Fidelius: Protecting User Secrets from Compromised Browsers

Saba Eskandarian, Jonathan Cogan, Sawyer Birnbaum, Peh Chang Wei Brandon, Dillon Franke, Forest Fraser, Gaspar Garcia, Eric Gong, Hung T. Nguyen, Taresh K. Sethi, Vishal Subbiah, Michael Backes, Giancarlo Pellegrino, Dan Boneh
In Browsers we Trust
Can we stop malware from reading the secrets we type in the browser window?
Hardware Enclaves

A trusted component in an untrusted system

- Protected memory isolates enclave from compromised OS
- Proves authenticity via *attestation*
- Enclaves in our implementation use Intel SGX
Challenges

1. Enclave only interacts with outside world through OS
Challenges

2. Browsers have a LOT of code and many bugs/vulnerabilities.
Challenges

2. Browsers have a LOT of code and many bugs/vulnerabilities.
   Vulnerable code in enclave → super-malware!
The Fidelius System

Goal: protect user keyboard inputs to browser from fully compromised OS
The Fidelius System

Keeps browser outside of hardware enclave

Related earlier approach: Microsoft Palladium...
The Fidelius System

Support for HTML forms, simple JavaScript, local storage, and XMLHttpRequests
The Fidelius System

Minimal changes for developers

```html
<script type="text/JavaScript"
src="validator.js"
secure="True" sign="Fi3Rt9mq2ff0">
</script>
```
The Fidelius System

Trusted path from enclave to secure I/O devices

Minimal changes for developers

```html
<script type="text/JavaScript" src="validator.js"
  secure="True" sign="Fi3Rt9mq2ff0">
</script>
```
Trusted Path to/from Enclave

Keyboard/display dongles built from Raspberry Pis

Dongles switch between trusted/untrusted modes
Trusted Path to/from Enclave

Keyboard/display dongles built from Raspberry Pis

Dongles switch between trusted/untrusted modes

Keyboard: encrypt keystrokes at constant rate
Trusted Path to/from Enclave

Keyboard/display dongles built from Raspberry PIs

Dongles switch between trusted/untrusted modes

Keyboard: encrypt keystrokes at constant rate

Display: decrypt overlays sent by enclave
Fidelius for Users

Security indicator lights for keyboard and display

Fidelius for Users

Security indicator lights for keyboard and display

Green overlay verifies who gets data and what data you are giving

Fidelius for Users

Security indicator lights for keyboard and display

Green overlay verifies who gets data and what data you are giving

Security relies on users watching indicators (in our prototype)

Example

User view (photograph)  Malware view (screen capture)

See video demo at https://crypto.stanford.edu/fidelius
What Fidelius Does

- Secure user I/O against tampering, eavesdropping, replay, etc.
- Give trusted Javascript local access to sensitive data
- Only allow data to be sent to designated destination
What Fidelius Does Not Do

- Secure hardware enclave against side-channel attacks
  [XCP'15, GESM'17, BMD+'17, WKPK'17, LSG+'17, CCX+'18, BMW+'18]
What Fidelius Does Not Do

- Secure hardware enclave against side-channel attacks
  [XCP’15,GESM’17,BMD+’17,WKPK’17,LSG+’17,CCX+’18,BMW+’18]
- Protect against dumb web sites

As a one-time security measure please verify the zip code of the property associated with this mortgage account, and your Social Security Number.

PROPERTY ZIP CODE
72947

SOCIAL SECURITY NUMBER
XXX-XX-XXXX

Please enter last four of SSN

[SUBMIT]

Submission success

Thanks! Your security number is 123-45-6789.
Performance

TCB: ~8,500 lines of C++
Performance

TCB: ~8,500 lines of C++

Display Latency Scaling

Doubling trusted display size only slightly increases display latency
Performance

TCB: ~8,500 lines of C++

Display Latency Scaling

Doubling trusted display size only slightly increases display latency

Display Bottlenecks

Expensive Render/Refresh due to implementation hacks, easily improvable

---

<table>
<thead>
<tr>
<th>Field size(s)</th>
<th>W</th>
<th>H</th>
<th>W×H px</th>
<th>Time (ms)</th>
<th>Incr. (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Small</td>
<td>171</td>
<td>50</td>
<td>8,550</td>
<td>195.83</td>
<td>-</td>
</tr>
<tr>
<td>1 Medium</td>
<td>342</td>
<td>50</td>
<td>17,100</td>
<td>199.20</td>
<td>3.38</td>
</tr>
<tr>
<td>1 Large</td>
<td>683</td>
<td>50</td>
<td>34,150</td>
<td>209.65</td>
<td>10.45</td>
</tr>
<tr>
<td>1 Extra large</td>
<td>911</td>
<td>50</td>
<td>45,550</td>
<td>214.74</td>
<td>-</td>
</tr>
<tr>
<td>2 Extra large</td>
<td>911</td>
<td>100</td>
<td>91,100</td>
<td>227.02</td>
<td>12.28</td>
</tr>
</tbody>
</table>

Fidelius Display Pipeline Costs

Latency (ms)

Refresh  
Decrypt  
Decode  
Transfer  
Render  

---

25
Performance

TCB: ~8,500 lines of C++

Display Latency Scaling

Doubling trusted display size only slightly increases display latency

Display Bottlenecks

Expensive Render/Refresh due to implementation hacks, easily improvable
Performance

Display Latency

(Unoptimized) refresh rate 2.8x faster than latest Kindle

Speed due to only sending small overlay rather than encrypting full display

Graph shows latency for Fidelius rendering a username/password login form
Summary

Fidelius uses enclave to protect user secrets even if entire OS compromised

Support for forms, JS, persistent local storage, and XmlHttpRequests

Trusted path to enclave for user I/O (other projects welcome to use)

https://crypto.stanford.edu/fidelius
https://github.com/SabaEskandarian/Fidelius