BLOCKCHAIN FUNDAMENTALS

Kostas Christidis • Duke MMCi 2017.06.23

About me

- Kostas Christidis
- Maintainer, Hyperledger Fabric
- Software Engineer, IBM Blockchain
- PhD Candidate, ECE NCSU

Plan of attack

1. Relevant **problem** in the **healthcare** industry
2. Let's try to **solve** it (**forget** about blockchains!)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Relevant problem[1]

- Balkanization of health records
- Lack of immediate access to health information
- 20% of preventable medical errors

[1] Source: L. J. Kish and E. J. Topol, “Unpatients—why patients should own their medical data,” Nat. Biotechnol., vol. 33, no. 9, pp. 921–924, Sep. 2015.

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Why is it that way?

1. No **incentives** for sharing
2. "We'll gladly let you **access** it"
3. No **interoperability** between EHR systems
4. **Easiest** option (keep the data locked)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

What should we aim for?

Make patient data easier to access

What's the easiest way to get there?

A **centralized** service

Holds every patient record out there

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

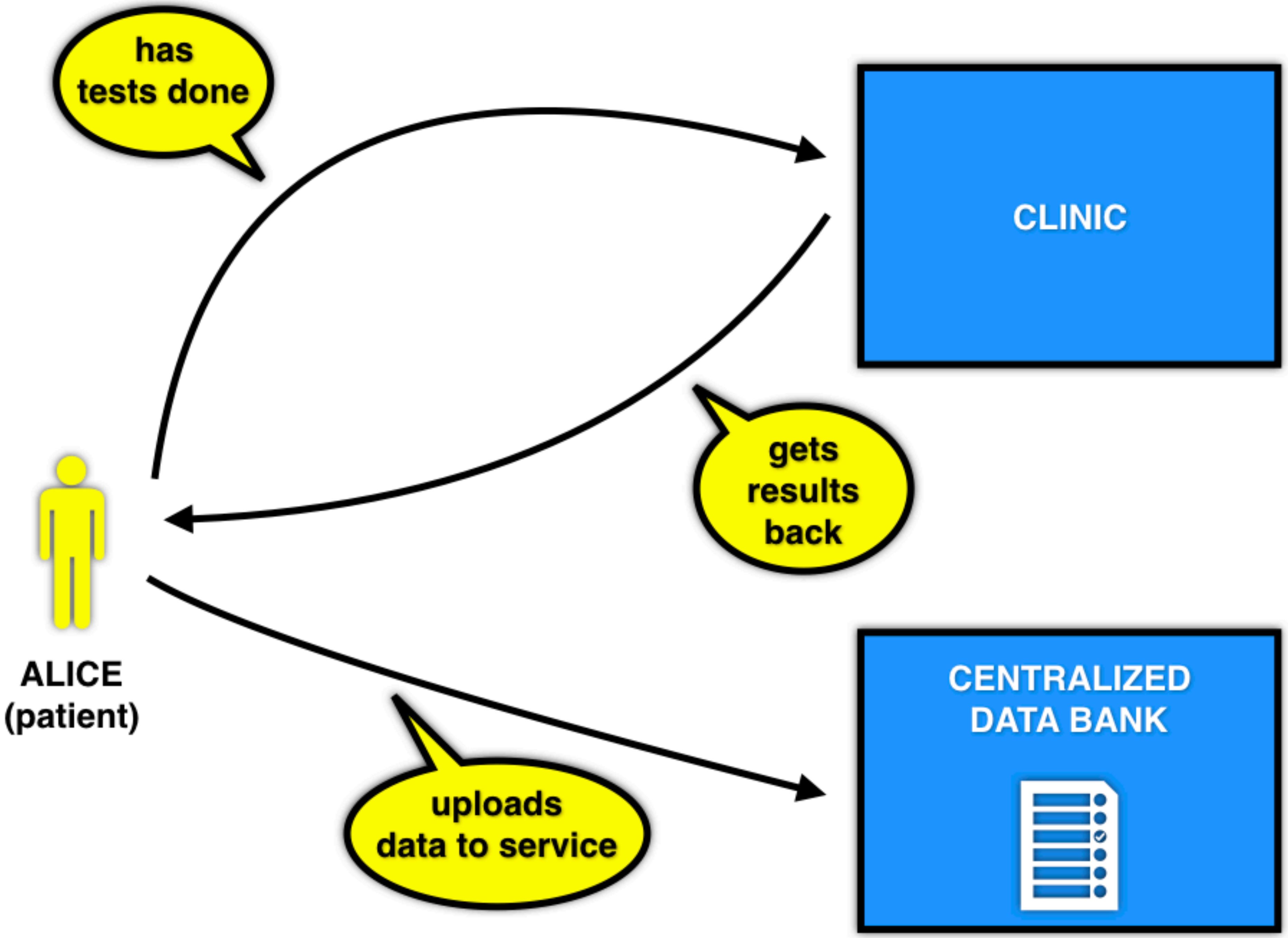
What's the easiest way to get there?

A **centralized** service

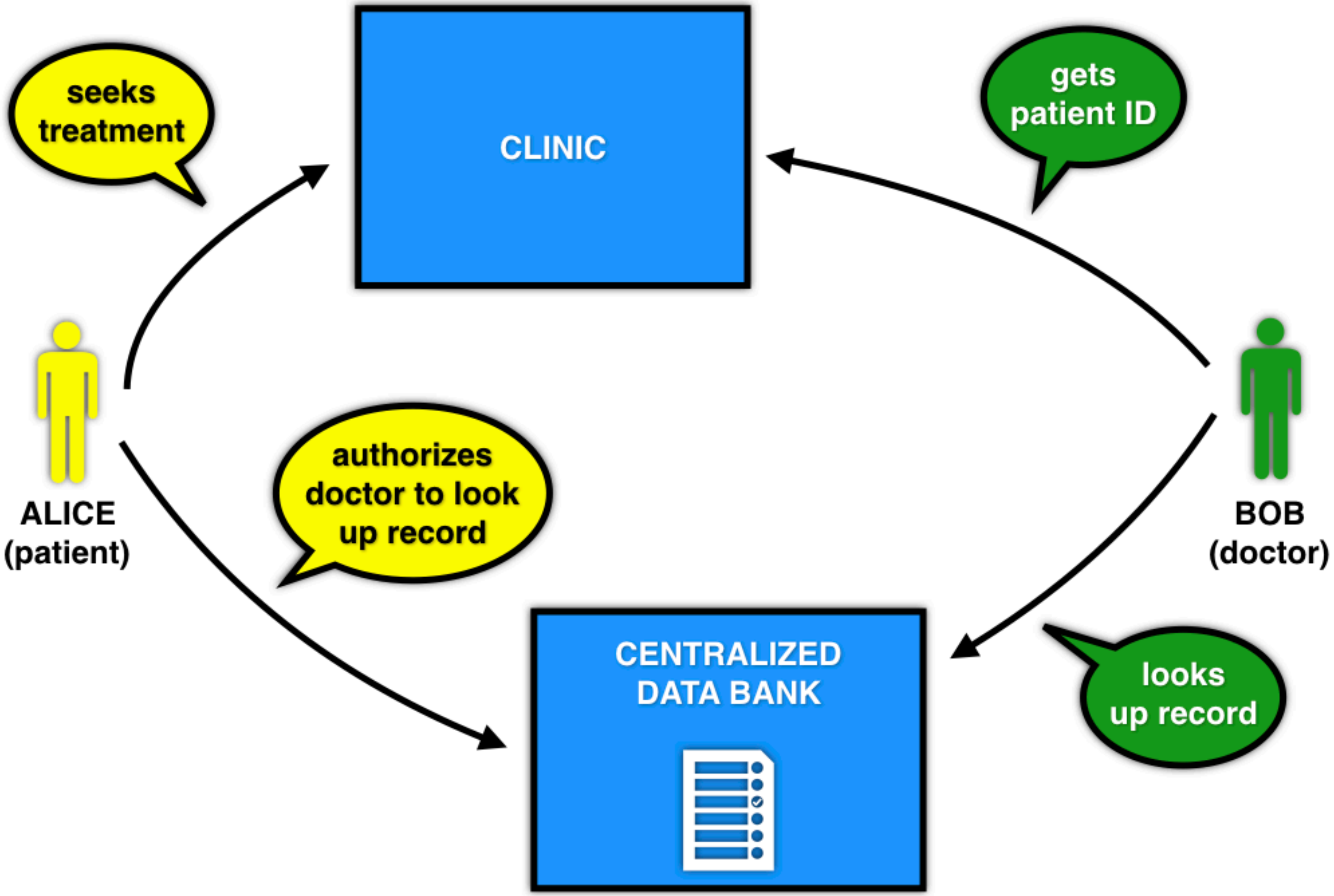
Holds every patient record out there

It would look somewhat like this...

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

This implies a table that looks like this

Patient Name	Record	Who Can Read	Who Can Write
Alice	Blood work @ Duke Hospital (2016/05/01)	Bob	Nobody

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

This implies a table that looks like this

Patient Name	Record	Who Can Read	Who Can Write
Alice	Blood work @ Duke Hospital (2016/05/01)	Bob	Nobody
John	Shoulder X-rays @ Rex Hospital (2016/02/04)	Carol	Carol

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Two concerns with that approach

1. A nightmare if it gets hacked
2. Who gets to run this service?

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

We need a different approach

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

If a centralized service won't do...

- Maybe a decentralized solution will?
- Every stakeholder is involved

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

If a centralized service won't do...

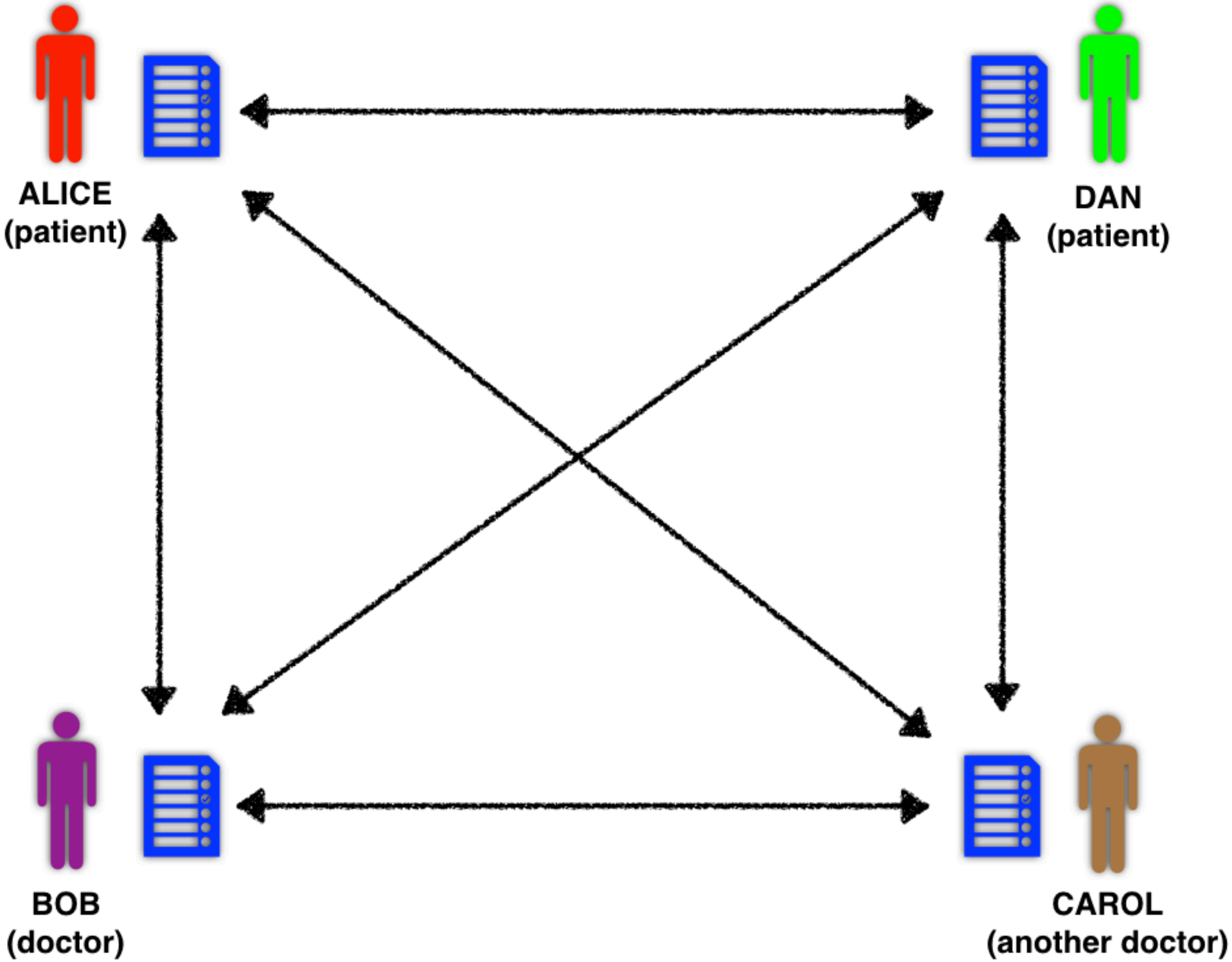
- Maybe a decentralized solution will?
- Every stakeholder is involved

But how would this work exactly?

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How would the decentralized service work?

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How would the decentralized service work?

1. Every stakeholder* has **a copy of that table**

∗ Hospital, patient, insurance agent, you-name-it

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How would the decentralized service work?

1. Every stakeholder* has **a copy of that table**

∗ Hospital, patient, insurance agent, you-name-it

Set aside any privacy concerns for now.

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

In fact, we don't need the table

Transaction logs *record all the changes made to the database. [...] appends are the only way to change the log. From this perspective, the contents of the database hold a caching of the latest record values in the logs.*

– Pat Helland, "Immutability Changes Everything"

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Let's look at an example

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Transaction log

```
t=0: k1=>v1
```

Database (table) state

Key	Value
k1	v1

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Transaction log

```
t=0: k1=>v1 | t=1: k2=>v1
```

Database state

Key	Value
k1	v1
k2	v2

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Transaction log

```
t=0: k1=>v1 | t=1: k2=>v2 | t=2: k1=>v1'
```

Database state

Key	Value
k1	v1'
k2	v2

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

The log is all that matters

1. Read it from start to finish
2. You have the most recent state of the table

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Again: how would the decentralized service work?

1. Every stakeholder has a copy of that ~~table~~ transaction log

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Again: how would the decentralized service work?

1. Every stakeholder has a copy of that ~~table~~ transaction log

How do we ensure the log is kept in sync across participants?

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

When there's no global ordering, we get:

- Participant A's log: **`foo|bar|baz`**
- Participant B's log: **`foo|baz|bar`**

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

When there's no global ordering, we get:

- Participant A's log: **`foo|bar|baz`**
- Participant B's log: **`foo|baz|bar`**

(^^ *This is what we call a fork*)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

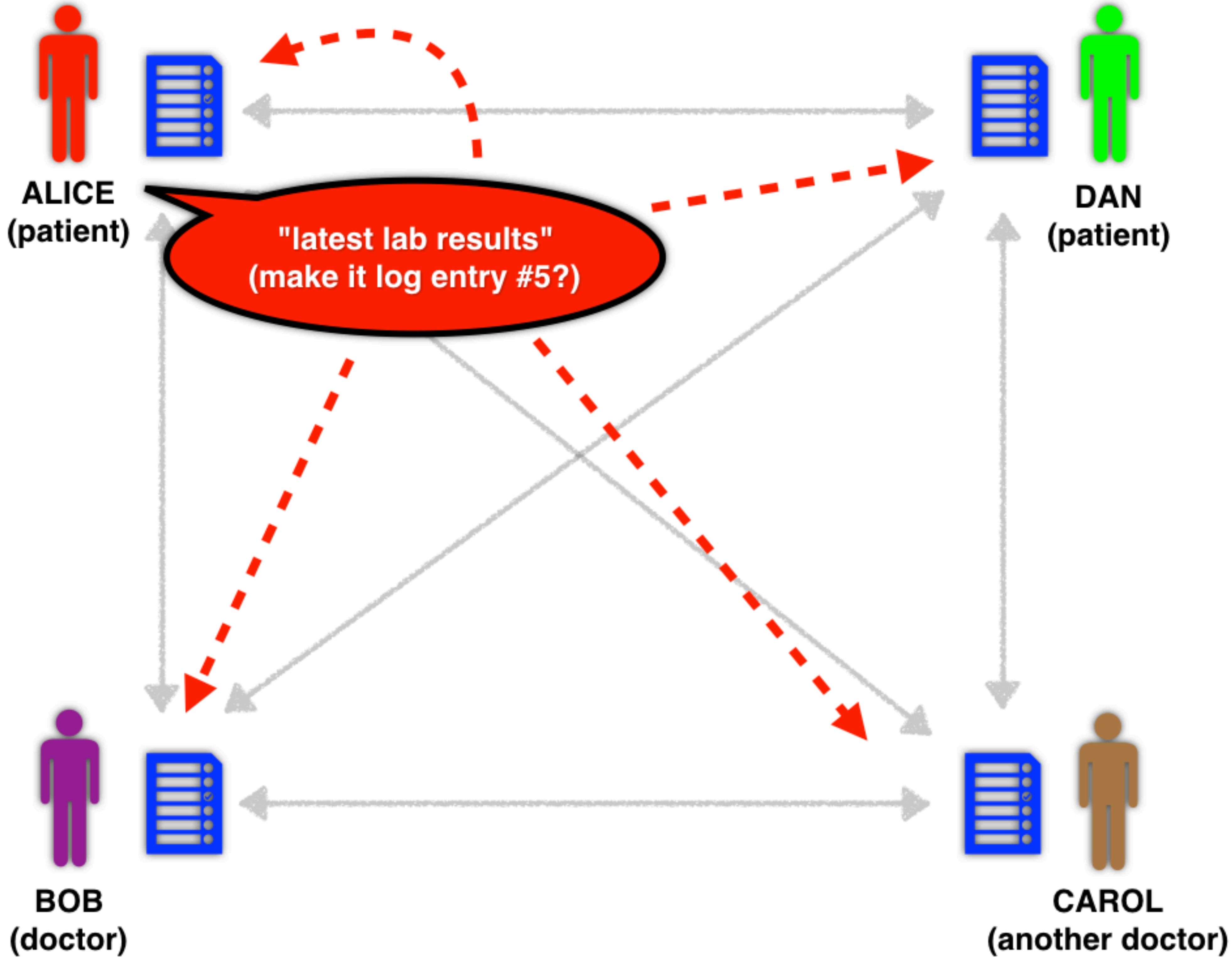
How do we order the writes?

- Need a marker
- Before you append to the log, determine its **height**
- Include this info in your broadcasted message

For example, Alice broadcasts this to the network:

These are my latest lab results. Make it ***log entry #5****?*

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

In fact, we can provide a better marker

A hash function is any function that can be used to map data of arbitrary size to data of fixed size.

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Google "sha-256 hash generator"

Input	Output
[foo,bar]	2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae
[foo,bar,baz]	b94d27b9934d3e08a52e52d7da7dabfac484efe37a5380ee9088f7ace2efcde9

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes?

Agree on a hashing function: `calc_hash()`

Assume the log looks like this: `[foo]`

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes?

Agree on a hashing function: **`calc_hash()`**

Assume the log looks like this: **`[foo]`**

```
hash := calc_hash(latest_log_entry)

message := ‹hash, original_message›

broadcast(message)
```

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes? (cont.)

- Log before: `[foo]`
- Log after: `[foo, ‹calc_hash(foo), original_message›]`

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes? (cont.)

- Log before: `[foo]`
- Log after: `[foo, ‹calc_hash(foo), original_message›]`

Let `bar := ‹calc_hash(foo), original_message›`

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes? (cont.)

- Log before: `[foo]`
- Log after: `[foo, <calc_hash(foo), original_message>]`

Let `bar := <calc_hash(foo), original_message>`

- New log is basically: `[foo, bar]`

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes now?

Used to be:

*These are my latest lab results. Make it **log entry #5**?*

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we order the writes now?

Used to be:

These are my latest lab results. Make it ***log entry #5****?*

Now:

These are my latest lab results. The hash of the log tip is ***2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae****.*

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Why is this a better marker than log height?

- Acts as a built-in consistency check
- You're not saying: "add to page 50"
- You're saying: "add to page 50 of a book that has this written in the previous 49 pages"

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Why is this a better marker than log height?

- Acts as a built-in consistency check
- You're not saying: "add to page 50"
- You're saying: "add to page 50 of a book that has this written in the previous 49 pages"

As we'll see below, hashing combined with digital signatures

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Status check

1. Every stakeholder has a copy of the transaction log
2. Writes ordered by including the hash of the tip of the log

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Status check

1. Every stakeholder has a copy of the transaction log
2. Writes ordered by including the hash of the tip of the log

Ignore collisions for now.
Will address them later.

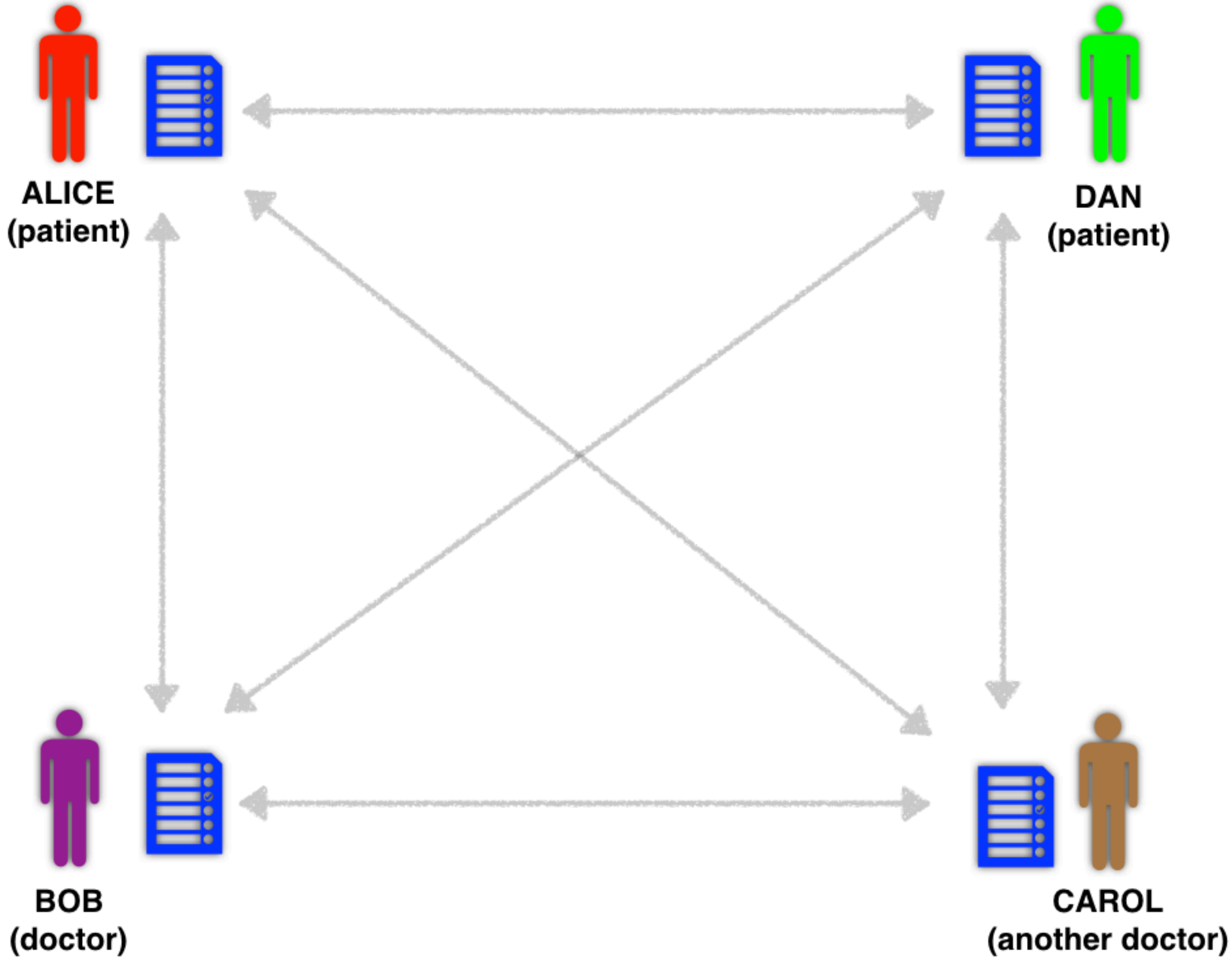
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Are we done?

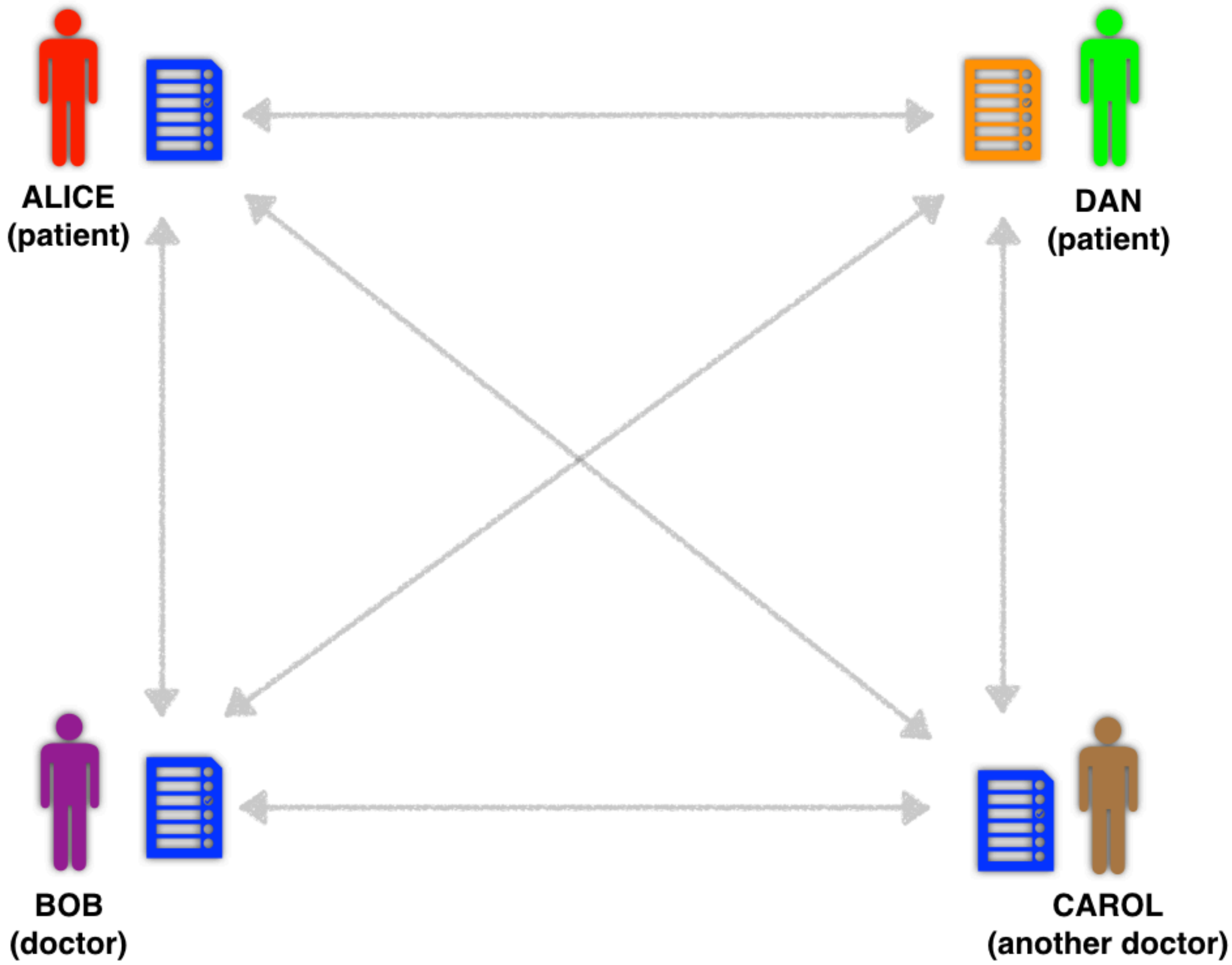
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Consider this scenario

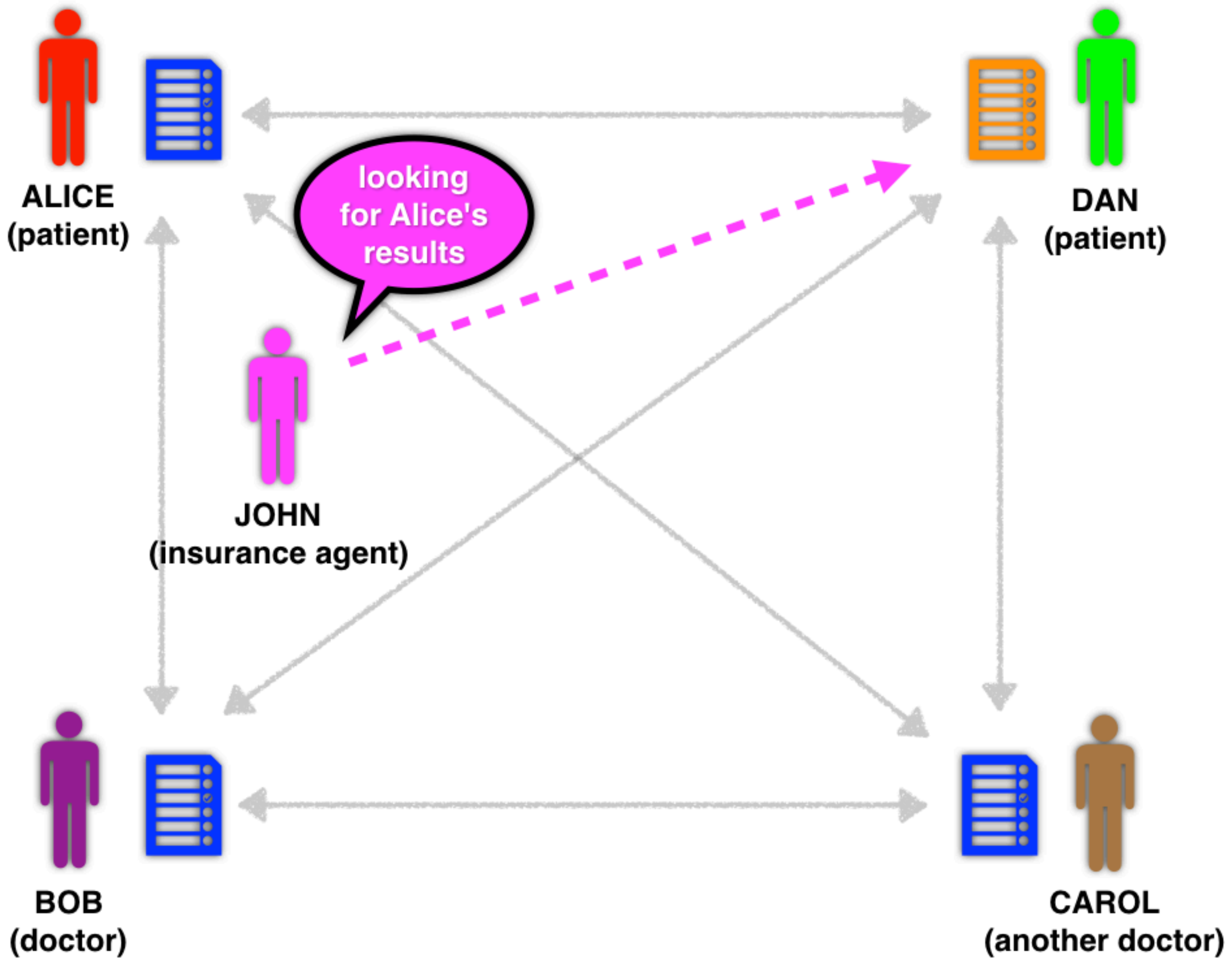
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



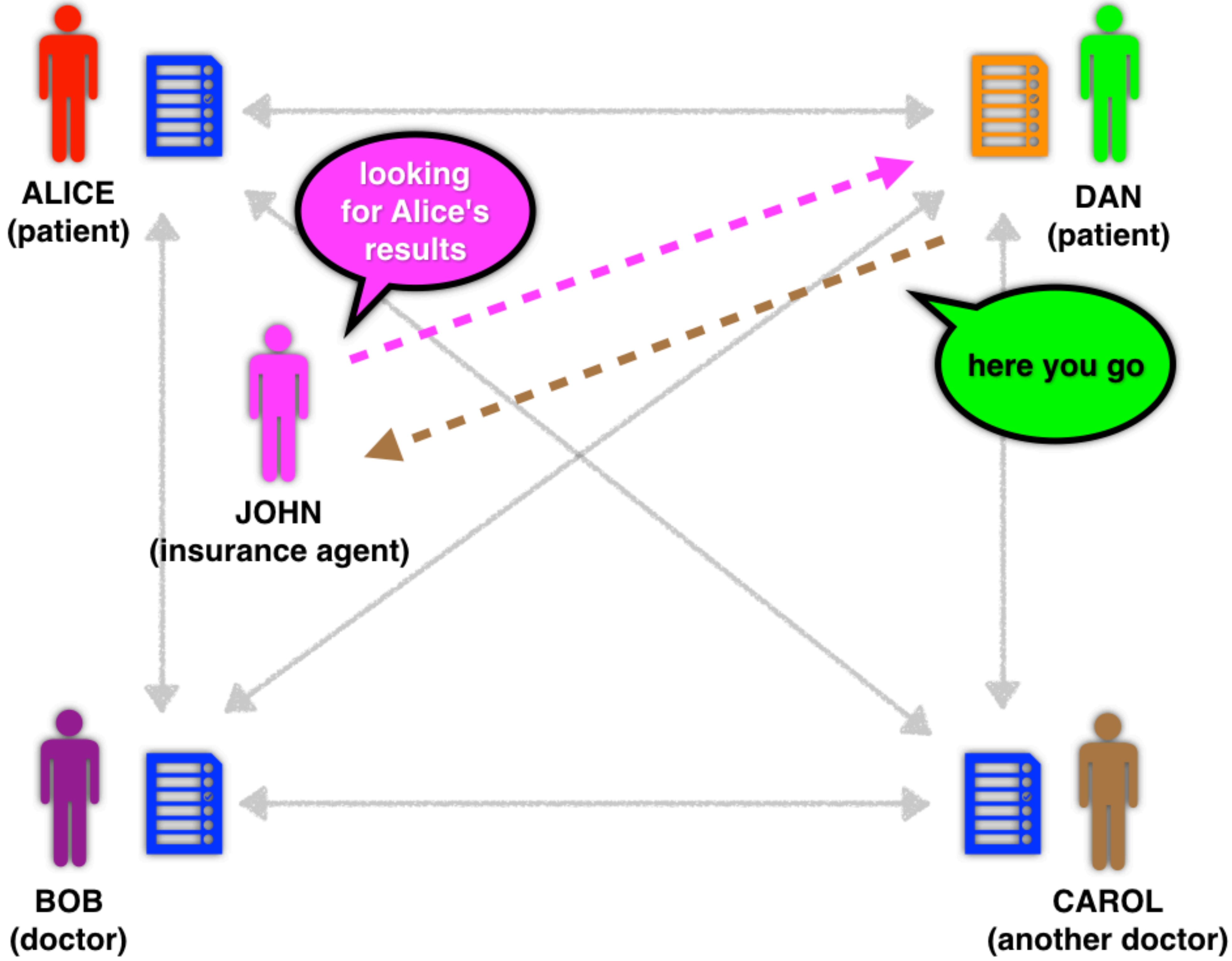
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Scenario summary

- Alice uploads the results of her latest checkup
- (She's in perfect health)
- Her insurance agent checks them to adjust her premiums
- Reads them from Bob's log
- Bob tampers Alice's update on purpose
- Alice's premiums get adjusted incorrectly

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

How do we avoid this?

Public key cryptography to the rescue

- Alice has a **known** public key
- Signs her updates with her private key
- Appends signature to her broadcasted message
- Reader looks for that signature

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Pseudocode

```
sign := calc_sign(private_key, message)

signed_update = [sign, message]

// ["835cda89", ‹2c26b46b, "LDL normal"›]
```

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

We now have a table that looks like this

Patient Name	Public Key	Record	Digital Signature	Who Can Read	Who Can Write
Alice	**b83da6c0 9893ad**	Blood work @ Duke Hospital (2016/05/01)	**835cda89 7d74c3**	Carol	Nobody

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

We now have a table that looks like this

Patient Name	Public Key	Record	Digital Signature	Who Can Read	Who Can Write
Alice	**b83da6c098 93ad**	Blood work @ Duke Hospital (2016/05/01)	**835cda897d 74c3**	Bob, Carol	Nobody
Alice	**b83da6c098 93ad**	Shoulder X-rays @ Rex Hospital (2016/02/04)	**dac089r967 856d**	Nobody	Nobody

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Digital signatures give us:

- Authentication
- Non-repudiation

Status check

1. Every stakeholder has a copy of the transaction log
2. All writes include the hash of the tip of the log for ordering
3. All writes are signed

Haven't addressed yet:

1. Privacy concerns
2. Collisions

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Let's start with collisions

Log currently at: **`[baz]`**

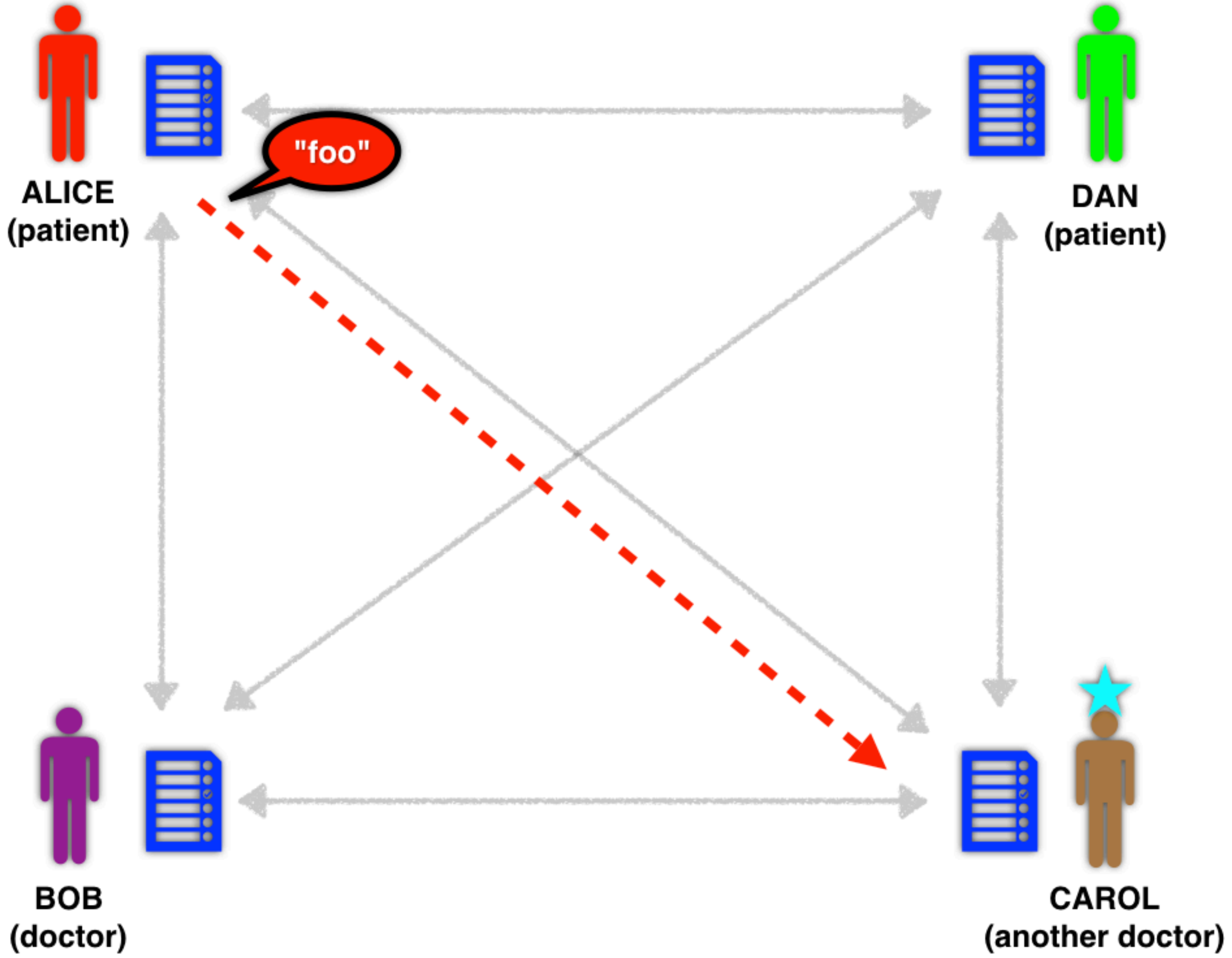
- Alice hashes `baz`, wants to add **`foo`**
- Bob hashes `baz`, wants to add **`bar`**

Who wins?

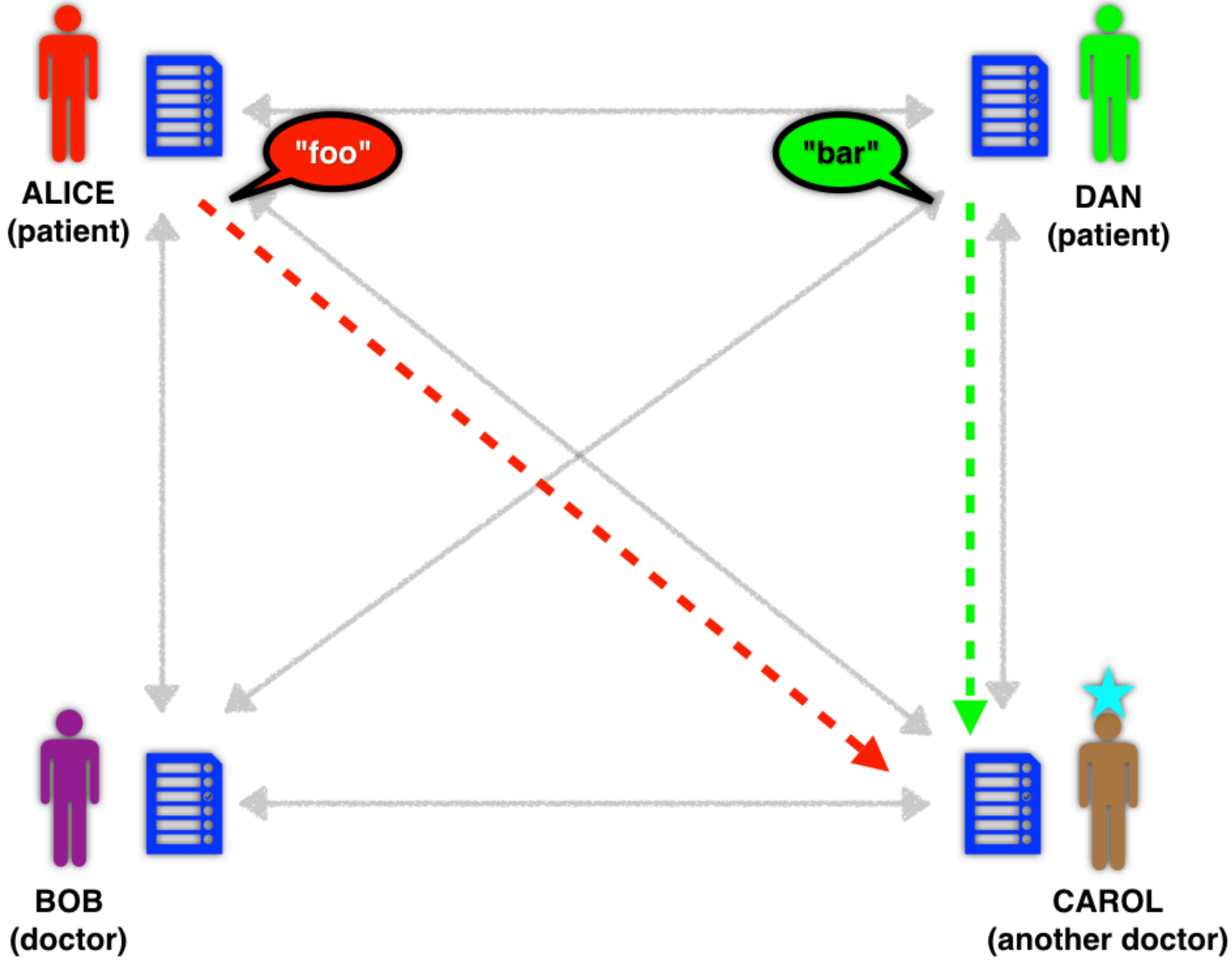
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Ordering is complicated

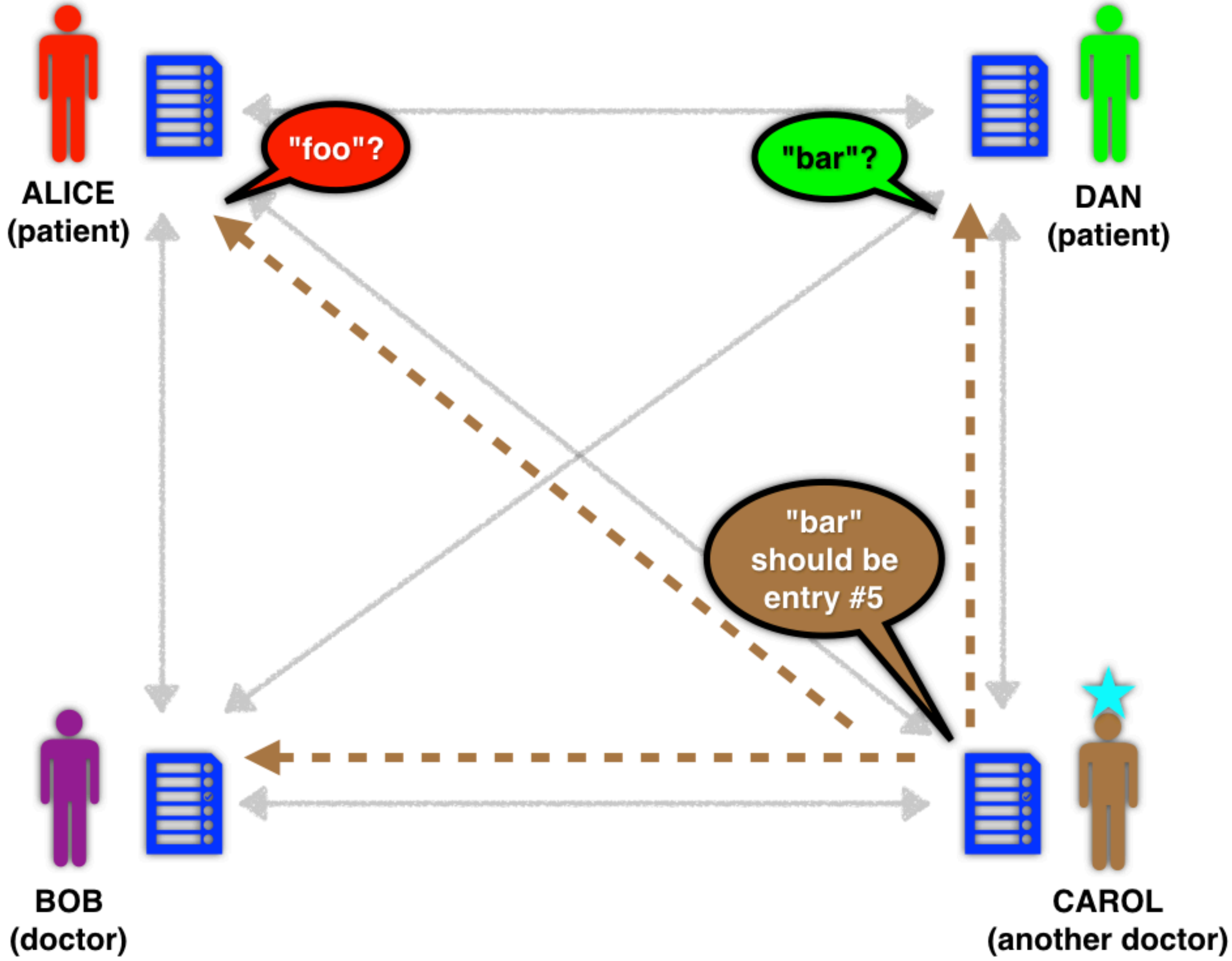
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Proper ordering flow

- Network assigns a leader who takes care of the ordering
- Alice and Bob no longer need to hash the log tip
- They just send their TXs (`foo`, `bar`) to the leader
- Leader decides on order arbitrarily (usually FIFO)
- Hashes log tip, broadcasts `<hash, orig_msg>` as usual
- Everybody goes along with it

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

In fact…

- If we just order one TX at a time, throughput will be low
- Leader waits to collect a batch of TXs and **orders the batch**
- So we now have a hashed chain of batches/**blocks**
- A **blockchain**! (Finally!)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

A couple of more nodes on ordering

- Leader cannot fake transactions
- They still need to carry valid signatures
- If leader doesn't play nicely (byzantine), can be voted off
- Leader may change periodically

Huge body of literature devoted to ***distributed consensus***

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Status check

1. Members have a copy of the transaction log
2. Members send signed writes to leader
3. Leader batches them, hashes previous block to establish order, then broadcasts this new batch to the network
4. Members inspect proposed batch, if everything looks good they adopt is as new addition to log

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Privacy concerns

- Shared log holding all patient records: a privacy nightmare
- Don't store the records or reveal any patient identifying info
- Replace everything with hashes — use these as identifiers

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Privacy concerns

- Shared log holding all patient records: a privacy nightmare
- Don't store the records or any patient identifying info
- Replace everything with hashes – use these as identifiers

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

For example

- Alice identified by the hash of her public key
- Alice's blood work represented simply by its hash

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Wrong way to do it

Patient Name	Public Key	Record	Digital Signature	Who Can Read	Who Can Write
Alice	**b83da6c0 9893ad**	Blood work @ Duke Hospital (2016/05/01)	**835cda89 7d74c3**	Carol	Nobody

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Right way to do it

Public Key	Record	Digital Signature	Who Can Read	Who Can Write
b83da6c0 9893ad	**149dca74 ef983a**	**835cda89 7d74c3**	**fe45208d 673a67**	Nobody

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

But *where* do you store the patient records?

- Use third-party **storage providers** built for this
- These services will have access to the blockchain
- They only process **signed requests** to read/write records
- They only accept them if the blockchain says that the associated public key is granted access to read/write on the blockchain

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

So how would it all work?

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Coinbase-like services for your EHRs

Expect companies to spring up that:

- *Both* act as **storage providers** for your EHRs, *and*
- Help you **interact** with the blockchain network

- Like Coinbase for buying/selling Bitcoin
- For most folks, their only interaction with the Bitcoin blockchain is via the Coinbase website or mobile app

Who's actually holding a copy of the transaction log?

All **major** stakeholders:

- hospitals
- insurance companies
- these third-party Coinbase-like companies

Unrealistic to expect individuals (as shown before) to run their own nodes

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

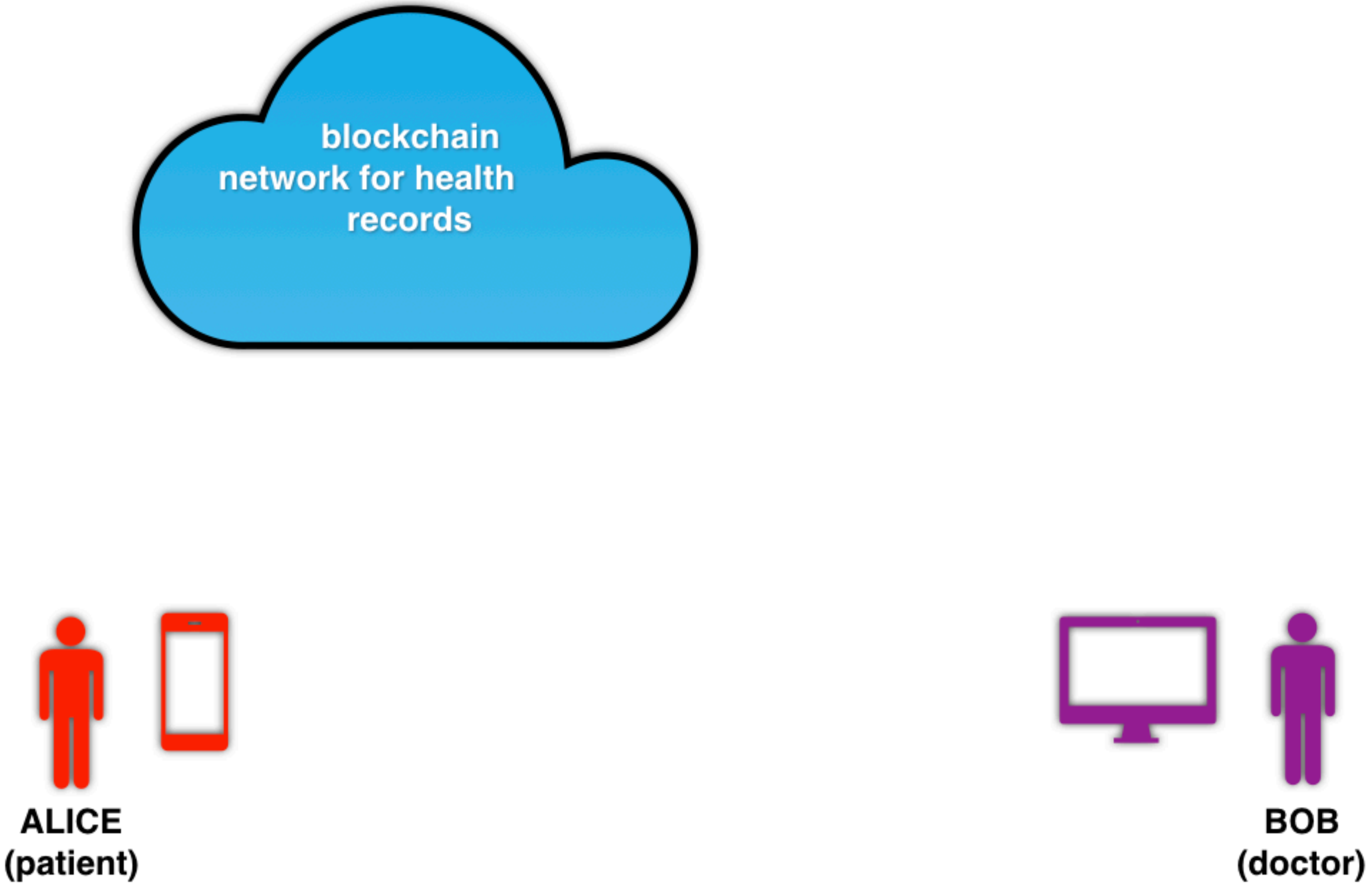
How does EHR data get into the system?

- Patients no longer get a hard copy of their lab work, but a digital one as well
- They upload these to their Coinbase-like service of choice
- They now have a record in that global table associated with their public key

Expect heavy usage of mobile apps and QR codes here

Then a visit to the doctor looks like this

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

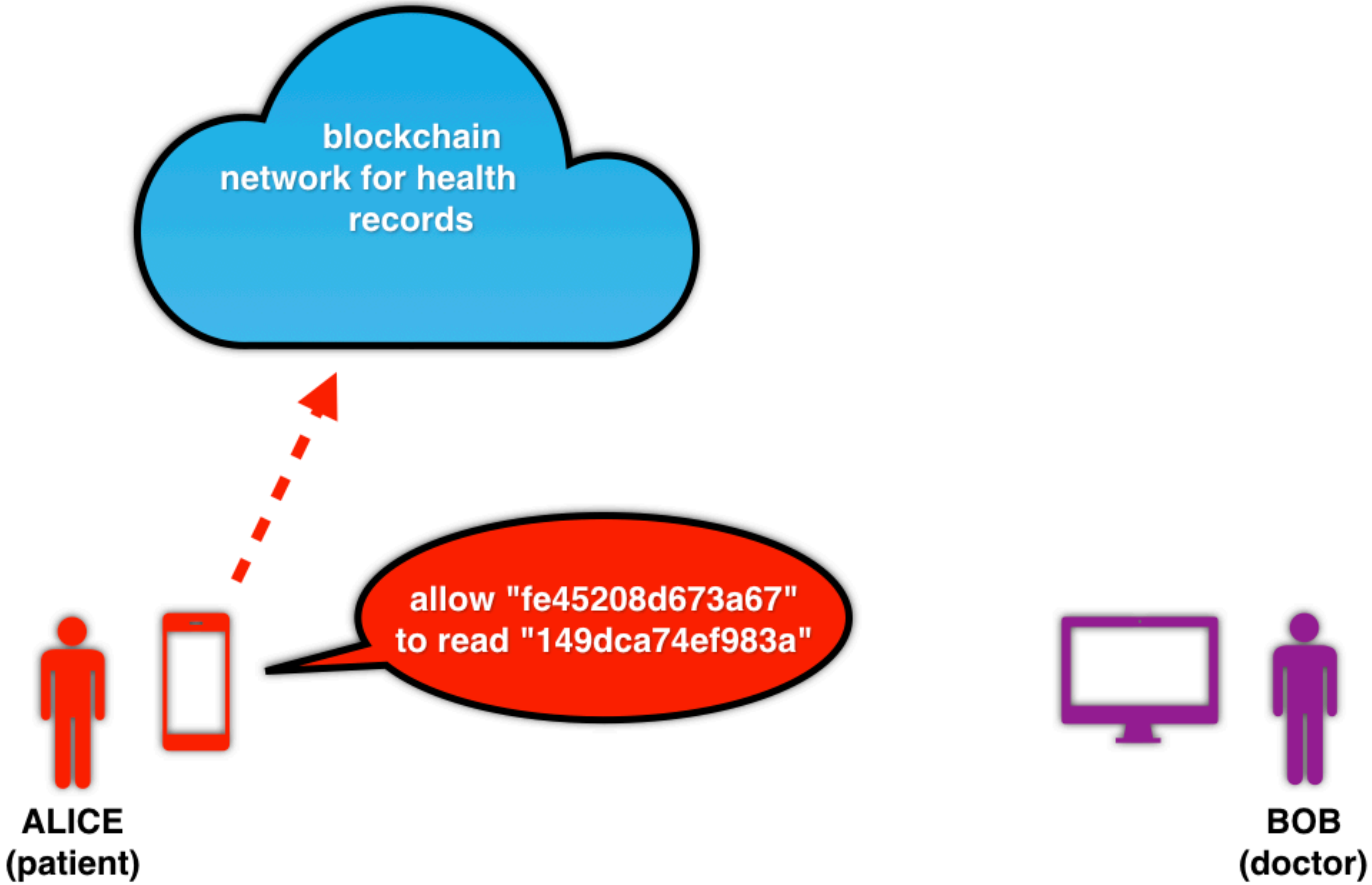


blockchain network for health records

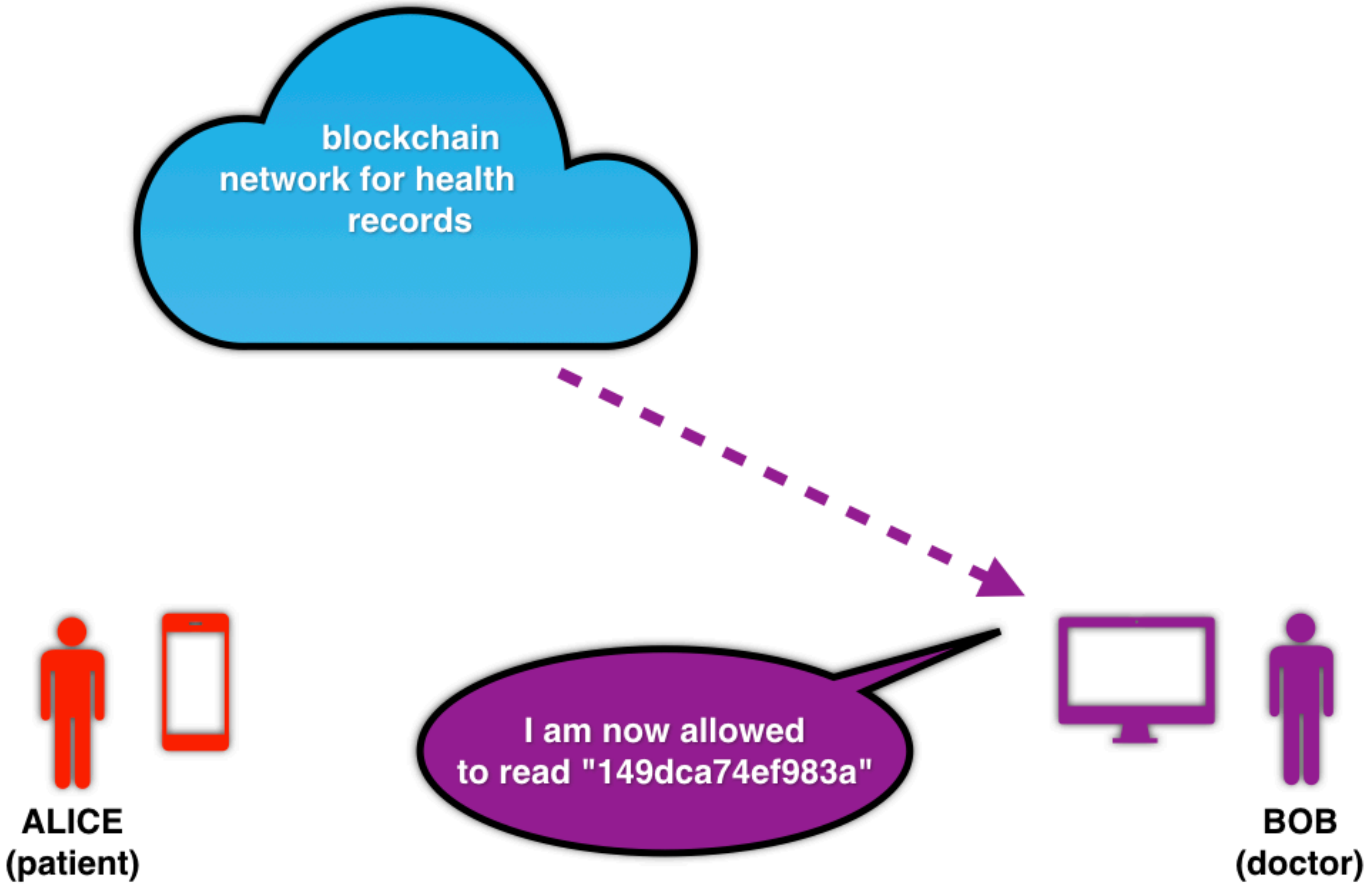
ALICE
(patient)

BOB
(doctor)

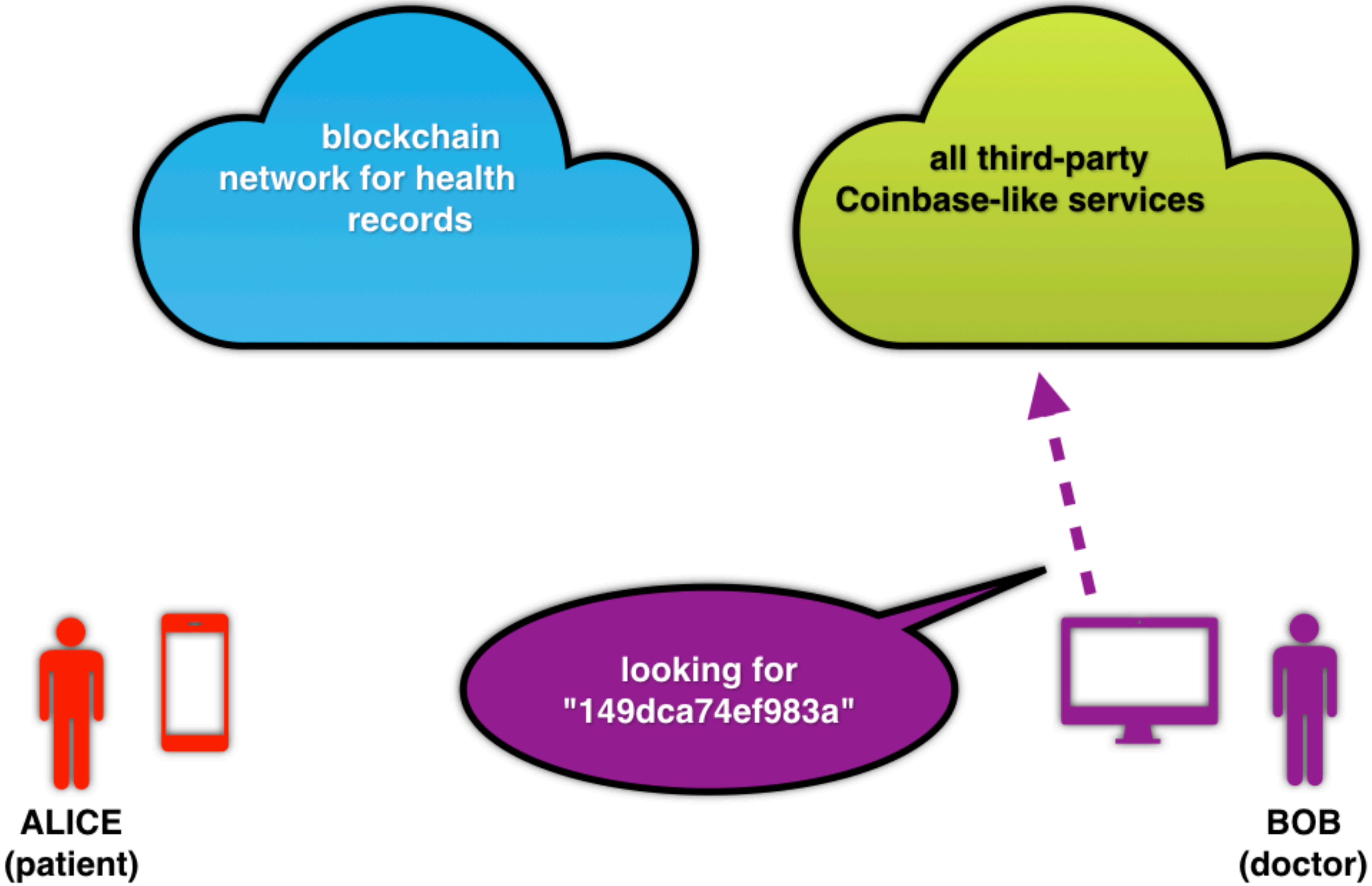
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.


blockchain network for health records
allow "fe45208d673a67" to read "149dca74ef983a"
ALICE (patient)
BOB (doctor)

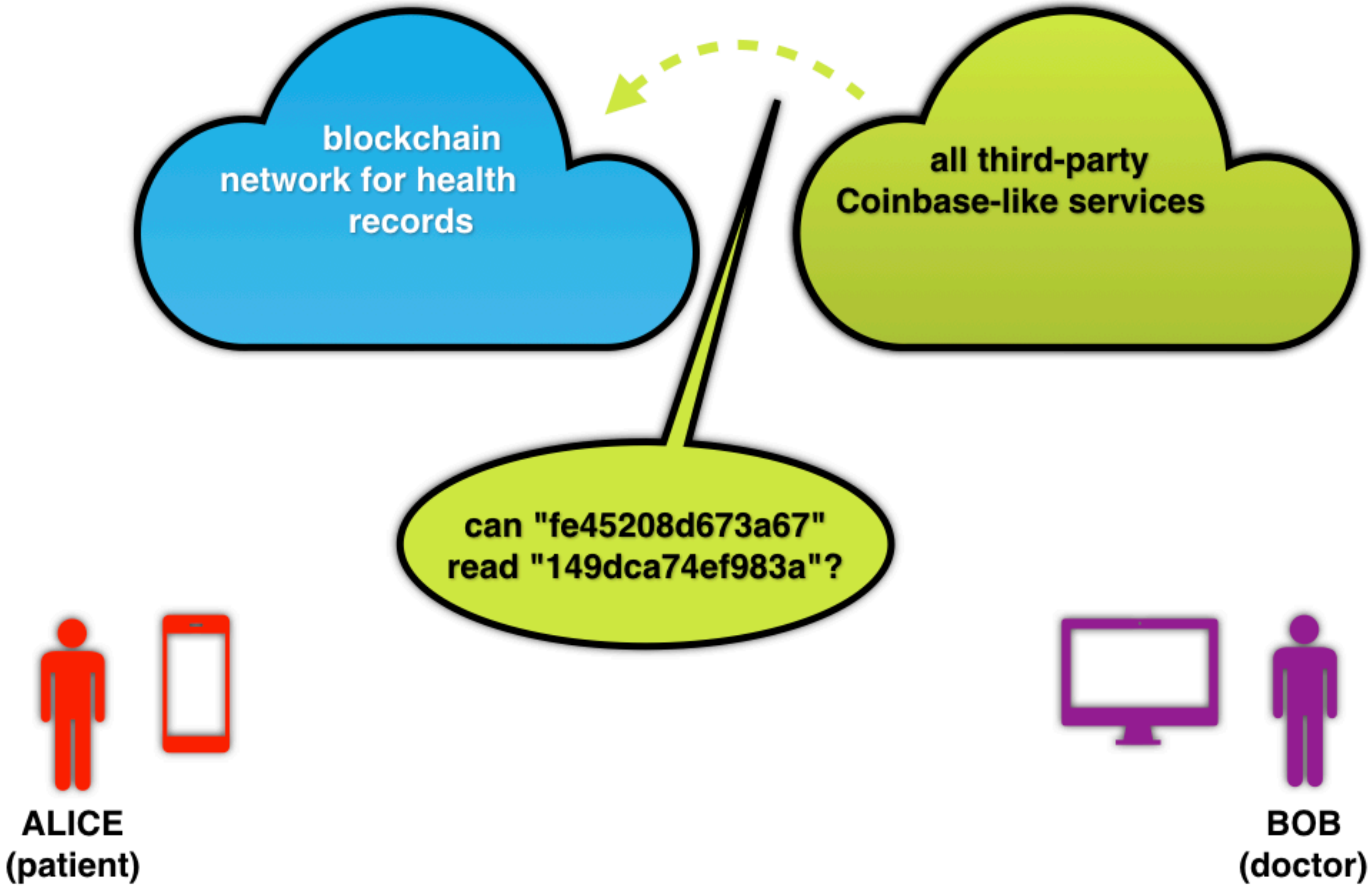
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.


blockchain network for health records
I am now allowed to read "149dca74ef983a"
ALICE (patient)
BOB (doctor)

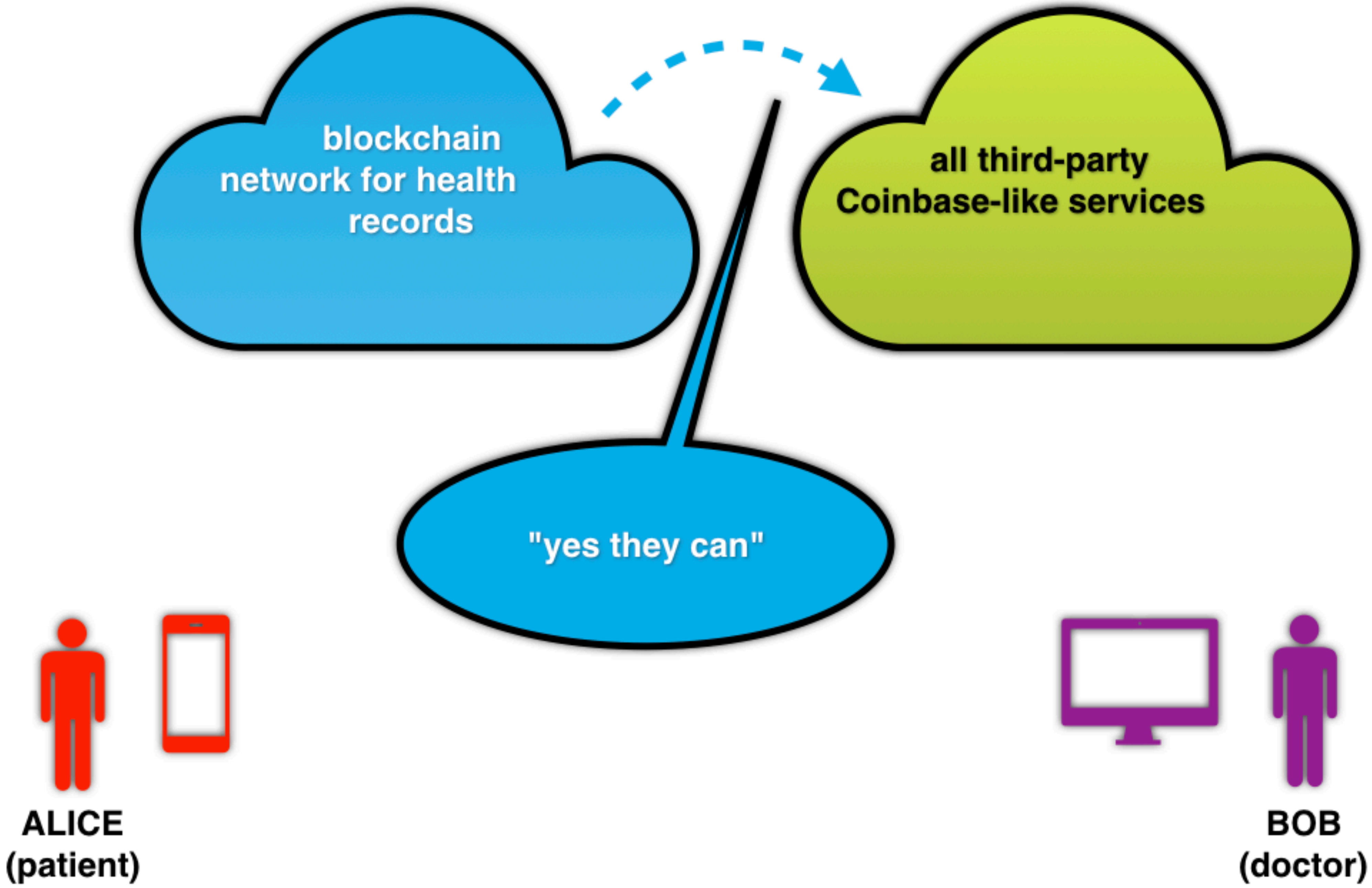
K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.


blockchain network for health records
all third-party Coinbase-like services
looking for "149dca74ef983a"
ALICE (patient)
BOB (doctor)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

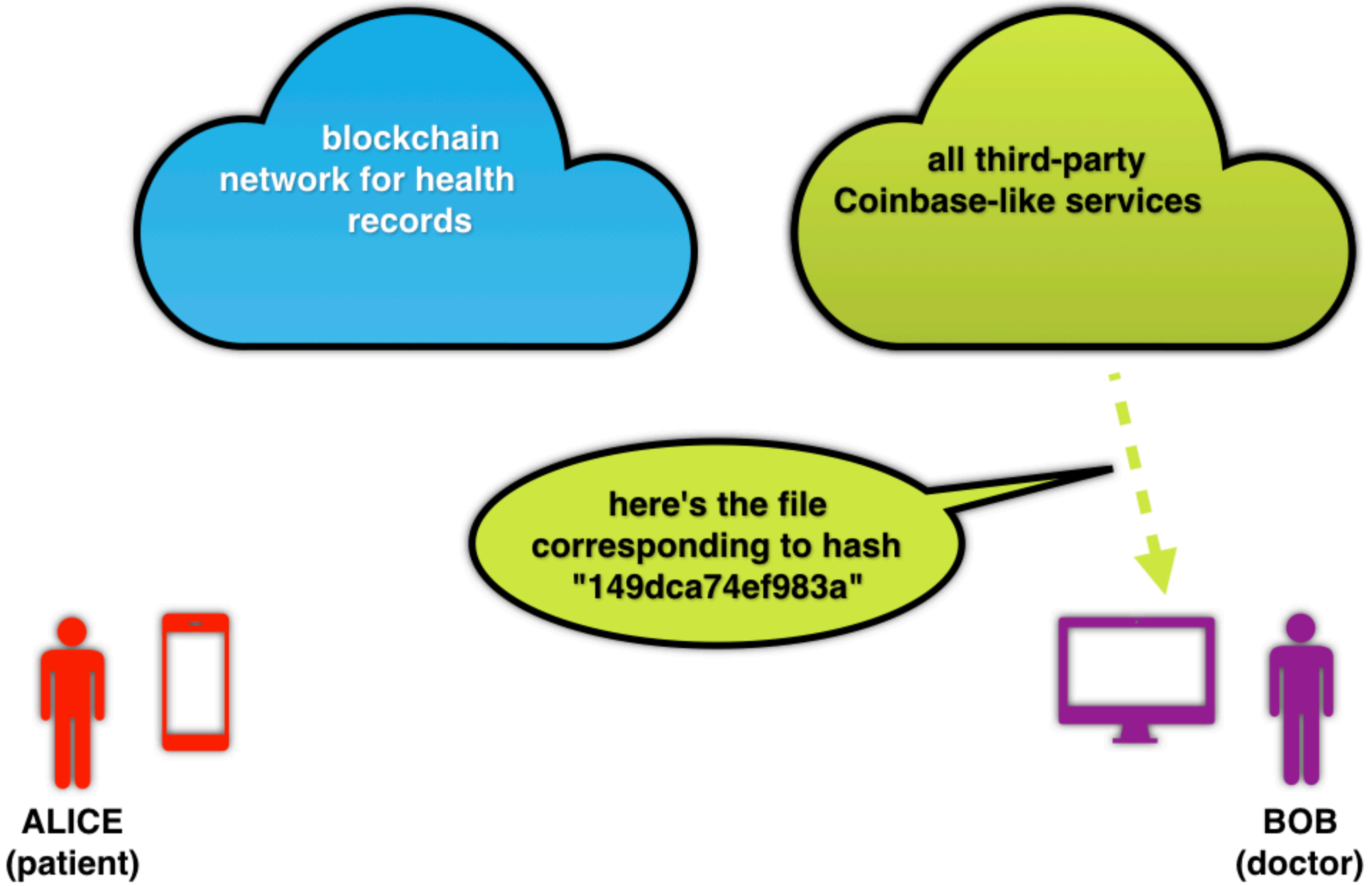

blockchain network for health records
all third-party Coinbase-like services
can "fe45208d673a67" read "149dca74ef983a"?
ALICE (patient)
BOB (doctor)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.



blockchain network for health records
all third-party Coinbase-like services
"yes they can"
ALICE (patient)
BOB (doctor)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.


blockchain network for health records
all third-party Coinbase-like services
here's the file corresponding to hash "149dca74ef983a"
ALICE (patient)
BOB (doctor)

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Visit to the doctor

- You visit the doctor & they need to see one of your records
- Using your mobile app, you:
    1. Scan the doctor's QR code (i.e. their public key)
    2. Then select the record you grant them access to
- Your mobile app then writes to the blockchain, updating the access list for the record

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Visit to the doctor (cont.)

What the doctor's app does:

1. Reads latest blockchain updates
2. Observes TX adding them to the read-list for the record
3. Seeks that record (via its hash) in all participating third-party services

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Visit to the doctor (cont.)

What the Coinbase-like third-party service then does:

1. Reads the blockchain to check if the request is allowed to access the record
2. If so, returns the file it stores locally to the reader

Epilogue

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Blockchain

A replicated ledger of signed blocks forming a hash chain. (Also, a set of rules on how these blocks are added to the chain to begin with.)

- **Only way** for multiple non-trusting **writers** to share a DB
- Minimizes the need for trust
- Maximizes auditability, facilitates provenance tracking

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Addendum

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Smart contracts

- A misnomer: **simple scripts**
- Think of them as an "**if this, then that**" mechanism for your blockchain data
- Example: "if someone writes this data to my row on the DB, update this other row that I own"
- A way to create simple workflows *on the blockchain*

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.

Questions?

K. Christidis, "Blockchain Fundamentals," Duke MMCi, June 2017.