Università degli studi di Modena e Reggio Emilia

Dipartimento di Ingegneria "Enzo Ferrari"

Corso di Laurea in Ingegneria Informatica - Sede di Mantova

Titolo: prima riga
Seconda riga
Terza riga
Quarta riga

Relatore:
Prof. Luca Ferretti

Candidato:
Giulio Barabino

Correlatore:
Ing. Federico Magnanini

Anno Accademico 2021/2022

Indice

1 Introduzione **1**

2 Conoscenze di Base **2**

2.1 Single Sign-On 2

2.2 Survivability 2

2.3 Passwordless 3

2.4 FIDO 4

2.4.1 Rilevamento tentativi di clonazione 4

3 Modellazione del sistema **6**

3.1 Attori 6

3.2 Flusso operativo 8

3.2.1 Fase di registrazione 8

3.2.2 Fase di autenticazione 9

4 Dettagli **11**

4.1 Modifica della libreria FIDO2 12

Capitolo 1

Introduzione

Capitolo 2

Conoscenze di Base

In questa sezione verranno introdotti i concetti necessari per proseguire con la lettura dei capitoli successivi.

2.1 Single Sign-On

Lo schema *Single Sign-On* è un protocollo molto diffuso utilizzato per fare il login a servizi Web. L'utente non effettua l'autenticazione direttamente presso il servizio desiderato ma passa prima da un ente terzo. Questo ente è chiamato Identity Provider, fornisce un Identity Server presso cui l'utente deve autenticarsi per farsi rilasciare un **token** di autenticazione. Tale token verrà poi presentato al servizio desiderato così che possa finalmente essere fruito.

2.2 Survivability

La centralizzazione dello schema SSO lascia spazio ad attacchi in cui un malintenzionato prenda il controllo dell'Identity Server e forgi token di autenticazione così da poter poi impersonare qualunque utente egli voglia. Vengono in aiuto gli schemi cosiddetti *survivable SSO* che possono limitare tali problematiche sfruttando più Identity Server. Un singolo Identity Provider gestirà quindi più Identity Server e l'utente dovrà aut-

enticarsi presso un sottoinsieme di questi, i quali rilasceranno poi una **attestazione** firmata collettivamente.

La componente survivable risiede nel fatto che viene tollerato un certo numero di Identity Server violati e, di conseguenza, viene richiesta una attestazione in funzione di questo numero. Con una soglia di tolleranza di server maligni sufficiente si riesce a garantire l'integrità del meccanismo di autenticazione e un overhead, dovuto alla reiterazione dei passaggi, trascurabile.

2.3 Passwordless

L'autenticazione passwordless è un metodo di autenticazione che permette ad un utente di effettuare il login ad un servizio senza la necessità di conoscere una password o più genericamente una conoscenza considerata segreta. Tipicamente utilizza una coppia di chiavi crittografiche, una **privata** e una **pubblica**: la prima viene generata e immagazzinata sul dispositivo dell'utente mentre la seconda viene inviata al server così che esso possa verificare l'autenticità dei messaggi ricevuti. La chiave privata, o segreta, non lascia mai il dispositivo su cui è stata creata e per accedervi è necessaria una qualche sorta di autenticazione da parte dell'utente [5]. L'autenticazione può riguardare:

- Qualcosa che l'utente **possiede** come un telefono cellulare, un token OTP, un autenticatore hardware
- Qualcosa che l'utente **è** come l'impronta digitale, la scansione retinica o il riconoscimento vocale

La registrazione passwordless e, conseguentemente, l'autenticazione vengono svolte seguendo un meccanismo *challenge-response*: al pervenire di una richiesta di registrazione il server invia una cosidetta *challenge*. L'utente che ha iniziato l'operazione avrà il compito di apporre, tramite propria chiave privata, una firma crittografica sulla challenge e di fornire in risposta al server la challenge firmata accompagnata dalla chiave pubblica. Così facendo il server verificherà l'autenticità della firma tramite la

chiave appena ricevuta e in caso di esito positivo, la immagazzinerà. La fase di autenticazione verrà svolta in modo analogo con la differenza che il server è già in possesso della chiave pubblica e non sarà quindi necessario inviarla.

Come si può vedere non viene scambiato alcun segreto e l'unica interazione richiesta all'utente è quella in fase di firma della challenge. Anche allora l'utilizzatore non dovrà inserire codici o password ma semplicemente autenticarsi al dispositivo contenente la chiave privata tramite uno dei fattori sopra elencati. Sfruttando l'autenticazione passwordless è possibile sopperire alle criticità tipiche dei segreti a bassa entropia come le password, quali phishing, brute forcing etc.

2.4 FIDO

FIDO Alliance è un'associazione nata nel 2013 con lo scopo di migliorare i sistemi di autenticazione. Sono gli autori di *FIDO*, un set di specifiche che include gli standard **CTAP** e **WebAuthn**. Nel corso degli anni vi sono stati un susseguirsi di iterazioni degli standard, prima conosciuta come Universal Authentication Factor per poi diventare Universal 2nd Factor e giungere infine alla versione corrente FIDO 2.0.

Il protocollo CTAP definisce le specifiche tramite cui vengono programmati gli autenticatori crittografici per fare in modo che possano operare con un client. Il protocollo WebAuthn si occupa invece di definire le specifiche tramite cui standardizzare l'autenticazione a servizi web sfruttando le chiavi crittografiche.

In particolare definiscono tutto il necessario per programmare un autenticatore e un server come: strutture dati, metodi, requisiti di funzionamento, encoding dei dati etc.

2.4.1 Rilevamento tentativi di clonazione

Compito dell'Identity Server è anche quello di rilevare eventuali tentativi di duplicazione dell'autenticatore fisico. Per fare ciò lo standard FIDO prevede un **contatore** gestito dall'autenticatore, sia esso globale o multiplo, che viene aggiornato ad ogni operazione avvenuta con successo. Il contatore prende il nome di *signature counter*. Il

server mantiene in memoria l'ultimo valore ricevuto e, all'interazione successiva, controlla che non vi siano discrepanze. In particolare se il valore ricevuto è minore di quello salvato in memoria dal server allora si può essere in presenza di un tentativo di clonazione. (L'accadere dell'inverso non è detto che costituisca un problema, motivo per il quale può essere utilizzato anche un contatore globale.) Ciò significa che un attaccante maligno ha duplicato l'autenticatore ad uno stato precedente mentre l'utilizzatore ha continuato ad autenticarsi incrementando il contatore.

Capitolo 3

Modellazione del sistema

In questo capitolo verranno inizialmente descritti gli attori che concorrono alle fasi di registrazione/autenticazione per poi definire il flusso delle operazioni stesse.

3.1 Attori

Lo schema di seguito rappresenta lo stato attuale dello standard FIDO in accordo all'implementazione descritta successivamente.

Figura 3.1: PLACEHOLDER.

Il protocollo include i seguenti attori: l'utente, Service Provider, Identity Provider, un numero n di Identity Server, l'autenticatore hardware e il FIDO Client.

Il **Service Provider** è un fornitore di un generico servizio a cui l'utente è interessato ad accedere. Può essere un qualunque servizio di streaming, banking, shopping etc. il quale fa uso di un intermediario per l'autenticazione dei propri utenti. Questo può avvenire per varie ragioni sia economiche che legate alla sicurezza. Il Service Provider definisce il **security level** per indicare il livello di *survivability* desiderato, cioè il numero di Identity Server necessari a completare le operazioni di autenti-

cazione/registrazione. Tale valore è un intero positivo e viene stabilito in funzione della confidenzialità del servizio erogato.

L'**Identity Provider** è un ente terzo che si occupa di fornire a un Service Provider il servizio di autenticazione. Fa parte del cosiddetto *Relying Party*, cioè quella parte che fornisce un accesso sicuro ad un servizio. Compito dell'Identity Provider è anche quello di fornire il token di autenticazione al client da presentare al Service Provider per fare in modo che l'utente possa accedere al servizio scelto. I vari Identity Server comunicano con il FIDO Client, tramite User Agent, per creare le credenziali degli utenti mantenendo in memoria le challenge generate e gli identificatori degli utenti registrati con le relative chiavi pubbliche ricevute.

L'**autenticatore** hardware è un dispositivo che, tramite l'interazione con l'utente, permette l'accesso al servizio web richiesto. Durante la fase di creazione delle credenziali, l'autenticatore si occupa di creare una coppia di chiavi crittografiche, una pubblica e una privata. Utilizzerà la chiave privata per firmare le challenge che gli vengono sottoposte mentre la pubblica verrà inviata al server così che esso possa verificare l'autenticità delle firme. Per la fase di creazione e le successive di autenticazione viene richiesto all'utente di compiere un'azione: essa può essere la pressione di un pulsante sulla chiavetta stessa, un collegamento NFC oppure ancora l'identificazione tramite impronta digitale. Così facendo l'operazione in corso viene autenticata.

Il FIDO **Client** è un dispositivo che sfrutta un **User Agent** conforme ad implmentare le specifiche FIDO per il dialogo con il Relying Party e con l'autenticatore, in collaborazione con l'hardware sottostante su cui è installato l'User Agent, tipicamente un sistema operativo. Il client è quindi interposto tra il FIDO Server e l'autenticatore fisico, agendo da intermediario. Il suo compito è duplice:

- Comunicare con il server al fine di iniziare, e successivamente terminare, le operazioni di autenticazione e di creazione delle credenziali
- Comunicare con l'autenticatore allo scopo di creare le chiavi crittografiche e firmare le challenge ricevute dal server

La comunicazione è bidirezionale, duplice e segue i costrutti specificati dalle API definite nei relativi standard: CTAP per l'interazione con l'autenticatore e WebAuthn per

la comunicazione con il FIDO Server. Nel caso particolare dell'estensione survivable il client si occupa di replicare le operazioni su n FIDO Server distinti, computando l'hash delle challenge ricevute e fornendolo all'autenticatore come un digest unico su cui apportare la firma.

3.2 Flusso operativo

Il flusso operativo si compone di due operazioni distinte: una fase di creazione delle credenziali e una fase di autenticazione dell'utente. Tali operazioni vengono svolte, rispettivamente, durante la registrazione al servizio del Service Provider e a tutti i login successivi.

3.2.1 Fase di registrazione

Vi partecipa: l'utente, il FIDO Client, lo User Agent, n Identity Server, l'autenticatore.

La fase di registrazione si origina a partire dalla richiesta dell'utente, utilizzando un User Agent, di registrarsi ad un servizio, offerto da un Service Provider, che supporti l'autenticazione passworldess, tramite degli Identity Provider. Il Service Provider fornisce all'utente il livello di sicurezza n necessario per completare l'operazione. La fase di creazione dovrà essere replicata dal client su tutti gli n Identity Server presenti. Il processo messo in atto è il seguente:

1. Ogni Identity Server crea il proprio stato interno e la challenge
2. Ogni Identity Server invia al Client una serie di requisiti secondo cui deve essere svolta la cerimonia di registrazione, e la challenge generata
3. Il Client salva tutte le challenge ricevute in un vettore e computa l'hash dello stesso
4. Il Client effettua una chiamata al metodo opportuno dell'autenticatore, fornendo i requisiti di creazione richiesti dal Relying Party e il digest computato come challenge

5. L'autenticatore procede a generare la coppia di chiavi crittografiche seguendo le imposizioni del server e invia al client la challenge firmata accompagnata dalla chiave pubblica e dal signature counter

6. Il Client invia ad ogni Identity Server il vettore con le challenge, l'hash dello stesso, la firma, la chiave pubblica e il signature counter ricevuto

7. Ogni Identity Server controlla che la challenge da lui generata sia presente all'interno del vettore e controlla, computando lui stesso l'hash del vettore, l'integrità di quanto ricevuto. Infine, verifica tramite la chiave pubblica fornitagli l'autenticità della firma.

8. Qualora il processo sia andato a buon fine, gli Identity Server salveranno le informazioni ricevute (contatore, chiave pubblica, identificatore del client) al proprio interno

3.2.2 Fase di autenticazione

La fase di autenticazione ricalca i passaggi di quella di registrazione con la differenza che gli Identity Server sono già in possesso della chiave pubblica con cui verificare l'autenticità della firma apportata alle challenge.

In questa fase viene comunicato un security level pari a $|Q|$ da parte del Service Provider, cioè la cardinalità del sottoinsieme di Identity Server $\mathbf{Q} \leq \mathbf{n}$ presso cui è necessario autenticarsi. Questo valore prende in considerazione la tollerabilità alle intrusione che ha il Service Provider. In particolare: $|Q|$ può essere *(2k+1)*, dove k rappresenta il numero di Identity Server di cui si può tollerare la compromissione, oppure può essere *(3k+1)* in caso di requisiti particolarmente stringenti.

1. Ogni Identity Server crea il proprio stato interno e la challenge

2. Ogni Identity Server invia al Client una serie di requisiti, secondo cui deve essere svolta la cerimonia di autenticazione, e la challenge generata

3. Il Client salva tutte le challenge ricevute in un vettore e computa l'hash dello stesso

4. Il Client effettua una chiamata al metodo opportuno dell'autenticatore, fornendo i requisiti di autenticazione richiesti dal Relying Party e il digest computato come challenge

5. L'autenticatore incrementa mono-atomicamente il contatore specifico al livello di sicurezza stabilito. Invia poi al Client la challenge firmata e il contatore aggiornato

6. Il Client invia ad ogni Identity Server il vettore con le challenge, l'hash dello stesso, la firma ricevuta e il signature counter

7. Ogni Identity Server controlla che la challenge da lui generata sia presente all'interno del vettore e controlla, computando lui stesso l'hash del vettore, l'integrità di quanto ricevuto. Infine, verifica tramite la chiave pubblica, memorizzata precedentemente, l'autenticità della firma

8. Ogni Identiy Server controlla che il *signature counter* ricevuto sia maggiore di quello memorizzato in precedenza e in tal caso aggiorna quest'ultimo

9. Se i passaggi precedenti sono avvenuti con successo rilasciano al Client l'attestazione tramite cui può completare l'autenticazione presso il Service Provider

Capitolo 4

Dettagli

In questo capitolo verranno trattati i dettagli implementativi relativi alle modifiche a:

- Una variante della libreria FIDO2 realizzata da Yubico modificata in una tesi precedente per accogliere il meccanismo survivable [6]
- Il codice sorgente dell'autenticatore Solokeys [4]

Nonostante la libreria Yubico fosse già stata modificata in precedenza per adottare la struttura survivable, è stato comunque necessario operare cambiamenti. La libreria FIDO2 si occupa di simulare l'interazione tra un Client FIDO2 e un Server FIDO2 per emulare la registrazione e la successiva autenticazione. Grazie alla modifica apportata precedentemente è possibile simulare un numero arbitrario di Server con cui stabilire la comunicazione e svolgere tali operazioni. Viene correttamente gestito tutto il funzionamento descritto nel capitolo precedente meno la parte di invio del security level.

Lato autenticatore invece si è reso necessario implementare diverse funzionalità: dal parsing del security level nel messaggio CBOR inviato dal client all'autenticatore fino ad arrivare a un contatore globale vero e proprio. Infatti, da standard FIDO2 il signature counter utilizzato per il controllo della clonazione dell'autenticatore può essere anche definito come globale e non specifico per credenziale [1]. Ciò è possibile per questioni legate alla natura degli autenticatori hardware: dispositivi constraint con limitazioni di memoria importanti.

4.1 Modifica della libreria FIDO2

Rispetto alla libreria modificata

Bibliografia

[1] FIDO Alliance. Signature counter considerations, 8 April 2021.

[2] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of cryptology*, 1(2):77–94, 1988.

[3] Google. Google scholar. `https://scholar.google.it/`, visited in Sep. 2016.

[4] Solokeys. Solokeys/solo1: Solo 1 firmware in C.

[5] Wikipedia contributors. Passwordless authentication — Wikipedia, the free encyclopedia, 2022. [Online; accessed 15-September-2022].

[6] Yubico. Yubico's FIDO2 Python Implementation.