Chain-Key Cryptography

Signatures

A ***digital signature scheme*** is a very traditional type of public-key cryptosystem, in which a secret key (held only by the signer) is used to generate a digital signature on a message, and a public key (available to everyone) may be used to efficiently verify a digital signature on a message. The basic security property achieved by such a scheme is that a valid signature on a message cannot be created without explicitly invoking the signing algorithm with the corresponding secret key.

A ***threshold signature scheme*** is a digital signature scheme where the secret signing key is never stored in one location (which would become a single point of failure). Rather, the secret key is effectively split up into ***secret shares***, and each secret share is stored on a different machine. To sign a message, these machines must agree to sign the message and coordinate with one another to generate a digital signature in a distributed fashion (importantly, without ever reconstructing the secret signing key in one location).

Chain-Key Technology

While threshold signature schemes is a technology that has been around for a long time, the IC is the first blockchain to fully integrate this technology in the core of its design. As described above, the technology enables ***chain-key cryptography*** and all of its benefits -- Efficient verification of blockchain outputs, autonomous evolution of the IC topology, a source of unpredictable and unbiasable pseudo-random numbers for canisters.

Each subnet in the IC is associated with the public verification key of such a threshold signature scheme.

1. Most importantly, this public key may be used to verify the outputs of the IC, including responses to ingress messages from external users, as well as messages from one canister to

another. This is one of the fundamental differences between the IC and other blockchains: the state of other blockchains can only be validated by running the entire protocol from the genesis block, whereas on the IC, it can be validated just by verifying a single digital signature. As such, this is one of the key technologies that enables unprecedented scalability on the Internet Computer.

2. This public key is also used to verify the entire state of a subnet at regular intervals, which enables a number of functions, such as adding new nodes to a subnet and allowing crashed nodes to quickly catch up to the rest. This enhances both the scalability of the IC and is crucial to enable the topology of the IC to autonomously evolve over time as orchestrated by the NNS.

In addition, these threshold signatures are used as a way to create a source of ***unpredictable pseudo-random numbers***, which is used in two ways:

1. as a source of unpredictable and unbiasable pseudo-random numbers available to any smart contract, which is a totally unique feature in the blockchain world that enables applications that would be impossible to implement on other blockchains (for example, an NFT raffle);
2. as a mechanism for pseudo-randomly selecting the leader in the IC consensus protocol, which enhances the efficiency and fairness properties of consensus.

Implementation

The threshold signature scheme implemented by the IC is a threshold version of the well-known **BLS signature scheme**. One reason for using the BLS signature scheme is that it is the only one that yields a threshold signing protocol that is very simple and efficient. Indeed, a machine holding a share of the secret signing key can very easily generate a share of a signature on a message, and these signature shares can be combined to form a BLS signature on a message – no further interaction between these machines is required.

Another reason for using the BLS signature scheme is that signatures are ***unique***, meaning that for a given public key and message, there is only one valid signature on that message. This unique-signature property is essential for the application to generating unpredictable and unbiased pseudo-random numbers for smart contracts: after a smart contract requests a pseudo-random number (and not before!), a signature on a special message is generated, and this signature is passed through a hash function to derive a seed from which the required

pseudo-random numbers are generated. By the security property of the signature scheme, neither this seed nor the derived pseudo-random numbers can be predicted or biased.

While signing with threshold BLS is quite straightforward, designing a secure, decentralized protocol for generating and distribution the shares of the secret signing key – that is, a DKG, or Distributed Key Generation protocol – remains a challenge. While there has been quite a bit of research on DKG design, the vast majority of DKG protocols in the literature do not meet the demanding requirements of the Internet Computer, in that they either assume a ***synchronous network*** (meaning that the protocols will fail or become insecure if messages are unexpectedly delayed) or provide ***no robustness*** (meaning that the ability to produce signatures is completely lost if a ***single*** node should crash) or ***both***. Neither of these assumptions are acceptable on the IC: security and liveness must hold even an an ***asynchronous network*** with many faulty nodes.

DFINITY has designed, analyzed, and implemented **a new DKG protocol** that works over an ***asynchronous network*** and is quite ***robust*** (it will still succeed if up to a third of the nodes in a subnet are crashed or corrupt) while still delivering acceptable performance. In addition to generating a new key, this protocol can also be used to reshare an existing key. This functionality is essential to enable autonomous evolution of the IC topology as subnet membership changes over time.

Chain Key Cryptography: The Scientific Breakthrough Behind the Internet Computer

Integrating The Internet Computer and Bitcoin Networks

Video on non-interactive distributed key generation

Applied Crypto: Introducing Noninteractive Distributed Key Generation

NIDKG White Paper