How to Delete Data for Realz:
This Presentation Will Self-Destruct In...

Davi Ottenheimer
President
Flyingpenguin
@davioffenheimer

Ian Smith
Research Scientist
University of Washington
@sesosek
Rashomon = Solving “Uncertainty of Fact”*

* “The Samurai Film,” Alain Silver, 1983 p 47
Rising Problem of "unDEAD" Data

The Ghost Kohada Koheiji
Katsushika Hokusai (1760-1849)
Data Lifecycle

Create → Store → Access → Modify → Share → Backup → Destroy
Snapchat Settles FTC Charges That Promises of Disappearing Messages Were False

Snapchat Also Transmitted Users' Location and Collected Their Address Without Notice or Consent

FOR RELEASE
May 8, 2014

5 alarming things that can be undeleted from your phone...

WASHINGTON (AP) — 2 people have been arrested in London in hacking of storage devices that record data from DC police surveillance cameras

Emails over 180 days old, to the extent they existed, were considered “abandoned property.”

The “180-day rule” is a relic. ECPA passed in 1986...
Example: Internet of Snitches

US Government collecting social media information from foreign travelers

Data from pacemaker used to arrest man for arson, insurance fraud
Example: Plaintext Distributed Data to Cloud
Example: War of Words
“Use Bleach” to Purge History

Ryan Adams
@fimystic

hi @IngrahamAngle Oops, looks like you missed a spot when you were purging your tweet history. (#TrumpTip: You're supposed to 'use bleach')
Broken Solutions

- Unlinking
- Overwriting
- Master Key
- Physical Destruction
Classic PGP (No Forward Secrecy) is Classic

\[ E_{pk_{bob}}(M) \rightarrow \text{Server} \rightarrow D_{sk_{bob}}(E_{pk_{bob}}(M)) \rightarrow M \]

sk = secret key
pk = public key
Classic PGP (No Forward Secrecy) is Classic

Attacker with $sk_{bob}$ can read all past messages
In Search of an Improved Trust Level
Distributed Expiring Auditable Data (DEAD)

1. Automatic expiration timers
2. Always, even when replicated or offline
3. Audited
Data, Prepare to be DEAD
DEAD Example Architecture

1. Automatic: Access gone after expiration
2. Always: Stored keys disappear over time, destroying data
3. Audited: Action required for initial data access
Split secret $S$ into $n$ pieces

- Knowledge of any $m$ of them makes $S$ easy to compute.
- Knowledge of any $m - 1$ or fewer leaves $S$ completely undetermined.
Automatic Key Expiration

• Store(secret, expiration time) → index
• Get(index) → secret, if not yet expired
Automatic Key Expiration

- Store(secret, expiration time) → index
- Get(index) → secret, if not yet expired
Always (Forward Secrecy)

\[ E_{pk_{bob}}(E_k(M), I) \rightarrow S \]

\[ E_{pk_{bob}}(k) \rightarrow S \]

\[ D_k(D_{sk_{bob}}(E_{pk_{bob}}(E_k(M), I))) \rightarrow M \]

\[ D_{sk_{bob}}(S) \rightarrow k \]
Always (Forward Secrecy)

Attacker with \( \text{sk}_\text{bob} \) cannot read any expired messages

\[ E_{\text{pk}_\text{bob}}(E_k(M), I) \rightarrow \]

\[ E_{\text{pk}_\text{bob}}(k) \rightarrow S \]

\[ D_{k}(D_{\text{sk}_\text{bob}}(E_{\text{pk}_\text{bob}}(E_k(M), I))) \rightarrow M \]

\[ D_{\text{sk}_\text{bob}}(S) \rightarrow k \]
Audited (e.g. Cloud Delete)

DEAD Service
(audits key requests)
Audited (e.g. Cloud Delete)

DEAD Service
(audits key requests)
Resilient to Attack: Privacy

DEAD secrets can not be read without the index
Resilient to Attack: Availability
Resilient to Attack: Availability
### Resilient to Attack: Privacy + Availability

#### Table:

<table>
<thead>
<tr>
<th>Num. storage locations</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Quorum threshold</strong></td>
<td>1/1</td>
<td>1/2</td>
<td>2/2</td>
<td>2/3</td>
<td>3/3</td>
<td>1/4</td>
</tr>
<tr>
<td></td>
<td>1/4</td>
<td>2/4</td>
<td>3/4</td>
<td>4/4</td>
<td>1/5</td>
<td>2/5</td>
</tr>
<tr>
<td></td>
<td>2/5</td>
<td>3/5</td>
<td>4/5</td>
<td>5/5</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Num. compromised to violate privacy</strong></td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td><strong>Num. failed to violate availability</strong></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>n - m + 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Diagram:

```
S
  S1  S2  S3  S4  S5
```
### Resilient to Attack: Privacy + Availability

<table>
<thead>
<tr>
<th>Num. storage locations</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Quorum threshold</strong></td>
<td>1/1</td>
<td>1/2</td>
<td>2/2</td>
<td>1/3</td>
<td>2/3</td>
<td>3/3</td>
</tr>
<tr>
<td>Num. compromised to violate privacy</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Num. failed to violate availability</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
## Resilient to Attack: Privacy + Availability

<table>
<thead>
<tr>
<th>Num. storage locations</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quorum threshold</td>
<td>1/1</td>
<td>1/2</td>
<td>2/2</td>
<td>2/3</td>
<td>3/3</td>
<td>1/4</td>
</tr>
<tr>
<td></td>
<td>2/4</td>
<td>3/4</td>
<td>4/4</td>
<td>1/5</td>
<td>2/5</td>
<td>3/5</td>
</tr>
<tr>
<td></td>
<td>4/5</td>
<td>5/5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Num. compromised to violate privacy</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Num. failed to violate availability</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ S = S_1 \cup S_2 \cup S_3 \cup S_4 \cup S_5 \]

\[ n - m + 1 \]
Resilient to Attack: Privacy + Availability

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. storage locations</td>
<td>15</td>
<td>n</td>
</tr>
<tr>
<td>Quorum threshold</td>
<td>8 / 15</td>
<td>m / n</td>
</tr>
<tr>
<td>Num. compromised to violate privacy</td>
<td>8</td>
<td>m</td>
</tr>
<tr>
<td>Num. failed to violate availability</td>
<td>8</td>
<td>n - m + 1</td>
</tr>
</tbody>
</table>

Example: 8/15
Distributed Expiring Auditable Data (DEAD)
1. Automatic expiration timers
2. Always, even when replicated or offline
3. Audited
Apply

1. Identify data flows in your organization
   a. Source code
   b. Customer and operations data
   c. Internal communications
   d. API tokens, TLS certs, SSH keys, DB credentials
   e. Internet of snitches
   f. Partners and service providers
2. Assess unDEAD data risk (severity x probability)
3. Flag processes where DEAD required
Apply: Russian Containers

My name is Dave. I am American. Like baseball and pie from apples.
Get DEAD

- Amazon KMS
- Fugue Credstash
- HashiCorp Vault
- Kubernetes secret objects
- Docker SwarmKit secrets
- OpenStack Barbican
DEAD Data Demo

Join our **Focus-On** session
Learn how to make data **DEAD**
- Automatic
- Always
- Audited

Time: 2:45 - 3:30 PM
Session Code: FON3-R11