Google

Becoming Unphishable

Towards simpler, stronger authentication

Grant Dasher

CIS 2017

Introduction and Agenda

Part of the team responsible for authentication at Google

Agenda

- Passwords are broken
- Introducing Security Key
- Google's Experience
  - Some numbers
  - We're not quite done
- How can you get started?

Passwords are broken

Passwords are broken

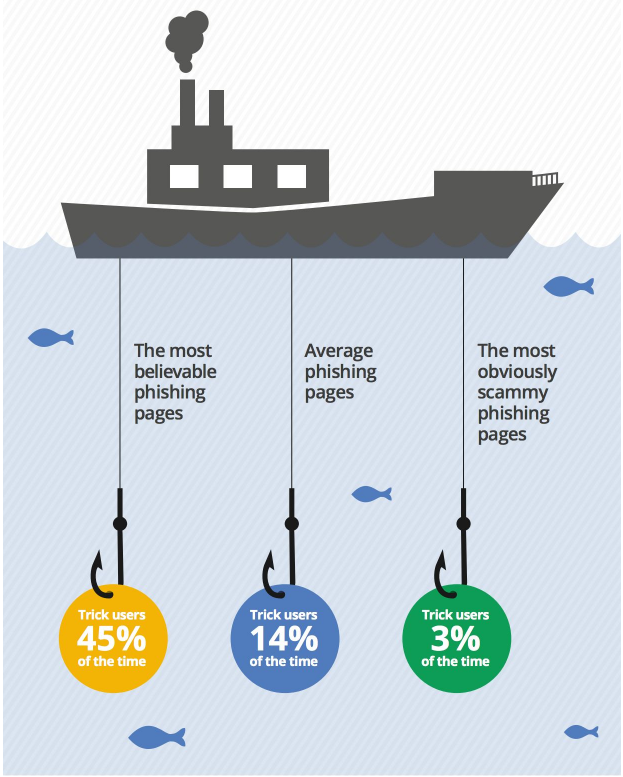
REUSED PHISHED KEYLOGGED

Phishing has become increasingly sophisticated

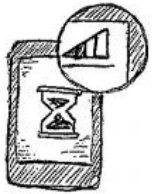
- More than ⅔ of incidents [in 2015] … involved phishing. With a 23% effectiveness rate*
- OTPs help against shared password, but it's not safe to rely on them for phishing

** http://www.verizonenterprise.com/DBIR/2015/*

Is Phishing Effective?



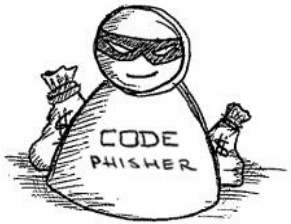
Today's solution: One Time Passwords



SMS USABILITY
Coverage Issues - Delay - User Cost

DEVICE USABILITY
One Per Site - Expensive - Fragile

USER EXPERIENCE
Users find it hard



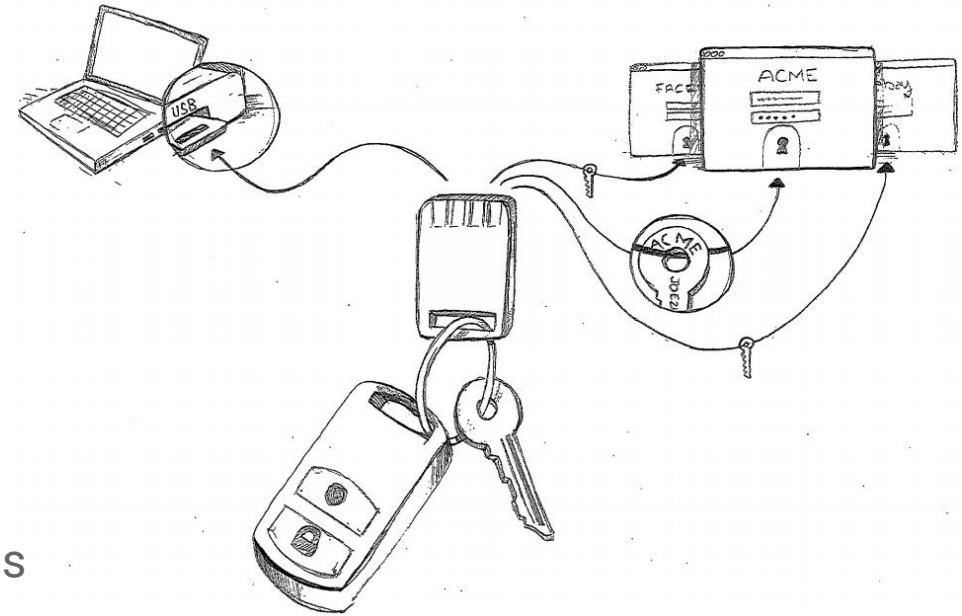
PHISHABLE
German Police re: iTan:
".. we still lose money"

Introducing Security Key

Introducing Security Key

Designed to solve authentication challenges

- For enterprises
- For consumers



Based on FIDO U2F standard

- *Safe:* Unphishable / UnMITMable
- *Easy:* Insert and press button
- *Compact:* One device, many services

Simple operation

1

Userid & Password

2

Insert, Press button

3

Successful Sign in

Based on Asymmetric Cryptography

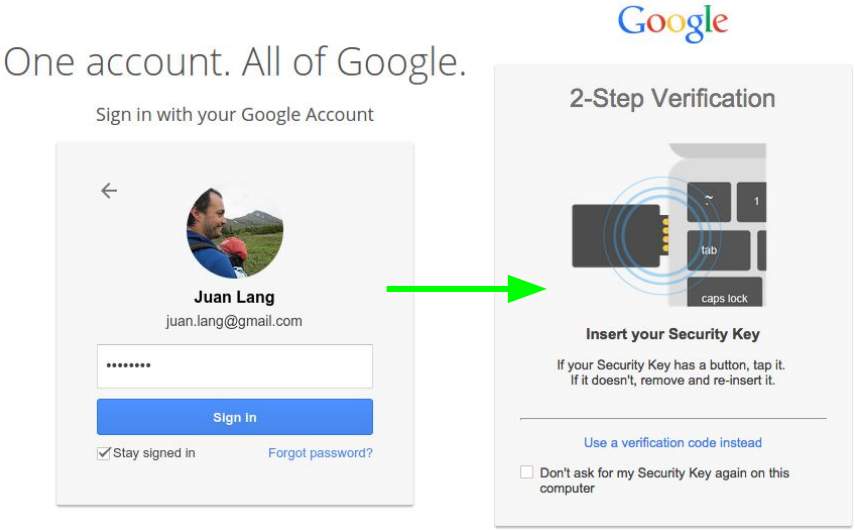
Core idea - Standard public key cryptography

- User's device mints new key pair, gives public key to server
- Server asks user's device to sign data to verify the user.
- **One device, many services, "bring your own device" enabled**

Google’s experience

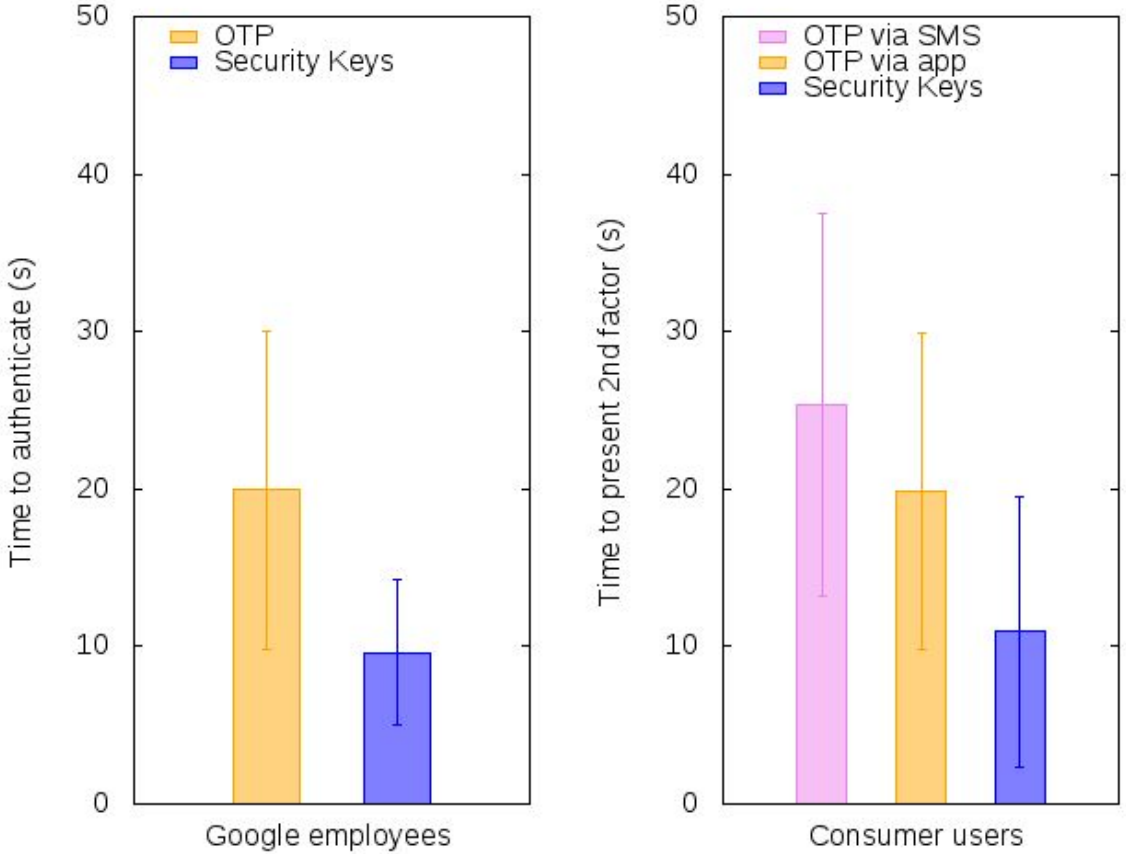
Deployment at Google

- **Enterprise use case**
  - Mandated for Google employees
  - Corporate SSO (Web)
  - SSH
  - Forms basis of all authentication

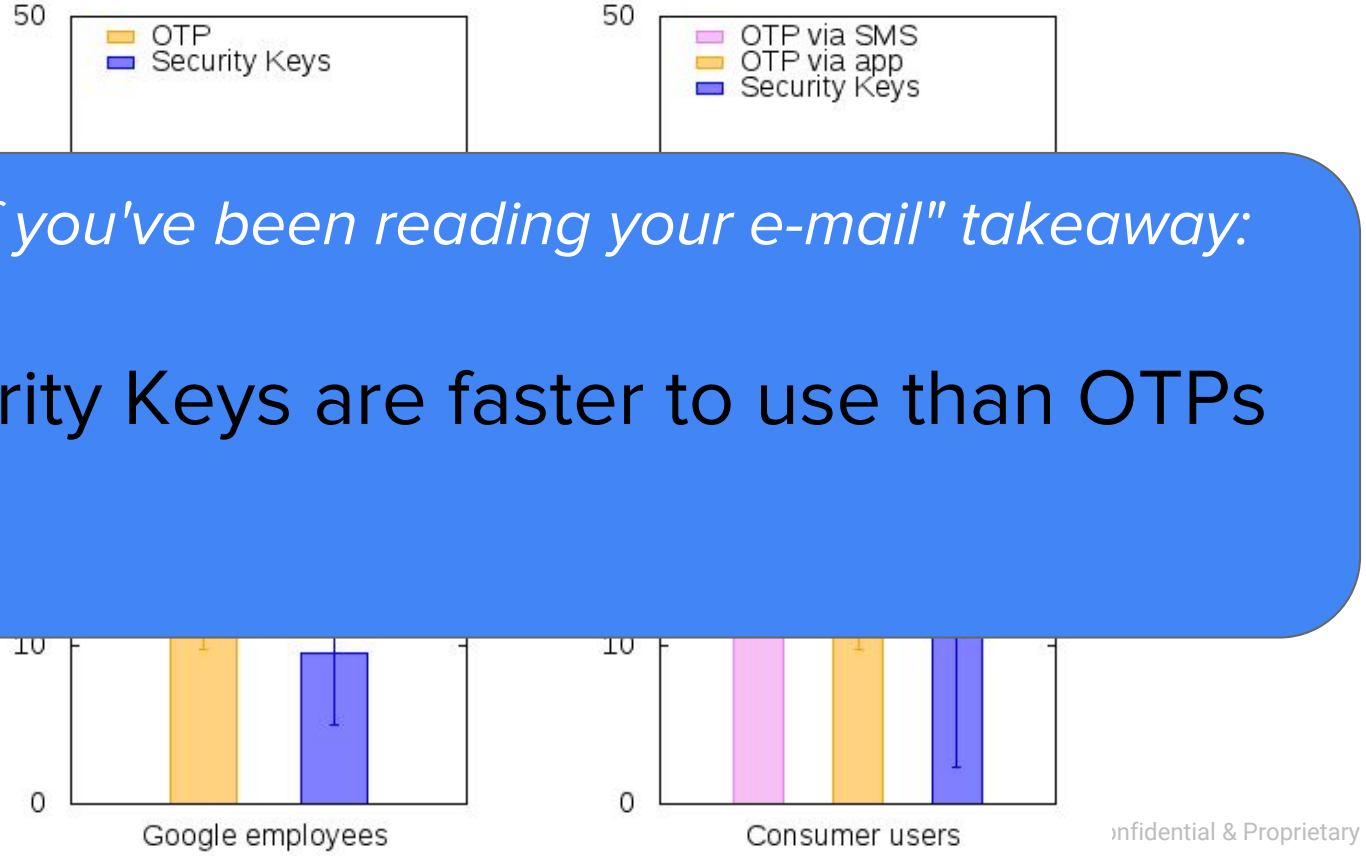


- **Consumer use case**
  - Available as opt-in for Google consumers
  - Adopted by other relying parties too: Dropbox, Github, Facebook

Time to authenticate



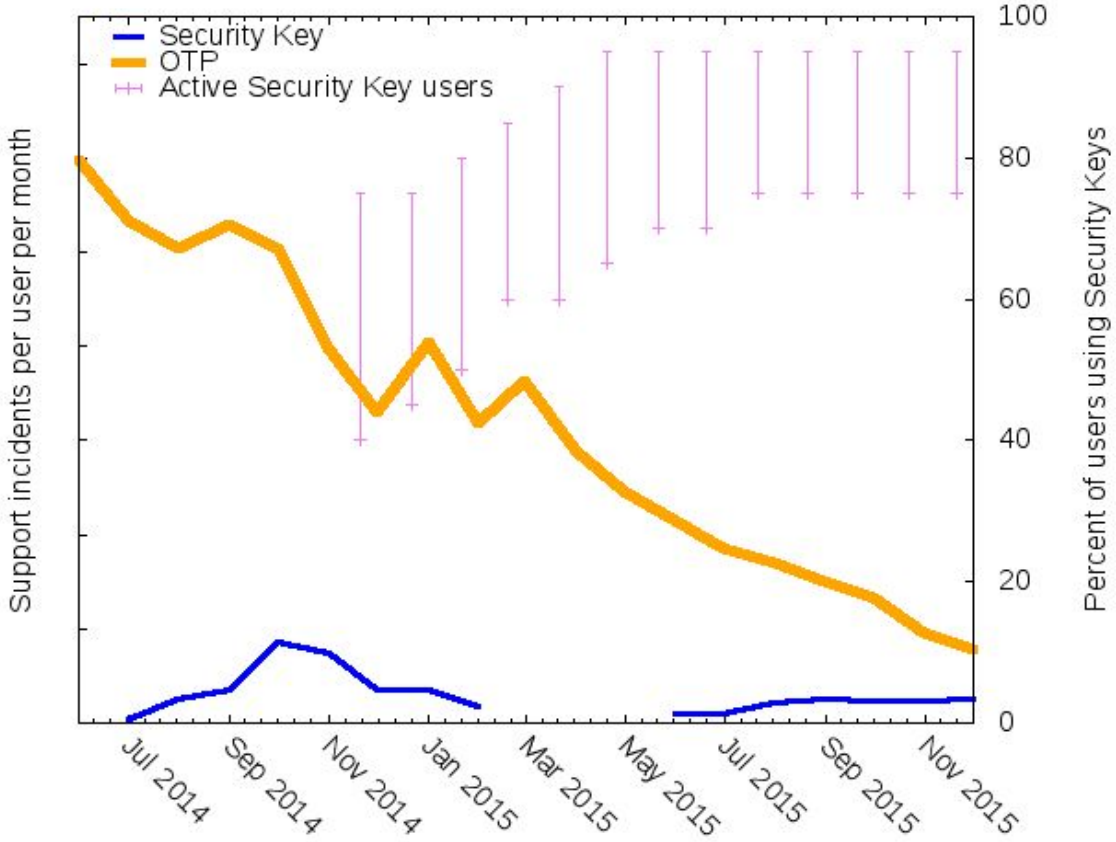
Time to authenticate



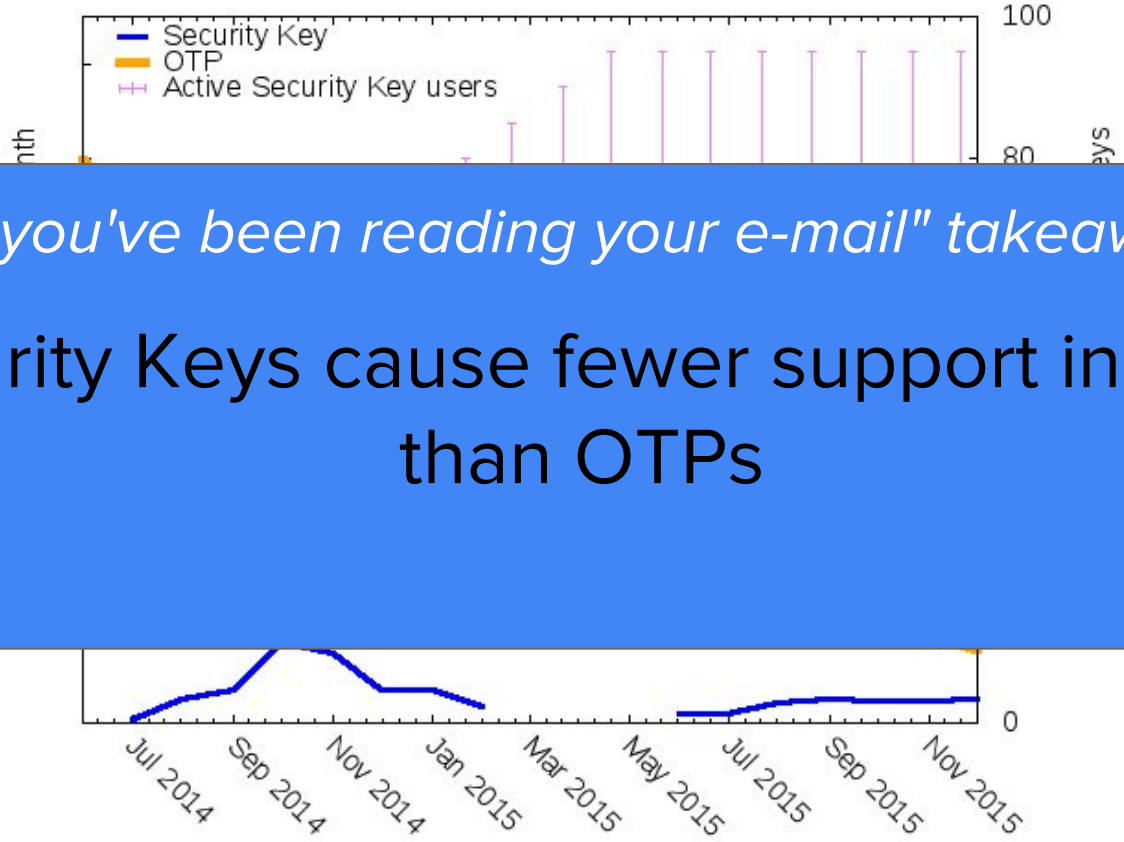
"If you've been reading your e-mail" takeaway:

Security Keys are faster to use than OTPs

Second Factor Support Incidents



Second Factor Support Incidents



"If you've been reading your e-mail" takeaway:

Security Keys cause fewer support incidents than OTPs

We're not quite done...

Ongoing work

- Wireless protocols
  - NFC, BLE
- More browsers
  - Firefox, Edge, more?
- More platforms
  - Android, Windows, OS X/iOS?
- V2 of the protocol
  - Device-centric authentication

How can you get started?

U2F use cases

- **Internal enterprise authentication (B2B)**
  Authenticate to your own web applications, mobile applications, etc

- **Authenticate to your service providers ("token necklace")**
  U2F works well in a non-federated environment
  Complete isolation between various RPs

- **External customer authentication**
  Authenticate your high-value customers using U2F

Resources

- **To use with Google**
  - Enable 2-Step Verification on your account
    - Go to: https://security.google.com
    - Click: 2-Step Verification
    - Click on the Security Keys tab

- **Also use with GitHub, Dropbox, Facebook**

- **And / or play with some code**
  - https://github.com/google/u2f-ref-code
  - https://github.com/google/pyu2f
  - https://developers.yubico.com/U2F/Libraries/List_of_libraries.html

Q & A