SpiderOak's Trusted Application Platform

A secure software platform for effortlessly building communication and collaboration software in mission critical environments.

Software for Mission Critical Collaboration

Meeting the mission needs in the modern world requires tools that enable collaboration, communication, and coordination with greater ease and flexibility. These tools need to not just serve individual teams, but must be flexible enough to work with mission partners, other companies, and other governmental organizations. No more can one team "go it alone" and expect success, but a wealth of knowledge and communication between a wide variety of actors is necessary.

Traditionally, it has been seen that making communication and collaboration easier has meant making it less secure, or greater security necessarily means decreasing ease of collaboration.

At SpiderOak we believe that the 21st-century mission requires tools enabling both greater collaboration and rock-solid assurances of data confidentiality, integrity, and authority.

What Needs to Change

At its core, the current client-server architecture is broken. In today's applications the server is at the center of trust and authority. To trust the server, you have to trust the millions of lines of code that form these servers and the operating systems they depend on. Far worse, we also have to depend on all the people that maintain the systems, the systems those people depend on, and so on. Internally, you have to trust your own IT staff, admins, and others who have the ability to access data they lack the legal or policy authority to see. In classified spaces, you see an example of this with system administrators being granted special clearances to be able to administer conventional services such as wikis and email.

Any one compromise within the system—from security bugs in the server source code, to a compromise with any of the innumerable vendors in use within the modern enterprise, to compromised or malicious internal privileged users, can bring the entire system crashing down. The data in the modern environment is too important to have single points of failure for the entire system.

Decentralization

The alternate solution to server based trust is the decentralized model. In a decentralized system, each part of the system is only given the ability to perform the actions required to complete its job (Figure 1). The server is only responsible for relaying bytes between clients, and is not able to forge or modify messages from clients. Content is only accessible to parties who have permission to see them; even IT staff cannot read messages not meant for them. Source code not critical for the implementation of the security system is no longer critical to maintain security. A bug in the operating system, or a dependent software component, no longer can reveal confidential data to an attacker.

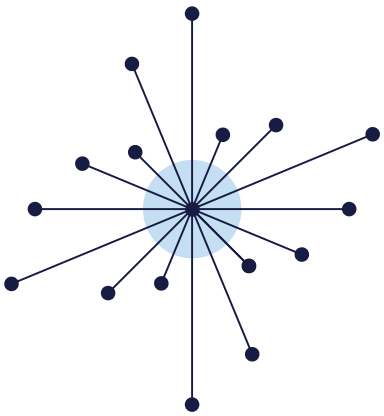
In a decentralized system it is still possible to implement all the required compliance and policies. Retention, escrow, and HR policies can be enforced and enacted in a way that gives stronger guarantees than are possible with the traditional systems. Using cryptographically-enforced decentralized authority, an organization can be sure of who has access to what data and for how long, who has the ability to take certain actions, and what actions have been taken by whom. With a decentralized approach, we gain the following advantages:

- The removal of millions of lines of code from the trust base.
- The removal of root authority from IT staff.
- The ability to keep our highest level permissions, used for disaster or data recovery, in a physical safe.
- The ability to create interlocks that require multiple parties to agree when performing important actions.
- The ability to know exactly who has visibility and control over what data with in your system.

These advantages allow your organization to remove whole classes of risk from your IT systems, and avoid being the next headline.

Figure 1

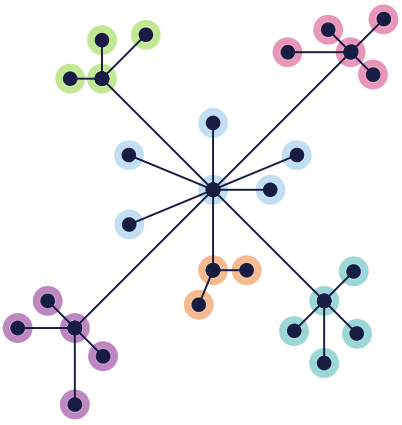
Centralized System



SINGLE POINT OF FAILURE

In a centralized system, any compromise in any part of the maze of corporate staff, developers, and IT, and any vendors, or their vendors, and so on, can result in the compromise of all of an organization's data. Absolute trust must be extended to everyone involved in the enterprise, even if seemingly unrelated.

Decentralized System



MULTIPLE POINTS OF AUTHORITY

In a decentralized system, data access is cryptographically enforced on a need-to-know basis. A compromise of one individual within an organization can only expose, at worst, the data that person has been granted the right to see. Through cryptography, the amount of data one needs to be able to see to perform their jobs is also greatly reduced. It separates the need to access systems for maintenance from being able to view the data on the system.

Blockchain

The most well-known application of this distributed authority model is the blockchain. Using blockchain, servers are not trusted and every client can validate the ledger. While these principles can be applied to general applications without compromise, blockchains provide ready-to-go structures and tools to work with data in this manner. Leveraging blockchain technology enables applications where one can reason about user authority and access in a more nuanced and more secure manner.

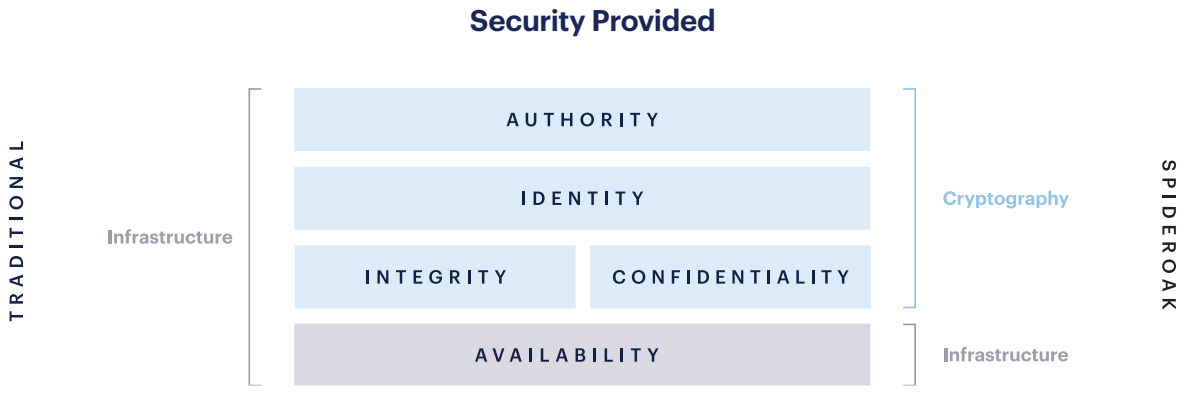
These properties can be leveraged to manage trust within an organization. Access to data and processes can be managed without having to tie trust to a central database, and scope of access can be limited to those who actually need to be able to see the data in question to do their jobs. Key management, a critical component to securing data cryptographically, can be easily handled through blockchain.

While the early application of blockchain was providing secure means to transfer value via cryptocurrency, it is undergoing rapid evolution today as it is applied to other fields needing distributed authority, and managing software security is a "killer app" for it in the foreseeable future.

The SpiderOak Solution

The SpiderOak Trusted Application Platform leverages the concept of decentralization to provide a means of rapidly developing collaborative multi-user applications that have inherent assurances of data confidentiality, integrity, and authority (Figure 2). Through the use of SpiderOak's own applications on the platform or custom built software on the platform, users can safely collaborate and share data to better execute on the mission without resorting to convoluted or expensive solutions to do so.

Figure 2



TRADITIONAL PROVIDERS

With traditional providers you rely entirely on infrastructure for every element of security, meaning more and more people to trust with every added level of security.

SPIDEROAK'S TRUSTED PLATFORM

SpiderOak moves many of these security elements out of the infrastructure, removing the need to trust so many different agents along the way. In this new model, information is available on a need-to-know cryptographically enforced basis.

Identity

User identity within the SpiderOak Platform is represented with a root account public/private keypair. The system assumes that an actor that can prove knowledge of the root account private key is the real-world user for that account. Normal operations within the platform are performed by individual device keys, which are constructed similarly to root account keys, and also signed by that root account. This separation of ultimate user identity and device identity enables administrators and users to securely and safely revoke individual devices without having to reset the entire account in the event of loss or breach.

SpiderOak client software strives to use the best available key storage mechanism available on client platforms. This includes available Trusted Platform Modules (TPMs), the Secure Enclave on iOS devices and select macOS computers, and PIV smart cards where available.

Channels and Blockchain

The core structure within the SpiderOak Platform is the channel. Channels can be thought of as persistent, cryptographically-compartmentalized, software-defined message busses. Each channel is made up of several linked blockchains, known as feeds, which provide the necessary security characteristics for sharing data within the system (Figure 3). All content, actions, and security information are tracked on feeds as part of a channel, providing an irrefutable, cryptographically-authenticated audit log.

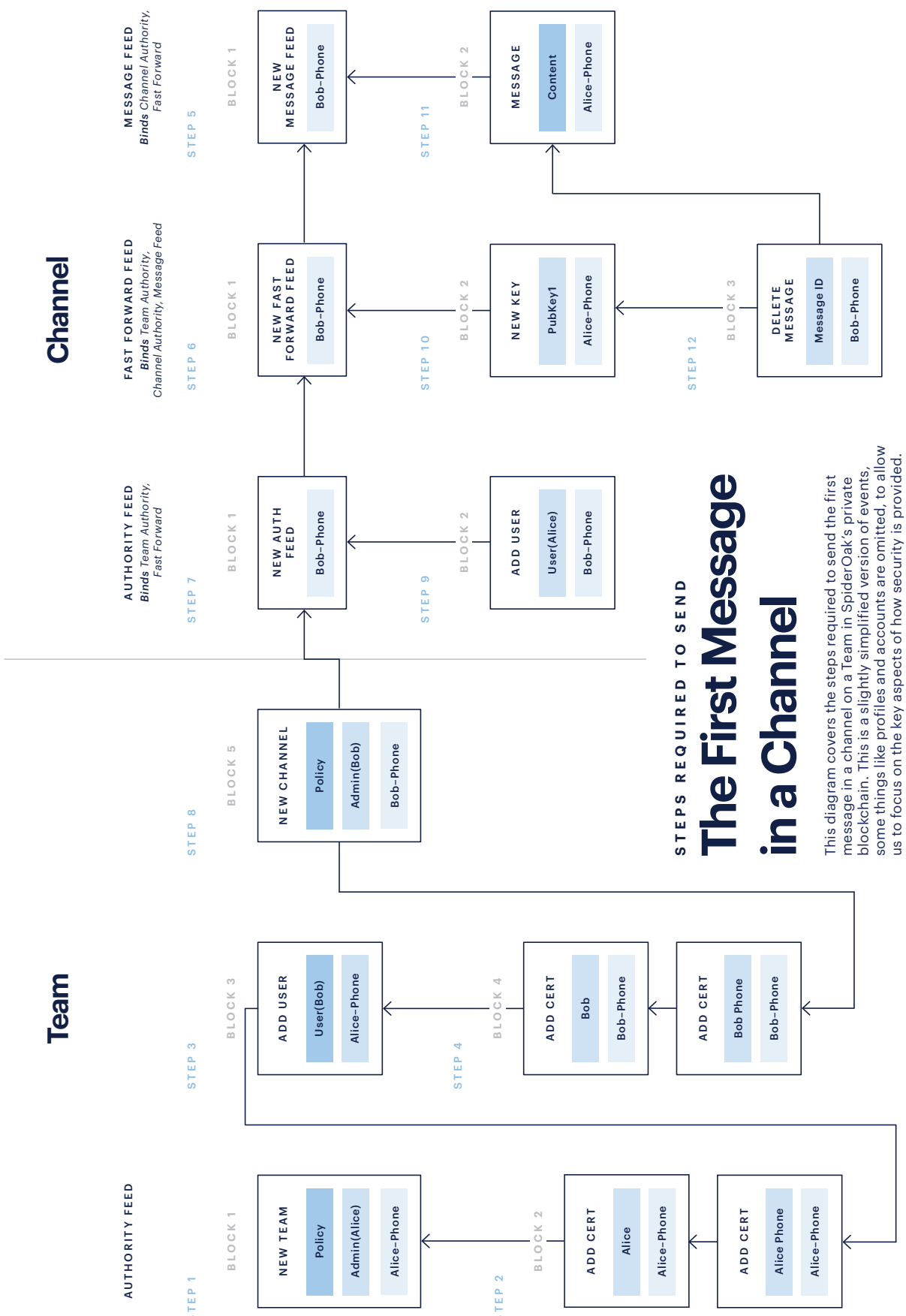
Rules implemented as policy objects enable controls as to what kinds of transactions and under what circumstances transactions can be added to a chain, ensuring consistency of the security system. Additionally, attempting to modify the history of the chain's transactions in order to modify the present state of the system will cause the chain to break, immediately alerting all clients within the system and prevent silent security failures.

Policy Engine

Unique to the SpiderOak private blockchain is the policy engine native to it. The policy engine, using technology similar to that of 'smart contracts' found in public blockchains such as Etherium, provides a means to intelligently validate every transaction made on the chain (Figure 4). Each policy object amounts to a miniature program that runs against transactions to be posted to the blockchain. This provides a simple and quick means of validating the correctness of transactions to be posted. All transactions made within the system need to be linked to a current validating policy for the compartment in question—either the team or a collaboration space.

Policy can range from simple—setting administrator rights for a user within a space, allowing that user to add or remove other users—to the complicated. For example, it is possible to create a policy enabling users of a space matching certain attributes to only have access for a limited amount of time, or only allowing users into a container who have a given set of clearances on their smart card.

Figure 3


STEPS REQUIRED TO SEND
The First Message in a Channel
This diagram covers the steps required to send the first message in a channel on a Team in SpiderOak's private blockchain. This is a slightly simplified version of events, some things like profiles and accounts are omitted, to allow us to focus on the key aspects of how security is provided.
Team
Channel
AUTHORITY FEED
STEP 1
BLOCK 1
NEW TEAM
Policy
Admin(Alice)
Alice-Phone
STEP 2
BLOCK 2
ADD CERT
Alice
Alice-Phone
ADD CERT
Alice Phone
Alice-Phone
STEP 3
BLOCK 3
ADD USER
User(Bob)
Alice-Phone
STEP 4
BLOCK 4
ADD CERT
Bob
Bob-Phone
ADD CERT
Bob Phone
Bob-Phone
STEP 5
BLOCK 1
MESSAGE FEED
Binds Channel Authority, Fast Forward
NEW MESSAGE FEED
Bob-Phone
STEP 6
BLOCK 1
FAST FORWARD FEED
Binds Team Authority, Channel Authority, Message Feed
NEW FAST FORWARD FEED
Bob-Phone
STEP 7
BLOCK 1
AUTHORITY FEED
Binds Team Authority, Fast Forward
NEW AUTH FEED
Bob-Phone
STEP 8
BLOCK 5
NEW CHANNEL
Policy
Admin(Bob)
Bob-Phone
STEP 9
BLOCK 2
ADD USER
User(Alice)
Bob-Phone
STEP 10
BLOCK 2
NEW KEY
PubKey1
Alice-Phone
STEP 11
BLOCK 2
MESSAGE
Content
Alice-Phone
STEP 12
BLOCK 3
DELETE MESSAGE
Message ID
Bob-Phone

Policy Engine, continued

An example of the policy engine language as it is used to implement default roles and permissions for teams within the product is below (Figure 4):

Figure 4

```
// Checks if this user exists, currently only checks for Member-Group-Fact
Macro.ActiveTeamMember(account) {
   TeamFact.Group.Member(account=$account)
}

// Admins can create users
Event.CreateUser {
   TeamFact.Group.Admin(account=$Event.Author)
} -> {
   TeamFact.Group.Member(account=$Event.User)
}

// Admins can promote others to Admins
SET DELETE TeamFact.Group.Admin(account) {
   Macro.ActiveTeamMember(account=$Event.Author)
   Macro.ActiveTeamMember(account=$account)
   TeamFact.Group.Admin(account=$Event.Author)
}
```

No Knowledge Encryption / Confidentiality

The other key piece to the collaboration suite is the use of end-to-end client-side encryption. Root account and device identity are kept secured on the device, preventing the server from masquerading as the user or reading data meant for the user. Each collaboration space has a set of cryptographic keys that are updated with every change in membership, making use of the known channel membership state to derive keys that all participants are aware of without the server being able to derive the shared secrets itself. At no point does the server receive the team or collaboration space decryption keys, preventing it from having knowledge of the content users are working on within the system.

Through this, physical or remote access to the server or storage components of the system cannot result in access to information by Information Technology (IT) staff, service vendors, or malicious actors. The product uses a Federal Information Processing Standard (FIPS) 140-2-certified implementation of the National Security Agency (NSA) Suite B cyphers. We are aware of the existing NSA guidance concerning Suite B in a post-quantum environment and will provide a forward-compatible update to the system once there is new guidance from the Government regarding acceptable secure quantum-resistant cyphers.

Authority

Through the use of blockchain with a policy engine, authority can be tightly managed within the system. When a collaboration space is created, by default only the creator of the space is granted administrative authority to the space on its blockchain. Authority does not automatically "flow down" from the team administrators to the collaboration space; it must be explicitly granted to users from the existing administrators from the space.

Authority, continued

It is through this mechanism that the actual authority to view and manipulate data is strictly and cryptographically enforced, instead of relying on expected compliance of information security policy and reactive solutions such as monitoring and audit logging. Due to the distributed nature of the system, a compromise of the central server infrastructure for the Collaboration Suite does not allow an attacker access to the collaboration spaces within the system.

If an attacker only had access to the server components of the system, to gain access to a space and receive new keys within a space they would need to insert themselves into the authority chain of the space. Without proper cryptographic signatures from an existing administrative user, however, this would be rejected as an invalid transaction from all the clients within the system. If an attacker instead attempted to alter the blockchain's history to insert themselves, that would invalidate the cryptographic fingerprint of that transaction, causing the chain to break and relevant clients to cease operating until proper history is restored.

Even if an attacker were to successfully gain control of a user's private keys or authorized endpoint device, the compromise would only spread as far as the user of the space is authorized to see within the organization.

Conclusion

The SpiderOak Trusted Application Suite provides a new way to reason about and consider security in the modern mission-oriented enterprise. By removing the nearly infinite number of single points of failure in the security system, you can trust that your critical data is only accessible by the people responsible for the mission success. Today's modern threat and regulatory environment does not need to mean roadblocks to fast and effective collaboration and communication.

For more information or to contact SpiderOak, please visit **www.spideroak.com/platform**.