

Home Address
112 Hancock Street
Somerville, MA 02144

Christopher T. Lesniewski-Laas
<http://lesniewski.org/>
ctl at mit dot edu
Updated January 2010.

Work Address
32 Vassar Street 32-G996
Cambridge, MA 02139
+1 617 253 0004

Experience

MIT CSAIL, Parallel and Distributed Operating Systems Cambridge, MA 2001 - 2010

- Research focus: computer systems, especially security of large-scale decentralized Internet systems.
- Thesis: [Whanau](#), a structured overlay routing protocol (DHT) which uses a social network to provide robustness against powerful pseudonym (Sybil) attacks. Advisor: [M. Frans Kaashoek](#).
- [UIA & Eyo](#): decentralized routing, naming, & storage in a zero-configuration, secure, ad-hoc network.
- [Alpaca](#): secure and flexible PKI based on a higher-order logical framework.
- Other work: distributed and dynamic compact routing for the Internet; coroutine-based asynchronous I/O programming framework; game theory, economics, mechanism design, and reputation in decentralized systems; distributed Web caching; RSA acceleration using a commodity GPU.
- Master's thesis: [SSL Splitting and Barnraising: Cooperative Caching with Authenticity Guarantees](#).
- Instructor, 6.033 Computer Systems Engineering, 2003-2005.
- Visiting scholar, Cambridge University Computer Lab, 2004.

Permabit Cambridge, MA 2001

- Developed highly available, robust, secure, scalable data storage system based on commodity hardware.

Microsoft Research Redmond, WA 2000

- I-Campus *Secure Successor to the MIT Card* project: cryptographic protocol design.

SensAble Technologies, Inc. Cambridge, MA 1999

- R&D: hardware and software development for the PHANToM haptic interface.

MIT AI Lab, Mathematics and Computation Cambridge, MA 1998

- Programmed randomly generated amorphous computers. Advisors: Hal Abelson, Gerry Sussman.

Education

Massachusetts Institute of Technology Cambridge, MA 1997 - Present

- Doctoral candidate, Computer Science, 2003 - present. (GPA 5.0)
- M.Eng. and B.S. Electrical Engineering and Computer Science, June 2003. (GPA 5.0)
- B.S. Mathematics (Minor in Physics), June 2001. (GPA 4.9)
- Topics: algorithms, complexity, compilers, software design, modeling, cryptography, architecture, digital design, signal processing, probability, algebra, quantum+stat physics, general relativity, economics.

Cohasset High School Cohasset, MA 1992 - 1997

- Valedictorian, early graduation, Harvard Extension School, Center for Talented Youth (CTY).

Societies

- MIT [Student Information Processing Board](#) (Chair, 2003-2004)
- [Eta Kappa Nu](#) (editor of UG6, 2000-2001)
- [Phi Beta Kappa](#)

Skills

- Languages: Python, Haskell, C, C++ STL/Boost, Java, Perl, Javascript, LISP, Matlab, VHDL, Postscript, various assembly, SQL, XML, HTML, CSS, LaTeX, GLSL, LF, Intercal, French, Chinese
- Network/system programming: TCP/IP, sockets, SSL/TLS, Kerberos, asynchronous, threads, load balancing, scheduling, consistency, kernels, compilers, JIT, virtualization, RDBMS, web apps, etc.
- Unix development: Make, GCC, git, Subversion, svk, VIM, X11, test suites, Ubuntu, Solaris, etc.
- Digital design: Xilinx FPGA development tools, use of oscilloscope, logic analyzer, datasheets, etc.
- Hobbies: coding, cycling, photography, cooking, SCUBA, travel, hiking, karate, economics

Software systems developed at MIT

Whanau

2010

- Designed and implemented a secure distributed hash table (DHT), a decentralized structured overlay network which can quickly look up the node responsible for a given key. (Existing DHT applications include distributed databases, filesystems, caching, rendezvous, and streaming multicast.)
- Novelty: Whanau uses an online social network to bootstrap a robust overlay network. It is secure against powerful denial of service (DoS) attacks, including the pseudonym-based “Sybil attack.”
- Implementation: high-performance in-memory simulator (C++/Boost), asynchronous network daemon (Python) deployed on PlanetLab testbed. Solo.
- Supervised Master’s thesis implementing secure SIP rendezvous over Whanau (Java).

UIA

2006

- Designed, implemented, debugged, and demoed a routing and naming system which ties together users’ many personal devices (e.g., laptops, phones, cameras, media players) into a coherent cluster. After devices are named and introduced to each other, UIA ensures that they can communicate whenever physically possible. Users can refer to each others’ devices by recursive names such as *phone.dad.bob*.
- Novelty: UIA maintains a shared, concurrently-modified namespace across intermittently-connected devices, and securely propagates peer-to-peer updates without relying on a master server.
- Implementation: routing module and kernel hooks (C++/Boost), UI (C++/QT), name database and resolver (Python). Team: 4 core developers, 2 PIs. Also incorporated into a Nokia product demo.

Eyo

2009

- Continues the UIA project. Eyo is a data storage system and API which provides a consistent view of a user’s data objects (e.g., photos, music, email) across all of her devices. Eyo tracks object updates, forwards changes to running applications, handles network partitions and concurrent updates, and proactively partitions and replicates data across heterogeneous devices.
- Novelty: Eyo separates objects’ metadata from their content and distributes all metadata to all devices, while partially replicating content to some devices.
- Implementation: storage system (Python), C API (C/D-Bus), example applications (Python and C). Team: 3 core developers, 1 collaborator, 2 PIs.

Alpaca

2007

- Invented and implemented a logic-based proof-carrying authorization protocol. Alpaca provides an API enabling network applications to state and prove logical assertions such as “the principal Alice says to delete the file X” using cryptographic operations specified in the accompanying proof.
- Novelty: verifiers don’t care how the proof is structured, as long as it is valid. Thus, Alpaca permits provers to use different cryptographic techniques (e.g., new hash functions or data transport mechanisms) without breaking compatibility with existing verifiers. Alpaca’s flexibility is more “future-proof” than crypto protocols such as Kerberos and TLS, which can only be updated by installing new software.
- Implementation: logic language, logic engine, cryptography, test suites, demos (Python). Solo.

Barnraising

2003

- Designed and implemented a peer-to-peer content distribution network (CDN). Barnraising enables Web sites to delegate some of their load to a distributed network of cooperating cache hosts.
- Novelty: Barnraising uses a new technique called SSL Splitting to securely serve data using untrusted caches. Because a malicious cache cannot send clients bogus data, Barnraising can safely permit any Internet host to contribute cache space. Other systems are limited to centrally-controlled cache servers.
- Implementation: SSL Splitting library (drop-in replacement for popular OpenSSL library, C), caching Web proxy, tracker, and DNS server (Perl). Solo.

See also

- CV: <http://lesniewski.org/cv.pdf>
- Twitter: <http://twitter.com/lesniewski>
- LinkedIn: <http://www.linkedin.com/in/chrislesniewski>
- Facebook: <http://www.facebook.com/lesniewski>