Stick Protocol

An end-to-end encryption protocol tailored for social network platforms, based on the Signal protocol. The Stick Protocol is the first of its kind to support re-establishable encryption sessions in an asynchronous and multi-device setting while preserving forward secrecy and introducing backward secrecy.

End-to-End Encrypted Content

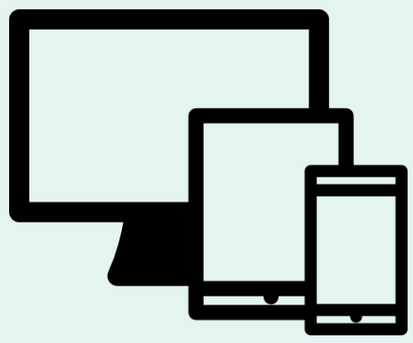
A social network app using Stick protocol will have all of its content end-to-end encrypted to the designated parties.

Re-establishable

A user will be able to securely re-establish their encryption sessions when re-installing the application.

Multi Device

Stick protocol provides end-to-end encryption for all of a user's devices.



Identity Keys Self-Healing

A user's identity key refreshes every while to mitigate the effect of an identity key compromise.

Perfect Forward & Backward Secrecy for Sharing Sender Keys

Provided by the multiple pairwise sessions.

Stick sessions Backward Secrecy

Sticky sessions provides backward secrecy every a maximum of N Encryptions.

Sticky Sessions Provides Perfect Forward Secrecy

Every post shared is encrypted with a unique key, which cannot be used to derive older keys.

Based on Signal Protocol

Which means that the Stick protocol benefits from the Signal protocol's security features as well.

Source Code • https://github.com/sticknet/stick-protocol

Omar Basem | omarbasem.com

Stick: an End-to-End Encryption Protocol Tailored for Social Network Platforms

1. **Motivation:** End-to-End Encryption (E2EE) has become a de facto standard in messengers, especially after the development of the secure messaging protocol – Signal. However, the adoption of E2EE has been limited to messengers, and has not yet seen a noticeable trace in social networks, despite the increase in users' privacy violations. The Stick protocol is an E2EE protocol, based on the Signal protocol, specifically designed for social networks. The Stick Protocol is the first of its kind to support re-establishable encryption sessions in an asynchronous and multi-device setting while preserving forward secrecy and introducing backward secrecy.

2. **Design Overview:** Stick's design includes several innovative features, including *sticky sessions*, multiple pairwise sessions, *double-hashing* and refreshing *identity keys*. Stick as its core relies on *sticky sessions,* which is an E2E encrypted session between a user and a party that can be re-established after STATE RESET* in an asynchronous and multi-device environment, and keeps track of the ratcheting Message Keys while preserving forward secrecy and introducing backward secrecy. Stick protocol provides the following benefits to a social network user, Alice:

- All of Alice's posts will be end-to-end encrypted to the designated parties.
- Alice can re-establish her encryption sessions and view her posts even after reinstalling the app, through the procedure shown in this diagram.
- Alice can view her content from any other device using her account.
- Alice still benefits from all the security features of the Signal protocol, such as X3DH.
- Alice benefits from identity keys self-healing.
- Sharing *sender keys* has perfect forward secrecy and perfect backward secrecy using multiple pairwise sessions.
- Sharing posts using sticky sessions provides perfect forward secrecy.
- Sharing posts using sticky sessions provides backward secrecy every maximum of N Encryptions.

3. **Formal Verification:** I verified Stick using Verifpal - a formal verification tool in the symbolic model. Security analysis shows the Stick protocol does achieve a form of post compromise security in many-to-many communications, the trait which most group protocols lack. Most importantly, the Stick protocol can re-establish encryption sessions while ensuring authentication and confidentiality.

4. **Implementation:** I implemented the protocol as a stand-alone API. The implementation is composed of 4 software libraries: Android library (Java), iOS library (Swift, Objective-C & C), server library (Python), and client-handlers library (JavaScript). These 4 libraries are open-source available on GitHub. Additionally, online usage documentation is available.

5. **Performance Evaluation:** Evaluation results shows the Stick protocol can be used in a real-world social network app with no noticeable compromise on usability or performance. The average overhead of sharing and receiving content is a mere 5.3%.

6. **Conclusion**: This work was an end-to-end process of developing the Stick protocol from design and verification to implementation and evaluation. Stick protocol is already being used in production in StickNet app. Stick protocol can be extended to areas other than social networking where E2E encrypted re-establishable sessions would be useful. This includes IoTs, health care and banking systems.

*For more information about the Stick protocol: https://www.sticknet.org/StickProtocol