

Notes on coding style and practice

February 1, 2006

I know. What's an ivory tower academic doing telling you l33t haxors how to code. It's lame. I'm sorry. If your code works 100%, I'll never have to look at it and I won't care.

Chances are, though, that you'll make the mistakes nearly *everyone* makes, and this sketch of notes may help. Some are C-specific in expression but are likely applicable to other languages.

- Your project code must compile using `-Wall -Werror -Wshadow -Wpointer-arith -Wwrite-strings`. Yes.
- No threads. Threads are easily abused and are wasteful. Non-blocking socket calls are okay.
- No signals. Signals lead to more bugs than cleanliness. Don't even think about calling `alarm()`.
- Use `const` annotations for buffers passed to functions. They make explicit that values will not change.
- Use `static` annotations for functions that should not be exported outside a module.
- Avoid modifying fields in-place so that each variable is either meaningless (uninitialized) or has a good value. Specifically, I never want to see:

```
packet->source = ntohs(packet->source);
```

In other words, fill in packet fields that should be in network byte order in network byte order, and read them out when needed. (You may choose to avoid this rule if you're really smart and intend to decode as you demultiplex protocols.)

- Avoid extra calls to `return`, `continue`, `next`, or `goto`. You may use `goto` as a means to jump to cleanup code on an error. Concurrent programs more often require function cleanup code—closing temporary sockets, releasing locks—and extra return calls make the code harder to maintain.
- Do not assume data objects are strings: `strlen()` is unlikely to serve you in this class. Every packet and every buffer will need explicit length information.
- I believe in the conditional assignment operator: `x = boolean ? true : false`.
- I believe in iterating (linked) lists using for loops.
- Variables should be named intuitively: higher values of “badness” (error rate, latency, cost) should be more bad, higher values of “goodness” (quality, reliability) should be more good.
- Variables with units should be named with those units: `seconds`, `milliseconds`, `microseconds` are common, and if confused, can cause very bad behavior. Use: `timeout_ms`, `rtt_us`, `hello_interval_s`, and so on.
- Though not precisely a style tip, please use a version control system, such as CVS, to track a history of somewhat working revisions. Don't copy your source files to many directories when you decide to try random edits to make the code work.