team pass
UNIVERSITY OF MARYLAND
HONORS COLLEGE

Usability and Formal Analysis of the FIDO2 Protocol

GEMSTONE
Honors College
University of Maryland

Zachary Breit, Hunter Dean, Tai-Juan Generette, Nathaniel Higgins, Samuel Howard, Balaji Kodali, Jim Kong, Jonah Tash, Philip Wang, John Wu

Mentor: Dr. John Baras

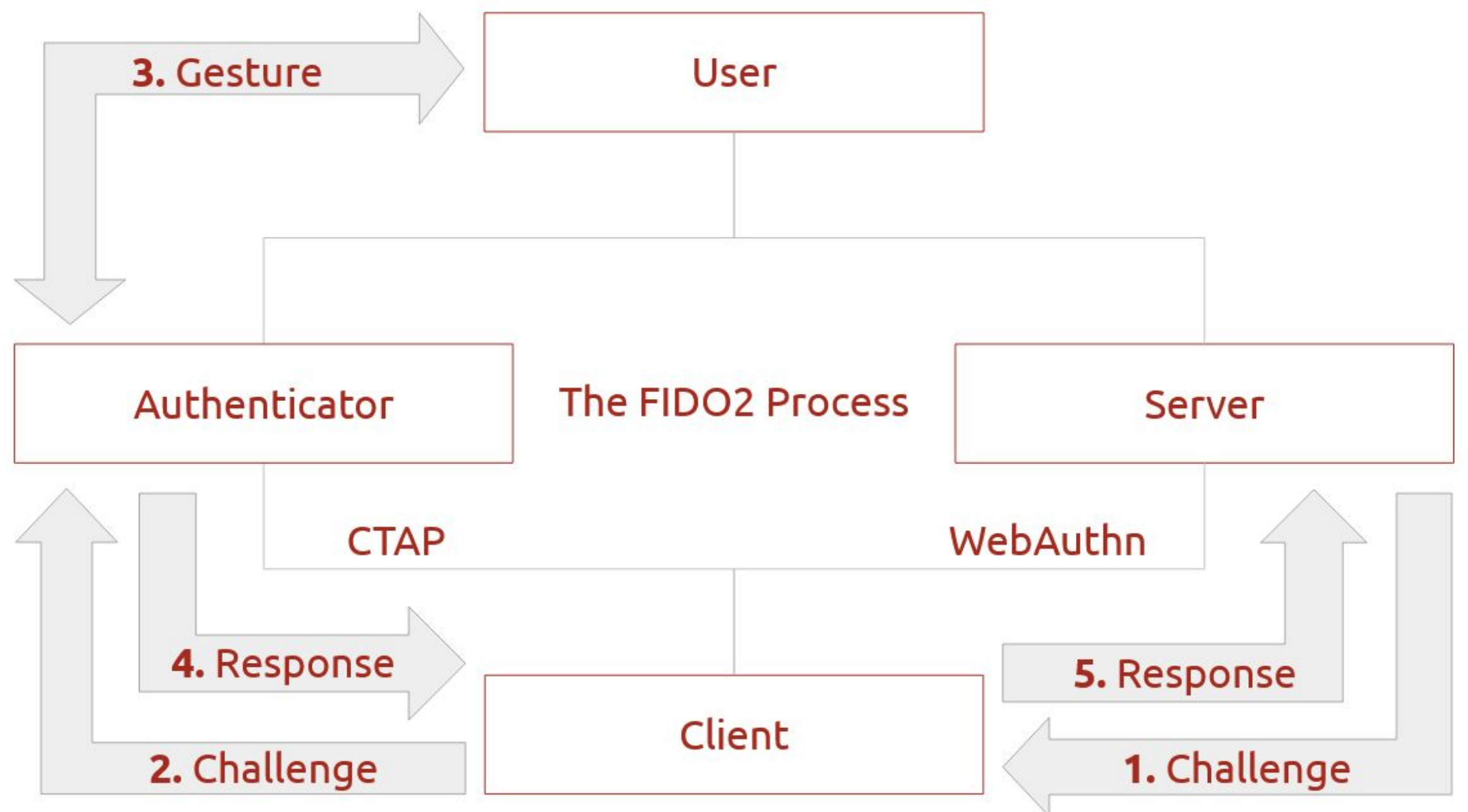
Abstract

FIDO2 is a passwordless authentication protocol for the web that leverages public key cryptography and trusted devices to avoid shared secrets on servers. It was recently standardized by the World Wide Web Consortium (W3C) into the Web Authentication API, which is supported by most modern browsers. The API integrates with many popular authenticators such as Windows Hello, YubiKey, and Apple TouchID/FaceID. In our research, we reviewed recent efforts to formally analyze FIDO2's security with symbolic and computational models. After interpreting these findings, we proposed a more secure version of FIDO's Client to Authenticator Protocol (CTAP2) that utilized a modified key exchange scheme to guarantee strong unforgeability of authentication gestures within the protocol. We also developed a procedure for a within-subjects experiment to measure the usability of the FIDO protocol. In this experiment, we will assess user satisfaction during the FIDO login process using automated page analytics and a System Usability Scale (SUS) survey.

Introduction to FIDO

FIDO2 is composed of two distinct parts: the Web Authentication protocol and the Client to Authenticator Protocol (CTAP) [1]. Web Authentication is a challenge/response protocol for authenticating a user to a web server. The user relies on a trusted authenticator (e.g., a fingerprint reader) and a potentially untrustworthy browser to communicate with the website on their behalf [2]. For each website that a user interacts with, the authenticator generates a distinct public key that is used for future interactions. An example usage of the Web Authentication protocol between a user Alice and the website `https://example.com` using a biometric authenticator is explained below:

1. Alice requests to log in to `https://example.com`.
2. `https://example.com` sends Alice a random challenge.
3. Alice signs the challenge using her trusted authenticator. The authenticator uses an embedded, private attestation key and a scoped public key for that service to sign the message.
4. Alice sends back her response.
5. `https://example.com` verifies Alice's response using the public key credential that they have on file for that particular authenticator.
6. Alice is granted access to the web service.



Overview of the FIDO2 protocol. Each arrow corresponds to a numbered step above. As you can see, the protocol is composed of two challenge/response phases. One occurs over a TLS connection on the web (WebAuthn) and the other occurs locally via a browser (CTAP).

The most recent CTAP, CTAP2 is a local protocol for ensuring that the web browser can only use the authenticator when given explicit authorization from the user [2]. The user grants authorization by gesturing to the authenticator. For example, the user could press a button on a hardware 2FA token or scan their finger.

Formal Analysis

Part of FIDO is the Client to Authenticator Protocol (CTAP), which is a local protocol for ensuring that the web browser can only use the authenticator when given explicit authorization from the user. [3]

Flaws in existing CTAP

- During the binding phase, the client and authenticator use *unauthenticated* Diffie Hellman. This scheme is vulnerable to Man-in-the-Middle (MitM attacks) since an attacker can sit in between the client and authenticator [2]
- The sPACA protocol introduces a password-authenticated key exchange (PAKE) to replace unauthenticated Diffie-Hellman, however it also introduces an additional gesture [2]

Our proposal

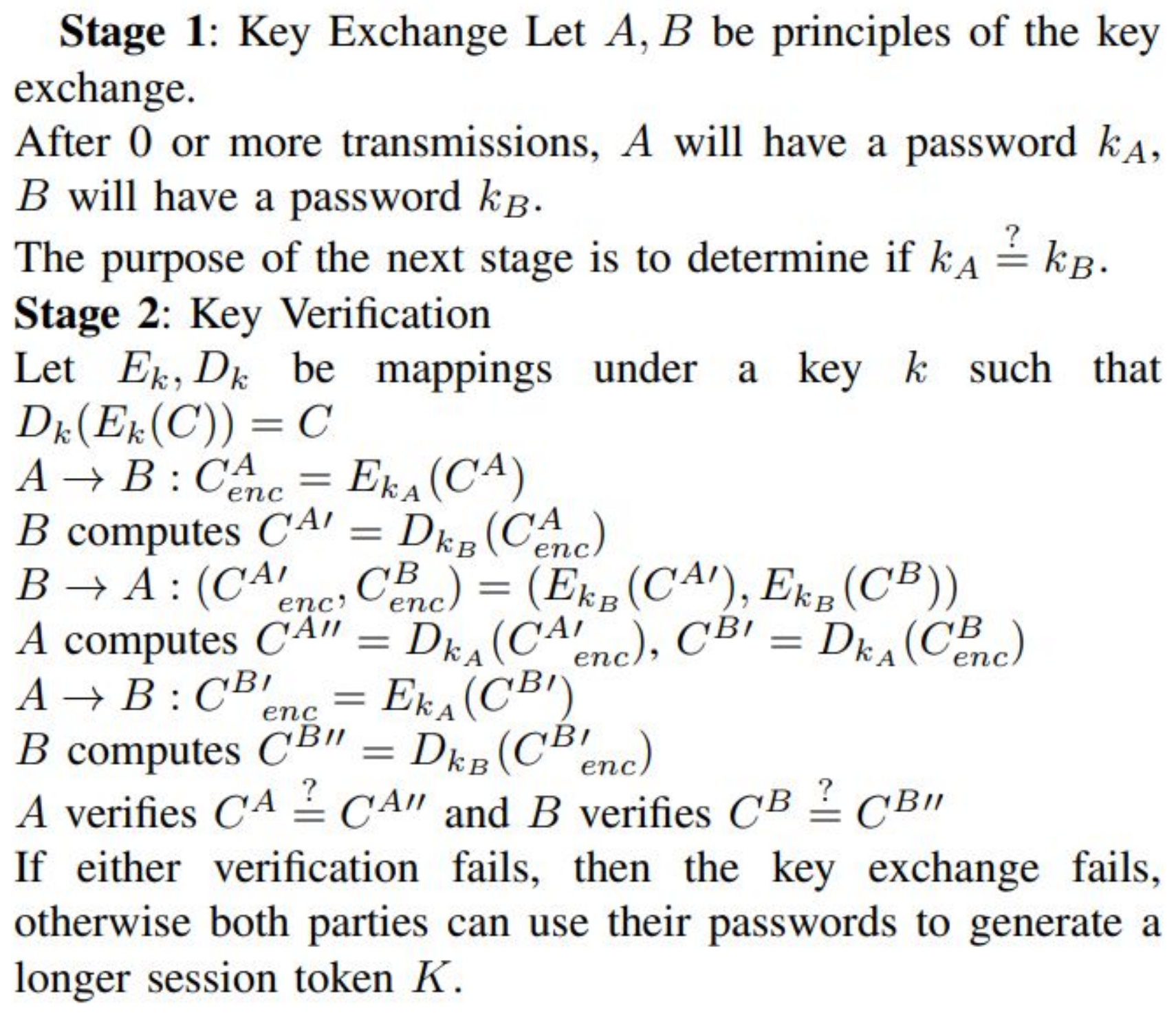
We want to answer the question:

Can multiple signals from the same authenticator be used in a PAKE-like authenticated protocol?

By introducing an aggregate signal, we hope to resolve the flaws of both CTAP and sPACA

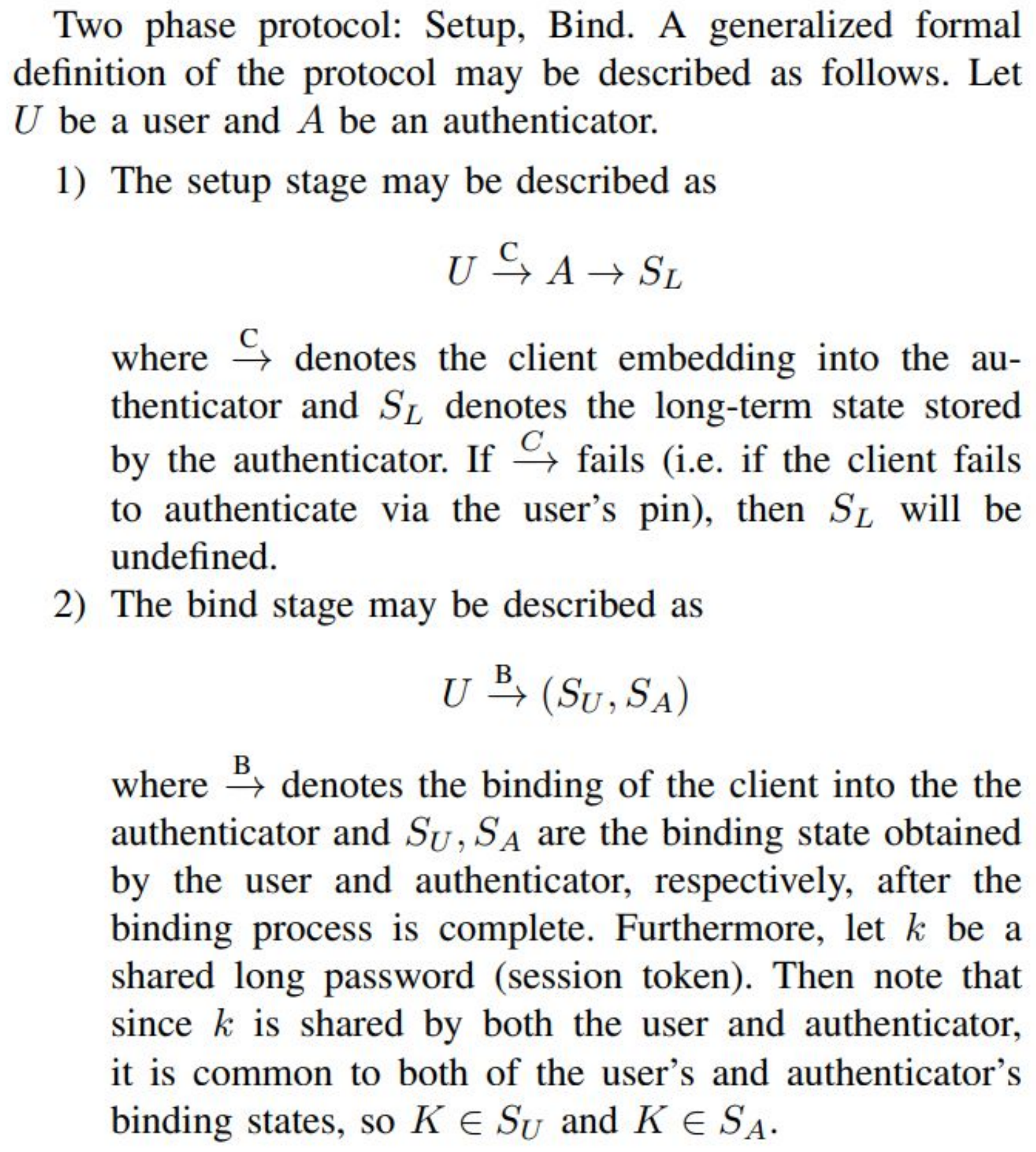
Formal Definitions

PAKE: A general two-stage PAKE protocol will be defined using a structure similar to the SPEKE protocol described in [2, 4]



Stage 1: Key Exchange Let A, B be principles of the key exchange.
After 0 or more transmissions, A will have a password k_A, B will have a password k_B.
The purpose of the next stage is to determine if $k_A \overset{?}{=} k_B$.
Stage 2: Key Verification
Let E_k, D_k be mappings under a key k such that $D_k(E_k(C)) = C$
$A \rightarrow B : C^A_{enc} = E_{k_A}(C^A)$
B computes $C^{A\prime} = D_{k_B}(C^A_{enc})$
$B \rightarrow A : ({C^{A\prime}}_{enc}, C^B_{enc}) = (E_{k_B}(C^{A\prime}), E_{k_B}(C^B))$
A computes $C^{A\prime\prime} = D_{k_A}({C^{A\prime}}_{enc})$, $C^{B\prime} = D_{k_A}(C^B_{enc})$
$A \rightarrow B : {C^{B\prime}}_{enc} = E_{k_A}(C^{B\prime})$
B computes $C^{B\prime\prime} = D_{k_B}({C^{B\prime}}_{enc})$
A verifies $C^A \overset{?}{=} C^{A\prime\prime}$ and B verifies $C^B \overset{?}{=} C^{B\prime\prime}$
If either verification fails, then the key exchange fails, otherwise both parties can use their passwords to generate a longer session token K.

aCTAP: An abstract authenticated CTAP based on [2]



Two phase protocol: Setup, Bind. A generalized formal definition of the protocol may be described as follows. Let U be a user and A be an authenticator.

1) The setup stage may be described as

$$U \xrightarrow{\text{C}} A \rightarrow S_L$$

where $\xrightarrow{\text{C}}$ denotes the client embedding into the authenticator and S_L denotes the long-term state stored by the authenticator. If $\xrightarrow{C}$ fails (i.e. if the client fails to authenticate via the user's pin), then S_L will be undefined.

2) The bind stage may be described as

$$U \xrightarrow{\text{B}} (S_U, S_A)$$

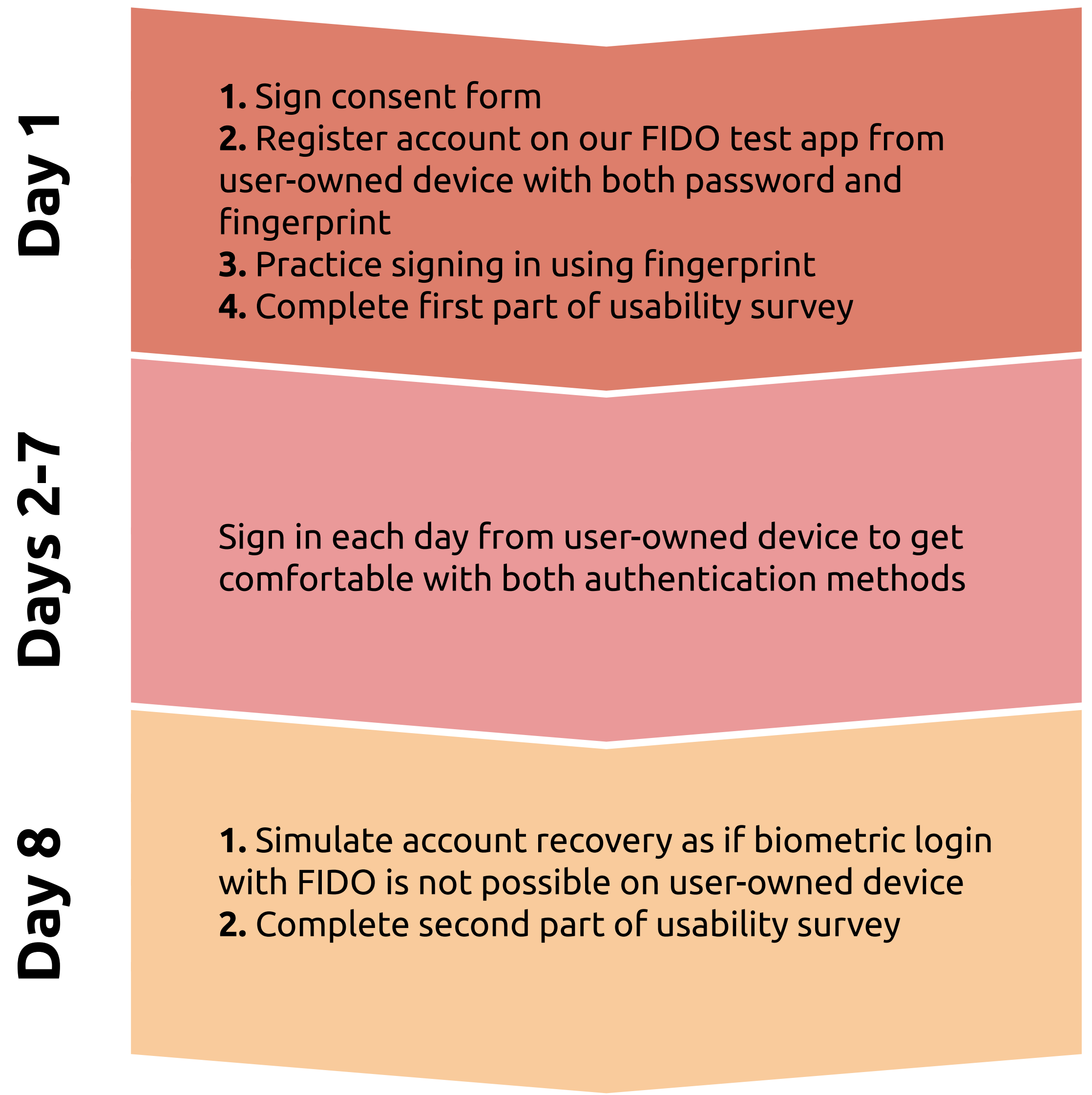
where $\xrightarrow{\text{B}}$ denotes the binding of the client into the the authenticator and S_U, S_A are the binding state obtained by the user and authenticator, respectively, after the binding process is complete. Furthermore, let k be a shared long password (session token). Then note that since k is shared by both the user and authenticator, it is common to both of the user's and authenticator's binding states, so $K \in S_U$ and $K \in S_A$.

Usability Study

We plan to conduct a usability study of FIDO as a single-factor authentication method by evaluating participants' experiences with a FIDO-based web app that we developed. Over an eight-day period, participants will access the app and authenticate themselves using a mobile device equipped with a biometric sensor. At the beginning and end of the period, we will administer a survey based on the System Usability Scale (SUS) to gather the opinions of the protocol's overall usability [5].

Our objective is to examine the experience of participants when using FIDO in comparison to using passwords. Some possible questions for the Day 8 survey are *"How would you rate the registration process for FIDO on a scale of 1-10 (10 being very simple)?"* and *"How would you compare FIDO to text-based passwords (less convenient, as convenient, more convenient)?"* as well as questions adapted from the System Usability Scale [5].

Using the data we gather from this study, we will be able to understand how FIDO compares against passwords in terms of usability. In particular, it will help us understand the areas in which the FIDO authentication scheme performs well and whether it is suitable as a single-factor replacement for passwords.



Day 1

1. Sign consent form
2. Register account on our FIDO test app from user-owned device with both password and fingerprint
3. Practice signing in using fingerprint
4. Complete first part of usability survey

Days 2-7

Sign in each day from user-owned device to get comfortable with both authentication methods

Day 8

1. Simulate account recovery as if biometric login with FIDO is not possible on user-owned device
2. Complete second part of usability survey

References