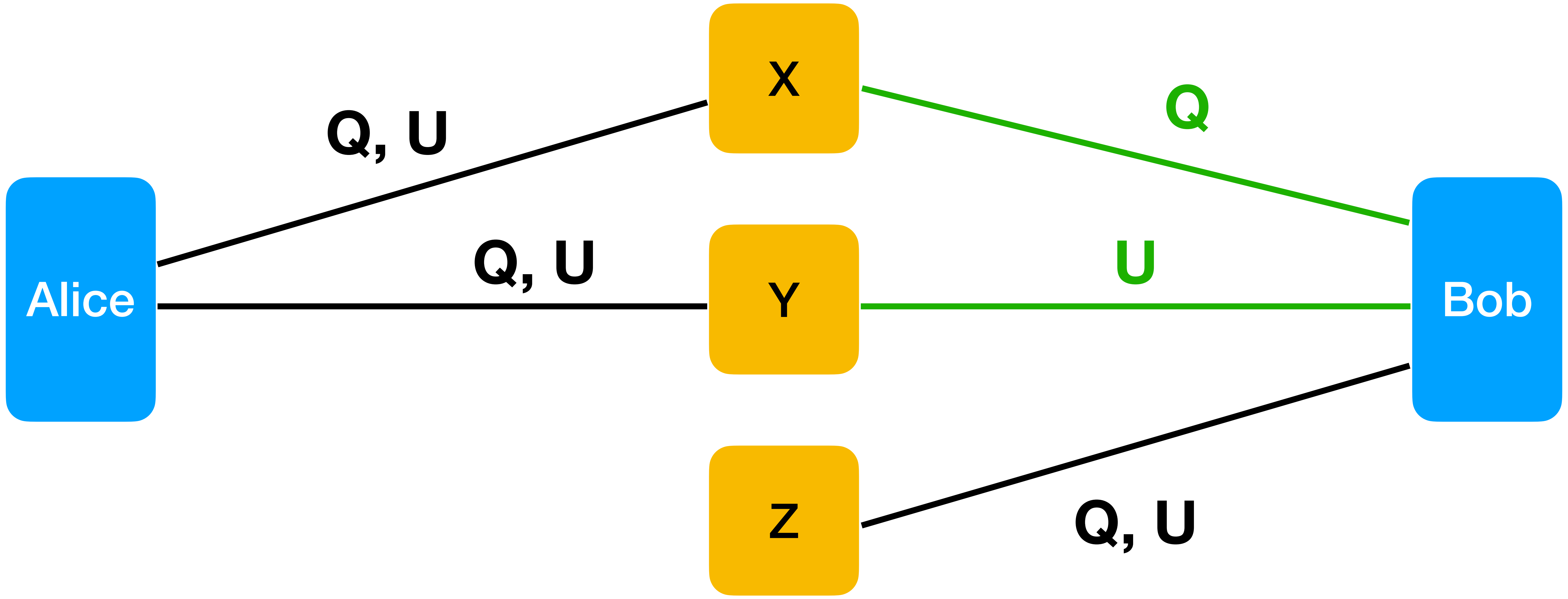
Delegations from different sources



Can Dave use both X and Y in a single invocation?

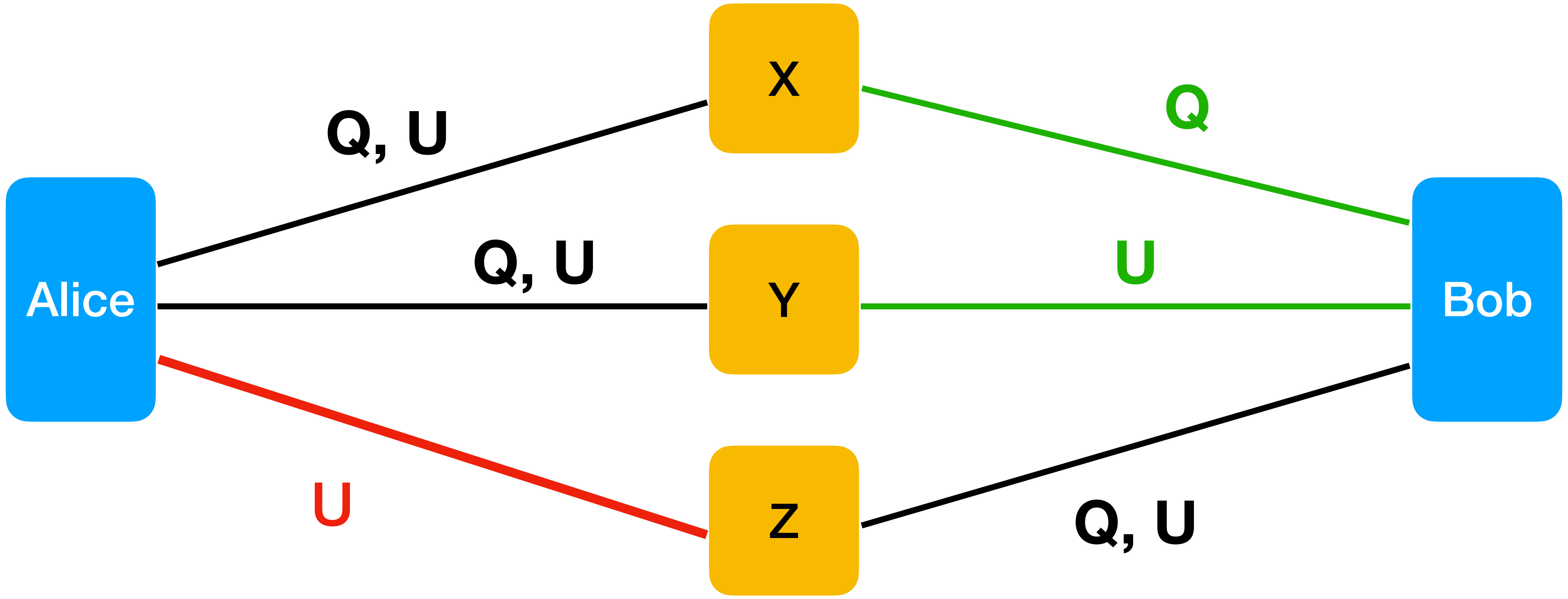
Multiple API Arguments

Bob expects, “Process X and put the output in Y.”



Hazard - Confused Deputy

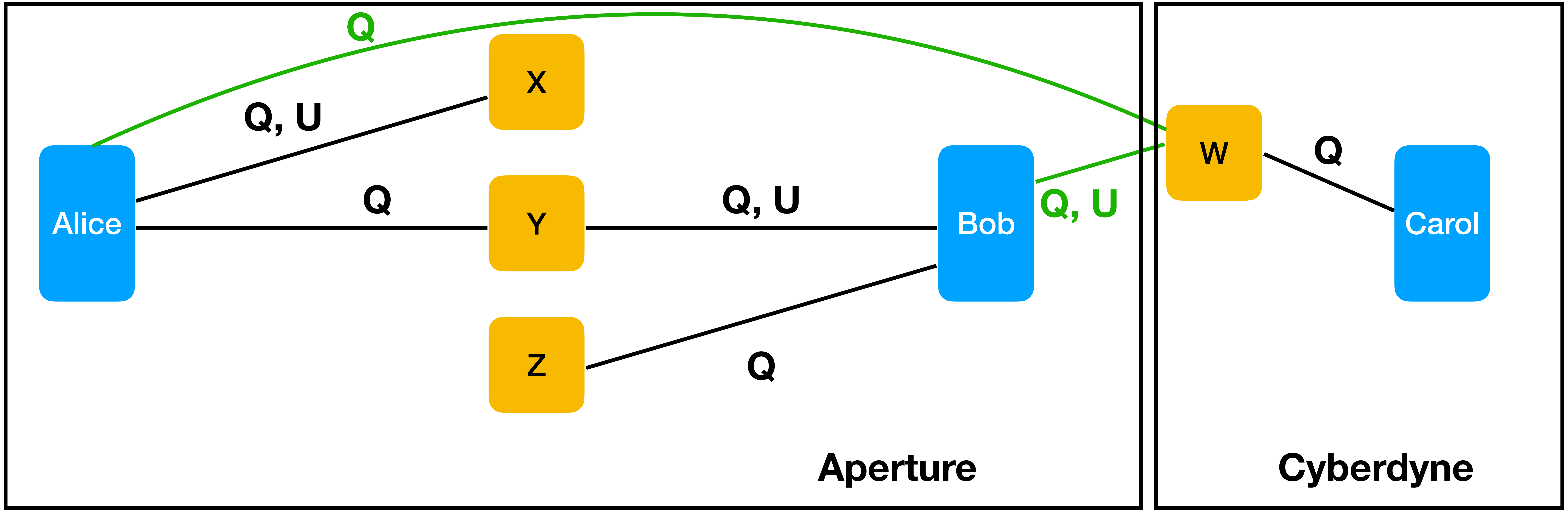
Alice actually says, “Process X and put the output in Z."



Oops! Bob just overwrote important data.

Cross Jurisdiction Delegation

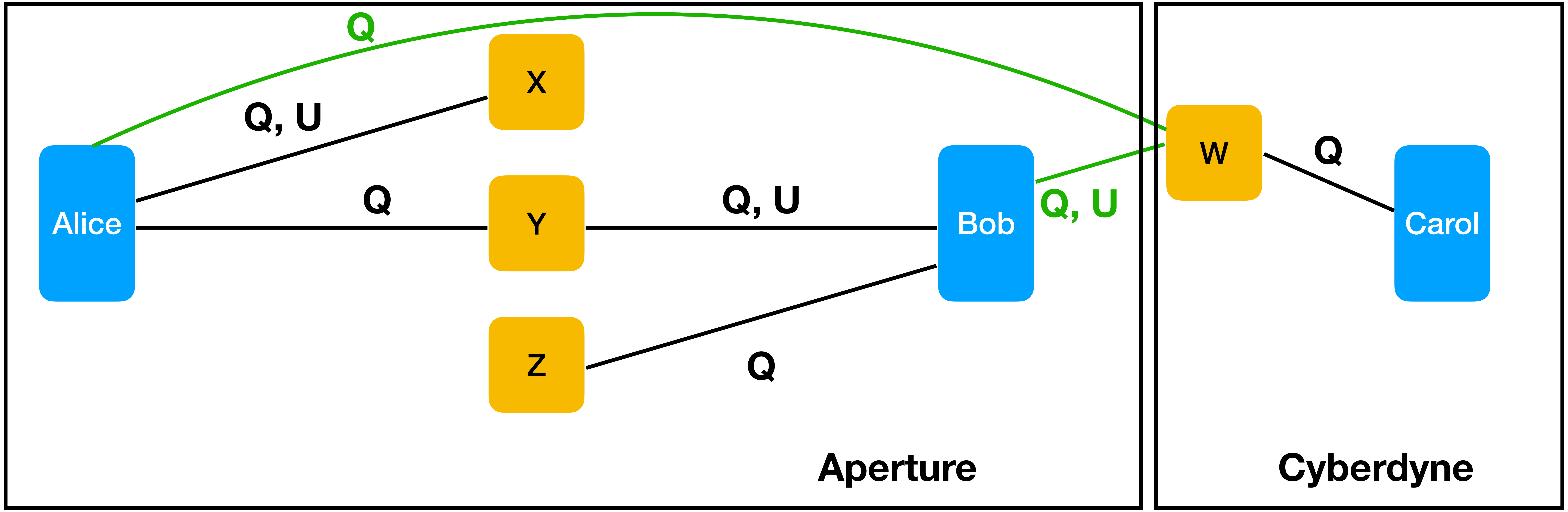
Different authentication domains



Bob delegates W to Alice

Hazard - Audit Failure

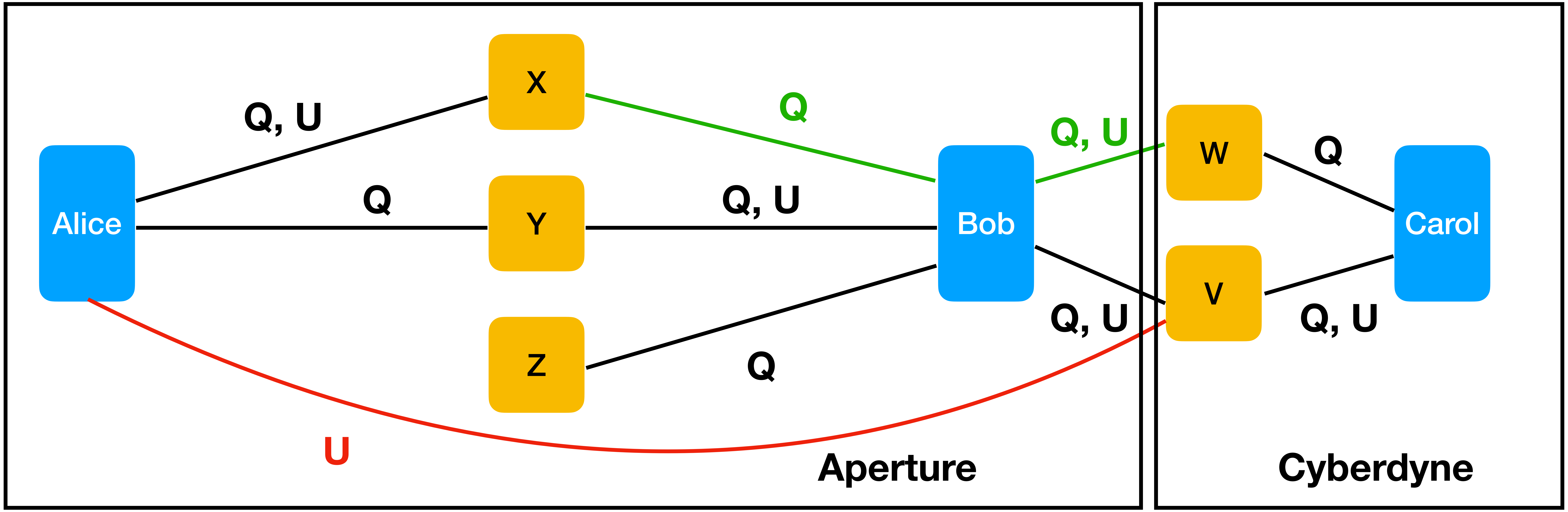
Carol has no idea who Alice is



How can Carol hold Alice responsible?

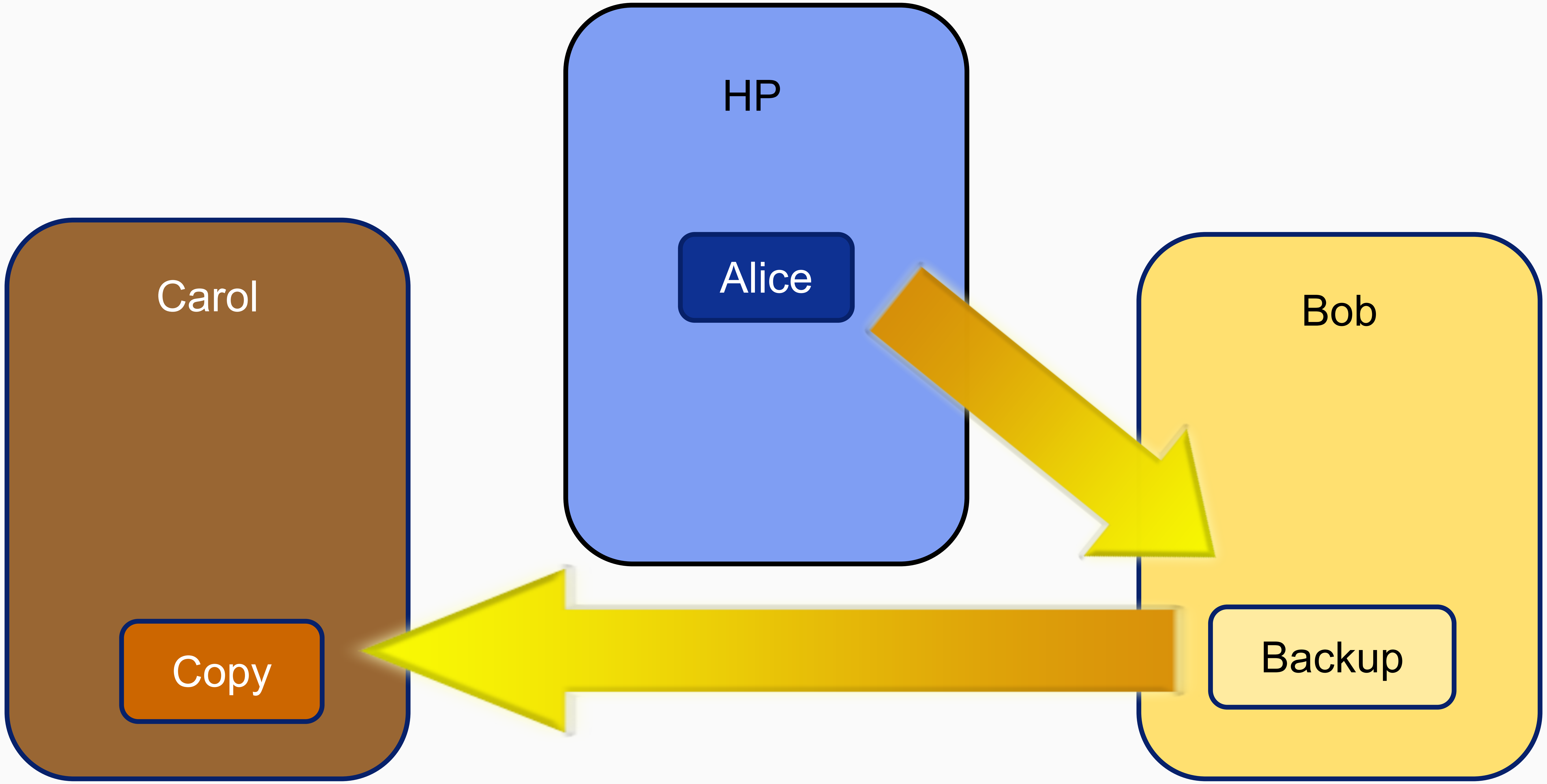
Cross Jurisdiction Confused Deputy

Different administrative domains



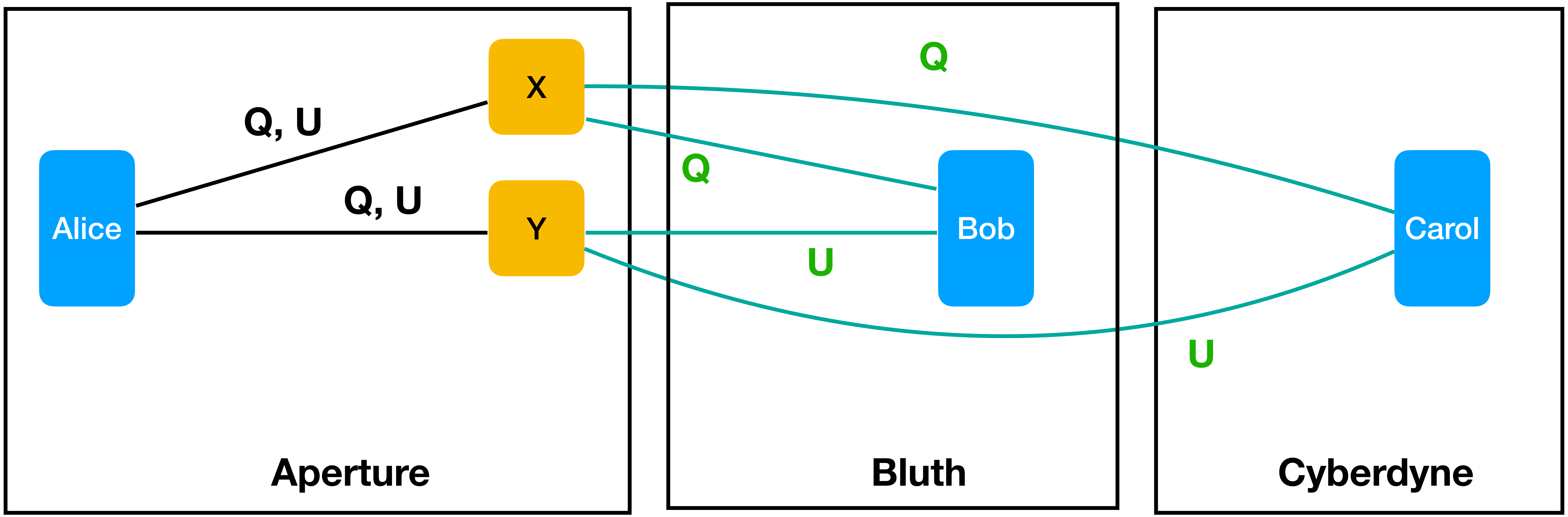
Bob has no way to know if Alice has access to V.

Service Composition



Transitive Access 1

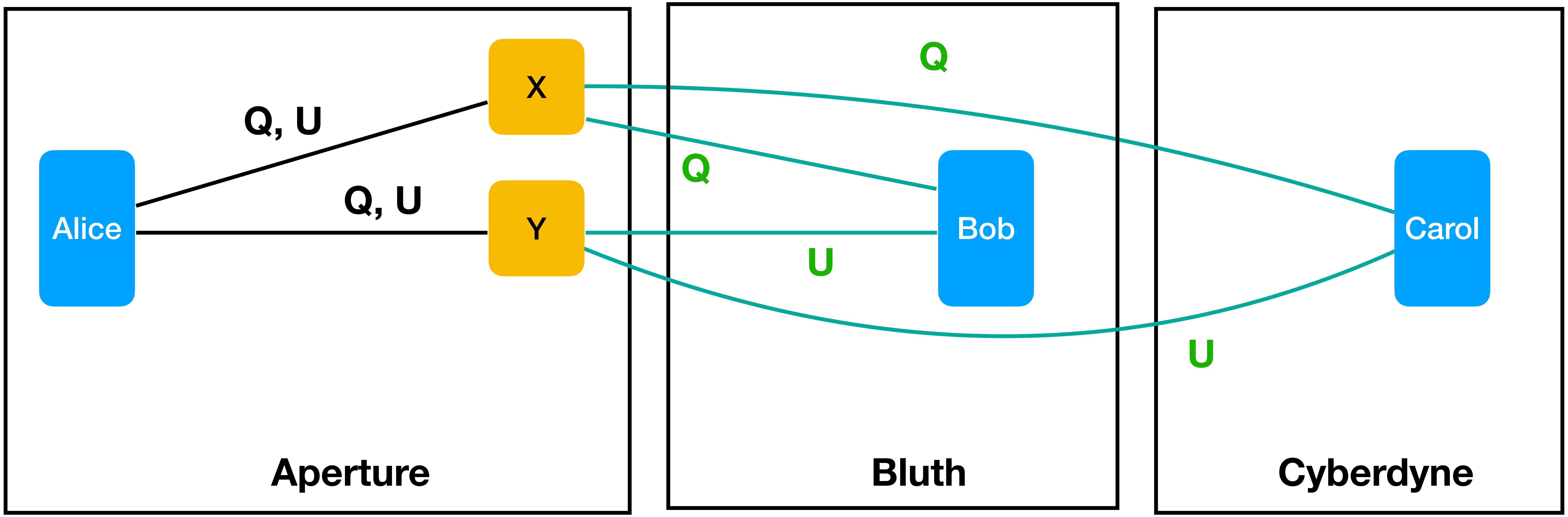
Alice: Bob, backup X to Y



Bob: Carol, copy X to Y

Hazard -Transitive Access 1

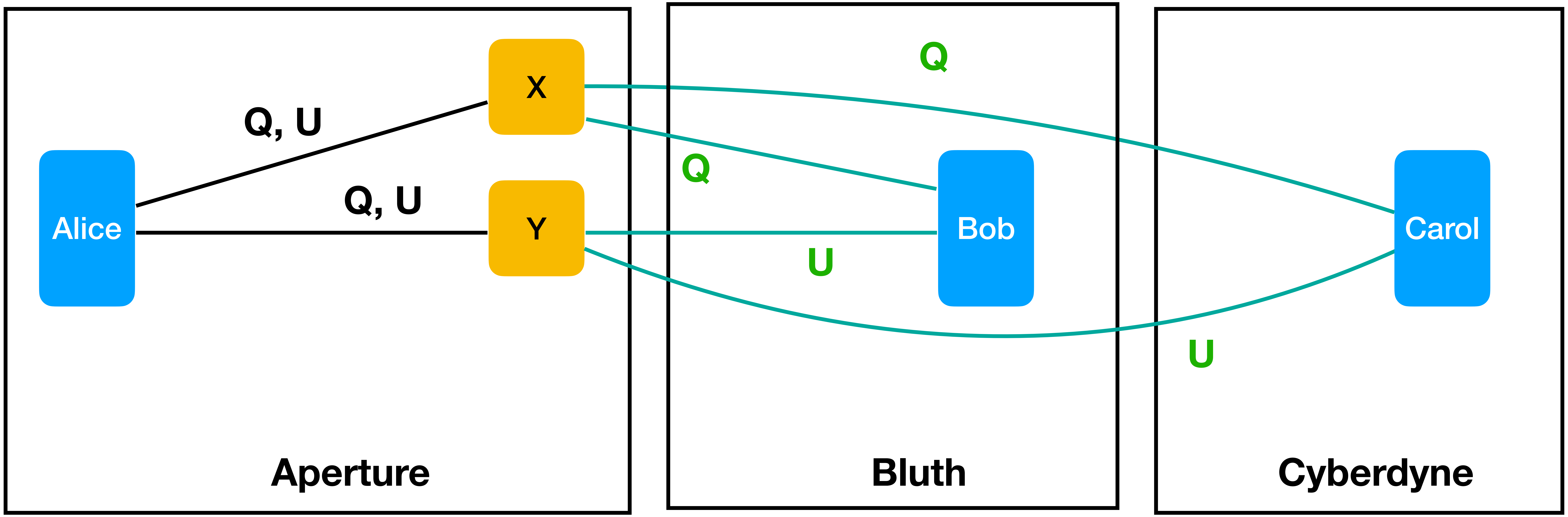
Federated identity failure



Essence of the Target breach

Hazard -Transitive Access 1

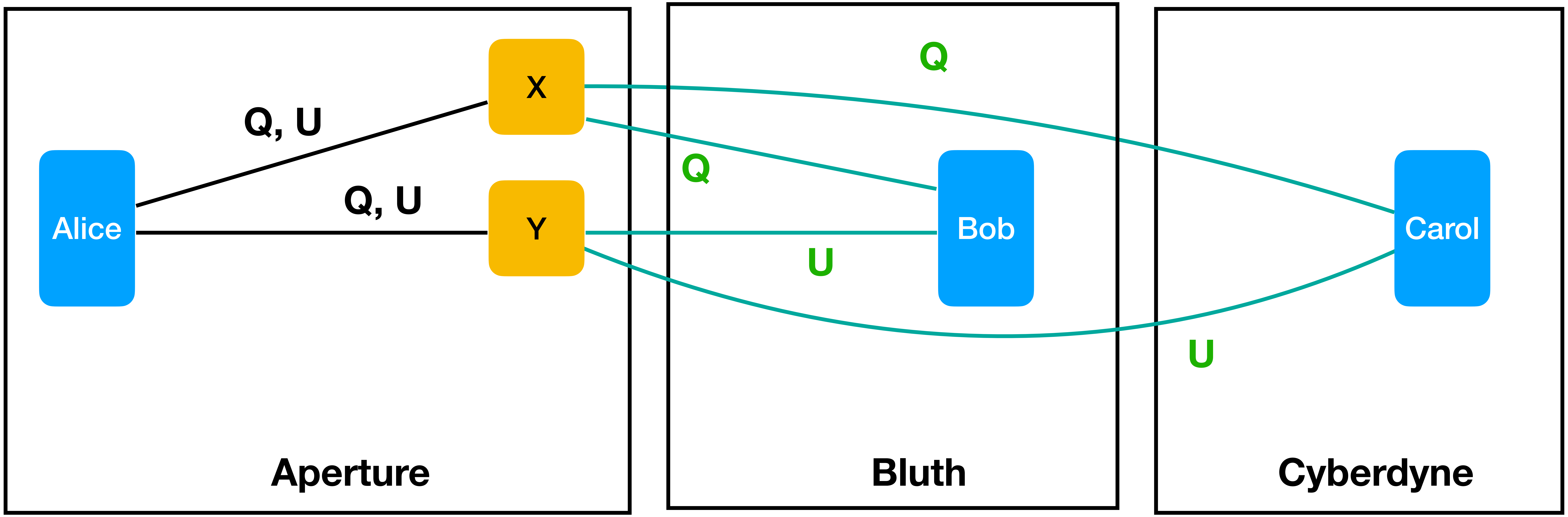
Excess Risk



Bob has permissions he doesn’t need.

Hazard -Transitive Access 1

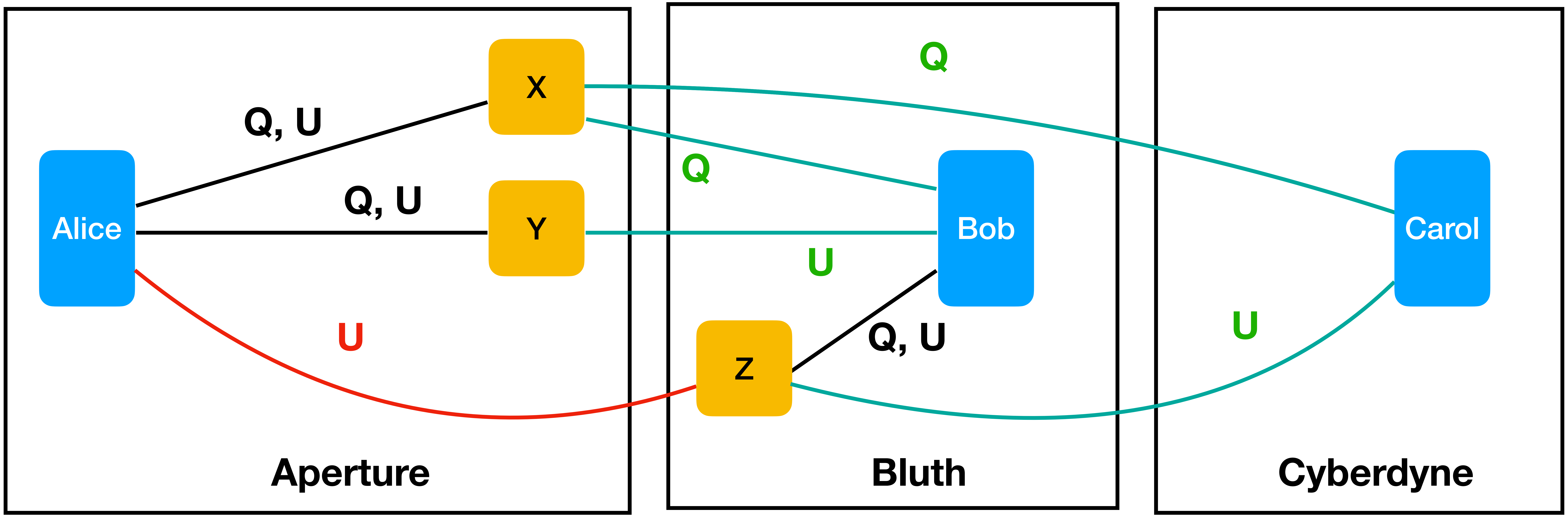
Audit failure



Alice never heard of Carol

Hazard -Transitive Access 1

Confused deputy

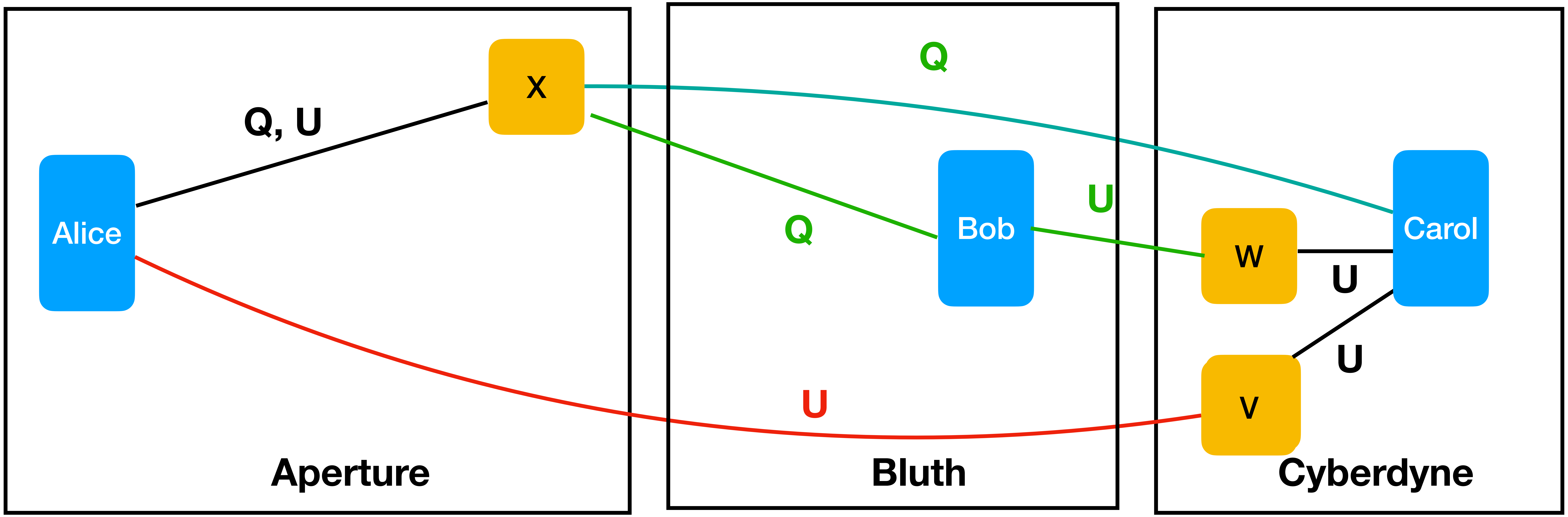


Alice: Bob, backup X to Z

Alice asked but Carol did it

Transitive Access 2

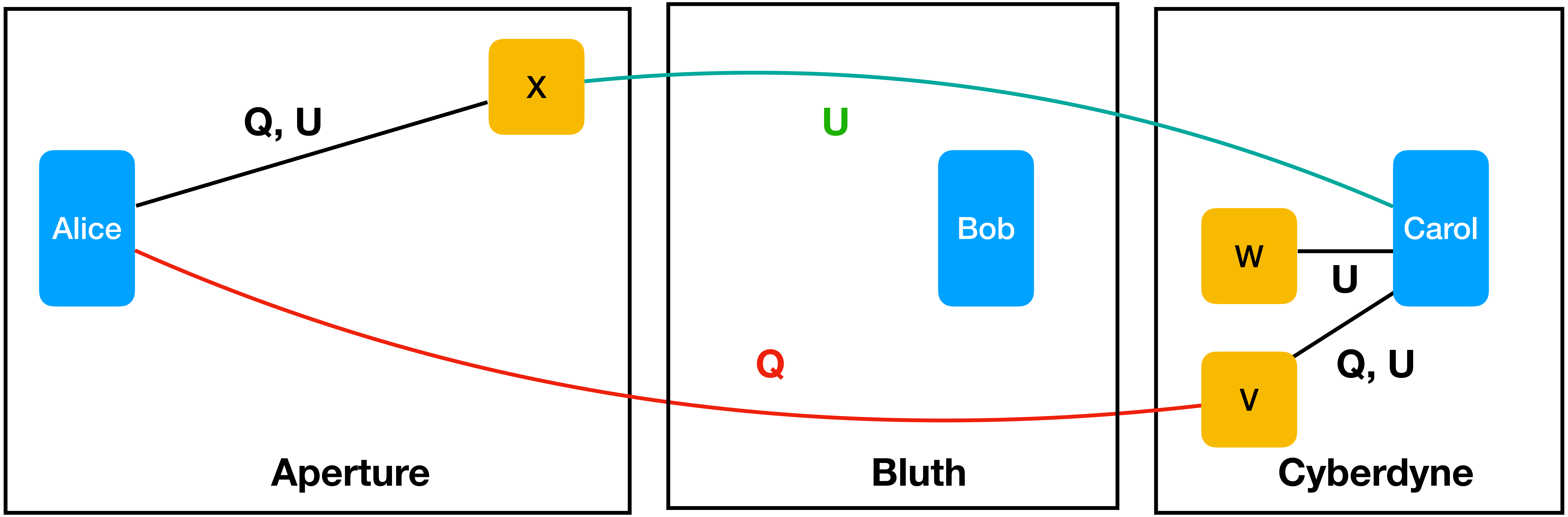
Expect Alice to say backup X to W



Alice manages to update a resourse at Cyberdyne

Hazard -Transitive Access 3

Alice says copy V to X



Alice can see any of Carol’s resources

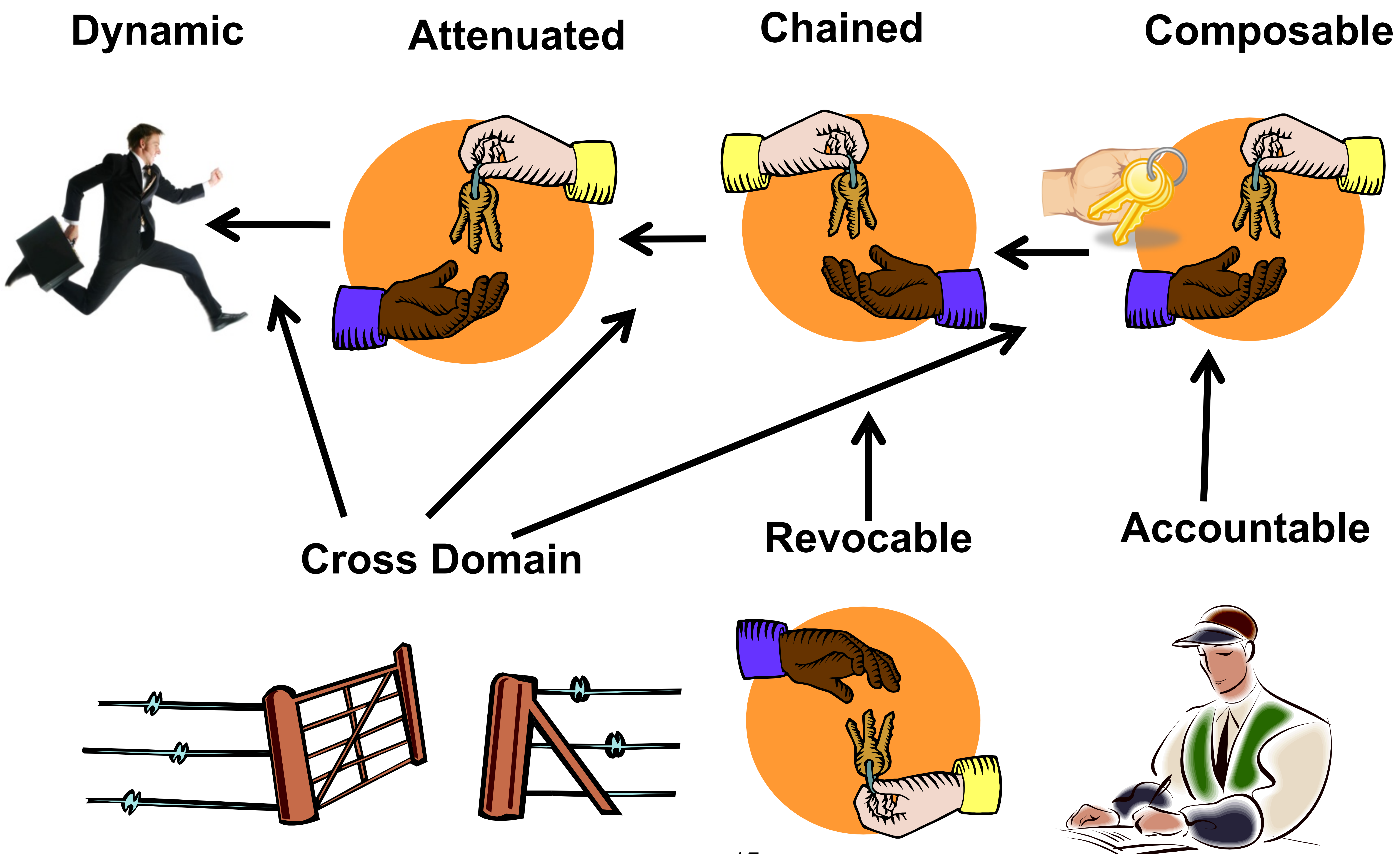
Lessons

- Incomplete set of use cases leads to unhandled hazards
- Unhandled hazards lead to vulnerabilities and usability issues
- Most hazards hard to address with authentication centric IAM
- Don’t start your design with identity

What to Do About It

Why Is IAM So Hard?

Seven Aspects of Sharing



Dynamic

Attenuated

Chained

Composable

Cross Domain

Revocable

Accountable

What Is Access Control?

Step	Action
Identify	Assign a responsible party
Authorize	Specify an access policy
Authenticate	Prove the right to use a specific policy
Decide	Should a request be honored

When
Before request
With request

Where
User domain
Service domain

What Is Access Control?

Step	AuthN	
	Where	When
Identify	User domain	Before request
Authorize	Service domain	Before request
Authenticate	Service domain	With request
Decide	Service domain	With request

What Is Access Control?

Step	AuthN		Step	AuthZ	
	Where	When		Where	When
Identify	User domain	Before request	Identify	User domain	Before request
Authorize	Service domain	Before request	Authenticate	User domain	Before request
Authenticate	Service domain	With request	Authorize	User domain	Before request
Decide	Service domain	With request	Decide	Service domain	With request

Capability

A capability is an

unforgeable

transferable

permission

to use the thing it designates.

Have You Ever Used a Capability?

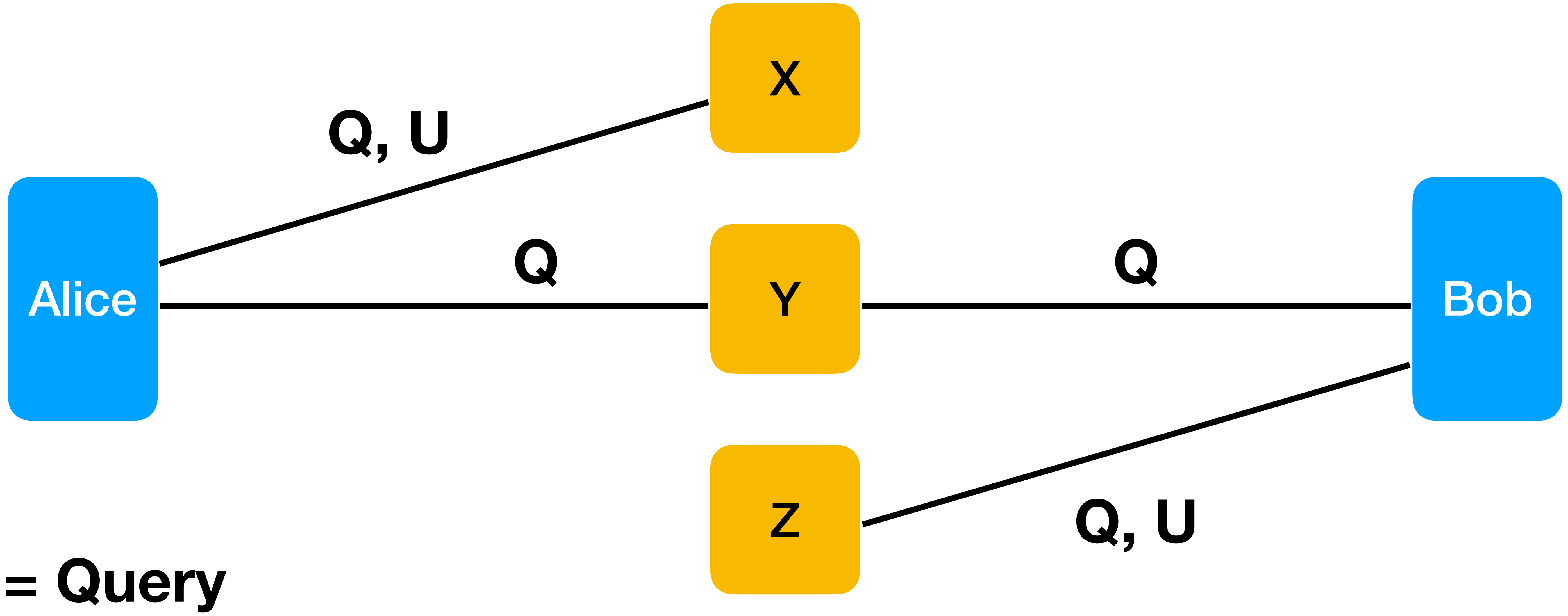
Have You Ever Used a Capability?

First implemented for computers in 1968

Use Cases Revisited

Basic Access Control

Different users have different permissions

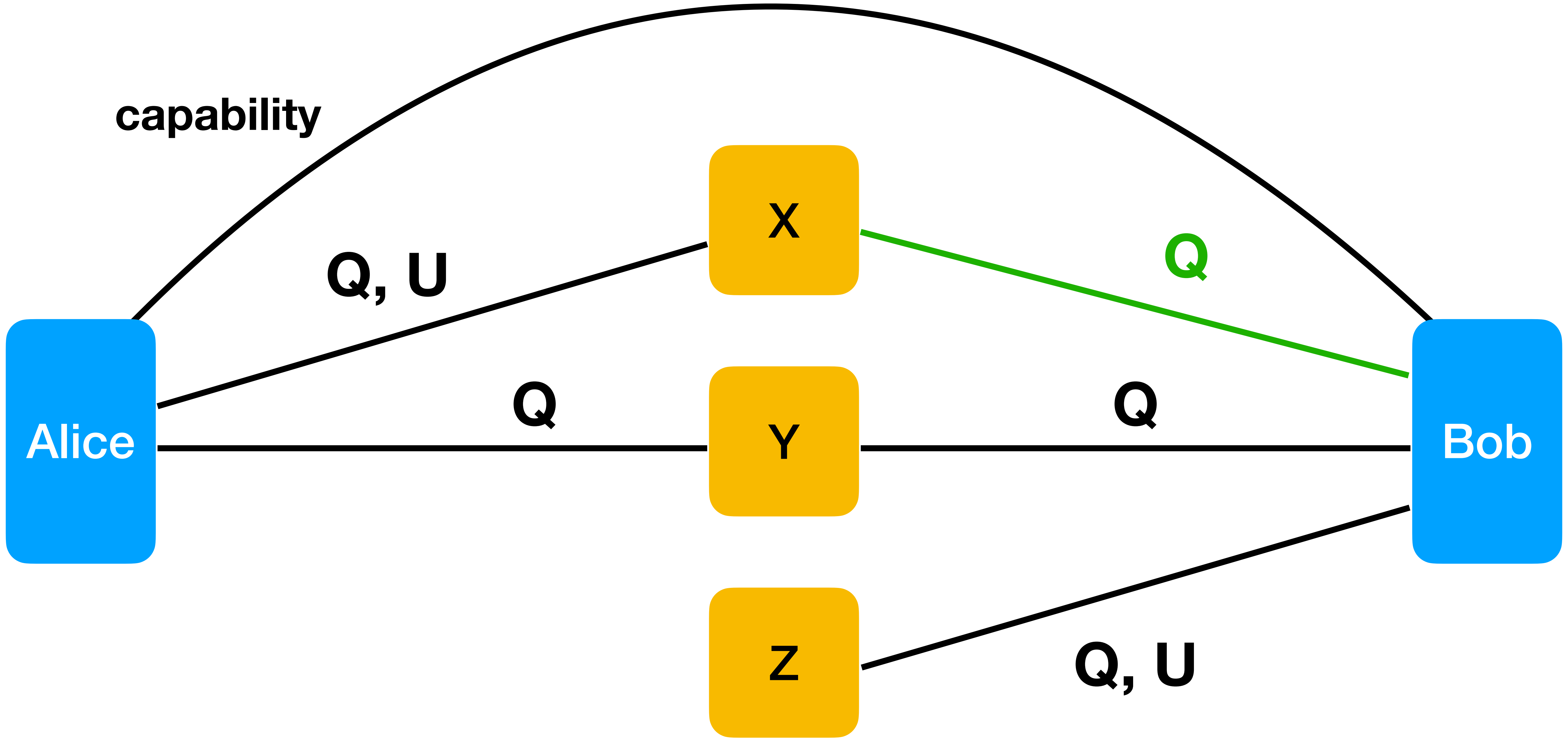


Q = Query

U = Update

No Hazard - Credential Sharing

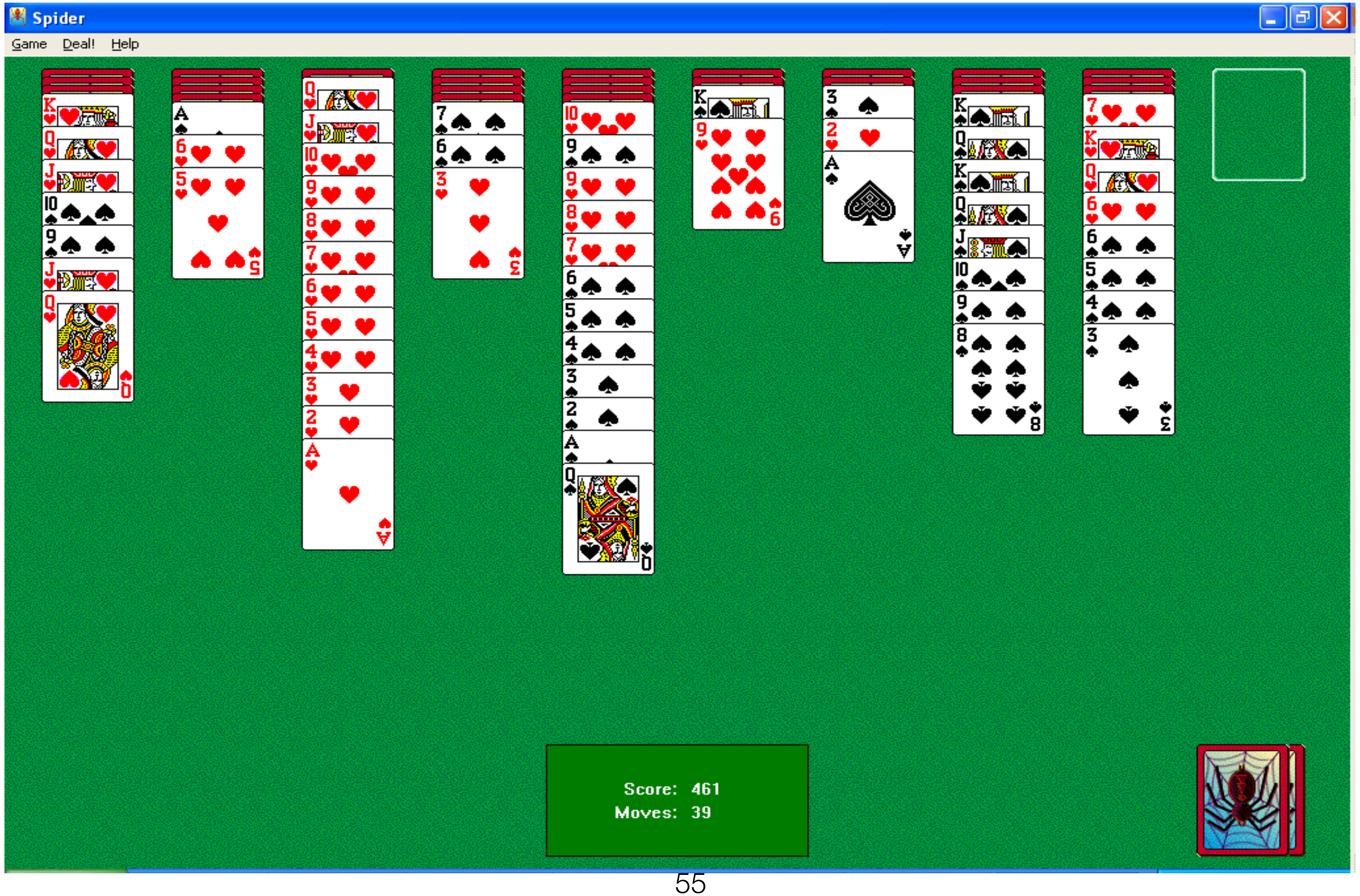
Alice wants Bob to help with X



No need to share credentials

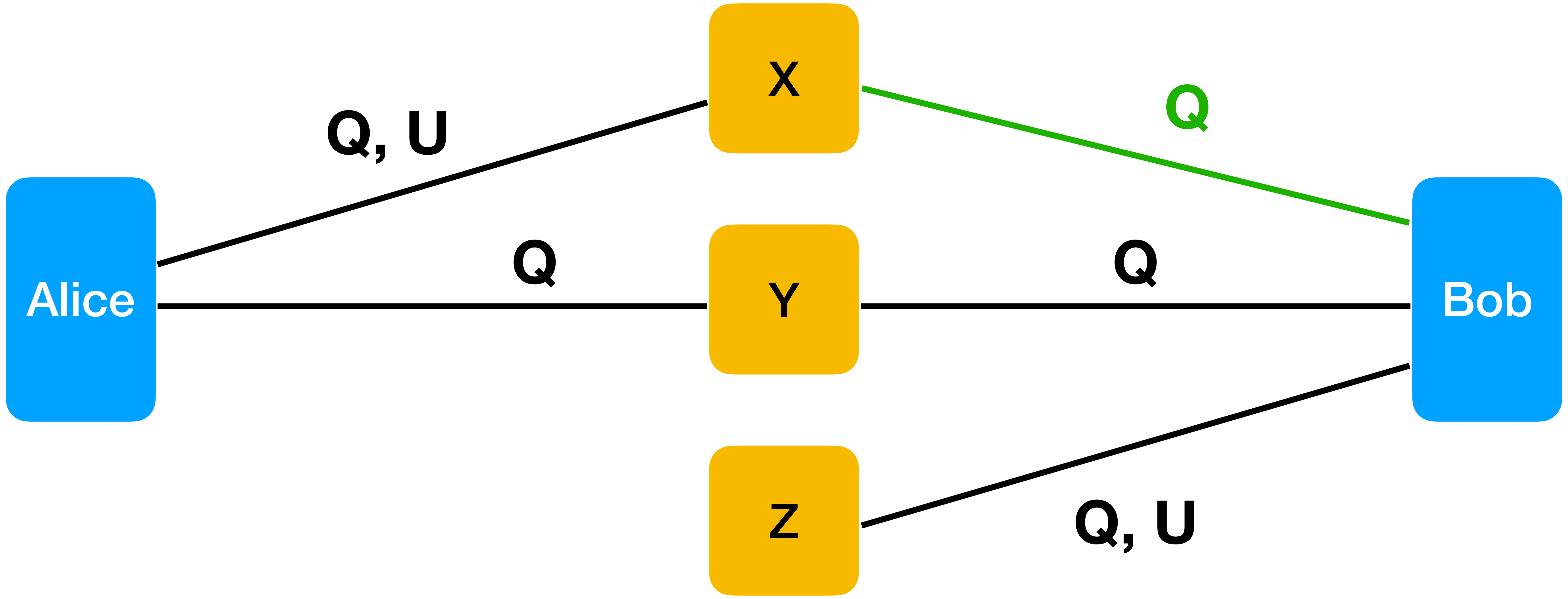
Hazard - Impersonation

Least Privilege



Hazard - Availability

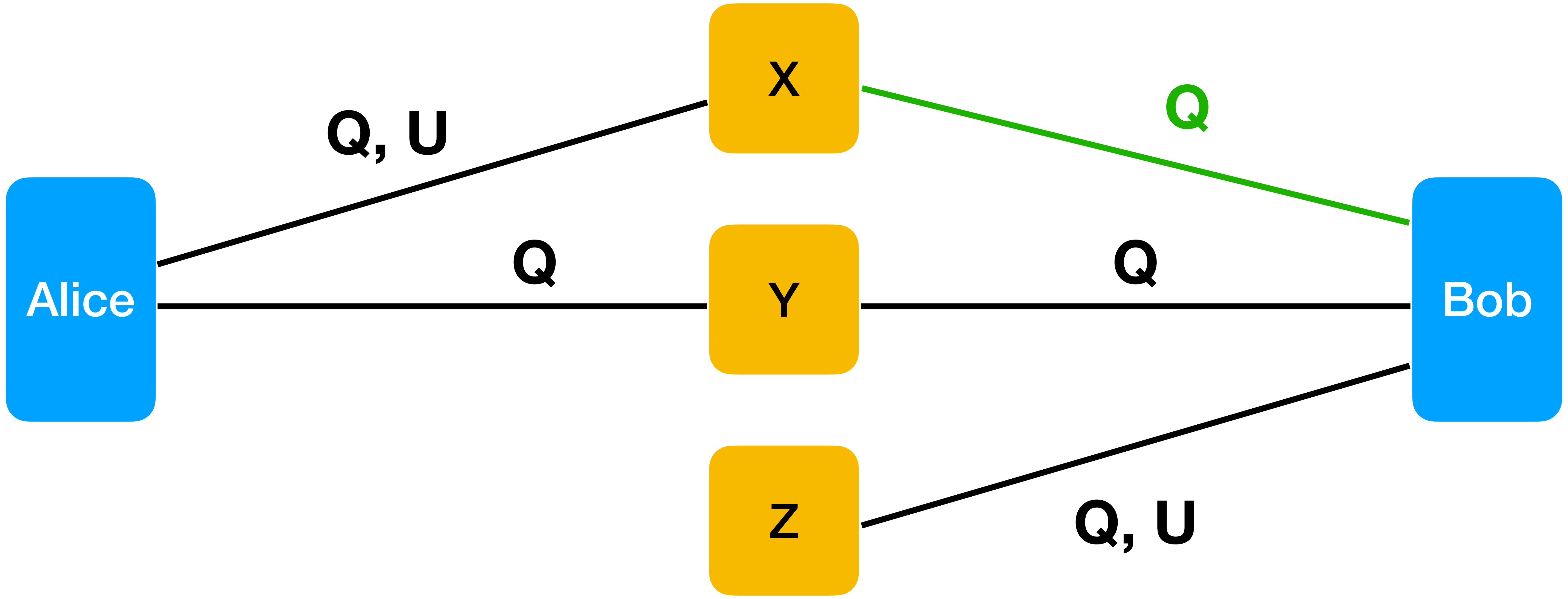
Alice can do the delegation



Alice can create a new capability
from an existing one

Hazard - Responsibility

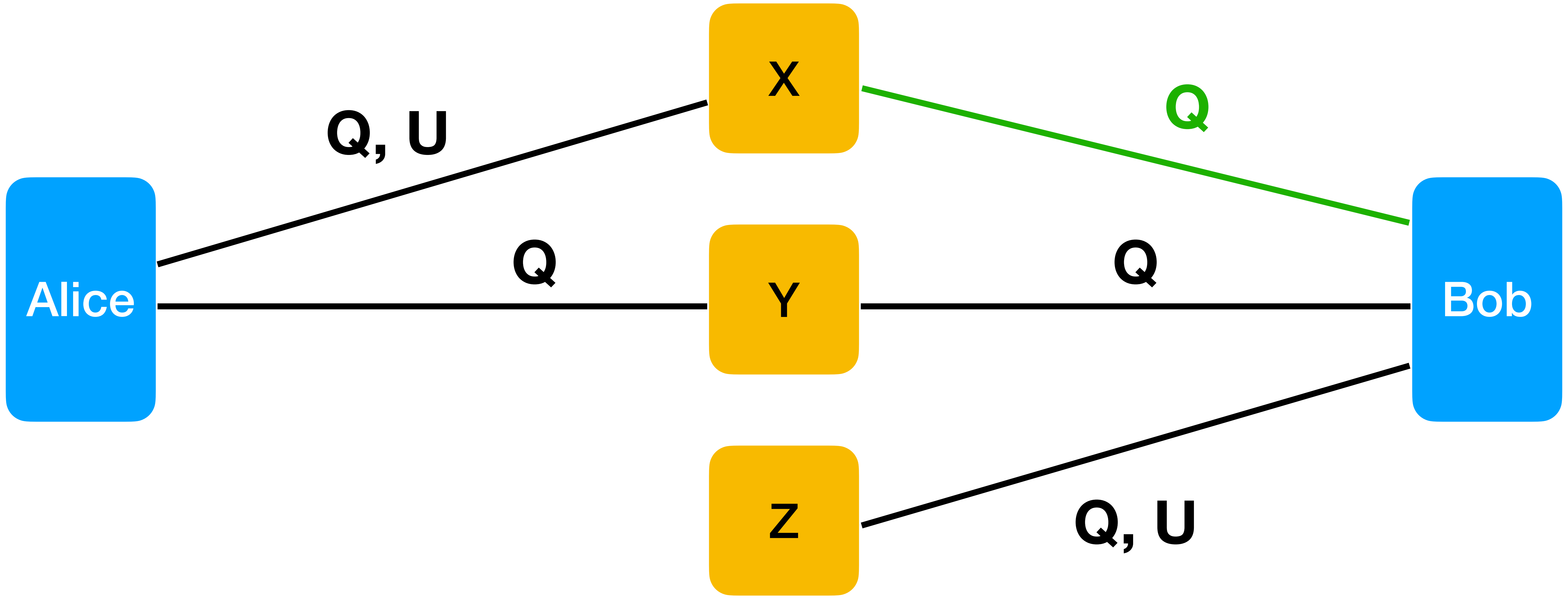
What if Bob does something bad?



Capability includes delegation chain

Hazard - Conditional Policy

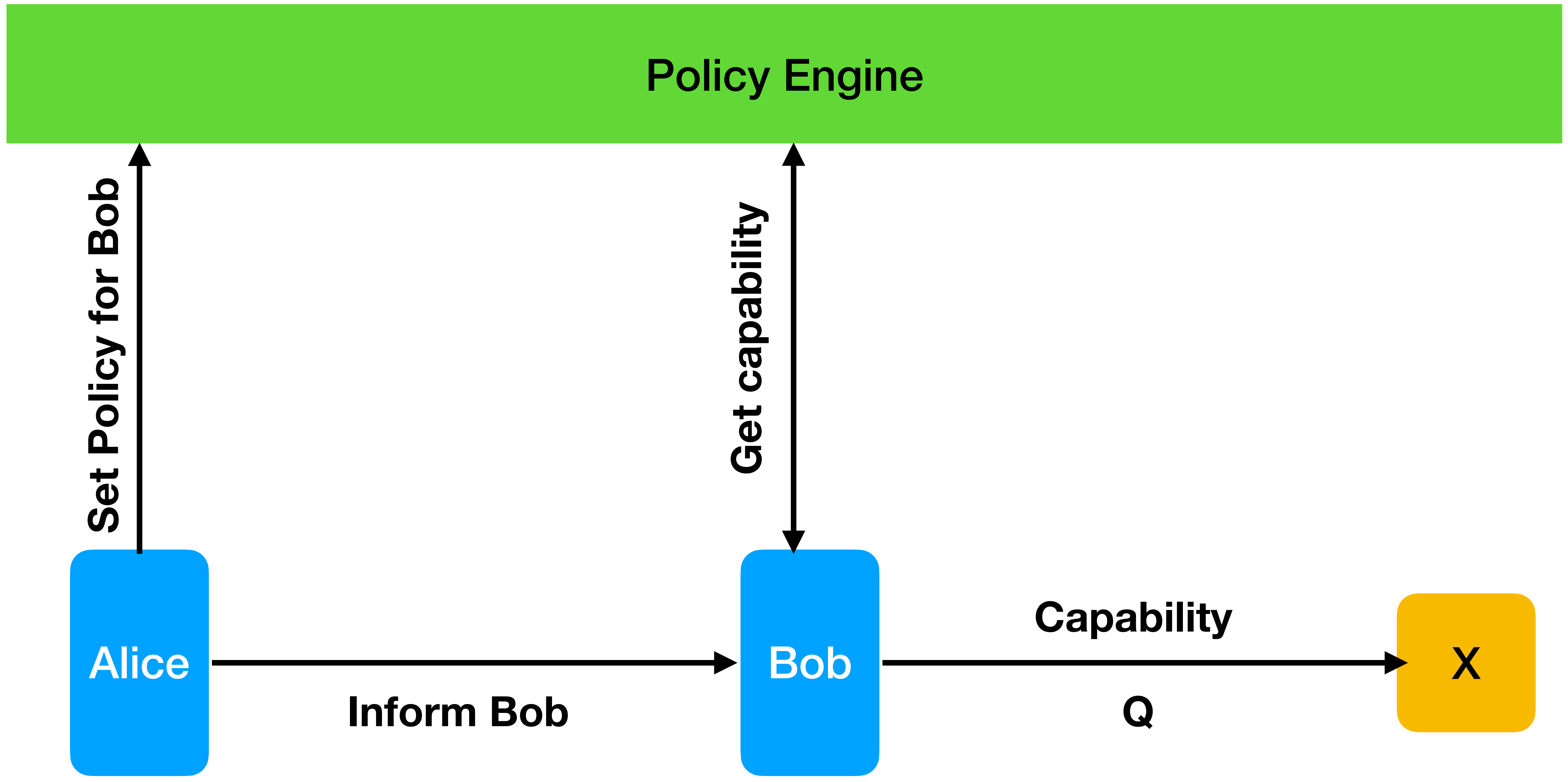
Alice wants to limit Bob's access to working hours.



Capability certificate includes policy conditions.

No Hazard - Performance

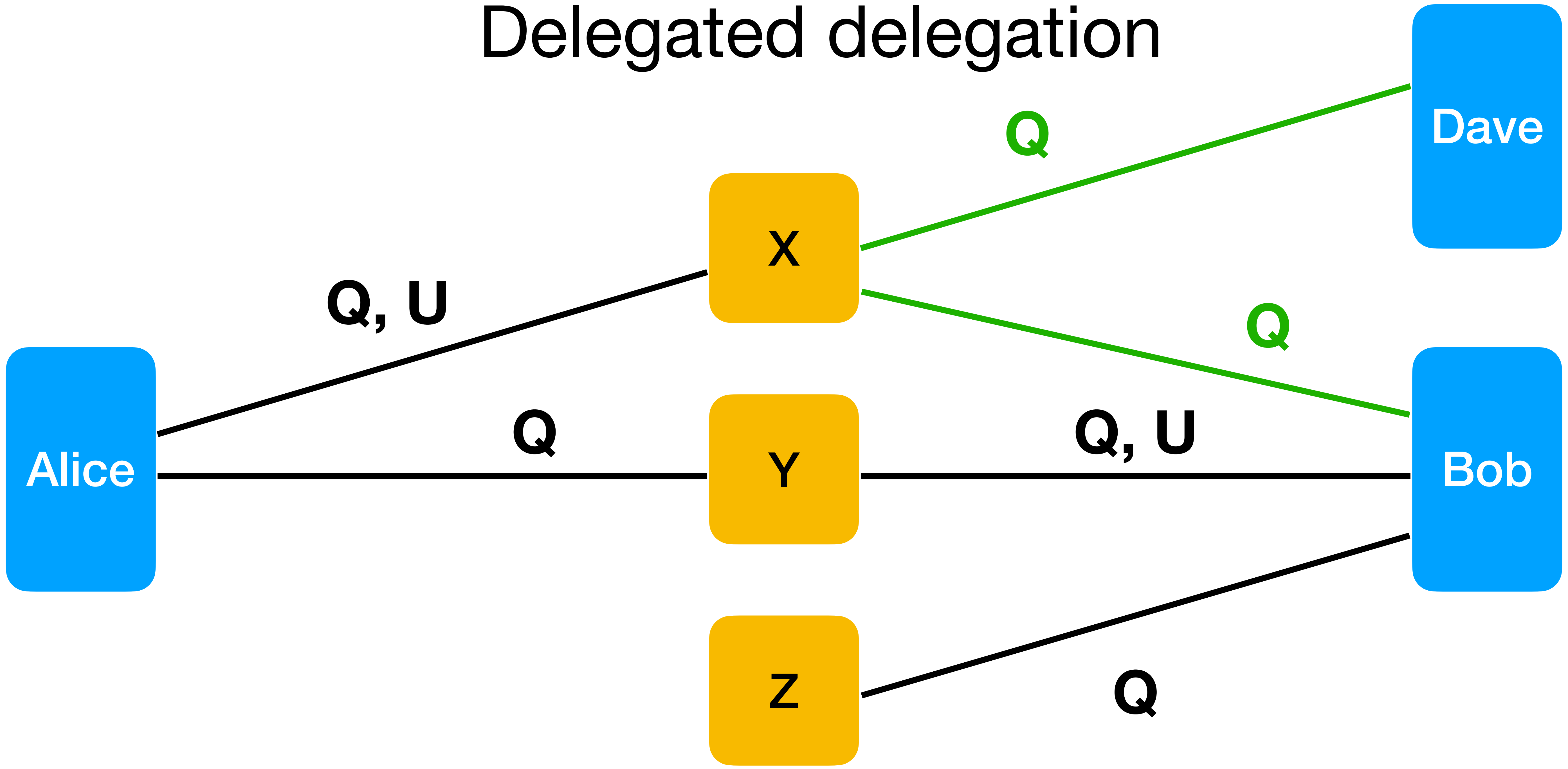
Delegation in Cedar



Capability is cache of policy calculation

Chained Delegations

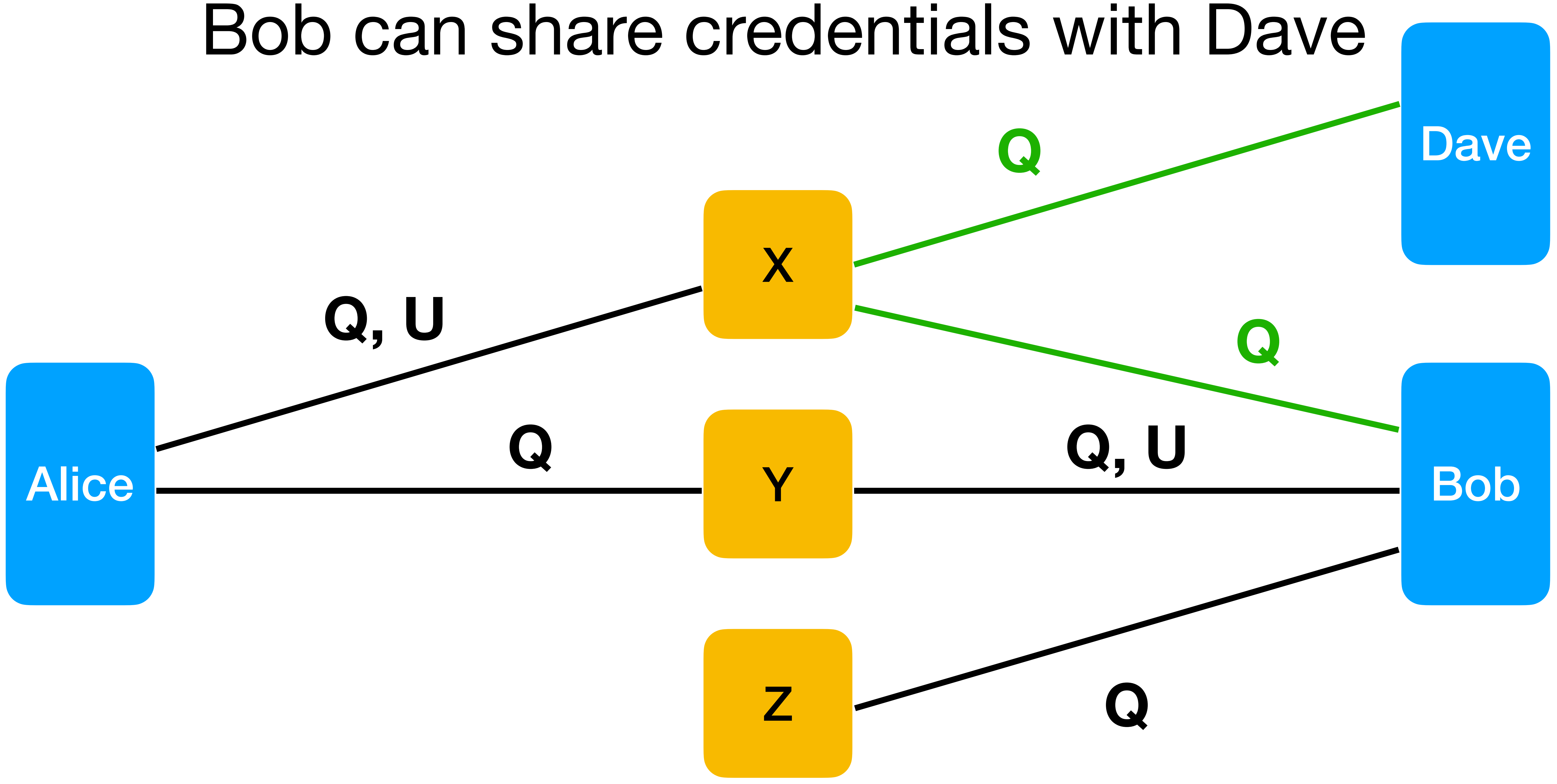
Delegated delegation



Bob delegates a delegated
permission to Dave

Hazard -“Oh, no. You’ve lost control!”

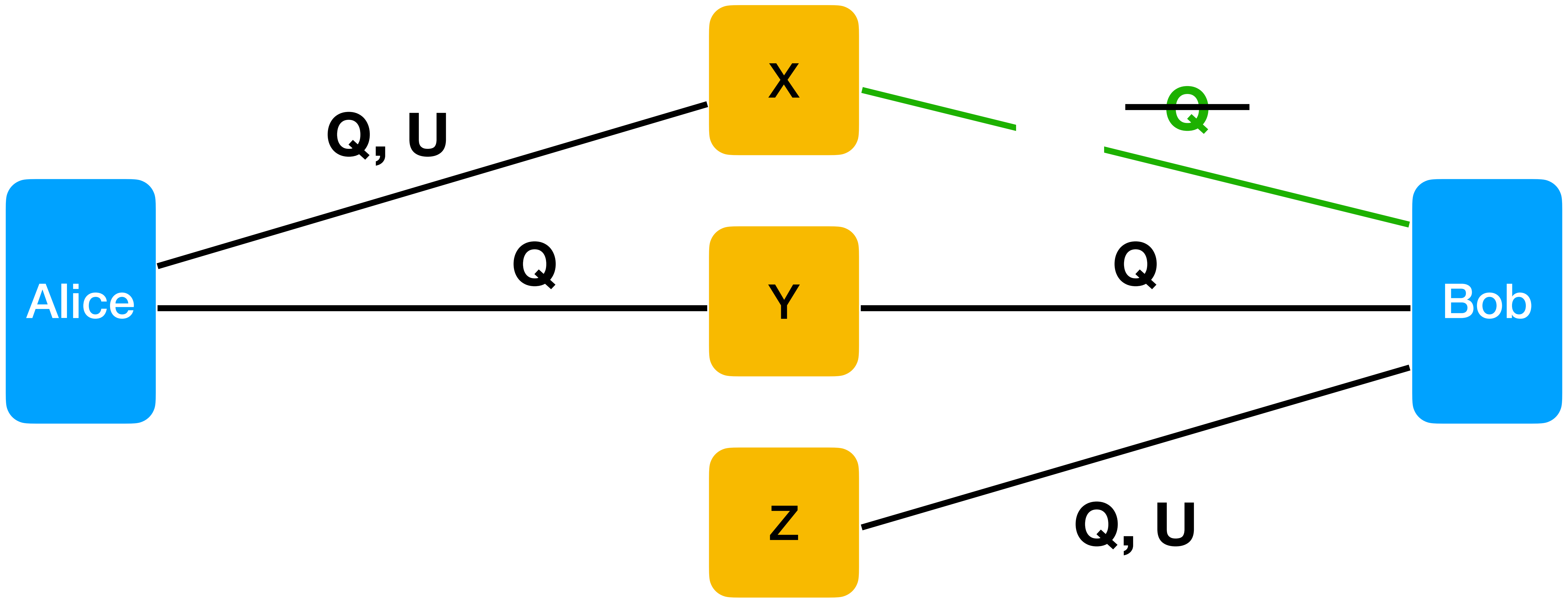
Bob can share credentials with Dave



Don’t prohibit what you can’t prevent

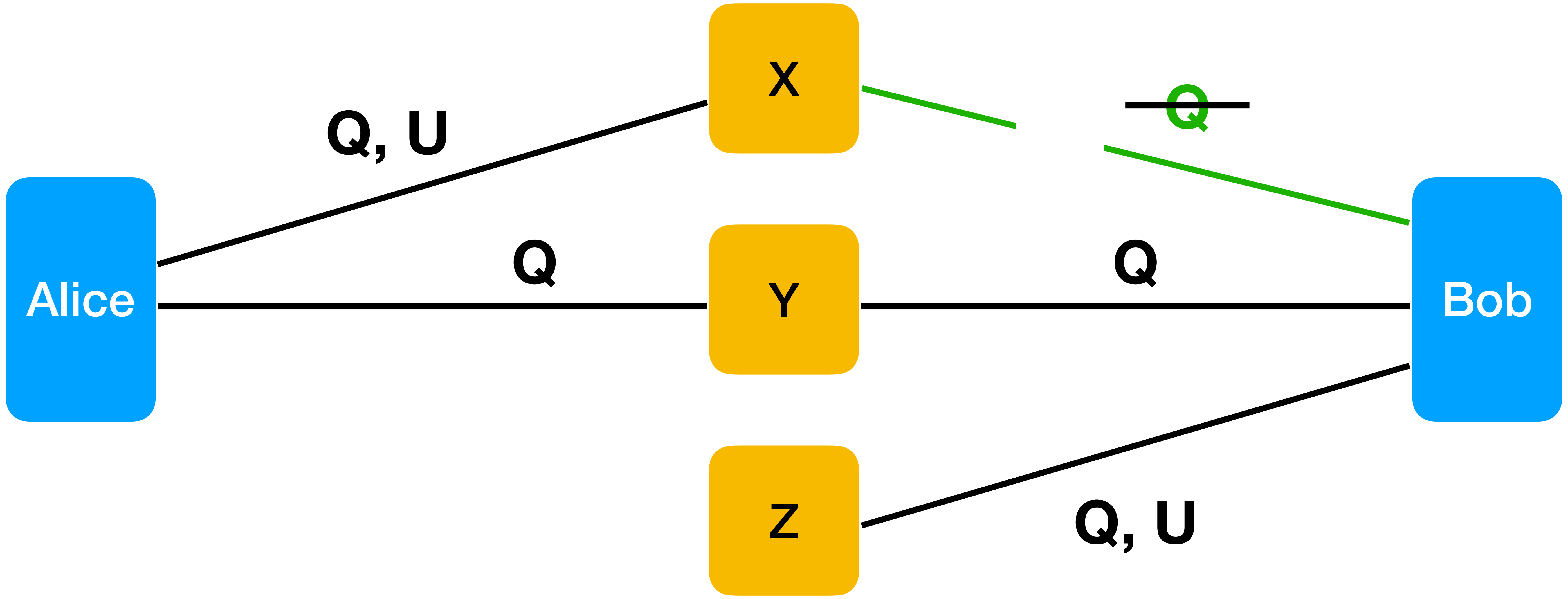
Revocation

Delegator wants to revoke



Hazard - Permission to Revoke

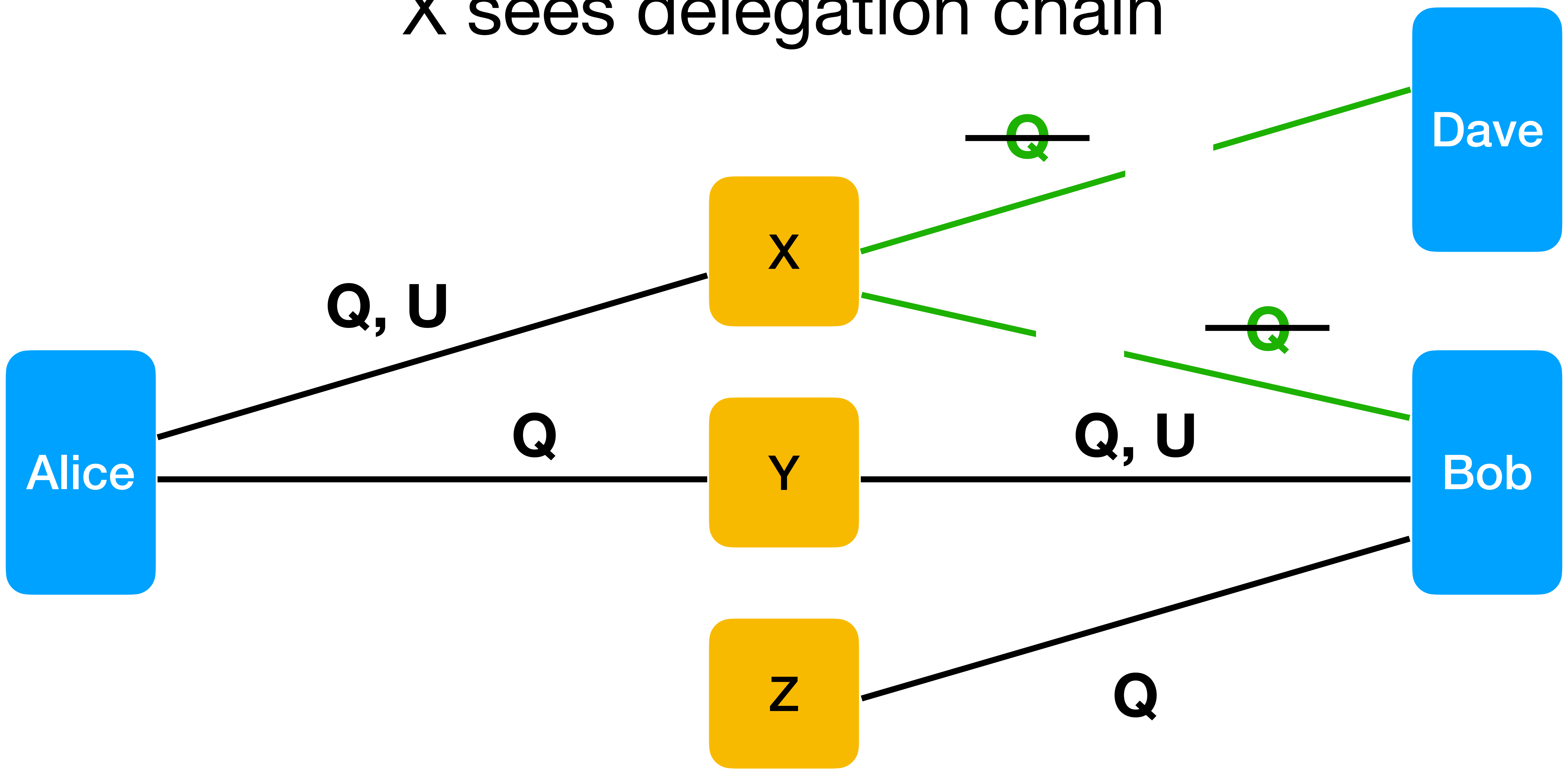
Who can revoke? Delegator



Alice to X: “Don’t honor Bob’s capability.”

Hazard - Sock Puppet

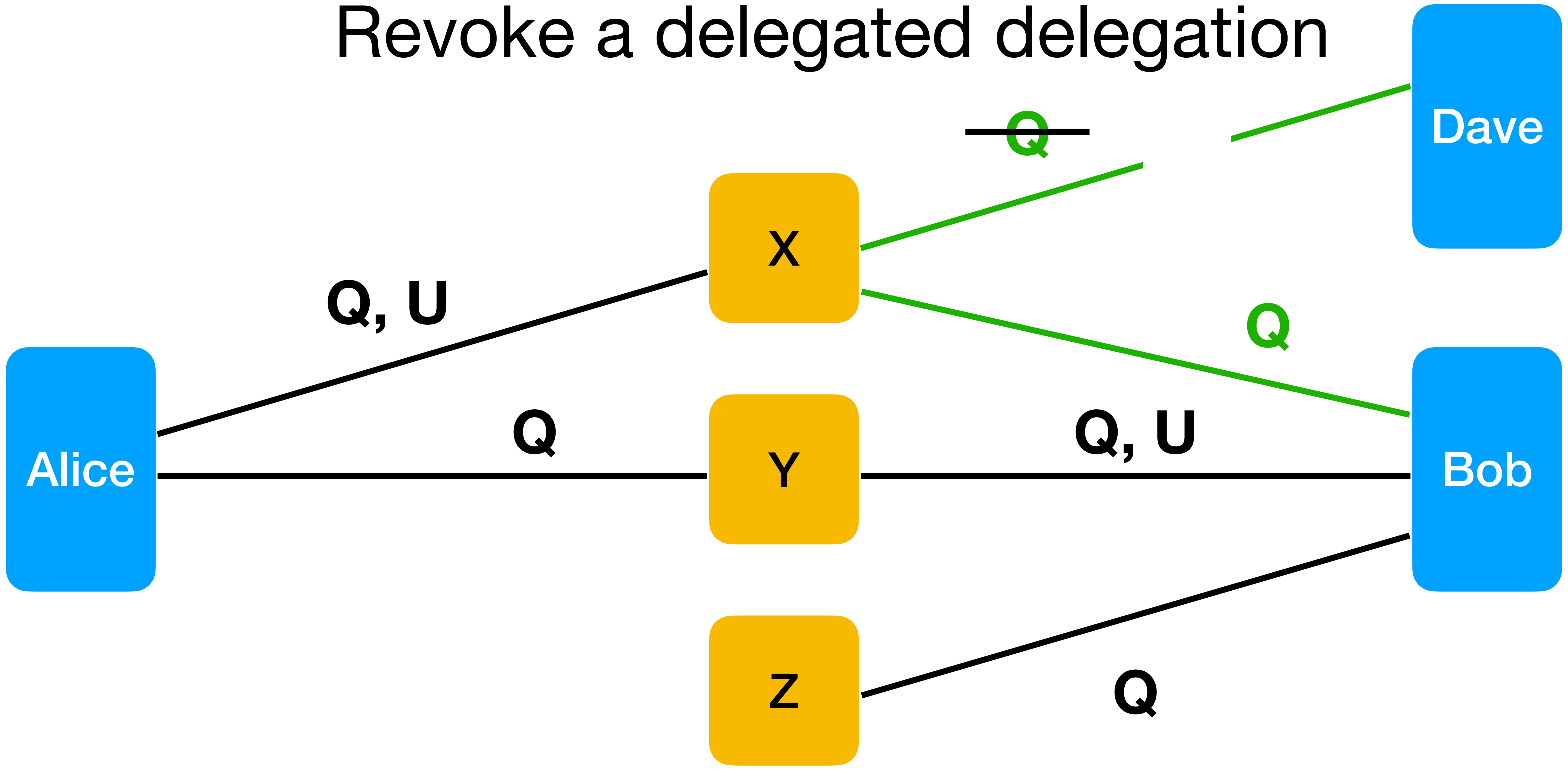
X sees delegation chain



Can revoke all downstream delegations

Hazard - Skip Revocation

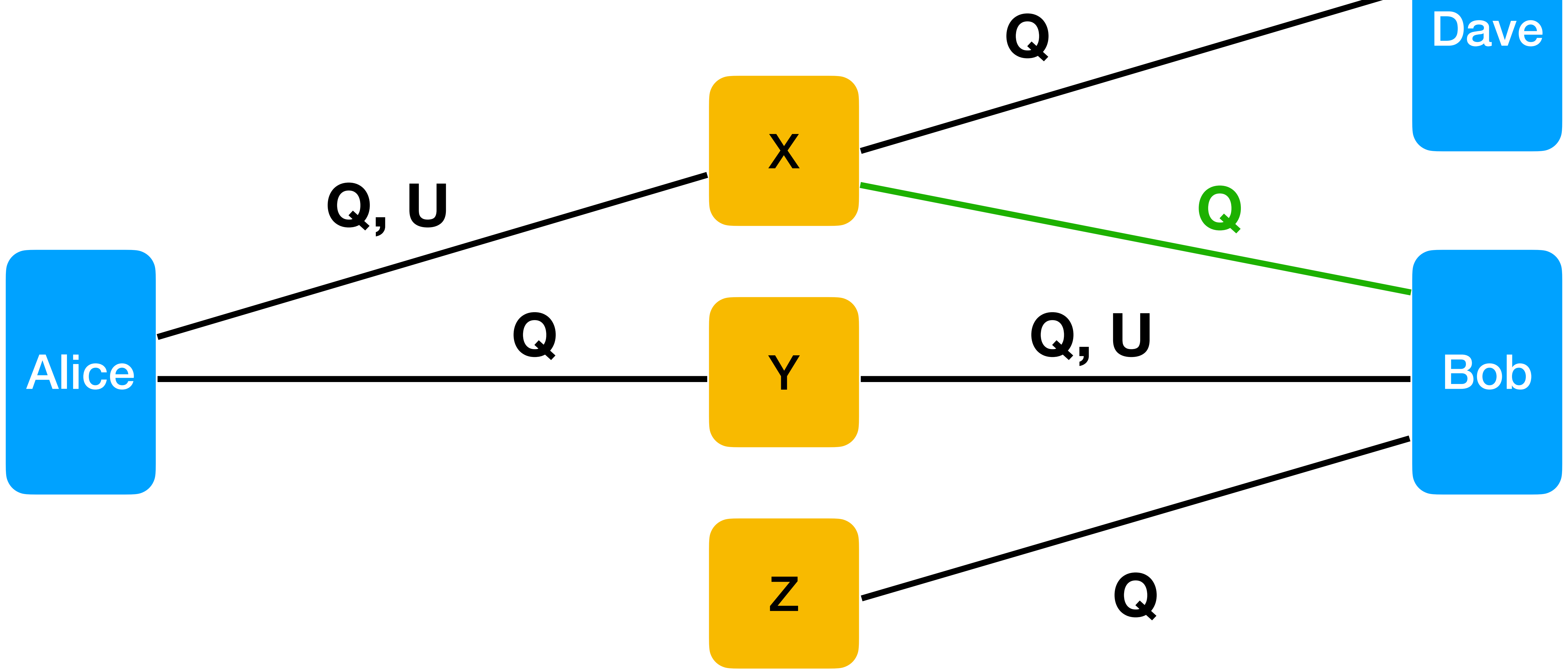
Revoke a delegated delegation



Bob delegates revoke permission to Alice

Independent Delegations

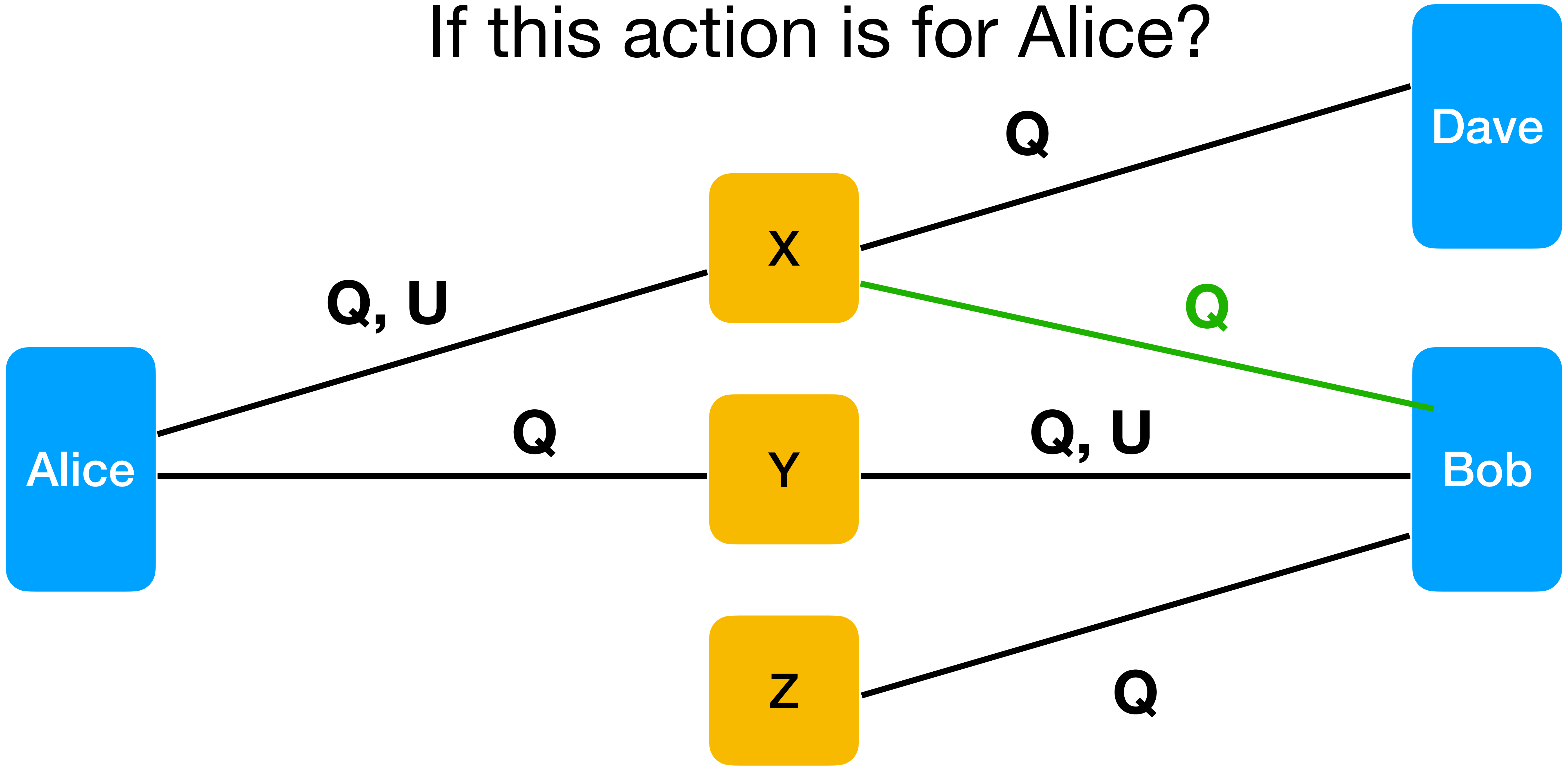
Overlapping delegations



Alice and Dave both delegate
access to X to Bob

Hazard - Accounting

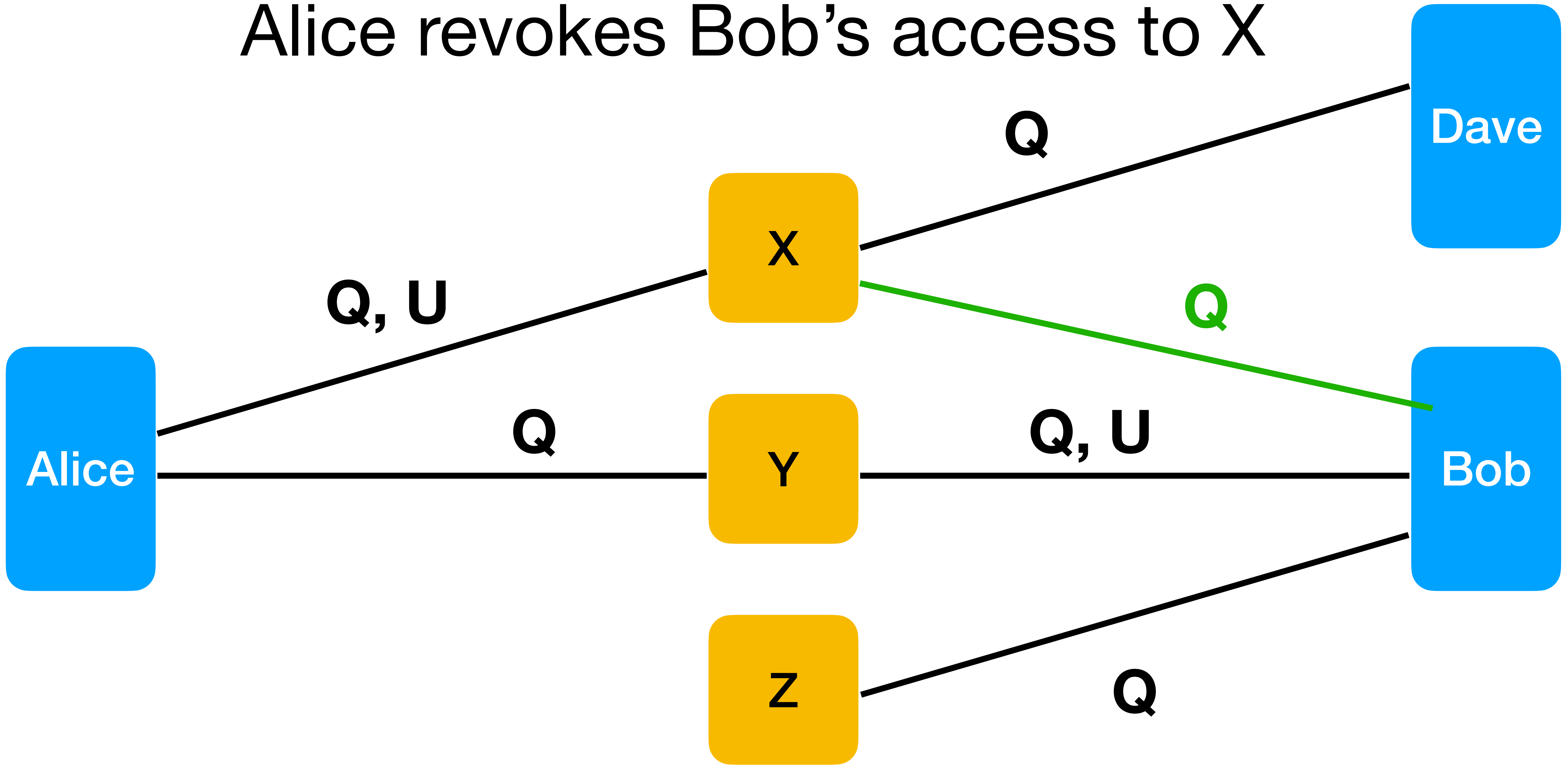
If this action is for Alice?



Bob uses the capability from Alice

Hazard - Lost Delegation

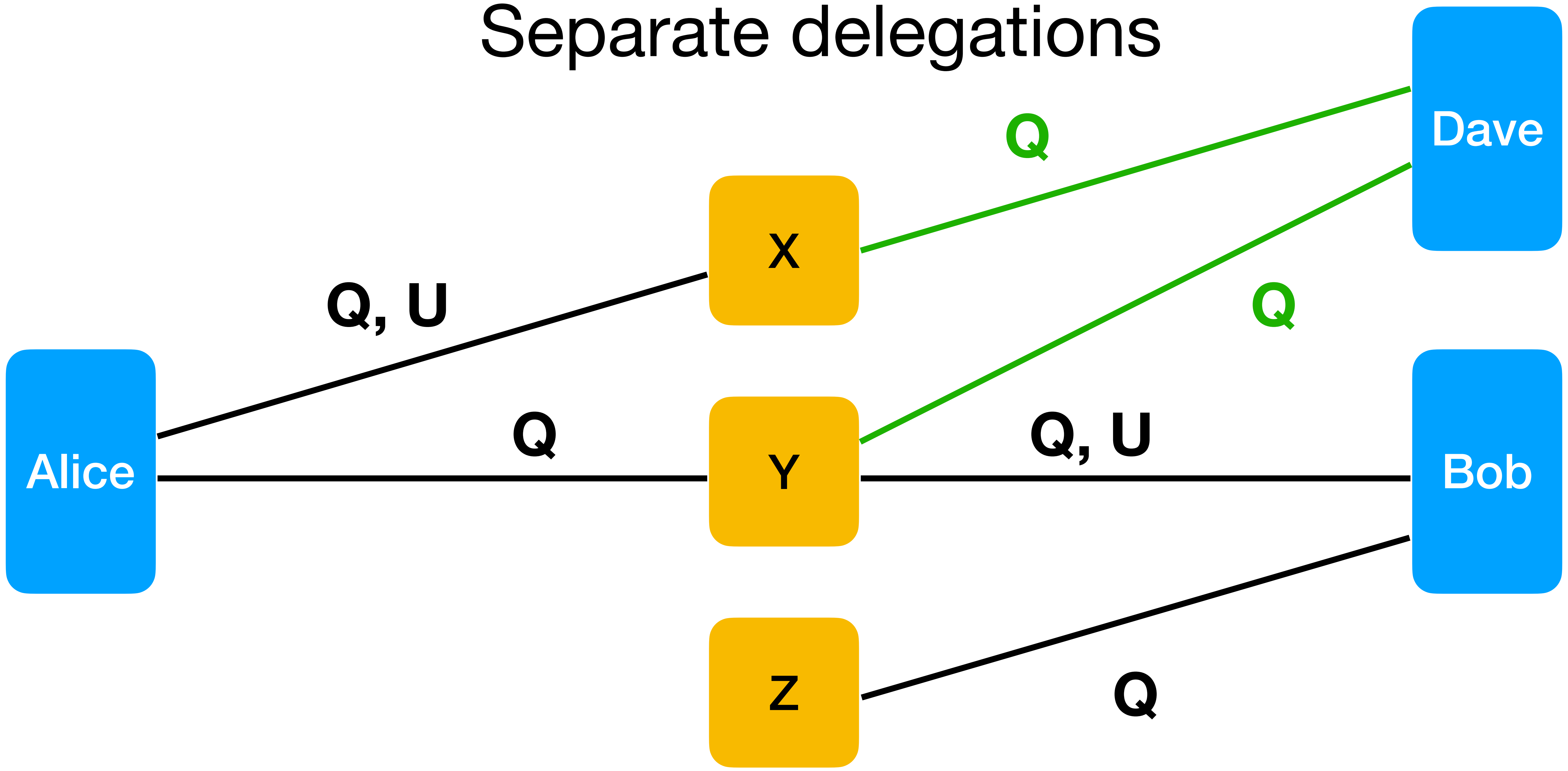
Alice revokes Bob’s access to X



Bob has a separate capability from Dave

Composed Delegations

Separate delegations

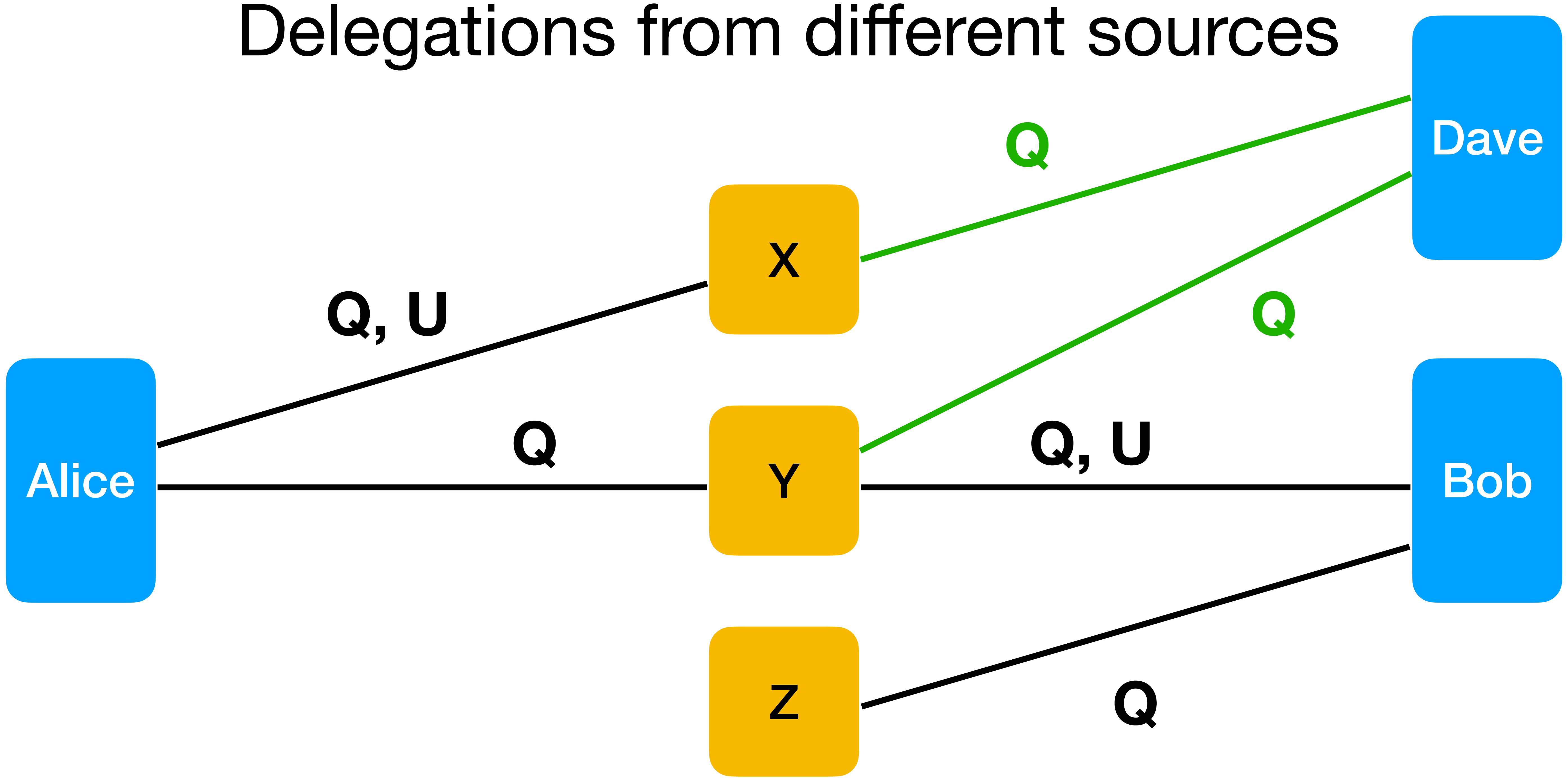


Alice delegates X to Dave

Bob delegates Y to Dave

Hazard - Composition

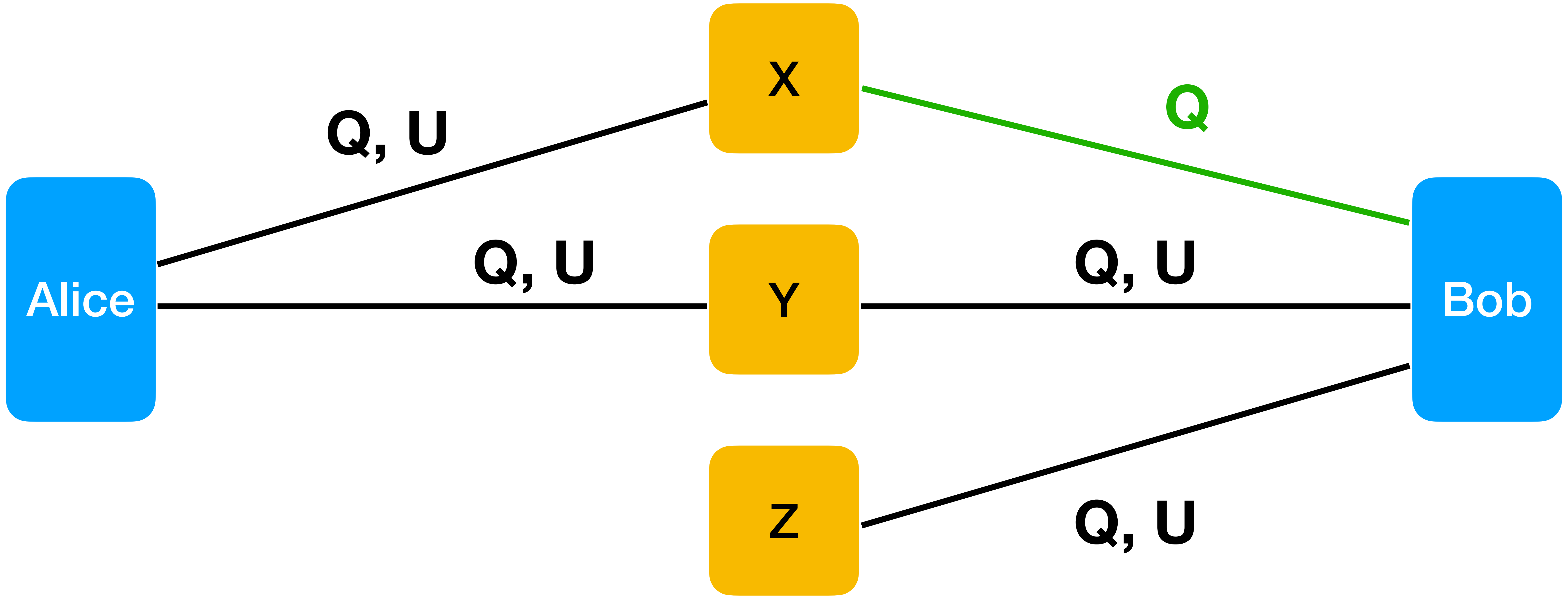
Delegations from different sources



Dave uses both capabilities in the request

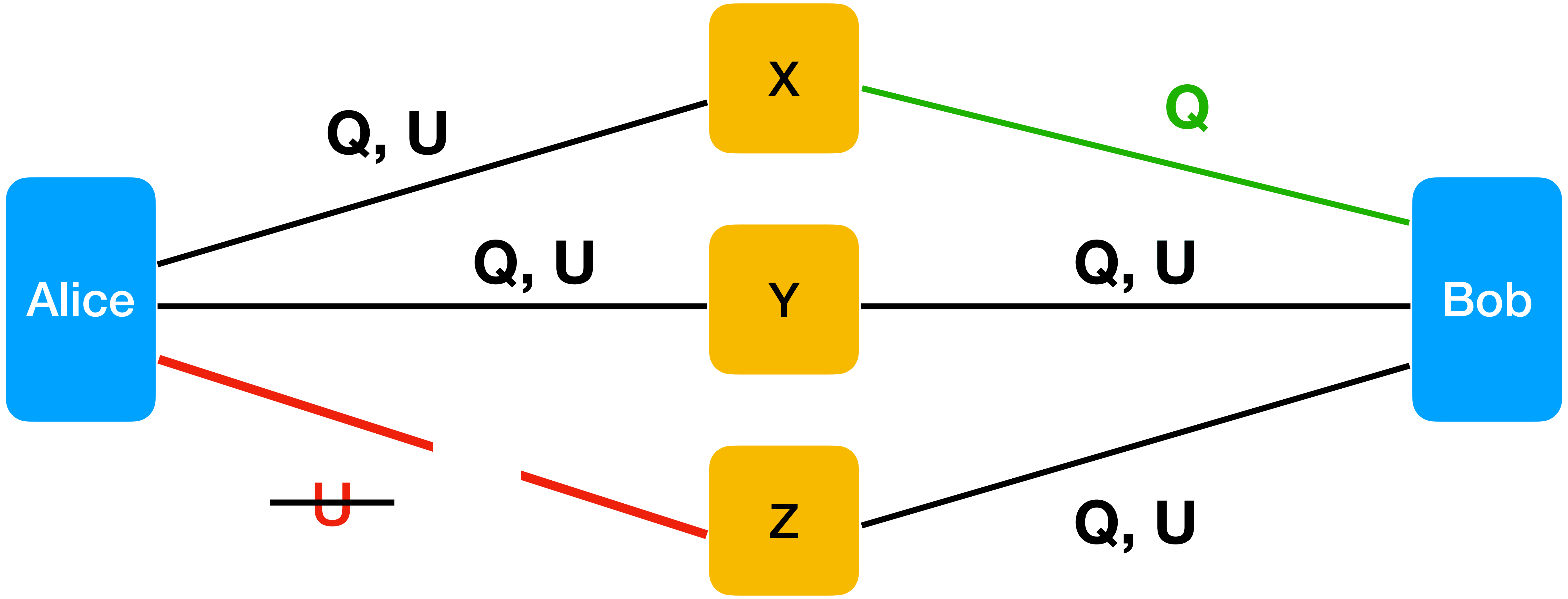
Multiple API Arguments

Bob expects, “Process X and put the output in Y.”



Hazard - Confused Deputy

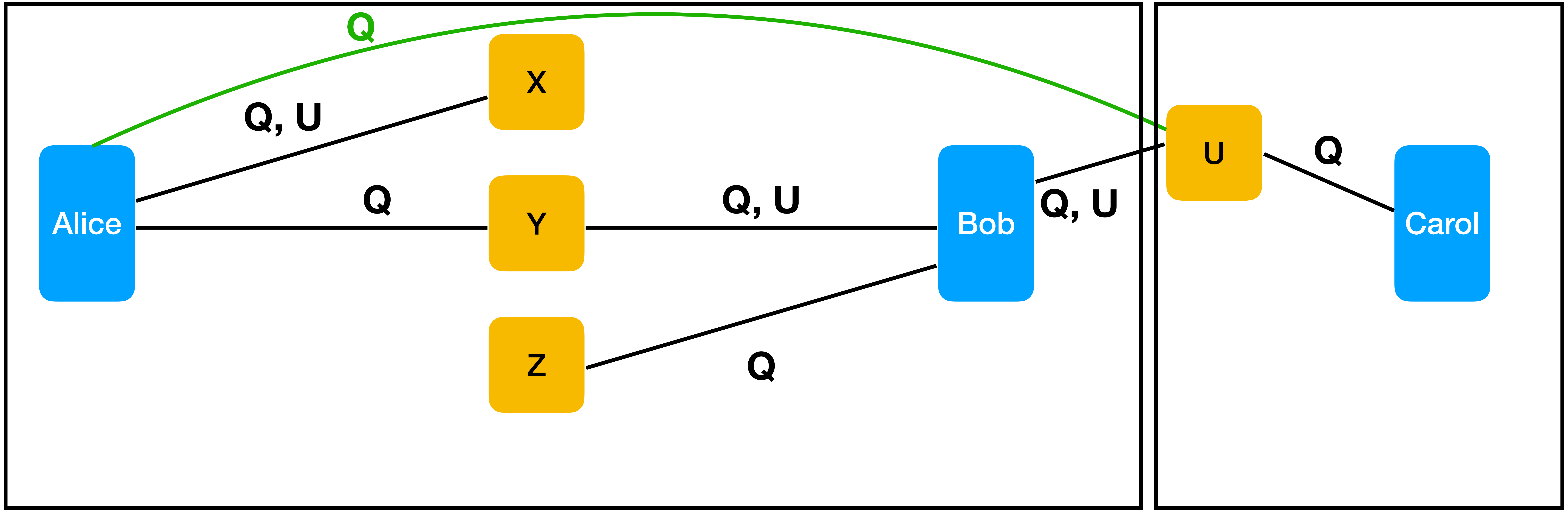
Alice actually says, “Process X and put the output in Z."



Alice doesn’t have U capability on Z

Cross Jurisdiction Delegation

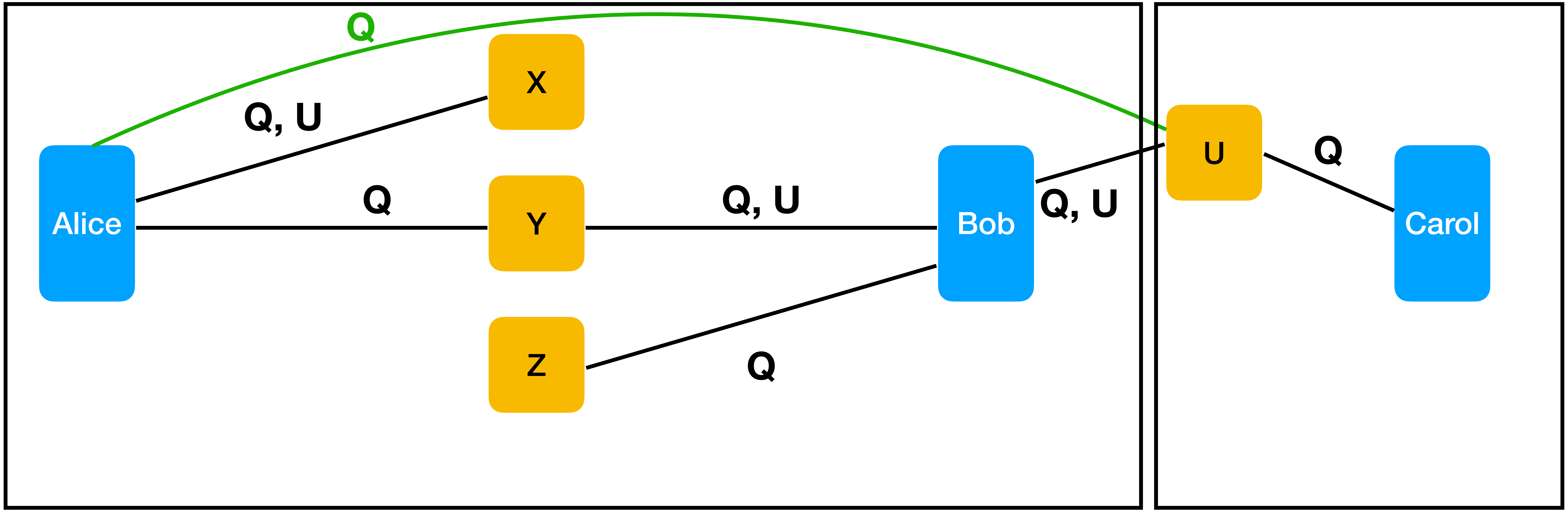
Different authentication domains



Bob delegates U to Alice

Hazard - Audit Failure

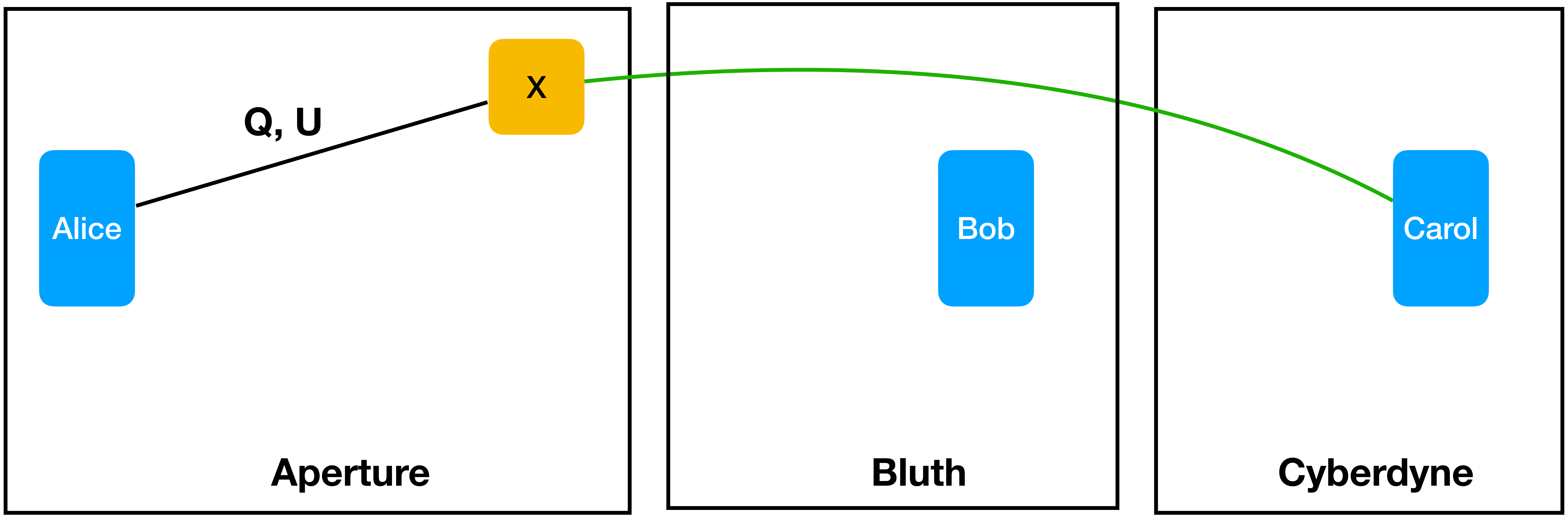
Carol has no idea who Alice is



Carol knows Bob delegated to Alice

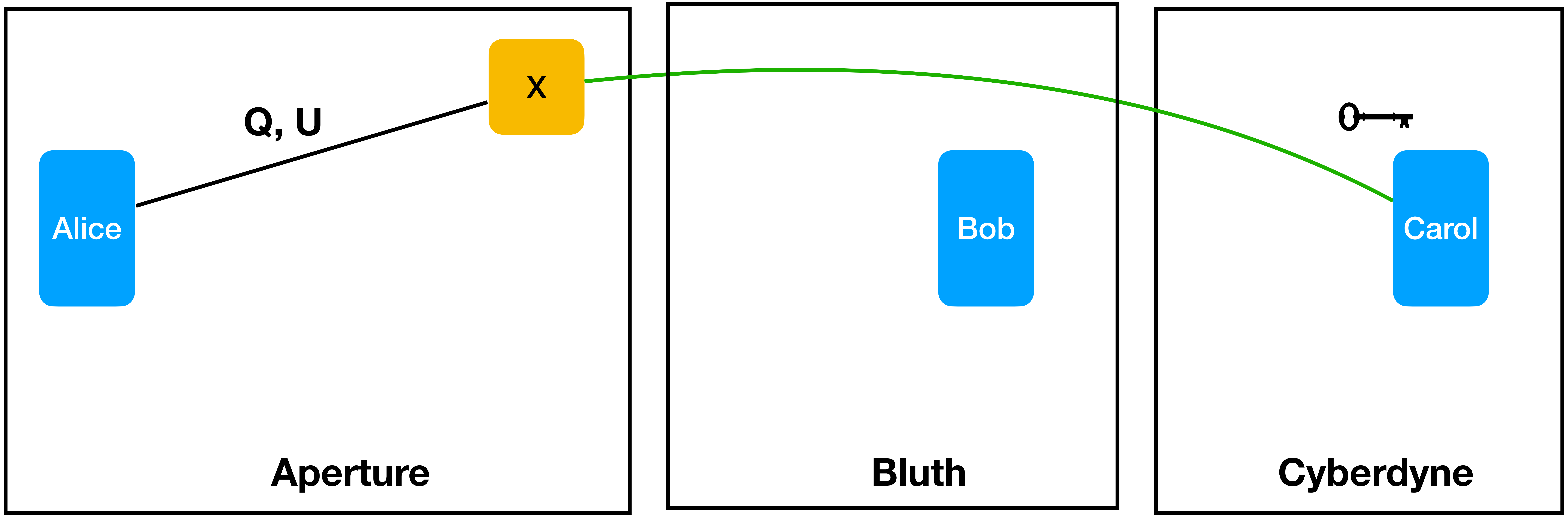
Oblivious Delegation

Alice delegates to Carol via Bob



Hazard - Untrused Intermediary

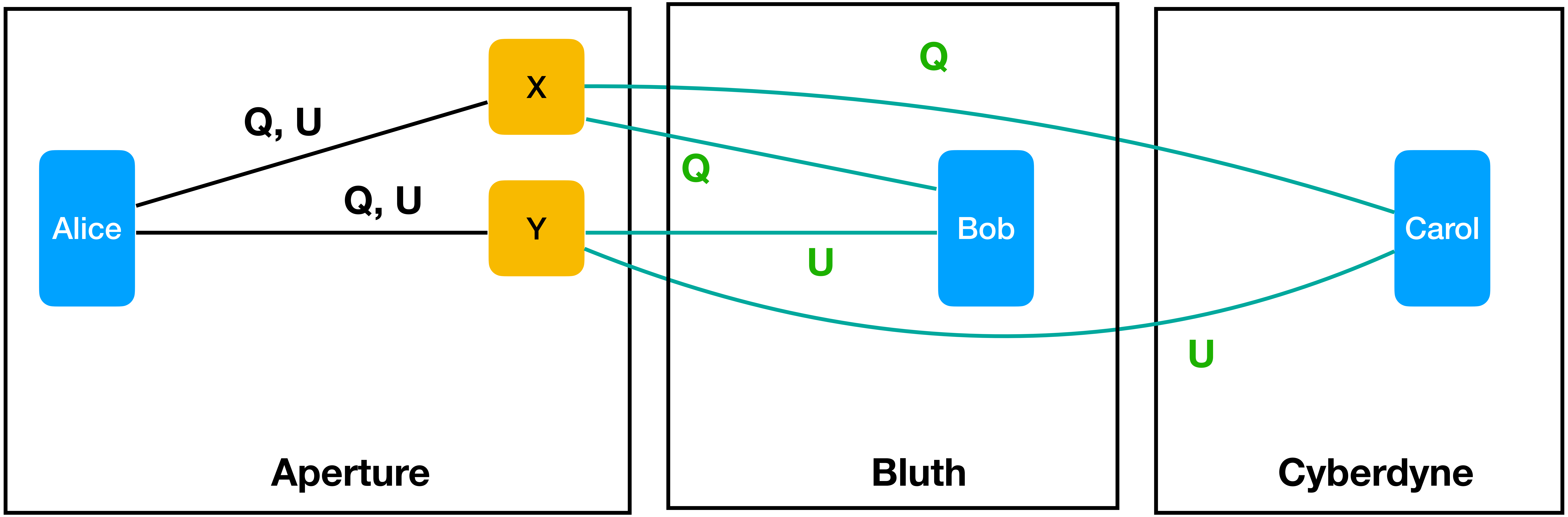
Alice delegates to Carol via Bob



Carol can use the capability; Bob can’t

Transitive Access 1

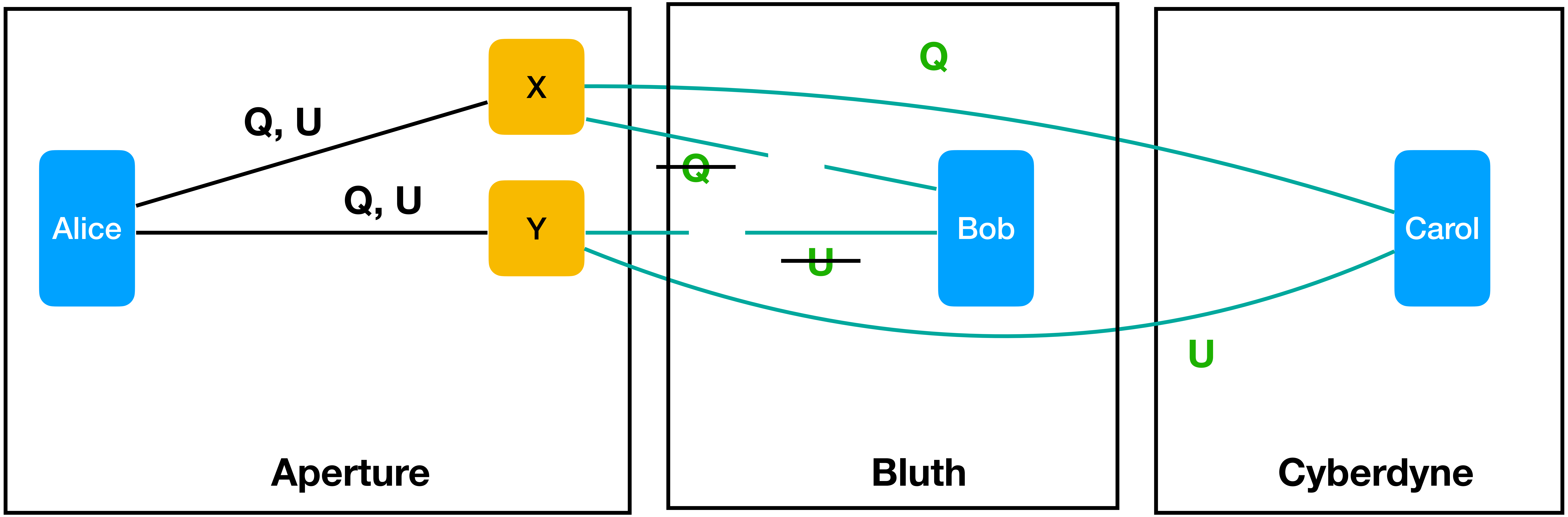
Alice: Bob, backup X to Y



Bob: Carol, copy X to Y

Hazard - Excess Authority

Alice: Bob, backup X to Y



Bob: Carol, copy X to Y

Lessons

- Incomplete set of use cases leads to unhandled hazards
- Unhandled hazards lead to vulnerabilities and usability issues
- Most hazards hard to address with authentication centric IAM
- All can more easily be addressed with authorization centric IAM

https://alanhkarp.com/UseCases.pdf

Questions?