CS4501 Cryptographic Protocols
Lecture 20: Perfect BA, PKI

https://jackdoerner.net/teaching/#2026/Spring/CS4501

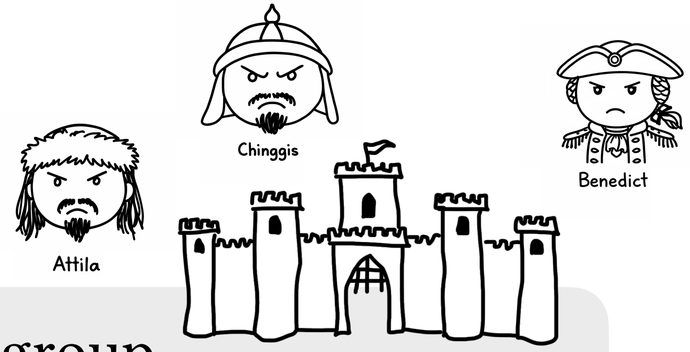
Byzantine Agreement

Definition 1. Let $n, t \in \mathbb{N}$ such that $t < n$, and let $P_1, \ldots, P_n$ be a group of parties that are connected by synchronous, authenticated channels. Let π be a protocol in which each P_i has an input bit $x_i \in \{0,1\}$ and an output bit $y_i \in \{0,1\}$. We say that π is a t-secure *Byzantine Agreement* protocol if the following conditions hold with all-but-negligible probability when $\mathcal{A}$ maliciously corrupts up to t parties:

1. **Consistency:** All honest parties output the same bit y.

2. **Validity:** If all honest parties start with the same input x, then $y = x$.

Byzantine Generals (Broadcast)



Definition 5. Let $n, t \in \mathbb{N}$ such that $t < n$, and let $P_1, \ldots, P_n$ be a group of parties that are connected by synchronous, authenticated channels. Let π be a protocol in which P_1 has an input $x \in \{0,1\}$ and every P_i produces an output $y_i \in \{0,1\}$. We say that π is a t-secure *Broadcast* protocol if the following conditions hold with all-but-negligible probability when $\mathcal{A}$ maliciously corrupts up to t parties:

1. **Consistency:** All honest parties output the same bit y.

2. **Validity:** If P_1 is honest, then $y = x$.

A broadcast *functionality* simply takes an input from one party and outputs it to all others. In other words, a simulation-secure broadcast protocol is one that securely computes $f(m, \lambda, \ldots, \lambda) = (m, m, \ldots, m)$.

Roadmap for the Next Few Lectures

Lecture 19, Theorem 2: In the setting where $t < n/2$, there exists a t-secure BA protocol if an only if there exists a t-secure broadcast protocol.

Lecture 20, Theorem 1: When $t \geq n/3$, there does not exist a t-secure BA protocol in the plain model, even assuming authenticated channels.

Lecture 20, Theorem 2: When $t < n/3$, there exists a perfectly t-secure BA protocol in the plain model, assuming authenticated channels.

Lecture 21, Theorem ??: Assuming the existence of one-way functions and a *public-key infrastructure*, there exists a (computationally) t-secure broadcast protocol in the setting where $t < n$.

Recap: BA $\Longrightarrow$ Broadcast

Lemma 1: Let $t < n$. If there exists t-secure BA, then there exists t-secure broadcast.

Proof: Consider the following putative broadcast protocol:

1. P_1 sends x to all parties. Let x_i denote the alleged value of x actually received by P_i, and let $x_i = 0$ if no message was received by P_i.

2. The parties $P_1, \ldots, P_n$ run BA on inputs $(x_1, \ldots, x_n)$.

Consistency of Broadcast: follows trivially from consistency of the BA protocol!

Validity of Broadcast: if P_1 is honest, then every P_i receives $x_i = x$; given this fact, validity of broadcast follows from validity of the BA protocol. ∎

Recap: Broadcast $\Longrightarrow$ BA

Lemma 2: Let $t < n/2$. If there exists t-secure broadcast, then there exists t-secure BA

Proof: Consider the following putative BA protocol:

1. Every P_i broadcasts x_i to all parties. Let $\hat{x}_i$ denote the alleged value of x_i actually received by P_j for all $j \in [n] \backslash \{i\}$, and let $\hat{x}_i = 0$ if P_i fails to broadcast at all.

2. The parties $P_1, \ldots, P_n$ output the *majority* function on $(\hat{x}_1, \ldots \hat{x}_n)$. If there is no majority, then they output 0.

Consistency of BA: By the consistency of the broadcast protocol, all honest parties agree on $(\hat{x}_1, \ldots \hat{x}_n)$, and the majority function is deterministic.

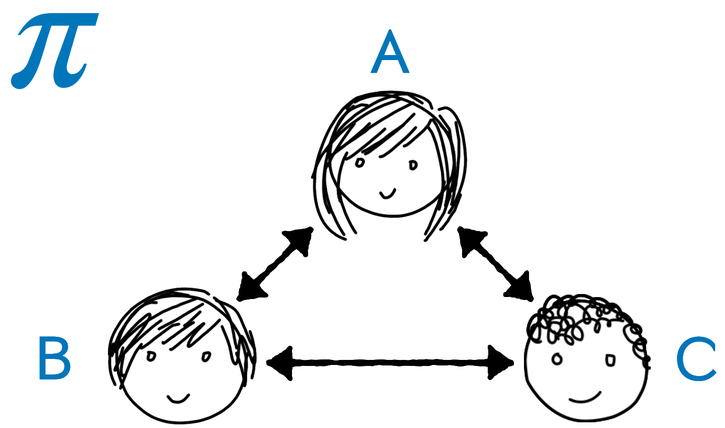
Validity of Broadcast: Suppose there is some x such that every honest P_i has $x_i = x$. Since $t < n/2$, the majority of $(\hat{x}_1, \ldots \hat{x}_n)$ must be x. ∎

First, an impossibility…

No Plain-Model BA for $t \geq n/3$

Theorem 1: There is no 3-party 1-perfectly-secure BA protocol in the plain model, even assuming authenticated channels.

Proof: Our proof will be by contradiction. Assume towards contradiction that a 3-party 1-perfectly-secure BA π exists.

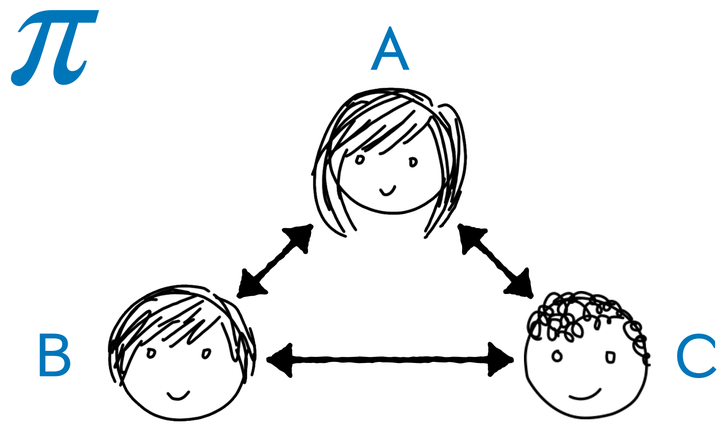


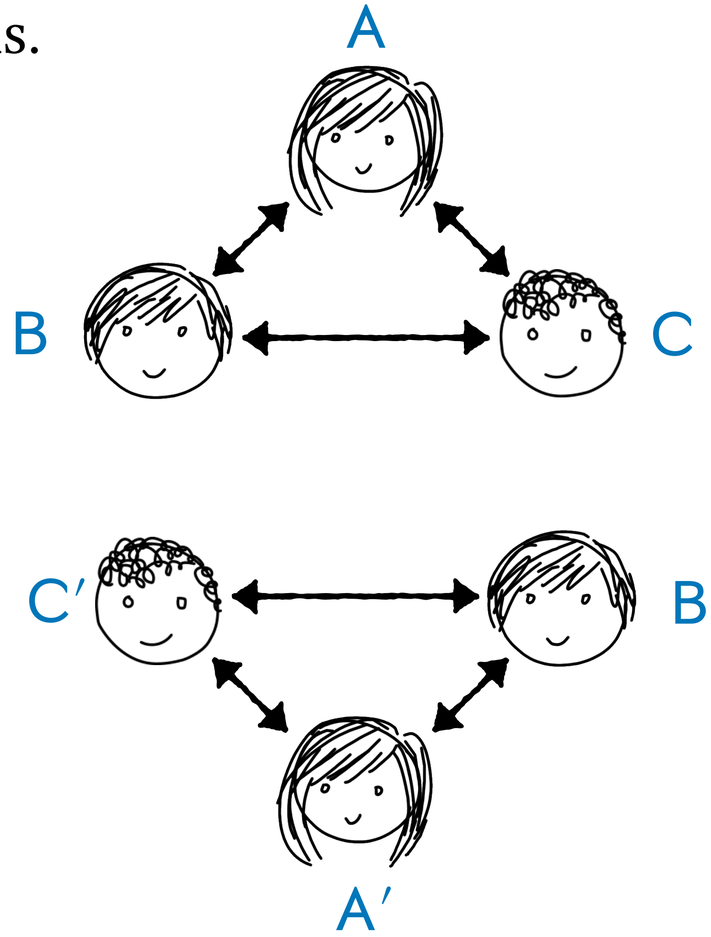
Next, consider two simultaneous instances of π...

No Plain-Model BA for $t \geq n/3$

Theorem 1: There is no 3-party 1-perfectly-secure BA protocol in the plain model, even assuming authenticated channels.

Proof: Our proof will be by contradiction. Assume towards contradiction that a 3-party 1-perfectly-secure BA π exists.





Next, consider two simultaneous instances of π...

No Plain-Model BA for $t \geq n/3$

Next, consider two simultaneous instances of π...

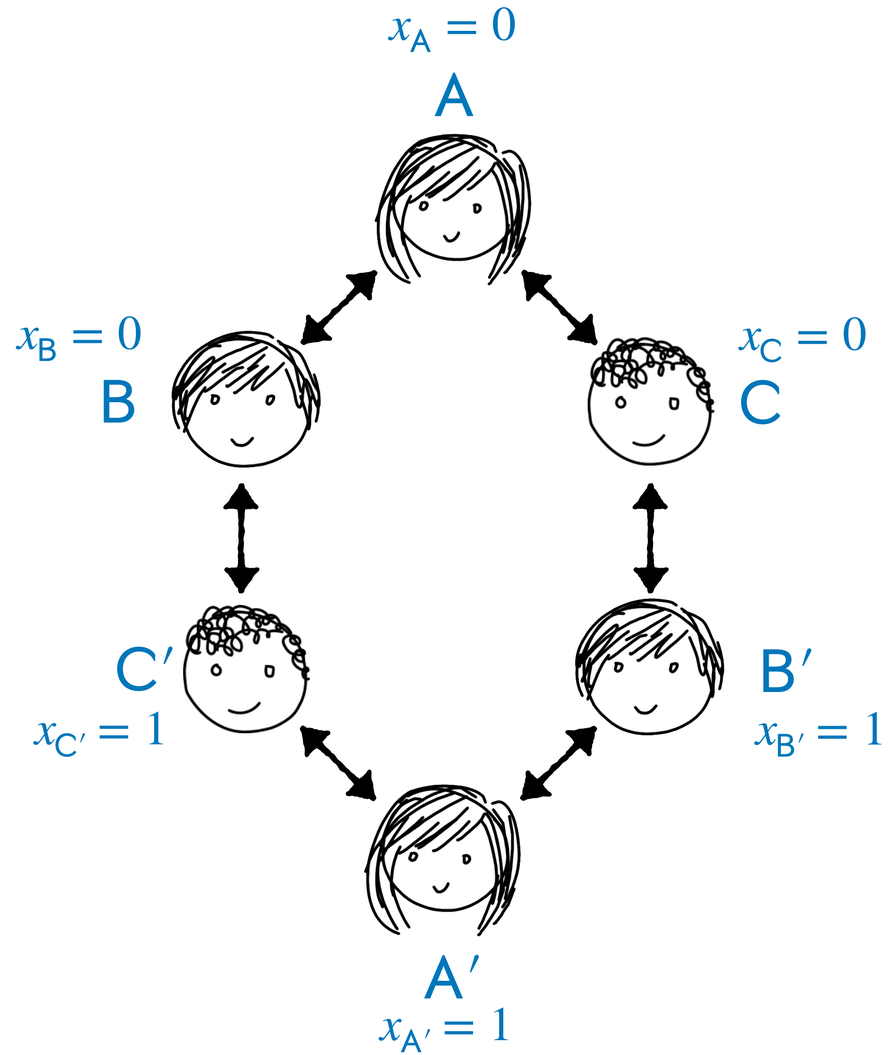
... and rewire them so B talks to C′ and B′ talks to C.

Q: *What does this protocol compute?*

A: *I don't know either.*

Even so, this is a *well-defined* protocol if the triangle protocol is. Each party sees the correct number channels and knows what to do when messages arrive on them.

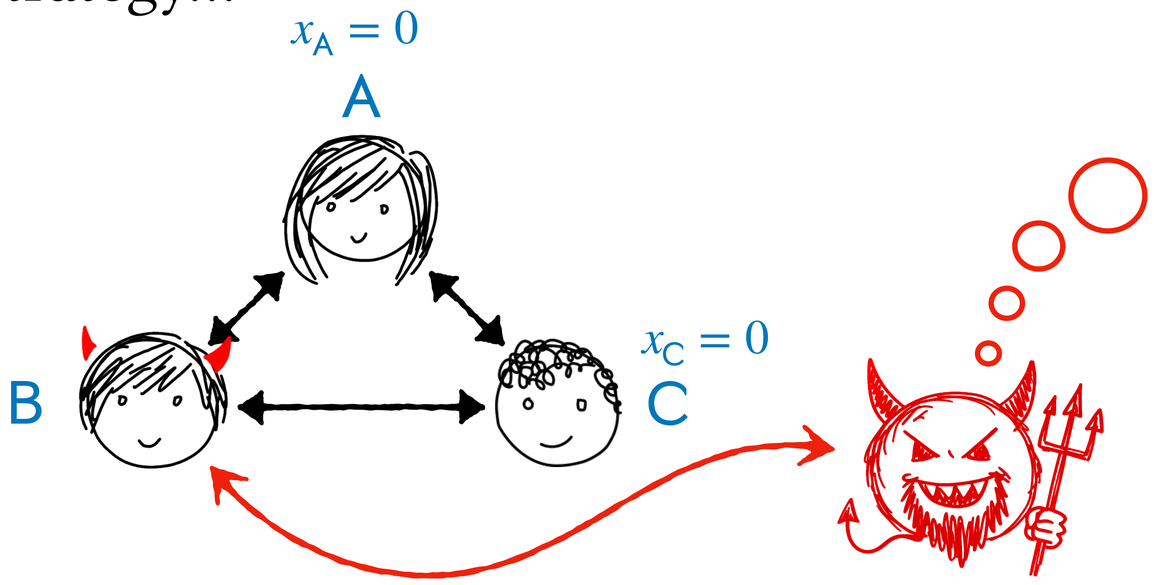
Let's set inputs for each of the six parties in our hexagon protocol.

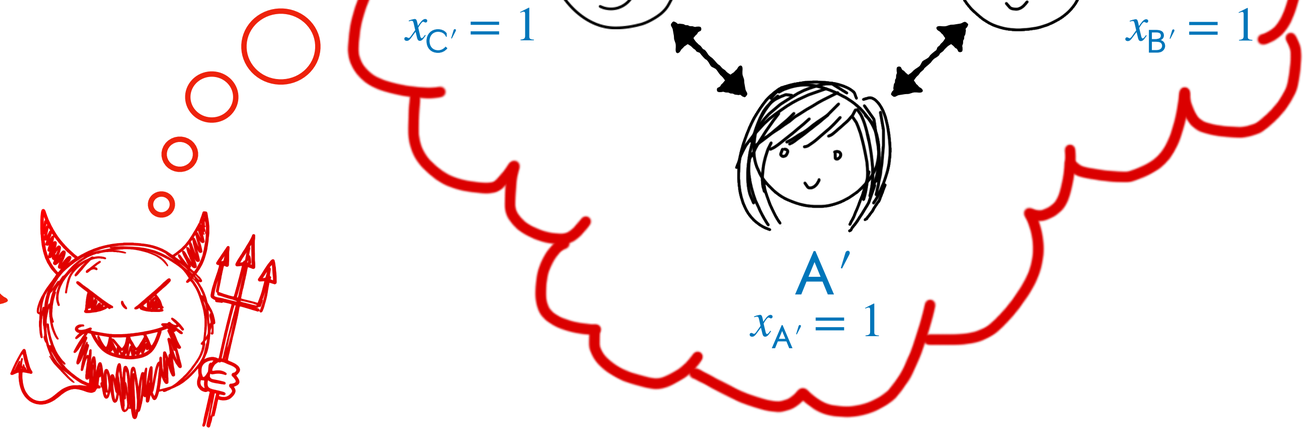


No Plain-Model BA for $t \geq n/3$

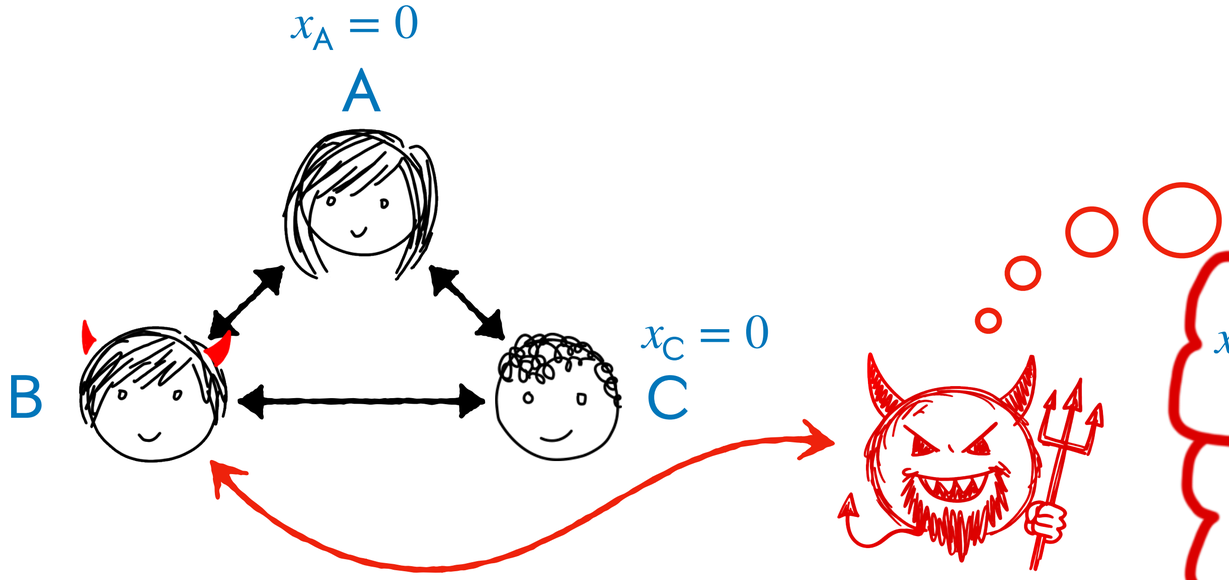
Claim 1: Parties A and C must both output 0 in the hexagon protocol.

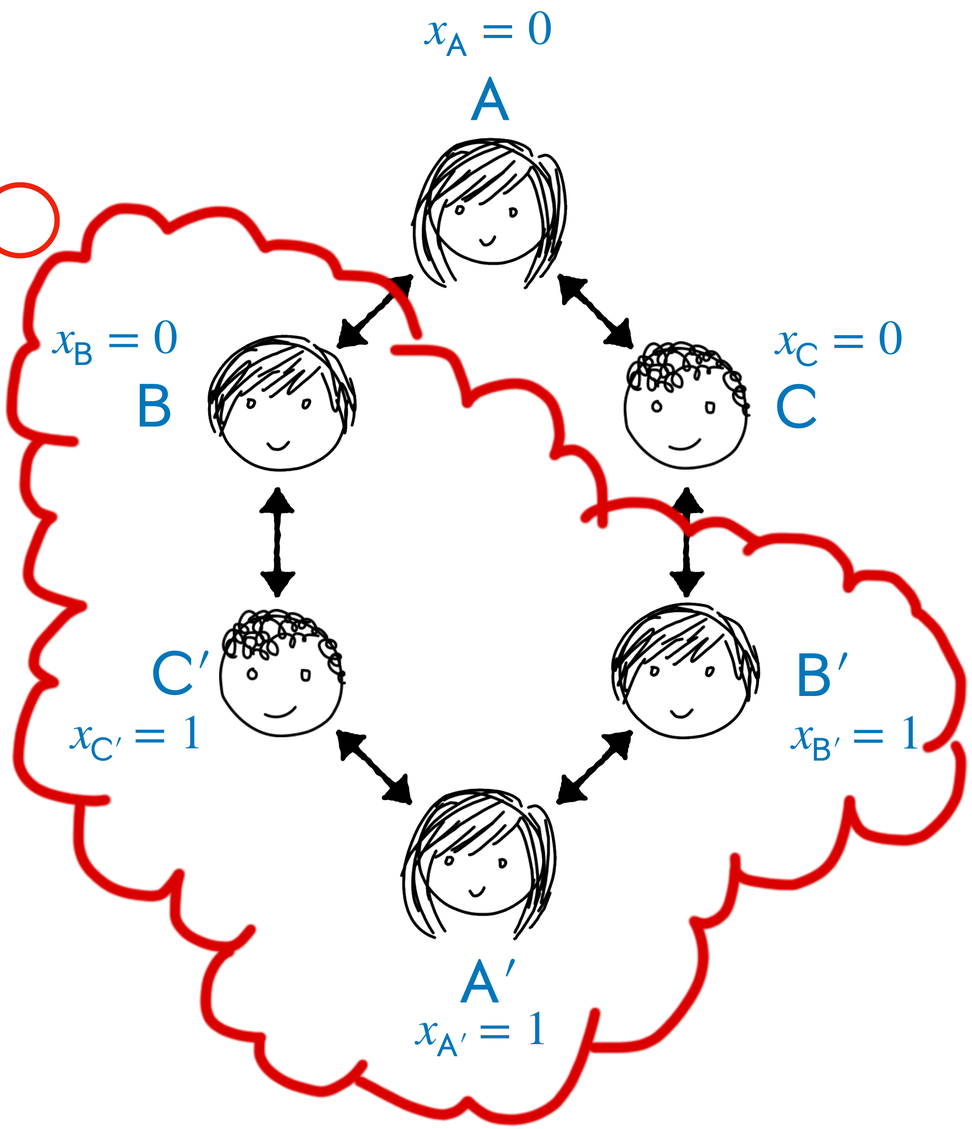
Proof: The joint view of A and C in the hexagon is *identically* distributed to the joint view of A and C in the triangle protocol when corrupt B uses the following strategy…





No Plain-Model BA for $t \geq n/3$





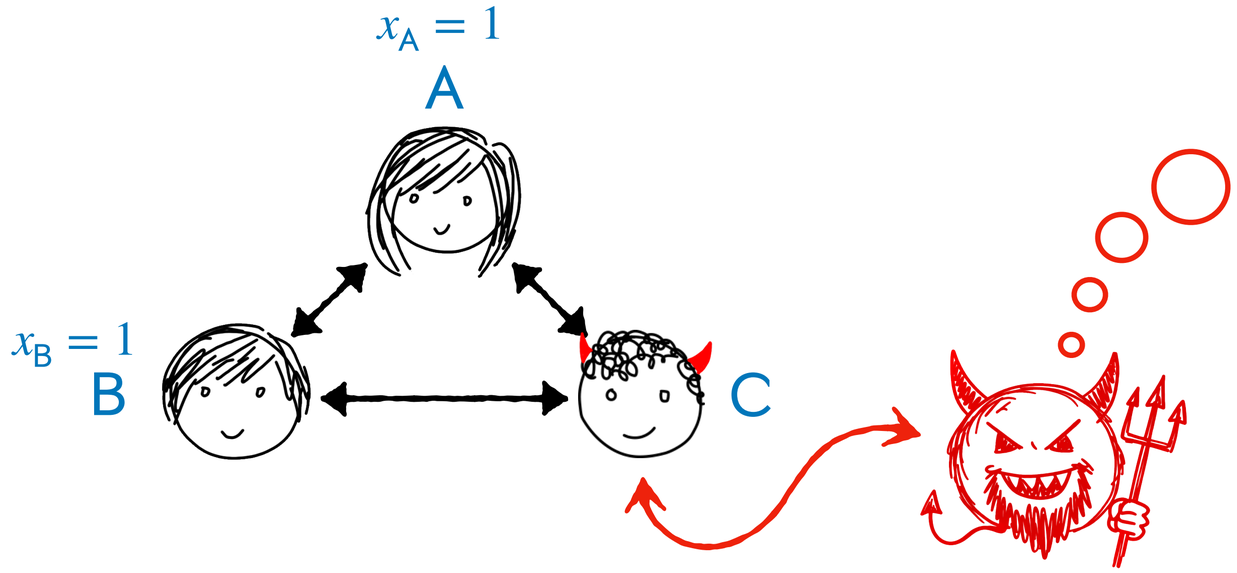
That is, $\mathcal{A}$ internally emulates an *honest* execution of A′, B, B′, C′ in the hexagon protocol on these inputs and sends the messages of hexagon parties B and B′ to triangle parties A and C.

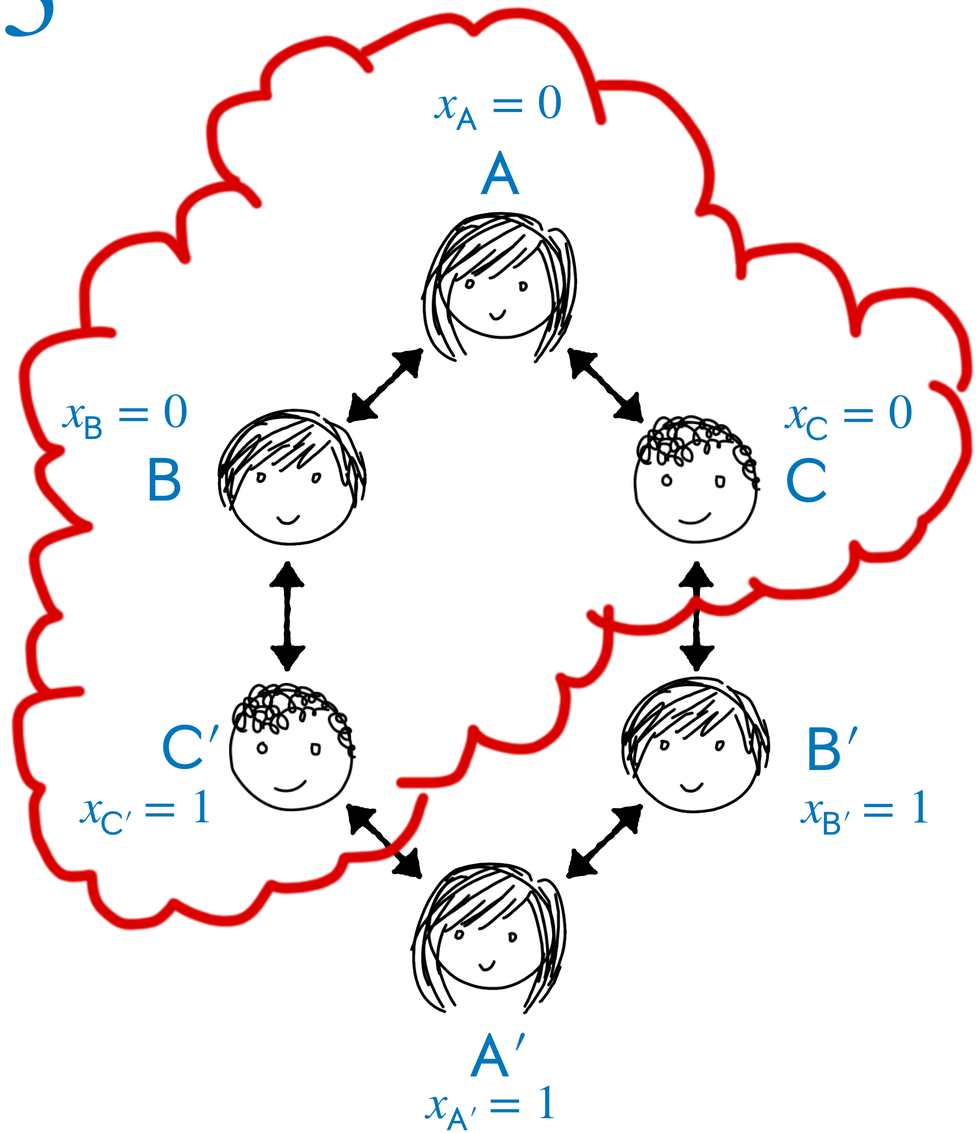
By the validity of π, A and C must both output 0 in the triangle, and therefore also in the hexagon. ∎

No Plain-Model BA for $t \geq n/3$

Claim 2: Parties A′ and B′ must both output 1 in the hexagon protocol.

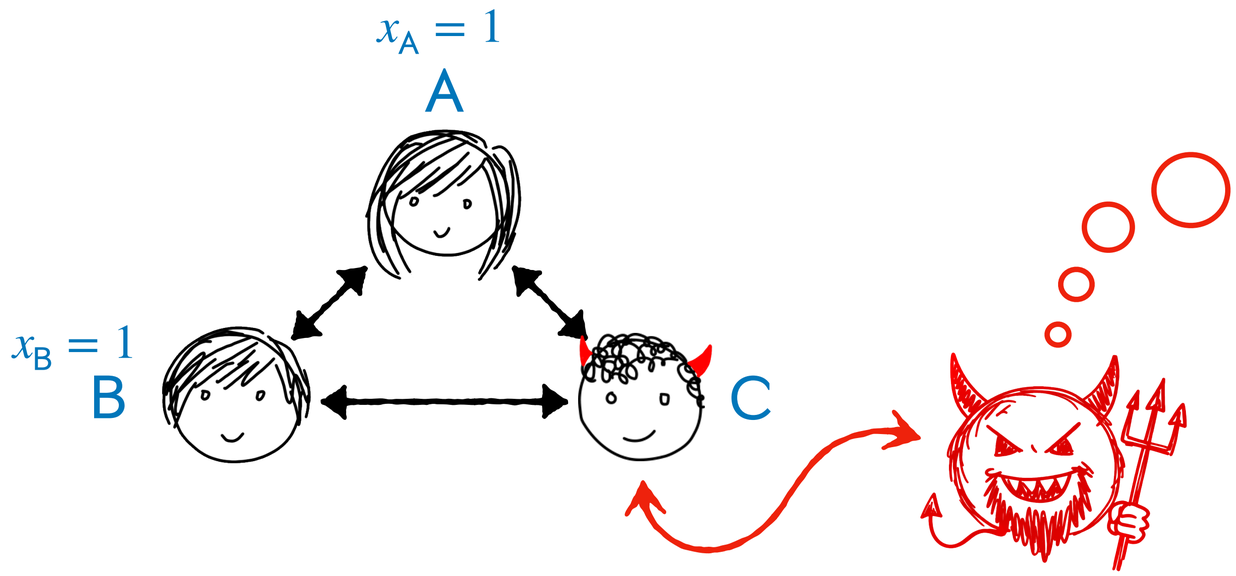
Proof: Symmetrically with the previous claim, the joint view of A′ and B′ in the hexagon is identically distributed to the joint view of A and B in the triangle protocol when corrupt C uses the following strategy…

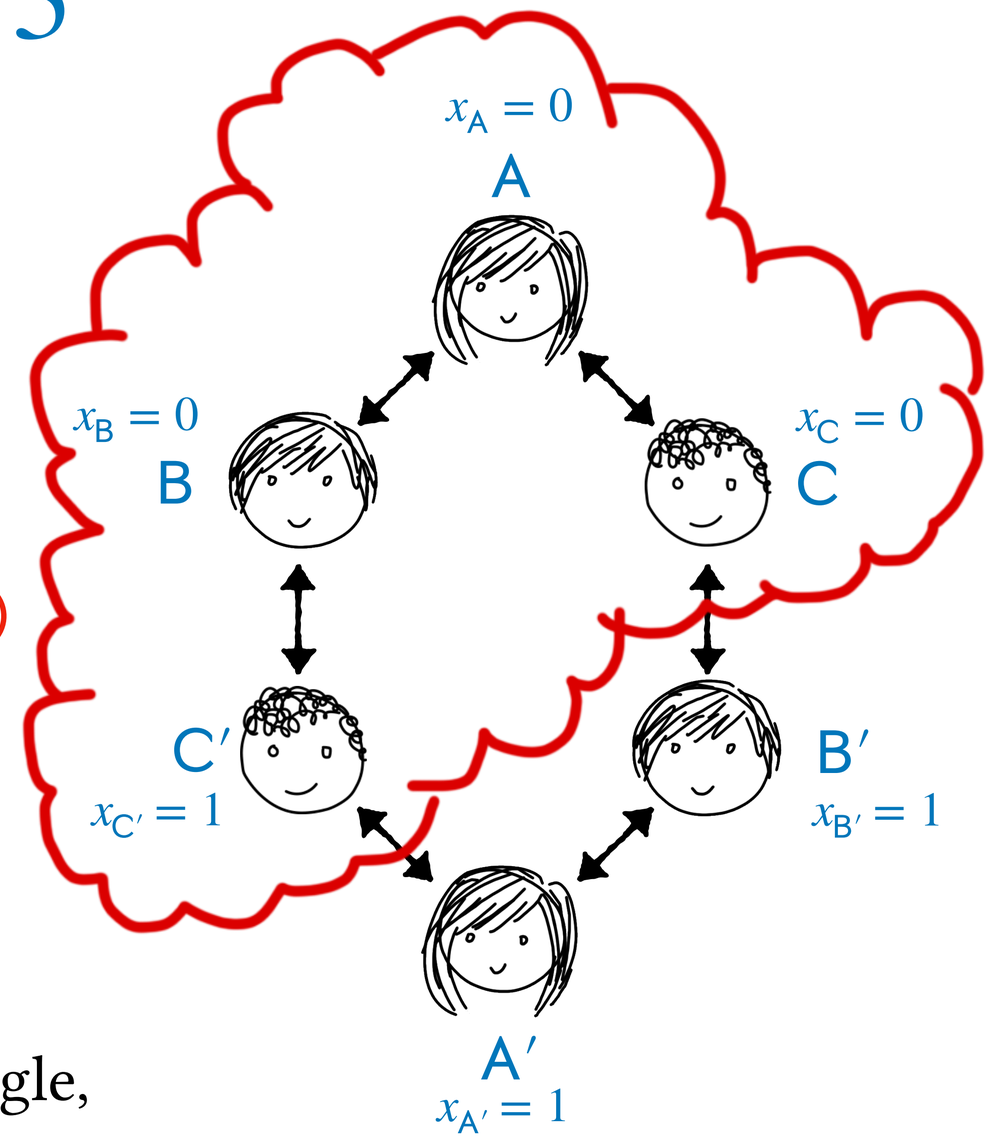




No Plain-Model BA for $t \geq n/3$

Proof: Symmetrically with the previous claim, the joint view of A′ and B′ in the hexagon is identically distributed to the joint view of A and B in the triangle protocol when corrupt C uses the following strategy…



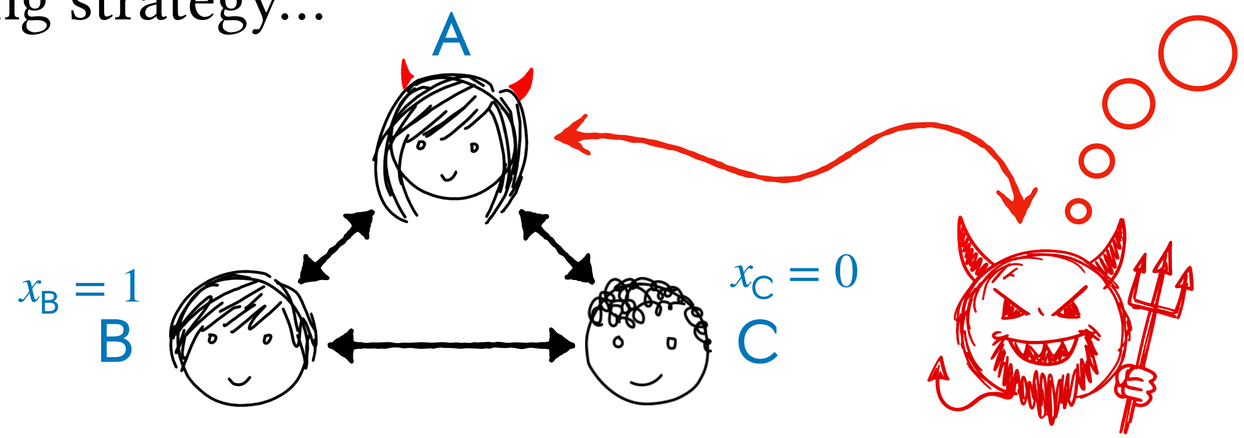


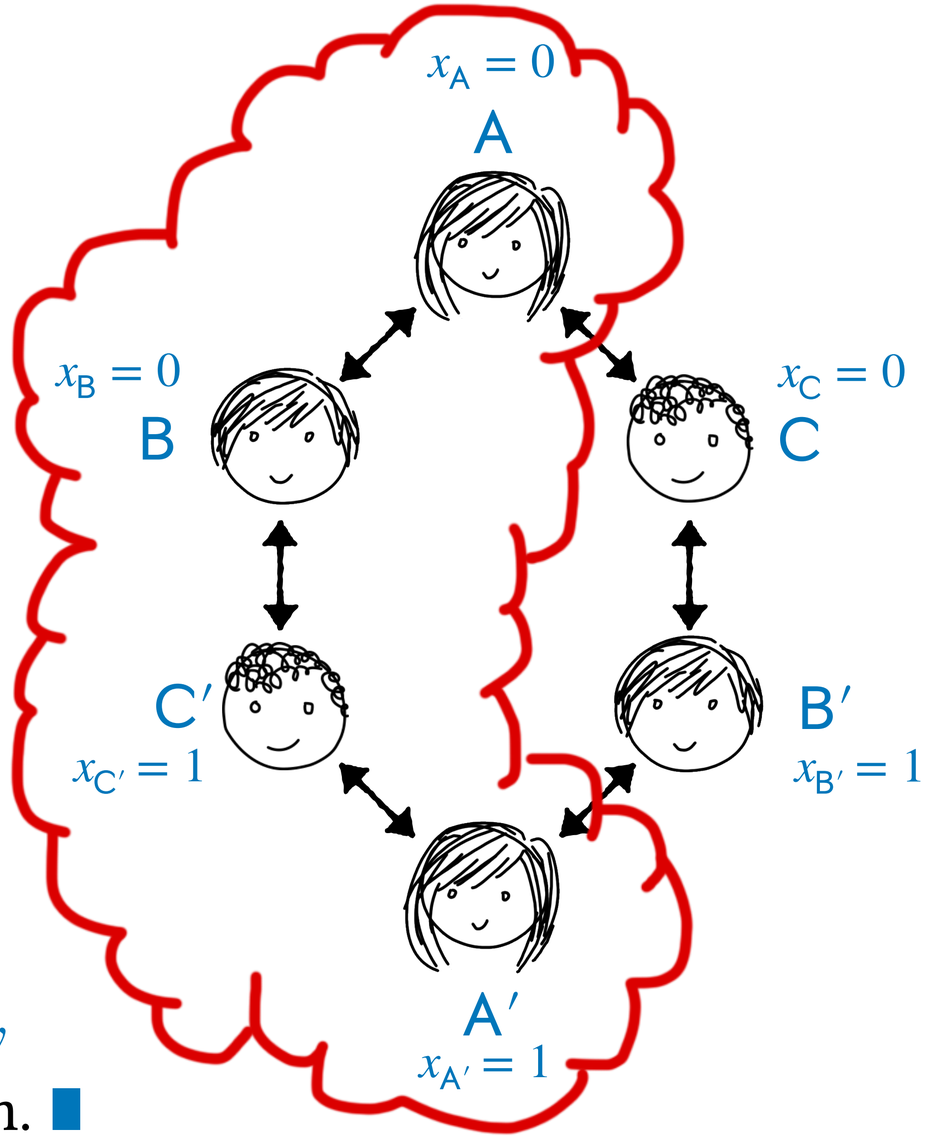
By the validity of π, A and B both output 1 in the triangle, and thus A′ and B′ must also do so in the hexagon. ∎

No Plain-Model BA for $t \geq n/3$

Claim 3: Parties B′ and C must output the same value in the hexagon protocol.

Proof: The joint view of B′ and C in the hexagon is identically distributed to the joint view of B and C in the triangle protocol when corrupt A uses the following strategy…





By the *consistency* of π, B and C both output the same y in the triangle; thus B′ and C must do so in the hexagon. ∎

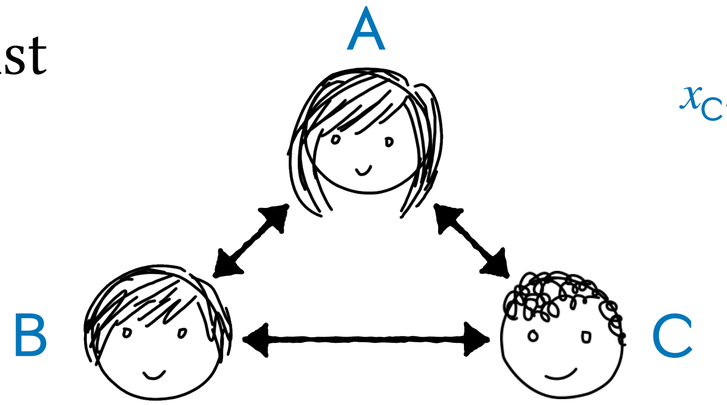
No Plain-Model BA for $t \geq n/3$

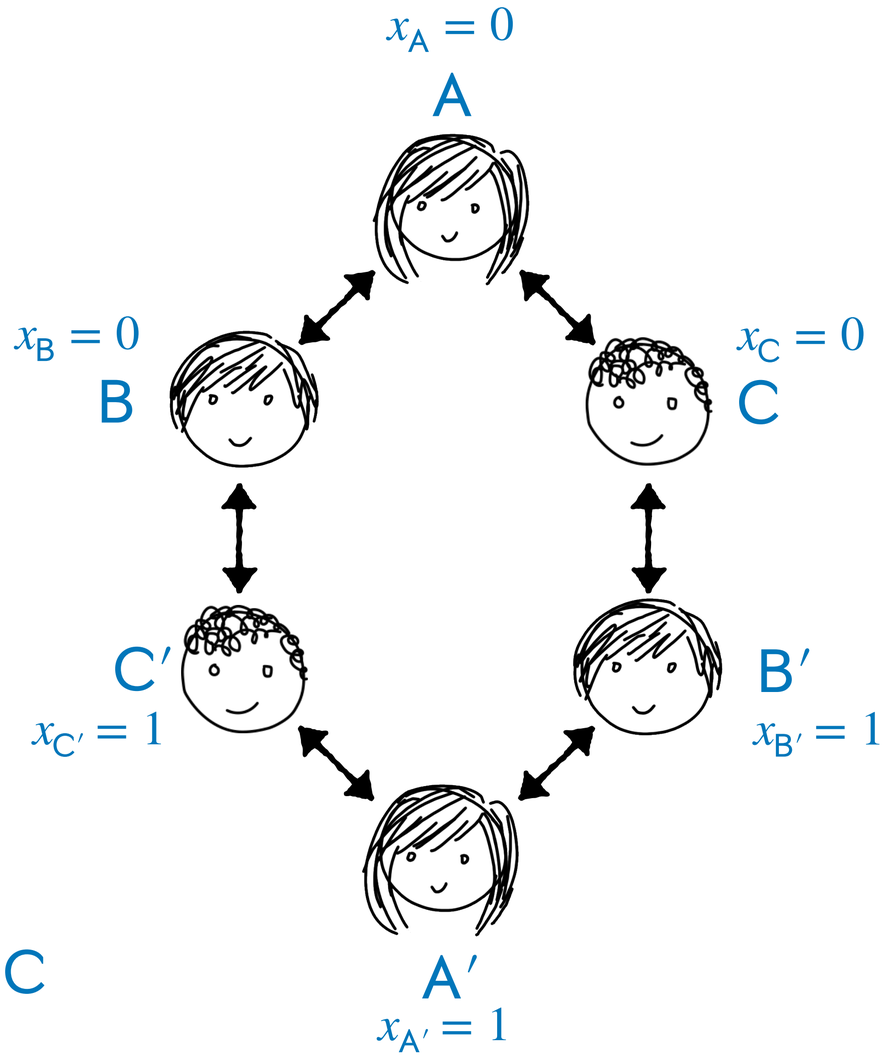
Putting it together:

By Claim 1, C must output 0 in the hexagon.
By Claim 2, B′ must output 1 in the hexagon.
By Claim 3, B′ and C must output the same thing.

This is a contradiction. The hexagon system cannot exist, and thus the triangle system cannot exist.

In other words, there cannot exist a 3-party 1-perfectly-secure BA protocol in the *plain* model. ∎





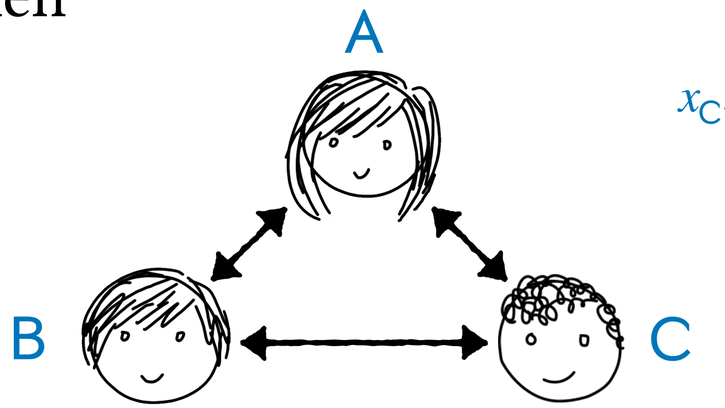
No Plain-Model BA for $t \geq n/3$

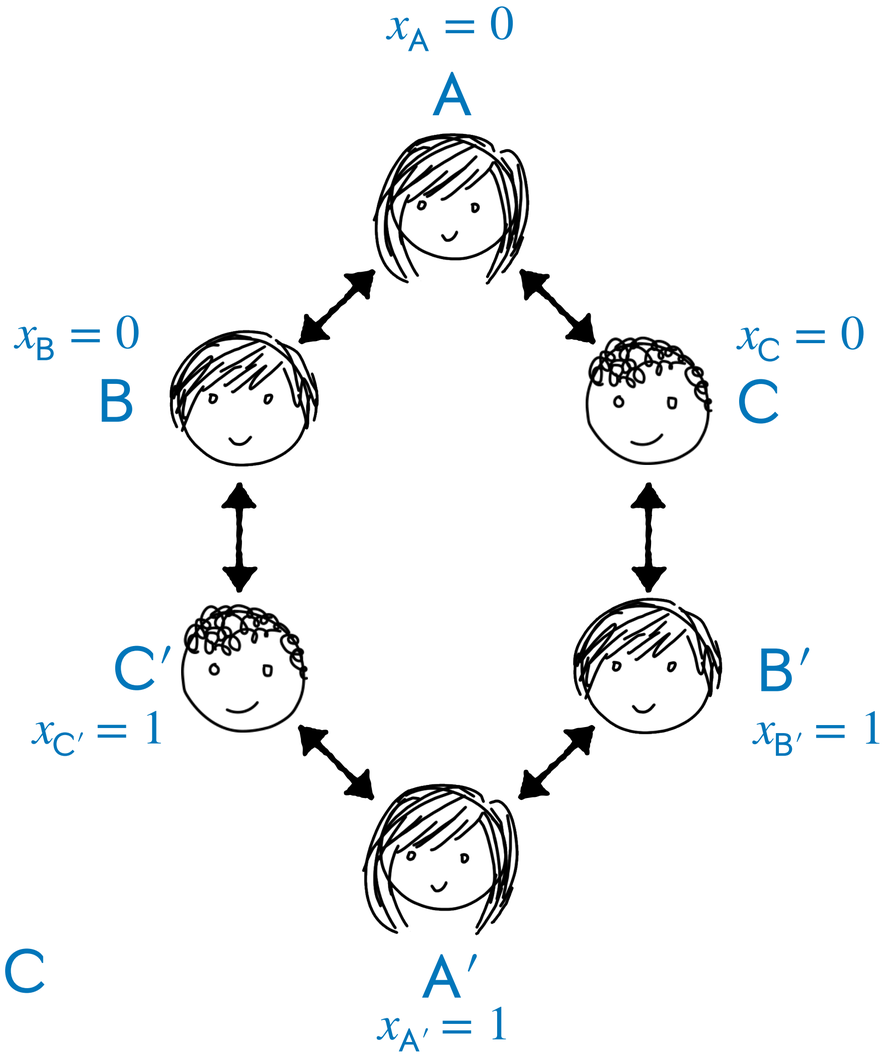
In other words, there cannot exist a 3-party 1-perfectly-secure BA protocol in the *plain* model. ∎

Nothing depends on the probabilities being exactly 0 or 1, or the adversary being bounded. Extending this proof to the computational case is easy.

Corollary 1: If $n/3 \leq t < n/2$ then there does not exist a t-secure n-party broadcast protocol in the plain model.

Proof: Thm 1 + Lem 2. ∎





Question: *what if* $n/2 \leq t < n$? **Hint:** t-security implies $(t-1)$-security.

…then, a possibility.

Plain-Model BA for $t < n/3$

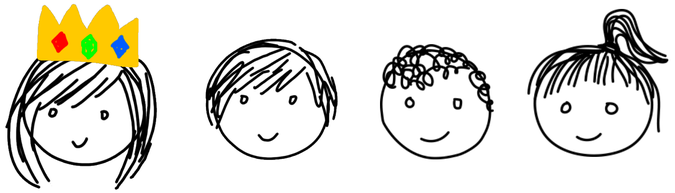
In in 1989, Berman, Garay, and Perry showed that the bound we just saw is tight.

(I couldn't find a photo of *or* a short story by Piotr Berman, so here's his RateMyProfessor profile instead)

Theorem 2: If at least $2n/3$ participants are honest (i.e. $t < n/3$), then there exists a t -perfectly-secure n-party BA protocol in the plain model, assuming auth. channels.

Proof: We will construct the protocol one piece at a time, and prove as we go along that it maintains certain important invariants. At the end we will show that these invariants yield consistency and validity.

Protocol π_1: In this protocol, P_1 is the king.



1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

Question: do we get *consistency* for $(v_1, \ldots, v_n)$?

Plain-Model BA for $t < n/3$

Protocol π_1: In this protocol, P_1 is the king.

1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

Claim 4: If all honest parties start with the same input b, then all honest $v_i = b$.

Proof: There are $n - t$ honest parties, by definition, and they all transmit b. Thus every honest P_i receives b from at least $n - t$ parties, and sets v_i to that value. ∎

Claim 5: If any honest P_i has $v_i = b$, then *every* honest P_j has either $v_j = b$ or $v_j = \perp$.

Proof: If an honest P_i has $v_i = b$, then it received b from at least $n - t \geq 2n/3$ parties. At most $t < n/3$ of these can be corrupt, which implies every *other* honest P_j received b from at least $n/3 \geq t + 1$ parties, and received $1 - b$ from at most $n - t - 1$ parties. ∎

Plain-Model BA for $t < n/3$

Protocol π_1: In this protocol, P_1 is the king.

1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

2. Each P_i sends v_i to everyone if $v_i \neq \perp$. If P_i receives a value b from $\geq n - t$ parties, then it sets $w_i := b,\ g_i = 2$. Otherwise, if P_i receives a value b from $\geq t + 1$ parties, it sets $w_i := b,\ g_i = 1$. Otherwise, P_i sets $w_i := x_i,\ g_i = 0$.

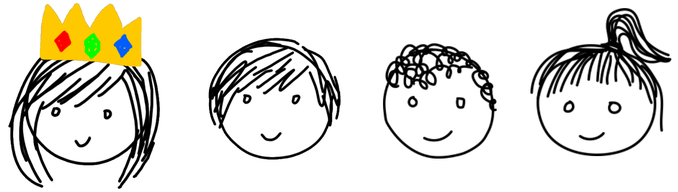
Claim 6: If all honest parties start with the same input b, then all honest $w_i = b,\ g_i = 2$.

Proof: Exactly like Claim 4. ∎

Note that we are beginning to see an invariant! If a round *starts* with validity, then it *ends* with validity. Now we just need to achieve consistency if we don't start that way.

Plain-Model BA for $t < n/3$

Protocol π_1: In this protocol, P_1 is the king.



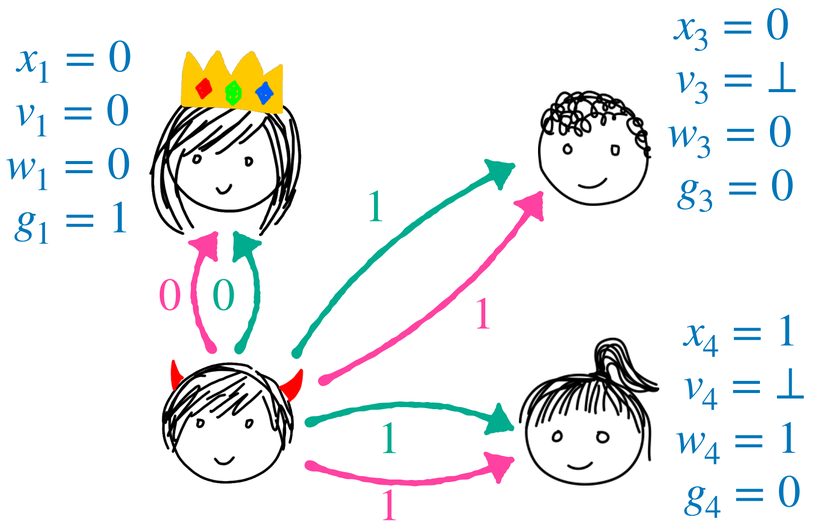
1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

2. Each P_i sends v_i to everyone if $v_i \neq \perp$. If P_i receives a value b from $\geq n - t$ parties, then it sets $w_i := b,\ g_i = 2$. Otherwise, if P_i receives a value b from $\geq t + 1$ parties, it sets $w_i := b,\ g_i = 1$. Otherwise, P_i sets $w_i := x_i,\ g_i = 0$.

Claim 7: If honest P_i has $w_i = b,\ g_i = 2$, then *every* honest P_j has $w_j = b$ and $g_j \in \{1,2\}$.

Proof: P_i *must* have received b from at least $n - t \geq 2n/3$ parties in round 2. As in claim 5, this implies that every *other* honest P_j received b from at least $n/3 \geq t + 1$ parties, which implies that every *other* honest P_j set $w_j = b$ and $g_j \in \{1,2\}$. ∎

Plain-Model BA for $t < n/3$



Protocol π_1: In this protocol, P_1 is the king.

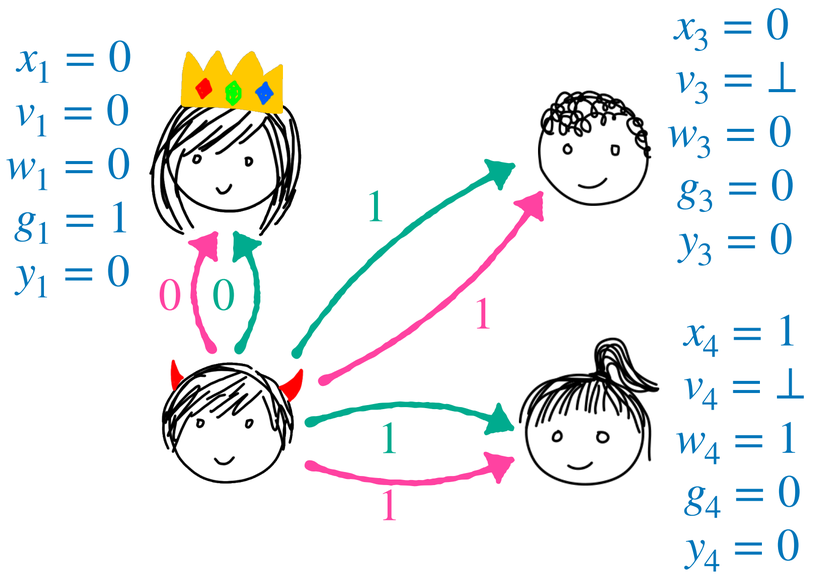
1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

2. Each P_i sends v_i to everyone if $v_i \neq \perp$. If P_i receives a value b from $\geq n - t$ parties, then it sets $w_i := b,\ g_i = 2$. Otherwise, if P_i receives a value b from $\geq t + 1$ parties, it sets $w_i := b,\ g_i = 1$. Otherwise, P_i sets $w_i := x_i,\ g_i = 0$.

Problem: An honest P_i might have $w_i = b,\ g_i = 1$ while honest P_j has $w_j = 1 - b,\ g_j = 0$.

Consider the above example with four parties and P_2 corrupt. In round 1, P_2 sends inconsistent messages. In round 2, the only honest party that speaks is P_1 (it sends $v_1 = 0$), but P_2 can send inconsistent messages again.

Plain-Model BA for $t < n/3$



Protocol π_1: In this protocol, P_1 is the king.

1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

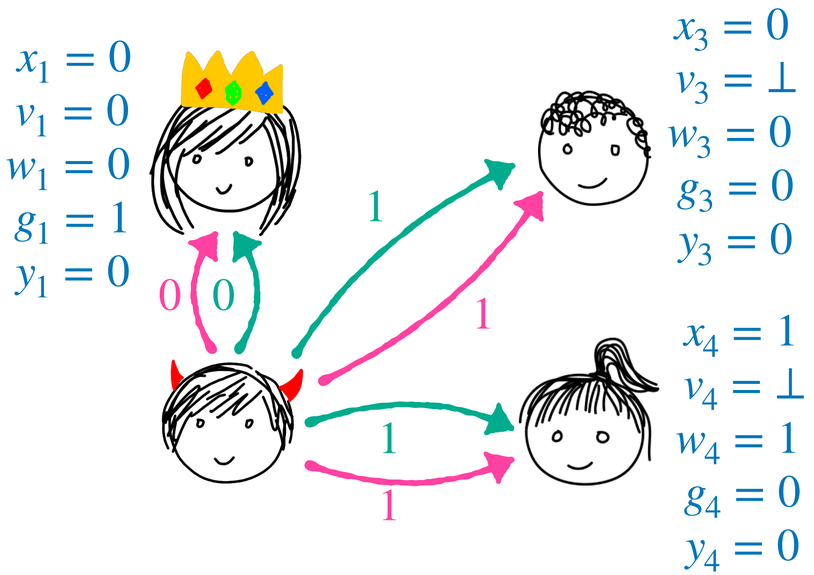
2. Each P_i sends v_i to everyone if $v_i \neq \perp$. If P_i receives a value b from $\geq n - t$ parties, then it sets $w_i := b,\ g_i = 2$. Otherwise, if P_i receives a value b from $\geq t + 1$ parties, it sets $w_i := b,\ g_i = 1$. Otherwise, P_i sets $w_i := x_i,\ g_i = 0$.

3. The king (P_1) sends w_1 to everyone. If P_i receives b from P_1, and $g_i < 2$, then P_i sets $y_i := b$. Otherwise $y_i := w_i$. Finally, every P_i outputs y_i.

Claim 8: If all honest parties start with the same input b, then all honest $y_i = b$.

Proof: By Claim 6, every honest party has $w_i = b,\ g_i = 2$ and round 3 produces no effect. ∎

Plain-Model BA for $t < n/3$



Protocol π_1: In this protocol, P_1 is the king.

1. Each P_i sends x_i to everyone. If P_i receives a value b from $\geq n - t$ parties, then it sets $v_i := b$. Otherwise it sets $v_i := \perp$ (undecided).

2. Each P_i sends v_i to everyone if $v_i \neq \perp$. If P_i receives a value b from $\geq n - t$ parties, then it sets $w_i := b,\ g_i = 2$. Otherwise, if P_i receives a value b from $\geq t + 1$ parties, it sets $w_i := b,\ g_i = 1$. Otherwise, P_i sets $w_i := x_i,\ g_i = 0$.

3. The king (P_1) sends w_1 to everyone. If P_i receives b from P_1, and $g_i < 2$, then P_i sets $y_i := b$. Otherwise $y_i := w_i$. Finally, every P_i outputs y_i.

Claim 9: If the king (P_1) is honest, then the values $(y_1, \ldots, y_n)$ are consistent!

Plain-Model BA for $t < n/3$

2.

Claim 7: If honest P_i has $w_i = b$, $g_i = 2$, then *every* honest P_j has $w_j = b$ and $g_j \in \{1,2\}$.

Proof: P_i *must* have received b from at least $n - t \geq 2n/3$ parties in round 2. As in claim 5, this implies that every *other* honest P_j received b from at least $n/3 \geq t + 1$ parties, which implies that every *other* honest P_j set $w_j = b$ and $g_j \in \{1,2\}$. ∎

3.

sets $y_i := b$. Otherwise $y_i := w_i$. Finally, every P_i outputs y_i.

Claim 9: If the king (P_1) is honest, then the values $(y_1, \ldots, y_n)$ are consistent!

Proof: If for some honest P_i we have $g_i < 2$, and P_1 is honest, then $y_i = w_1$.

If for *all* honest P_i we have $g_i < 2$, and P_1 is honest, then *all* honest $y_i = w_1$.

If for *some* honest P_i we have $w_i = b'$, $g_i = 2$, then, as in our argument for Claim 7, P_i *must* have received b' from at least $n - t \geq 2n/3$ parties in round 2, which implies that P_1 received b' from at least $n/3 \geq t + 1$ parties. Thus if P_1 is honest, it set $w_1 := b'$ and transmitted that value in round 3. Consistency then follows from Claim 7.

Plain-Model BA for $t < n/3$

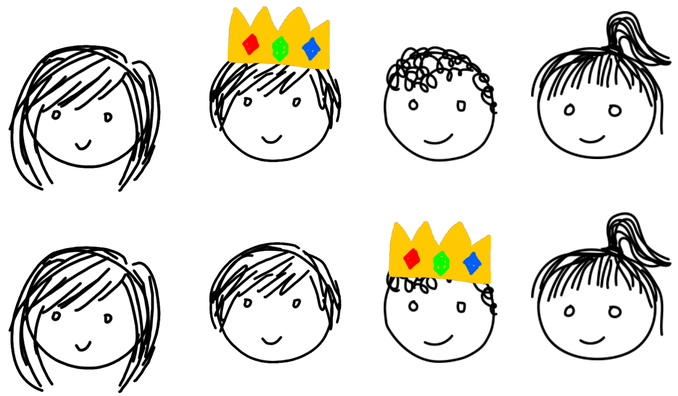
So by Claims 8 and 9, we have the following invariants:

- If the inputs of the honest parties to π_1 are the *same* value b, then all honest parties in π_1 output b. In other words π_1 achieves validity.
- If the king (P_1) is honest, then the outputs of of the honest parties from π_1 are always consistent.

But what do we do if P_1 is corrupt? We achieve validity but not consistency...

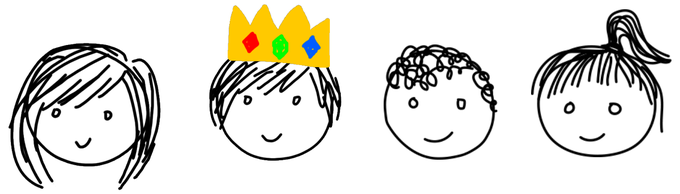
What if we use the outputs of π_1 as the inputs for π_2, which is exactly the same as π_1 except that P_2 is king.

Then we use the outputs of π_2 as the inputs for π_3, and so on until π_{t+1} finishes (its outputs are the true outputs).

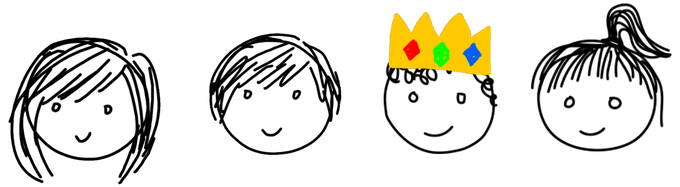


Plain-Model BA for $t < n/3$

What if we use the outputs of π_1 as the inputs for π_2, which is exactly the same as π_1 except that P_2 is king.



Then we use the outputs of π_2 as the inputs for π_3, and so on until π_{t+1} finishes (its outputs are the true outputs).



This is the BGP protocol. Does it have the properties we want?

Validity is preserved through all iterations, and thus the overall protocol achieves it.

Since we run $t + 1$ iterations of the protocol, and there are at most t corruptions, at least *one* king must be honest. In the iteration where the king is honest, *consistency* is achieved. Consistency is *preserved* due to the validity of all subsequent iterations.

Thus the overall protocol achieves **Consistency** as well, and is t-perfectly-secure BA. ∎

Plain-Model *Broadcast* for $t < n/3$

Recall **Theorem 2:** If $t < n/3$, then there exists a t-perfectly-secure n-party BA protocol in the plain model, assuming authenticated channels.

Recall **Lemma 1:** Let $t < n$. If there exists t-secure BA, then there exists t-secure broadcast.

Corollary 2: If $t < n/3$ then there exists a t-secure n-party *broadcast* protocol in the plain model, assuming authenticated channels.

Proof: By Theorem 2 and Lemma 1. ∎

Where we Stand

- We have shown that no plain-model protocol can achieve the property-based notions of BA or broadcast if $t \geq n/3$ parties are corrupt.
- This is a strong bound: it rules out any protocol that realizes any functionality that *implies* these properties, regardless of whether OWFs exist.
- The broadcast functionality obviously implies these properties. *Can you think of other functionalities that do?*
- Consider the property of *identifiable abort:* If a functionality has identifiable abort, then a corrupt party can prevent it from producing output, but the honest parties learn the identity of the party that did so. It turns out that *any* functionality with this property implies broadcast.
- So this bound tells us a pretty important category of functionalities is unrealizable!

What can we Do About it?

- Last time we ran into an impossibility, we changed the model by *bounding* the computational power of the adversary and then making *assumptions* about what the bounded adversary is able to compute.

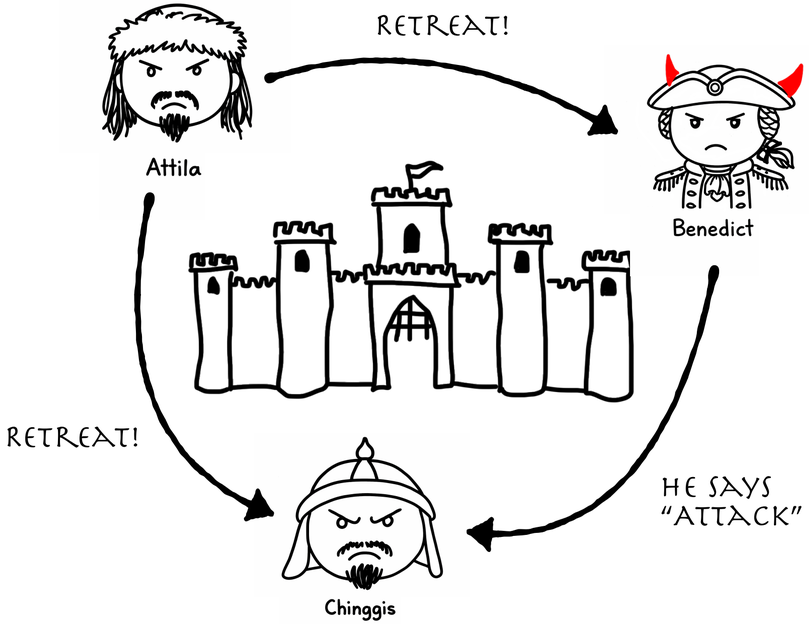
- This time, our impossibility proof wasn't sensitive to the computational power of the adversary. The adversary in the hexagon protocol only had to imagine a constant number of honest parties running with particular inputs. As long as the adversary has roughly as much computational power as the honest parties do, this strategy will work for it!

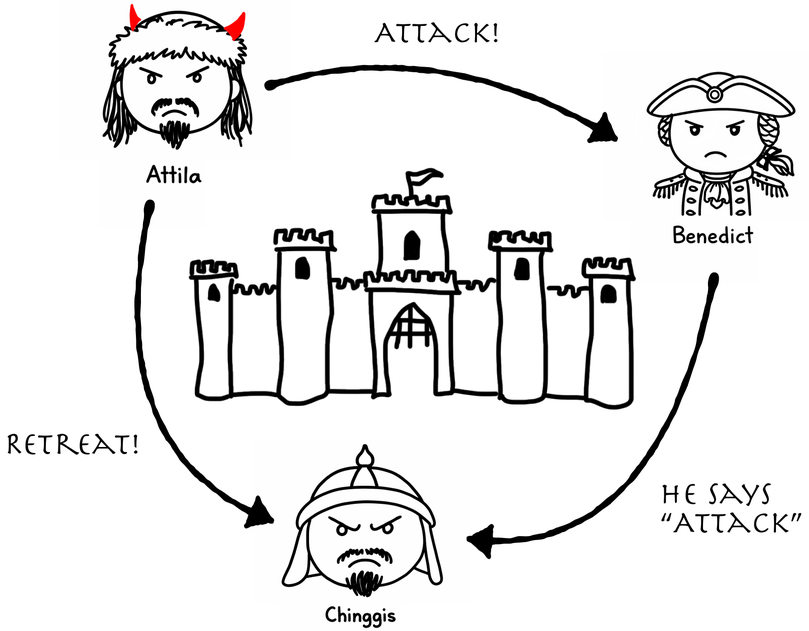
- We will have to change our model in another way.

- To get some inspiration, let's look back at the generals.

MPC Lays Siege

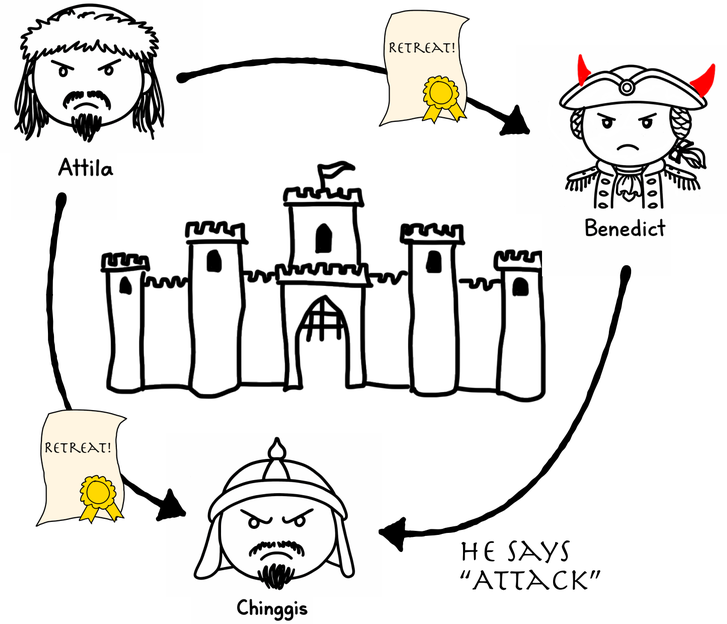
- Last class we proposed that to Chinggis, the two scenarios at left appear identical.
- Implicitly we were using something called the “oral message model,” in which of the messages are relayed vocally by trusted messengers. Benedict can *repeat* Attila’s message, but Chinggis can’t verify that the message was originally sent from Attila.

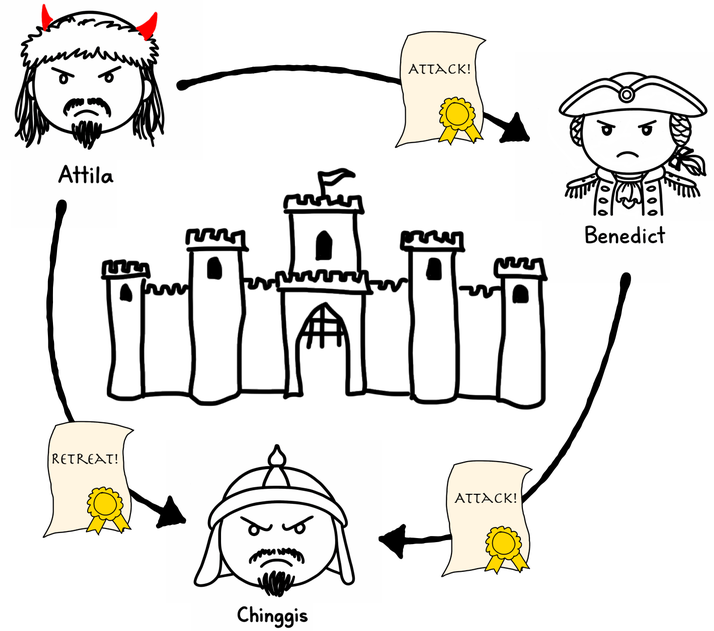




MPC Lays Siege

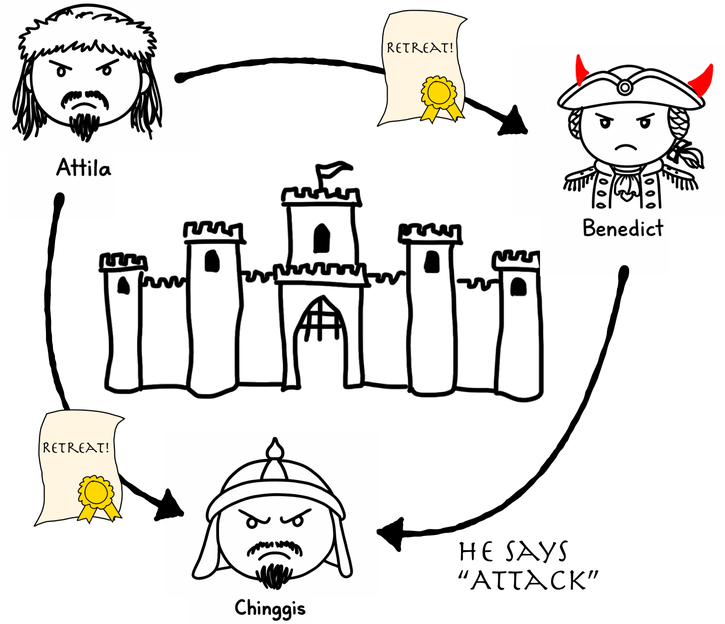
- Last class we proposed that to Chinggis, the two scenarios at left appear identical.
- Implicitly we were using something called the "oral message model," in which of the messages are relayed vocally by trusted messengers. Benedict can *repeat* Attila's message, but Chinggis can't verify that the message was originally sent from Attila.
- What if they only trust *written* messages that have been signed by the sending general? Benedict cannot *forge* Attila's signature, but he can *forward* a signed document to Chinggis. *Does this help?*
- Cleary the generals need to meet *in advance* and learn one another's signatures. *Is this enough?*
- Chinggis and Benedict also need to be sure that Attila will use the *same* signature when communicating with each of them!

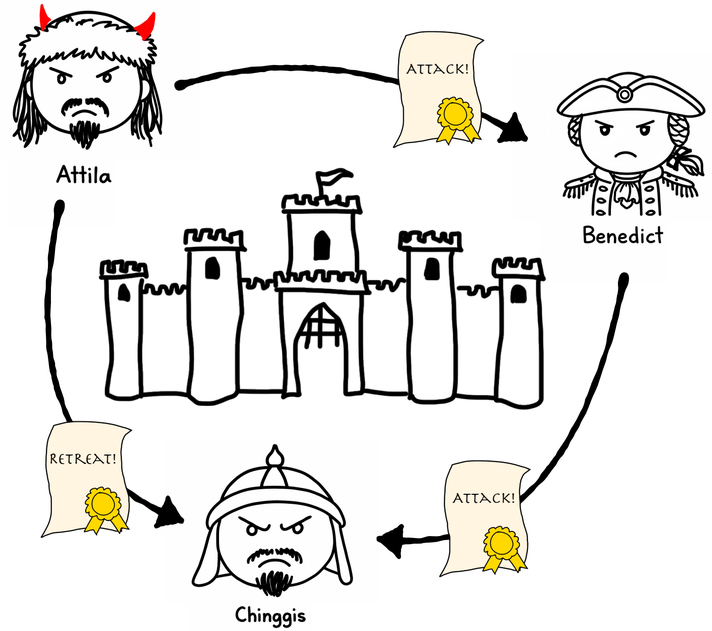




MPC Lays Siege

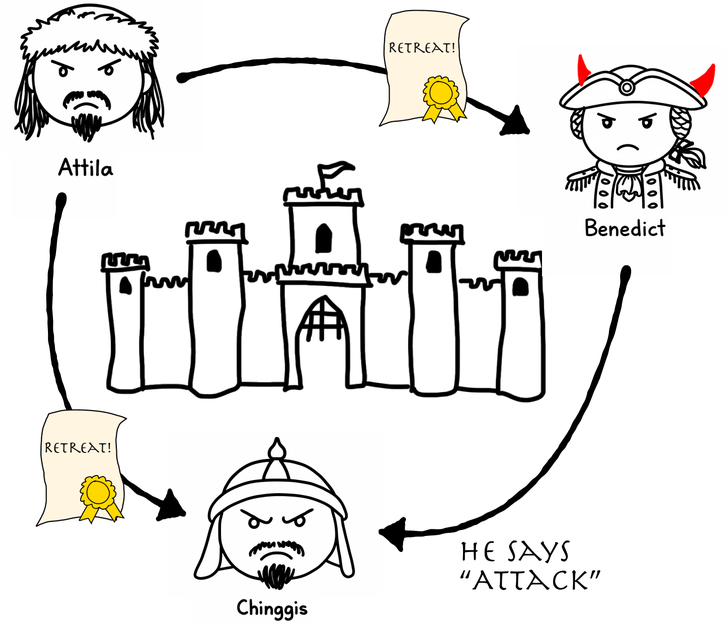
- We'll formalize these ideas and show how they can be used to achieve broadcast in the presence of a dishonest majority.
- First, we'll introduce the notion of *signature schemes*. Like public-key encryption, this is a fundamentally cryptographic notion, but instead of guaranteeing privacy, we guarantee *authenticity*.
- In particular, signature schemes use a *public key* to verify that a matching *secret key* was used to sign a message, to which the signature is attached.
- The second piece of the puzzle is a way to reliably *distribute* and *agree upon* public keys.
- The latter task is equivalent to broadcast! So we're back where we started, and we know that no protocol will help us. Instead, we'll change the model to encode the assumption that it can be done.

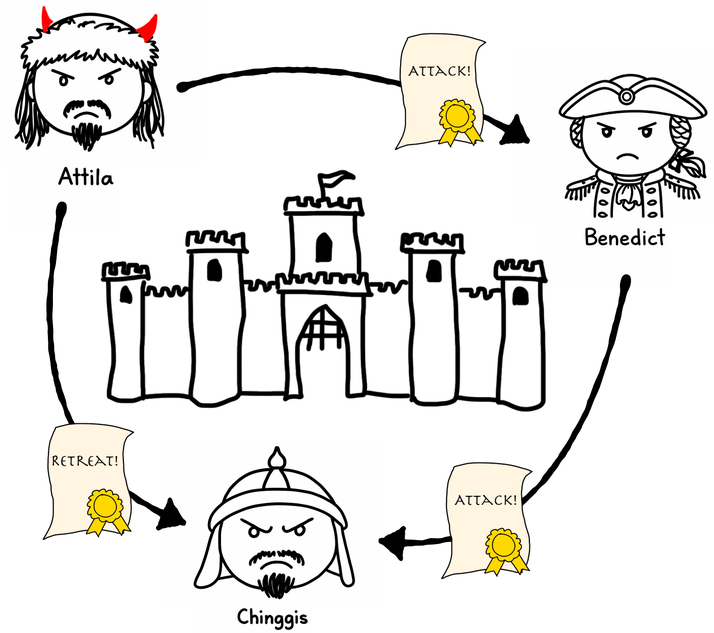




MPC Lays Siege

- So, if we're assuming a *public-key infrastructure* with broadcast-like properties, what have we achieved?
- We can move the trusted broadcast to *setup* time, before the protocol runs or the inputs are known. If our signature scheme has *reusable keys*, then we only have to do this setup once.
- How do public-key infrastructures work in practice?
- For small groups of participants, physical meetings might be possible.
- For larger groups, we often use trusted distributors. A small number of public keys usually ship with your operating system. These are used to fetch additional public keys from trusted sources.
- These are heuristics (and there are others). The PKI model allows us to fit them into our proofs cleanly.





The PKI Model

- In this class, we will use two slightly different versions of *the PKI model.*
- Informally, we will assume that every party begins the experiment knowing the public keys of all other parties, and that all keys have been sampled appropriately.
- When we need to formalize this, we will use a simple *PKI functionality* $\mathcal{F}_{\text{PKI}}$ which accepts a key from one party and transmits it reliably to all of the others.
- This is exactly the same as the broadcast functionality, except for its name. In the literature, there are other kinds of PKI functionalities, but we won't worry about them in class.
- Thus, when we construct broadcast in the PKI model, we're effectively doing *broadcast extension.*
- OT extension turns κ OT instances into polynomially many OT instances. Broadcast extension turns n broadcasts (one for each party, at the beginning of the experiment) into polynomially many broadcasts.

CS4501 Cryptographic Protocols
Lecture 20: Perfect BA, PKI

https://jackdoerner.net/teaching/#2026/Spring/CS4501