A TALK ON LEGIBLE COMPUTING

Lisp-steading

Staying above water when everything is bugged or pwned



[Your name] · [Event] · [Date]

THE WATER LINE · OCTOBER 2026

Everything is bugged or pwned. Patching faster isn't keeping us dry.

> *"if it isn't formally proven, asi is going to zero day it"*
> @ynniv, August 2025

Oct 13 · Microsoft
1,413 flaws in one Patch Tuesday, more than its old annual record

Oct 21 · PHP
PHP2Shell: unauthenticated RCE in the engine itself, exploited before the patch shipped

Oct 21 · Node.js
Out-of-bounds write in the Brotli decoder it bundles

Oct 13 · Ivanti
Root on a fully patched VPN

Oct 17 · eMudhra CA
Eight intermediate CA keys turned out to be guessable

Oct 21 · LiteLLM
Poisoned PyPI release steals cloud and AI keys

Oct 28 · UEFI firmware
EDK II network-boot overflow runs code before the OS loads

THE PROBLEM

Every layer below you is a place to hide

Your app	You read it
Frameworks and packages	Skimmed, once
Language runtime	Its maintainers
C libraries: libc, OpenSSL, SQLite	A few experts
Kernel and drivers	Fewer still
Firmware and blobs	The vendor, maybe
Management engines, secure enclaves	Nobody you can ask

THE IDEA

Lisp-steading: own the floor, grow what you eat

> *Homesteading for computing. A stack small enough for one person to read, written in one language you can inspect while it runs.*

Fewer layers

Collapse the stack until nothing sits between you and the hardware.

One language

The same Lisp from the boot vector to the browser engine.

Outside judges

Every claim checked by something you didn’t write.

WHY LISP

The right soil for a legible system

`'(code is data)`

Programs you can read with programs

Source is a list. The tools that walk, check and rewrite it are the same tools you use for everything else.

`(inspect *system*)`

A running image you can open

Boot to a REPL. Inspect and change the live system without a rebuild, a reboot or a deploy.

`#'genera`

It has been done before

Lisp machines ran whole workstations in Lisp. The idea was abandoned for economics, not refuted.

THE PROOF

Modus

A personal computer written from scratch in Common Lisp, from the bare-metal kernel up.

modus-lisp.github.io

BY THE NUMBERS

Counted from the commit log, not estimated

62

repositories, clean-room and portable Common Lisp

6,234

commits across 34 working weeks

100%

of the unmodified ANSI suite: 21,681 of 21,681 (x86-64)

0

divergence from Bitcoin Core’s own library on the taproot vectors

THE FLOOR

It boots on what's in the closet

Pi Zero 2 W

Booted over USB with no SD card. A stock OpenSSH client logs in.

An iPhone

Runs natively: 11 of 11 runtime tests on the device. It draws on a watch, too.

A 486

Thrift-store hardware is a target, not an afterthought.

AMD SEV-SNP

Measured boot in a confidential VM, attested end to end.

A browser tab

The same Lisp, running inside a web page.

Nine CPU backends

One portable bytecode ISA underneath all of them.

COLLAPSE THE STACK

One dependency at a time, until the C is gone

Usually	In the workspace, pure Lisp
OpenSSL	seal: TLS 1.3 and 1.2, clean-room
OpenSSH	conch client, plus Modus’s own server
SQLite (C)	sqlite-pure: shares a live file with 3.40
The git binary	cairn: validated against real git
libvpx, ffmpeg	reel: eight codecs, bit-exact
Bitcoin Core	cl-consensus: a full node, zero divergence

Each replacement is **tested against the C it replaced**.

BUILD THE ORACLE FIRST

Nothing is graded by the thing that wrote it

What was built	What grades it
Common Lisp	Paul Dietz’s ANSI suite, unmodified
Bitcoin consensus	Core’s own compiled library
Video codecs	ffmpeg and conformance suites
SVG renderer	Chromium
Fonts	ttx and HarfBuzz
Lightning	Core Lightning and LND, on signet
SSH server	A stock OpenSSH client

THE HONEST FIGURE

Retracting a number is part of the security model

17,587

tests behind a reported 99.994% had never actually run. The harness was rebuilt until it lost none.

150× → 5×

a speedup claim, retracted once SBCL and libvpx ran on identical silicon.

ATTESTED, END TO END

Trust you can check, not trust you're told

The chip signs
An AMD processor signs a report about the machine it runs.

→

Modus checks
AMD's whole certificate chain, verified in pure Lisp.

→

It matches
The launch measurement equals the value computed offline.

→

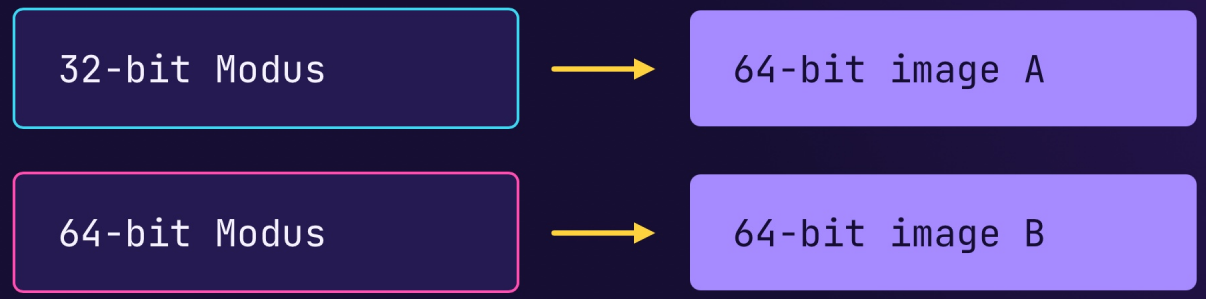
Only then
Key-only SSH. The agent refuses routes that are encrypted but not attested.

Flip one bit of the measurement and it refuses.

witness publishes who attested which machine, on Nostr, as a web of trust.

THE FIXPOINT OF THESEUS

The compiler leaves no fingerprint of the machine that ran it



A = B

Bit for bit: five SHA-256 comparisons across three architectures.

Since July, Modus compiles Modus with no host Lisp anywhere in the loop.

WHY NOW

From scratch used to cost a company. Now it costs attention.

One person, an AI, closet hardware

Symbolics needed a building full of engineers and custom silicon. This took a subscription and a person who knows what matters.

But the floor does not compound

Output across the workspace accelerated. The kernel underneath never did. The bottom stays the hard part.

OPEN PROBLEMS

What’s still underwater

- ⚠ Blob-free Wi-Fi is still a goal, not a result.
- ⚠ The full desktop doesn’t run on Modus yet: a viewer sees the framebuffer, not the window manager.
- ⚠ The attested build’s core is attested by hash, not yet reproducible.
- ⚠ Full ANSI conformance is proven on x86-64; [other architectures: status].
- 💡 Legible isn’t proven yet. A stack this small is what makes proof affordable.

YOUR TURN

Start your own lisp-stead

01 **Pick a floor you own**

A machine you can boot and replace: a Pi, an old laptop, a VM.

02 **Build the oracle first**

Before the feature, the thing that grades it, written by someone else.

03 **Pull one root at a time**

Audit where C and blobs get in, then replace the deepest one you can.

04 **Publish legibly**

Counts, not estimates. Plain-text mirrors, and retractions in public.

Simple enough to read is simple enough to defend.



modus-lisp.github.io

github.com/modus-lisp

[Your name] · [contact]