Building an Internet of Things We Can Trust

Ben Zorn
Principal Researcher and Research Manager
Research in Software Engineering (RiSE) Group
Microsoft Research, Redmond
RiSE – Research in Software Engineering

http://research.microsoft.com/en-us/groups/rise/

Zorn, UWMSR 2017
Rapidly Expanding World of Computing

- Medicine and Global Health
- Energy and Sustainability
- Security and Privacy
- Technology for Development
- Interacting with the Physical World
- Scientific Discovery
- Transportation
- Elder Care
- Neural Engineering
- Accessibility

CORE CSE
- Natural Language Processing
- Sensors
- Big Data
- Cloud Computing
- Mobile
- HCI
- Machine Learning

Graphic: Lazowska

Zorn, UWMSR 2017
Implication:
Every company is a software company
Every object is a computer

Yes, this is a computer too
IoT Devices Cause Unintended Consequences

Mirai malware used to create 380,000 node device botnet

Botnet was leveraged to deliver massive DDOS attack on KrebsOnSecurity

Did I Mention Companies Cheat?

You can hide a lot in 100M lines of code

How They Did It: An Analysis of Emission Defeat Devices in Modern Automobiles

Moritz Contag*, Guo Li†, Andre Pawlowski*, Felix Domke†, Kirill Levchenko†, Thorsten Holz*, and Stefan Savage†

* Ruhr-Universität Bochum, Germany, {moritz.contag, andre.pawlowski, thorsten.holz}@rub.de
† University of California, San Diego, {gul027, klevchen, savage}@cs.ucsd.edu
tmbinc@elitedvb.net

https://cseweb.ucsd.edu/~klevchen/diesel-sp17.pdf
Takeaway: Creating a trustworthy IoT is hard... and we don’t even do the easy stuff well.
The Cathedral and the Skyscraper

Heroic effort, amazing engineering, one of a kind...

Stronger materials, reusable components, mathematical analysis...

Zorn, UWMSR 2017
Trust.... But **Verify**

Two Expeditions: DeepSpec and Everest

Scalable reasoning meets software verification at scale

---

DeepSpec is an Expedition in Computing funded by the National Science Foundation. We focus on the specification and verification of full functional correctness of software and hardware.

**http://deepspec.org/**

$10M NSF Expedition in Computing
Awarded 2016

https://project-everest.github.io/

---

Zorn, UWMSR 2017
Weaknesses in Critical Infrastructure Components: HTTPS

2014

HOW HEARTBLEED BROKE THE INTERNET — AND WHY IT CAN HAPPEN AGAIN

https://www.wired.com/2014/04/heartbleedslesson/

The DROWN Attack

2016

DROWN is a serious vulnerability that affects HTTPS and other services that rely on SSL and TLS, some of the essential cryptographic protocols for Internet security. These protocols allow everyone on the Internet to browse the web, use email, shop online, and send instant messages without third-parties being able to read the communication.

DROWN allows attackers to break the encryption and read or steal sensitive communications, including passwords, credit card numbers, trade secrets, or financial data. Our measurements indicate 33% of all HTTPS servers are vulnerable to the attack.

https://drownattack.com/

Zorn, UWMSR 2017
Everest Expedition: Microsoft Research

Goal: drop-in verified HTTPS replacement

Challenges:
- scalability of verification
- performance
- usable tool chain

Slide courtesy of Cedric Fournet
Subgoal: Verified low-level crypto

Slides courtesy of Nikhil Swamy
Why verify crypto implementations?

Bugs are real, and potentially devastating!
3 bugs in OpenSSL’s Poly1305 this year!

“These produce wrong results. The first example does so only on 32 bit, the other three also on 64 bit.”

“I believe this affects both the SSE2 and AVX2 code. It does seem to be dependent on this input pattern.”

“I’m probably going to write something to generate random inputs and stress all your other poly1305 code paths against a reference implementation.”
Efficient crypto requires customizations

- **Poly1305:** Uses the prime field with $p = 2^{130} - 5$
  - Need 130 bits to represent a number
  - Efficient implementations require custom bignum libraries to delay carries
  - On X86: use 5 32-bit words, but using only **26 bits in each word**
  - On X64: use 3 64-bit words, but using only **44 bits in each word**

- **Curve25519:** Uses the prime field with $p = 2^{255} - 19$
  - On X64: use 5 64-bit words, but using only **51 bits per word**

- OpenSSL has **12 unverified bignum libraries** optimized for each case

**Everest goal:** generic, efficient bignum libraries
Performance of Everest’s High Assurance Crypto Library

- Several complete TLS ciphersuites
- Verification can scale up!

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Spec (F^* loc)</th>
<th>Code+Proofs (Low^* loc)</th>
<th>C Code (C loc)</th>
<th>Verification (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Salsa20</td>
<td>70</td>
<td>651</td>
<td>372</td>
<td>280</td>
</tr>
<tr>
<td>Chacha20</td>
<td>70</td>
<td>691</td>
<td>243</td>
<td>336</td>
</tr>
<tr>
<td>Chacha20-Vec</td>
<td>100</td>
<td>1656</td>
<td>355</td>
<td>614</td>
</tr>
<tr>
<td>SHA-256</td>
<td>96</td>
<td>622</td>
<td>313</td>
<td>798</td>
</tr>
<tr>
<td>SHA-512</td>
<td>120</td>
<td>737</td>
<td>357</td>
<td>1565</td>
</tr>
<tr>
<td>HMAC</td>
<td>38</td>
<td>215</td>
<td>28</td>
<td>512</td>
</tr>
<tr>
<td>Bignum-lib</td>
<td>-</td>
<td>1508</td>
<td>-</td>
<td>264</td>
</tr>
<tr>
<td>Poly1305</td>
<td>45</td>
<td>3208</td>
<td>451</td>
<td>915</td>
</tr>
<tr>
<td>X25519-lib</td>
<td>-</td>
<td>3849</td>
<td>-</td>
<td>768</td>
</tr>
<tr>
<td>Curve25519</td>
<td>73</td>
<td>1901</td>
<td>798</td>
<td>246</td>
</tr>
<tr>
<td>Ed25519</td>
<td>148</td>
<td>7219</td>
<td>2479</td>
<td>2118</td>
</tr>
<tr>
<td>AEAD</td>
<td>41</td>
<td>309</td>
<td>100</td>
<td>606</td>
</tr>
<tr>
<td>SecretBox</td>
<td>-</td>
<td>171</td>
<td>132</td>
<td>62</td>
</tr>
<tr>
<td>Box</td>
<td>-</td>
<td>188</td>
<td>270</td>
<td>43</td>
</tr>
<tr>
<td>Total</td>
<td>801</td>
<td>22,926</td>
<td>7,225</td>
<td>9127</td>
</tr>
</tbody>
</table>

Table 1: HACL^* code size and verification times

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>HACL^* (cycles/ECDH)</th>
<th>OpenSSL (cycles/ECDH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHA-256</td>
<td>13.43</td>
<td>16.11</td>
</tr>
<tr>
<td>SHA-512</td>
<td>8.09</td>
<td>10.34</td>
</tr>
<tr>
<td>Salsa20</td>
<td>6.26</td>
<td>-</td>
</tr>
<tr>
<td>ChaCha20</td>
<td>6.37 (ref)</td>
<td>7.84</td>
</tr>
<tr>
<td>Poly1305</td>
<td>2.19</td>
<td>2.16</td>
</tr>
<tr>
<td>Curve25519</td>
<td>154,580</td>
<td>358,764</td>
</tr>
<tr>
<td>Ed25519 sign</td>
<td>63.80</td>
<td>-</td>
</tr>
<tr>
<td>Ed25519 verify</td>
<td>57.42</td>
<td>-</td>
</tr>
<tr>
<td>AEAD</td>
<td>8.56 (ref)</td>
<td>8.55</td>
</tr>
<tr>
<td></td>
<td>5.05 (vec)</td>
<td></td>
</tr>
</tbody>
</table>

- With performance as good as or better than hand-written C
Everest Impact on the TLS 1.3 Standard

Everest verification efforts led to many of their proposals being included in the standard:

- #4  log-based key separation
- extended session hashes
  (fixing attacks we found on 1.2)
- #11  stream terminators
  (eventually fixing an attack)
- #14  downgrade resilience
- #15  session ticket format
- #17  simplified key schedule
- pre-shared-key 0RTT
- #18  PSK binding (fixing an attack)

Takeaway 4: Verification at scale is real and already having impact
Challenges Beyond Verifying Code

• Is code obsolete?
  • How do we test ML/AI models?
  • What’s the equivalent of code coverage?
  • What math is needed to “verify” an ML model?

• Program analysis focuses on code.
  What tools do we have to debug data?
  • Data wrangling is a hot topic, about getting data fast
  • What debugging tools exist for finding bad data?
Addressing Human Understanding

• We need to retrain everyone to think about smart things differently

• Developer and engineers
  • Design for adversaries – every object is smart and can be hacked
  • Design for failsafe operation – like a limit switch on centrifuge

• End users
  • What is the minimum level of knowledge to be IoT-savvy?
  • Can we really assume people know nothing? What are best practices in smart object hygiene?
IoT is Happening Now

• **Smart objects** will replace dumb objects
• The software embedded in this objects will be written in the next 5 years, but will have implications for the next 50 years
• Our lives will depend on these objects...
• We need languages, tools, and processes to make these objects safe and secure
• **Verification at scale** is possible, necessary, and is already having impact
Thank you!

Research in Software Engineering (RiSE) at Microsoft Research

CCC Computing in the Physical World Task Force
http://cra.org/ccc/task-forces/computing-in-the-physical-world/

Follow me: @benzorn https://twitter.com/benzorn