

# WhatsApp End-to-End Encrypted Backups Kevin Lewi

## (Unencrypted) Cloud Backups





<b>〈</b> Chats	Chat Backup	
La To	st Backup: Today, 09:02 tal Size: 705.7 MB	
Back up your of lose your iPhor history is safe. media when you messages you end-to-end en	hat history and media to ne or switch to a new one You can restore your cha ou reinstall WhatsApp. Me back up are not protecte cryption while in iCloud.	iCloud so if you , your chat at history and edia and d by WhatsApp
Back Up Now	/	
Auto Backup		Daily >
Include Video	DS	$\bigcirc$
To avoid excess Fi or disable cel Cellular > iCloud	ive data charges, connect lular data for iCloud: iPhon d Drive > OFF.	your phone to Wi- e Settings >

## End-to-End Encrypted (E2EE) Backups

Goal: Full privacy of message content

- Including from backup storage providers (Apple / Google) Including even from WhatsApp / Meta

### Solutions:

**This talk** 

- Ask users to write down the bytes of their encryption key Ask users to remember a password + enforce attempt limit







Creating an E2EE backup































## **Client Authentication**

## How to securely authenticate client to HSMs?

Option 1: "Hash-then-Encrypt" or "Password-over-TLS"

- Client hashes their PIN, and then PK-encrypts to the HSM
- HSM decrypts and verifies hash

### **Option 2: Password-Authenticated Key Exchange**

- Establish a secure channel between client and HSM based on PIN
- Transmit backup key through this channel  $\bigcirc$





# **OPAQUE** [Jarecki, Krawczyk, Xu '18]

• OPAQUE is a **strong**, *asymmetric* password-authenticated key exchange (saPAKE) protocol

 Theorem: Oblivious PRF + Authenticated KE -> saPAKE

• We use DH-OPRF:  $F(k,x) = H(x, H(x)^k)$ 

#### **OPAQUE:** An Asymmetric PAKE Protocol Secure Against Pre-Computation Attacks \*

Stanislaw Jarecki<sup>1</sup>, Hugo Krawczyk<sup>2</sup>, and Jiayu Xu<sup>1</sup>

<sup>1</sup> University of California, Irvine. Email: {stasio@ics.,jiayux@}uci.edu. <sup>2</sup> IBM Research. Email: hugo@ee.technion.ac.il.

Abstract. Password-Authenticated Key Exchange (PAKE) protocols allow two parties that only share a password to establish a shared key in a way that is immune to offline attacks. Asymmetric PAKE (aPAKE) strengthens this notion for the more common client-server setting where the server stores a mapping of the password and security is required even upon server compromise; that is, the only allowed attack in this case is an (inevitable) offline exhaustive dictionary attack against individual user passwords. Unfortunately, current aPAKE protocols (that do not rely on PKI) allow for pre-computation attacks that lead to the instantaneous compromise of user passwords upon server compromise, thus forgoing much of the intended aPAKE security. Indeed, these protocols use - in essential ways - deterministic password mappings or use random "salt transmitted in the clear from servers to users, and thus are vulnerable to pre-computation attacks.

We initiate the study of Strong aPAKE protocols that are secure as aPAKE's but are also secure against pre-computation attacks. We formalize this notion in the Universally Composable (UC) settings and present two modular constructions using an Oblivious PRF as a main tool. The first builds a Strong aPAKE from any aPAKE (which in turn can be constructed from any PAKE [26]) while the second builds a Strong aPAKE from any authenticated key-exchange protocol secure against KCI attacks. Using the latter transformation, we show a practical instantiation of a UC-secure Strong aPAKE in the Random Oracle model. The protocol ("OPAQUE") consists of 3 messages requires 3 and 4 exponentiations for server and client, respectively (including a multi-exponentiation and 1 or 2 fixed-base per party) provides forward secrecy and explicit mutual authentication, is PKI-free, supports user-side password hardening, has a built-in facility for password-based storage-and-retrieval of secrets and credentials, and accommodates a user-transparent server-side threshold implementation.

#### 1 Introduction

Passwords constitute the most ubiquitous form of authentication in the Internet, from the mundane to the most sensitive applications. The almost



<sup>\*</sup> This is a revised ePrint version of the paper which appeared in Eurocrypt 2018 [33]. See revision notes in Sec. 1.2.

# **OPAQUE in E2EE Backups**





### On backup recovery:





K, pwd, HSM\_PK

<u>Client</u>



### HSM\_SK

#### HSM Server



K, pwd, HSM\_PK

### <u>Client</u>

1. Pick a random scalar r

2. Send  $\alpha = H(pwd)^r$ 





#### HSM Server



K, pwd, HSM\_PK

### <u>Client</u>

1. Pick a random scalar r

4. Send  $\beta = \alpha^{K'}$ , nonce



### HSM\_SK

#### HSM Server

2. Send  $\alpha = H(pwd)^{r}$ 

3. Pick a random OPRF key K' and nonce



K, pwd, HSM\_PK

### <u>Client</u>

1. Pick a random scalar r

5. Compute (export\_key, client\_SK) = PBKDF(pwd,  $\beta^{(1/r)}$ )

6. Compute K\* = AES-128(export\_key, K) and client\_PK = g^client\_SK



### HSM\_SK

### HSM Server

2. Send  $a = H(pwd)^r$ 

4. Send  $\beta = \alpha^{K'}$ , nonce

3. Pick a random OPRF key K' and nonce



K, pwd, HSM\_PK

### <u>Client</u>

1. Pick a random scalar r

5. Compute (export\_key, client\_SK) = PBKDF(pwd,  $\beta^{(1/r)}$ )

6. Compute  $K^* = AES-128(export_key, K)$  and client\_PK = g^client\_SK

7. Send E = RSA-OAEP(HSM\_PK, K\* || client\_PK || transcript)



### HSM\_SK

### HSM Server

2. Send  $a = H(pwd)^r$ 

4. Send  $\beta = \alpha^{K'}$ , nonce

3. Pick a random OPRF key K' and nonce

8. Decrypt E and verify transcript, then store K<sup>\*</sup>, K<sup>'</sup>, and client\_PK for user















### K, pwd, HSM\_PK

#### Client

1. Pick a random sca

- 5. Compute (export\_
- 6. Compute  $K^* = AE$

### PRF F(k, x) = PBKDF(x, H(x)^k)

### Client has backup key K and PIN.

For each client, server stores:

- K', a freshly generated PRF key
- $K^* = AES128(F(K', pwd), K)$
- client PK

7. Send E = RSA-OAEP(HSM\_PK, K\* || client\_PK || transcript)



### HSM\_SK

### HSM Server

3. Pick a random OPRF key K' and nonce

8. Decrypt E and verify transcript, then store K<sup>\*</sup>, K<sup>'</sup>, and client\_PK for user















pwd, HSM\_PK

### <u>Client</u>





### HSM\_SK, K', K\*, client\_PK

HSM Server



pwd, HSM\_PK

### <u>Client</u>

1. Pick a random scalar r and client\_e\_SK





### HSM\_SK, K', K\*, client\_PK

### HSM Server

### 2. Send $\alpha = H(pwd)^r$ , client\_e\_PK



pwd, HSM\_PK

### <u>Client</u>

1. Pick a random scalar r and client\_e\_SK

4. Send  $\beta = \alpha^{K'}$ , server\_e\_PK,  $\sigma = \text{Sign}(\text{HSM}_SK, \beta)$ 



### HSM\_SK, K', K\*, client\_PK

HSM Server

3. Pick a server\_e\_SK,

decrement

attempt\_counter

2. Send  $\alpha = H(pwd)^r$ , client\_e\_PK





pwd, HSM\_PK

### Client

1. Pick a random scalar r and client\_e\_SK

4. Send  $\beta = \alpha^{K'}$ , server\_e\_PK,  $\sigma = \text{Sign}(\text{HSM}_SK, \beta)$ 

- 5. Compute (export\_key, client\_SK) = PBKDF(pwd,  $\beta^{(1/r)}$ )
- 6. Derive shared\_secret\_key from KE protocol

7. < Complete KE with server >



### HSM\_SK, K', K\*, client\_PK

HSM Server

3. Pick a server\_e\_SK,

decrement

attempt\_counter

2. Send  $a = H(pwd)^r$ , client\_e\_PK





pwd, HSM\_PK

### Client

1. Pick a random scalar r and client\_e\_SK

4. Send  $\beta = \alpha^{K'}$ , server\_e\_PK,  $\sigma = \text{Sign}(\text{HSM}_SK, \beta)$ 

- 5. Compute (export\_key, client\_SK) = PBKDF(pwd,  $\beta^{(1/r)}$ )
- 6. Derive shared\_secret\_key from KE protocol

7. < Complete KE with server >

9. Send  $C = AES(shared secret key, K^*)$ 

10. Decrypt C with shared\_secret\_key, then decrypt result with export\_key to obtain K





### HSM\_SK, K', K\*, client\_PK

HSM Server

3. Pick a server e SK, decrement attempt\_counter 8. Verify KE completion, reset attempt\_counter, and obtain shared\_secret\_key







## **More Resources**

- Security audit from NCC Group in 2021
- We released a whitepaper on the E2EE backup design
- Open-source Rust OPAQUE library



#### nccgroup

#### End-to-End Encrypted Backups Security Assessment

#### WhatsApp

October 27, 2021 – Version 1.2

Prepared by Gérald Doussot Marie-Sarah Lacharité Eric Schorn

©2021 - NCC Group

Prepared by NCC Group Security Services, Inc. for WhatsApp. Portions of this document and the templates used in its production are the property of NCC Group and cannot be copied (in full or in part) without NCC Group's permission.

While precautions have been taken in the preparation of this document, NCC Group the publisher, and the author(s) assume no responsibility for errors, omissions, or for damages resulting from the use of the information contained herein. Use of NCC Group's services does not guarantee the security of a system, or that computer intrusions will not occur.



WhatsApp

#### **Security of End-To-End Encrypted Backups**

WhatsApp Security Whitepaper

Version 1 Originally published September 10, 2021



## Future Work

1. Alternatives to HSMs?

2. Threshold OPRFs / OPAQUE?



## WhatsApp End-to-End Encrypted Backups

<b>14:44</b> <ul> <li>✓ Search</li> </ul>		14:43 ◀ Search	
Chats Chat Backup		K Back	Encrypted Backup
Last Backup: Yesterday, 04:16 Total Size: 1.51 GB Back up your chat history and media to iCloud so if you lose your iPhone or switch to a new one, your chat history is safe. You can restore your chat history and media when you reinstall WhatsApp.		End-to-e	nd Encrypted Backup is On
		To restore y you will nee	your end-to-end encrypted backup, ed your password.
Back Up Now		Change Pas	ssword
Auto Backup	Daily >	Your backu iCloud. No	p is end-to-end encrypted on one, not even Apple or WhatsApp, it
Include Videos To avoid excessive data charges, connect your Fi or disable cellular data for iCloud: iPhone Set Cellular > iCloud Drive > OFF.	phone to Wi- ttings >	Turn Off	
End-to-end Encrypted Backup For another layer of security, protect your back to-end encryption.	On >		



### As of December 2022:



Will Cathcart 🤣 @wcathcart

Also excited to see Apple bring end-to-end encrypted backups to iCloud. We launched this on WhatsApp over a year ago and have seen more than 100 million users turn it on already. People want privacy.

🙎 Will Cathcart 🤣 @wcathcart · Sep 10, 2021

We're very excited to be launching end-to-end encrypted backups on WhatsApp.

Show this thread





