Contract Signing
Contract Signing

Contract signing protocols enable two parties, Alice and Bob, to sign a binding contract.

It is required that after the execution of the protocol:

- Alice will be able to prove that Bob signed.
- Bob will be able to prove that Alice signed.
- **Fairness**: Alice can prove that Bob signed if and only if Bob can prove that Alice signed.
  - If one party cheats, he/she will not be able to take advantage of that in order to gain an advantage over the other party.
  - He/she will not be able to prove that the other party signed, if the other party is not able to prove that he/she signed.
Example: An insecure protocol for signing contracts.

Given a contract $m$, Alice sends her signature on the contract, $S_A(m)$, to Bob and then Bob sends his signature, $S_B(m)$, to Alice.

This protocol is not secure because Bob can choose not to send his signature to Alice, and therefore he will be able to prove that the contract is valid, if he wants to, and when it’s in his best interests. On the other hand, Alice cannot prove that the contract is valid, but also cannot cancel the contract.
“Improved” (still insecure) protocols:

1. Alice and Bob sign. Alice sends the signature to Bob, then Bob sends to Alice, but the signature becomes valid only if Alice signs that she got Bob’s signature.

2. Signing 50 times in 50 rounds, such that in each round Alice and Bob provide another signature to the contract (mixed with the round number), and all the 50 are required to prove signature.

3. Signing 50 times, but each time the signature is on the message along with all previous signatures, i.e., Alice signs the contract, Bob signs the contract with Alice’s signature, Alice signs the contract with the two signatures, etc.
Theorem: A deterministic protocol for signing contracts without the participation of a third party does not exist.

Proof: Assume that such a protocol exists. While running the protocol, there should be some step $n$, after which both parties have the signatures, and none of them know it before it.

Without loss of generality, the $n$’th message is sent from Alice to Bob. Since Alice does not receive any information in step $n$, she knows the signature already after step $n - 1$. Contradiction.

Therefore, there is no protocol where both parties learn the signatures together at exactly the same time.

On the other hand, if one of the learns it first, he has no reason to continue the protocol and let the other party learn it as well.

QED
Using a Trusted Center for Contract Signing

When the third party knows about all the contracts which are being signed, it is possible to use the following simple protocol:

- Alice and Bob are interested in signing a contract \( m \).
- Alice signs \( m \) and sends \( m, S_A(m) \) to the center.
- Bob signs \( m \) and sends \( m, S_B(m) \) to the center.
- The center signs the information he received and sends

\[
m, S_A(m), S_B(m), S_C(m)
\]


to Alice and to Bob.
- The contract is valid only if the center’s signature is given in addition to Alice’s and Bob’s signatures.

**Disadvantage:** The center is involved in all contract signing, and knows all their contents.
There is no need for the center to be involved in all contract signing. It is enough to involve him in canceling contracts whose signing was not completed:

- Alice and Bob agree on a contract $m$ and add to it an additional item which includes the contract number, the validity date and a statement that the contract becomes binding on this date only if it was not canceled by Alice or Bob prior to this date (this date is usually 24 hours or some other reasonable period of time after the time of agreement).

- Alice signs $m$ and sends $S_A(m)$ to Bob.

- Bob signs $m$ and sends $S_B(m)$ to Alice.
Using a Trusted Center for Contract Signing (cont.)

- If one of the parties did not receive the other’s signature a reasonable time before the contract becomes valid (e.g., one hour), he sends a cancellation message to the center, which includes the canceled contract number and the parties involved. The center informs the other side about the cancellation of the contract.

Advantages:

- The center does not know about contracts which were signed successfully.

- The center knows about canceled contracts, but does not know their contents.
Bad Contract Signing Without a Trusted Center

The idea is to ensure that no party know much more than the other.

The protocol thus sends one bit of each signature at a time, i.e., Alice sends the first bit of the signature, then Bob sends the first bit of his signature, then Alice sends the second bit, etc.

Such a protocol ensures that both parties know about the same number of bits of the signature of the other party, and thus, if one quits the protocol, the difference in information is small.

If one finds the signature and quits the protocol, the other party misses only one bits, which can be easily guessed.

If he quits with \( l \) unknown bits, and should spend \( 2^l \) time to search for the full signature, the other party has at most \( l + 1 \) missing bits, and can thus find the signature in at most \( 2^{l+1} \) time, which is at most twice as the time required for the quitting party.
Bad Contract Signing Without a Trusted Center (cont.)

Why is this a bad protocol?
Contract Signing Without a Trusted Center

The following protocol enables contract signing without using a trusted center.

The protocol is designed such that if one of the parties quits the protocol in any given time his advantage will be small, i.e., both parties will be able to complete the signatures at about the same time or it will be hard for both to complete the signature.

The protocol uses $OT_2^1$ and a public key cryptosystem. Also it assumes that both parties have the same computational power.

See:

Contract Signing Without a Trusted Center (cont.)

The protocol:

1. Alice chooses $2n$ random $\ell$-bit keys $a_1, a_2, \ldots, a_{2n}$ and commits to each of them by the commitment $c_i^A$, e.g., by encrypting a standard message under these keys:

   $$c_i^A = \text{DES}_{a_i}(s), i = 1, \ldots, 2n.$$

2. Alice declares that: “I am obligated to this contract if Bob can present two values $K_i^A, K_{i+n}^A$ for some $i, 1 \leq i \leq n$, such that

   $$c_i^A = \text{DES}_{K_i^A}(s) \text{ and } c_{i+n}^A = \text{DES}_{K_{i+n}^A}(s).$$

   Alice signs her declaration with a public key signature, and sends the declaration and the signature to Bob.

3. Bob performs steps 1,2 similarly to Alice, but with the values $b_1, \ldots, b_{2n}$ and the commitments $c_1^B, \ldots, c_{2n}^B$. 
Contract Signing Without a Trusted Center (cont.)

4. For every value \( i, 1 \leq i \leq n \):
   - Alice sends \( a_i \) and \( a_{i+n} \) to Bob using \( OT_2^1 \).
   - Bob sends \( b_i \) and \( b_{i+n} \) to Alice using \( OT_2^1 \).

Each of them checks the value that he got, e.g., whether \( c_i^A = DES_{a_i}(s) \), or \( c_i^B = DES_{b_i}(s) \).

5. For every value \( j, 1 \leq j \leq \ell \):
   - Alice sends the \( j \)’th bit of all the \( a_i \)'s to Bob (and Bob checks whether the new bits match the bits he got earlier).
   - Bob sends the \( j \)’th bit of all the \( b_i \)'s to Alice (and Alice checks whether the new bits match the bits she got earlier).

When one of the parties discovers that the other one is trying to cheat (in step 4 or in step 5), he immediately quits the protocol (as self defense).
Security:

- At the end of the protocol Alice and Bob know all the $a_i$'s and $b_i$'s, thus they have valid signatures.

- If one of the parties quits the protocol prior to step 5, each party would need $2^\ell$ trials in order to find a valid signature. For a large enough $\ell$ none of the parties will be able to find a valid signature prior to step 5.
Contract Signing Without a Trusted Center (cont.)

- If one of the parties quits the protocol during step 5 then:

  1. Either Alice and Bob sent the same number of bits for all $a_i$’s and $b_i$’s, thus in order to complete the signature both of them need the same amount of work, or

  2. Alice sent an extra bit (of at least one $a_i$) to Bob, thus she needs twice as much work as required by Bob in order to complete the signature.

Therefore, if one of the parties can complete the signature in time $t$, then the other one can complete the signature in no more than $2t$. 
Contract Signing Without a Trusted Center (cont.)

- If one of the parties is cheating, and sending false values during step 4, he will be caught with probability of at least $\frac{1}{2}$ per cheating (if he cheats in both values $a_i$ and $a_{i+n}$, he will always be caught). If the other party will get a false value during step 4, he will stop the execution of the protocol, and therefore, both parties will not be able to get signatures.

- If one of the parties cheats during step 5, i.e., he sends a bit $a'_i \neq a_i$ or $b'_i \neq b_i$, he will be caught for any such bit with probability $\frac{1}{2}$, but in order to prevent the other party from receiving a valid signature he must cheat in all $n$ values $a_i$ or $b_i$.

- Therefore, the cheater must cheat at least $n$ times, thus his chances of getting caught are $1 - 2^{-n}$. If he cheats only in $\frac{1}{2}$ of the $n$ iterations, his chances to get caught are $1 - 2^{-\frac{n}{2}}$, but this still enables the receiving side to know the right values in half of the cases.
Contract Signing Without a Trusted Center (cont.)

• It is important to mention that the probability of catching an act of dishonesty and the complexity for finding a valid signature in case of premature termination of the protocol are not average values. They are the actual probabilities for all executions of the protocol, so the cheater cannot hope for an easy cheating to occur even rarely.

• Therefore, at any point in which the protocol halts, if one of the parties can complete a signature, then with a high probability \((1 - 2^{-n})\) the other party can also complete the signature in about the same time complexity.
Weakness of the Protocol

This protocol seemed fine, and over the years it pushed aside other protocols based on other ideas due to its advantages.

In time it became clear that apart from the (impractical) assumption that both parties have the same computational power, there is another problem due to a premature termination of the protocol.
Weakness of the Protocol (cont.)

Assume that Alice and Bob sign an agreement which states that Alice should buy a million dollar worth of stocks for Bob within a week, and that Bob is supposed to pay her for the stocks within a month.

Assume that Bob stopped the protocol prematurely. Now Alice needs a year in order to complete Bob’s signature or to find that Bob cheated in choosing the $b_i$’s.

Alice does not know if Bob will complete the signature, when he will do so, or even if he is capable of doing so.

What should she do?

- If she buys the stock she may lose the money.
- If she doesn’t she could be sued by Bob (a year later).
Weakness of the Protocol (cont.)

The problem here arises from the fact that on the validity date, the two parties do not know whether it is valid or not, because it will take them a year to get the signatures.

To solve this problem a trusted third party must be involved in order to make the decision whether or not the contract is valid. The third party’s involvement is not necessary if the protocol was completed successfully, it is needed only in case of a premature termination.
The following protocol suggests a probabilistic method to sign contracts, which does not require heavy computations, even if the protocol was terminated prematurely, but involves a trusted third party, who will be utilized in order to solve disputes.

Rabin and Rivest suggested probabilistic methods which differ mainly in the role of the third party.

In Rabin’s method the third party is active and chooses random values daily.

In Rivest’s method the third party is involved only when the protocol is terminated prematurely, and then he chooses a random value.

The idea: Probabilistic fairness

With a high probability: Alice can prove that Bob signed if and only if Bob can prove that Alice signed.
Rabin’s method:

- The center publishes regularly every fixed period of time (e.g., daily) a random number in a known range (e.g., 1, ..., 100).

- When Alice and Bob want to sign a contract $m$, they agree on a validity date, and incrementally commit to the document:

  For $i = 1$ to 100:

  - Alice commits to: “If the center chooses the number $i$ on the validity date, I’m binded by contract $m$.”.
  - Bob commits to a similar statement.
Comments:

• In the validity date Alice and Bob can compare the other party’s commitment to the number published on that day to prove that the other party signed the contract.

• In any given time the probability for one party being committed while the other is not is not more than $\frac{1}{100}$. Thus, Bob can cheat with probability $\frac{1}{100}$ (or any other chosen value).
Rivest’s method:

- Rivest’s protocol’s commitment: “I’m committed to contract \( m \) with probability \( p \).”

- Alice and Bob increase the probability slightly in every iteration, until it grows to 1.

- If the protocol is terminated prematurely, then the probability gap is small.

- When the center is involved, he chooses a random number \( r \) between 0 and 1 and compares to the probabilities he received. If he is asked again on the same contract, he should select the *same* random number.
Contract Signing with a Trusted Center — Revisited (cont.)

- If $r \leq p$: the contract is binding. If $r > p$: the contract is not binding.
- The center informs the parties about his decision.
- If one of the parties comes with a higher probability the center compares the same $r$ to the new value $p$. 

Comments:

• In this protocol the center is not involved when there is no need to solve a dispute.

• The center does not need to keep a list of all random value for all the contracts he processed, if he chooses $r$ by computing a function (known only to him) on the contract.