Selecting and Maintaining a Cryptographic Module for ISO 19790 and CC

Iain Holness
CygnaCom Solutions

CygnaCom Solutions Laboratories

- Accredited FIPS and Common Criteria laboratories
- Consultation Services
- Professional Services

Topics

- selecting a module or DIY: pros and cons of each
- combinations
- entropy
- keeping up to date
- maintaining disclosure

Assumptions

- maintaining = you're not creating a one-off product, i.e. you're in it for a multi-year commitment
- You may be doing additional testing beyond FIPS and CC (UCR and STIG fun)
- Your resource investment (personnel, $$) is significant
- You recognize that the greater the complexity, the more scrutiny will be cranked up

Module choices: DIY versus pre-existing

CARTOONSTOCK .com
Search ID: lcan850
CHAPMAN
"Just a pinch, Helga ... spicy eye of newt doesn't agree with me."

DIY versus pre-existing: Pros and Cons

	DIY		Pre-existing	
	Pros	**Cons**	**Pros**	**Cons**
Source code	You own it	you need to implement the published algorithms correctly	Much of the legwork is done	
Reported bugs	People look to you for answers on your crypto	You do the full research to resolve all issues	You look to see what the originator is doing	You're stuck until a patch comes out, unless there's a mitigation strategy
Maintenance and updates	You keep abreast of all issues and issue updates	If it's a big issue, you need to be proactive to check	You look for updates	

Combinations: SW vs. HW vs. hybrid

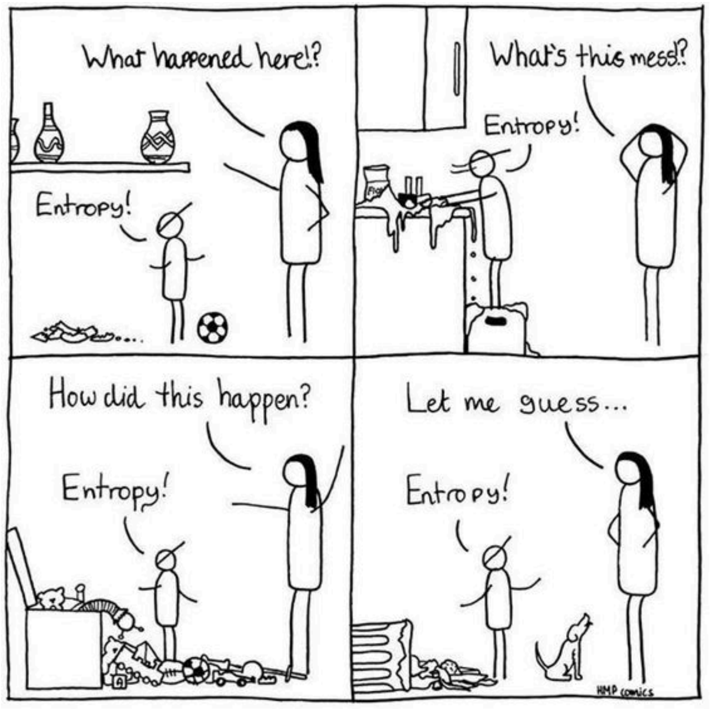
- remember the CAVP aspect of ISO 19790: crypto in HW must be stable for years to justify the choice
- SW - much easier to update
- SW & HW - isolate exactly what needs to update, and consider wrappers in the future
- HW - expensive to update, so minimize what is in HW
- DIY approach will mostly go with SW or SW&HW
- going with already-approved typically means SW, except if you're already maintaining a previously-validated hybrid module

homemade crypto vs published standards

- stick with the published implementations that NIST provides, otherwise expect LENGTHY discussions proving yourself
  - It can be significant rework if you lose the argument(s)

- U.S. & CDN federal governments / DOD / large organizations are looking for the "FIPS validated crypto" checkbox, nothing else
  - Johnny's super-fast, better-than-AES crypto??

Entropy: hard to prove at the best of times



This is why we don't teach our children about entropy until much later...

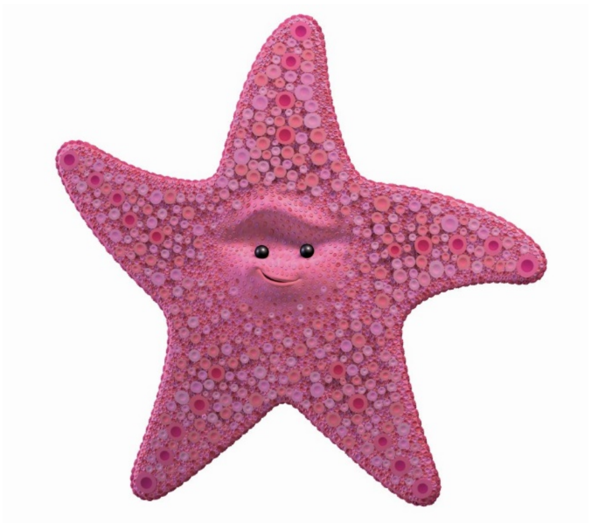
THE FILE ROOM AT THE CHAOS THEORY & ENTROPY INSTITUTE.

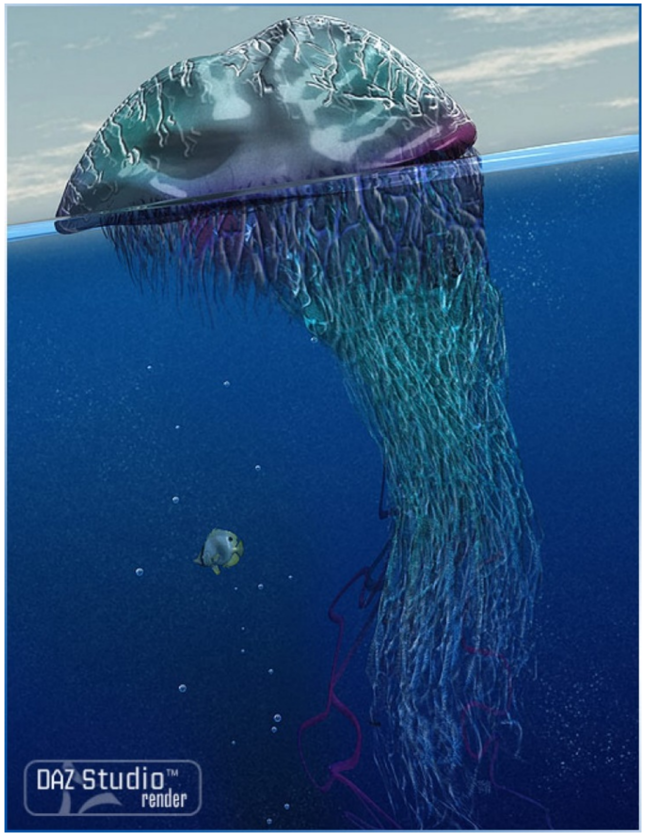
Entropy: hard to prove at the best of times

- expected by CMVP and NIAP
- need to look at hardware used as part of this
- collecting the raw data for analysis takes time (many weeks)
- writing the EAR takes time!
- reviewed by IAD group within NSA (weeks to months)
  - fudging the data is a BAD idea (even if by accident)
- DIY: expect to open up more of your code

Keeping Up to Date

- How many points on each?





Keeping Up to Date – CERT advisories

- CMVP and NIAP expect that you check CERT advisories regularly

- You need to be able to track down EVERY point in your own code that interacts with a part of the crypto module itself or its library / libraries
  - minimize these points as MUCH as possible
  - vetting CERT advisories becomes more efficient
  - when someone comes running and screaming "the code is broken!", you can be Yoda and say "fret not youngling"

Keeping Up to Date – how many modules??

We're assuming you are using the same crypto module and crypto library (OpenSSL, wolfSSL, LibreSSL)

- if you use one combination of module and library in one place, then another combination in several others, expect a LOT more effort
  - For example, issues affect OpenSSL 1.0.1x may not affect 1.0.2x, but you may have 1.0.1x everywhere expect in two spots where you've got the latest-and-greatest 1.0.2x
- if you have mixed flavours of the same crypto module, consider architecture changes in the future to simplify your codebase
  - You'll need to plan this a good 6-12 months in advance, and you'll need your software architects and senior developers onboard
- if you are already in a hybrid situation with multiple crypto modules (Mocana, OpenSSL, etc.), then your security architect is busy!

Keeping Up to Date - Updating your certification

- You know you can't keep the certified version frozen
  - At some point algorithms, modes, key sizes change
    - Either NIST determines they are no longer secure, or have a timeline determined when they will be

    OR
    - A researcher finds a weakness or outright breaks something

- You need to be vigilant about changes that might touch on the source code for the crypto module (both SW and HW)
  - Don't find this out when running the FIPS test hardness and things break unexpectedly in front of the lab guy
    - It's not the end of the world, but you may end up having a mechanism or two non-certified for one revision

Keeping Up to Date - Updating your certification

- FIPS test harness
  - Plan to check this at least at every major revision
    - Sanity check to see if something unexpected pops up
      - Hopefully the break is simple
  - Rotate the developers who work on this
    - Fresh eyes are always a good thing
    - It removes a single point of failure if an issue pops up and the principal developer is legitimately MIA
- CAVP testing: what if you get failures back?
  - No, it does NOT necessarily mean your code is broken
    - Have a seasoned developer review the results (hint!), and it is possible to push back
      - The NIST tool gets updated often

Keeping Up to Date – when to change

Mechanisms, settings and key sizes

- You should be changing your FIPS mode settings at least 6 months prior to when NIST says they're no longer valid
- When a HW crypto mechanism has less than 12 months left, begin shifting the more complex portions of the mechanism into SW and minimize the calls to keep core computations stable in the HW
- SW is obviously quicker to update, but don't short-change the necessary QA cycles
- Plan the scope of the effort with your lab, so they have time in their schedule and you're not caught in a last-minute scramble

Disclosure (OMG don’t tell them!!!)

Non-technical **BAD** example:

“This all came about through the discovery of a single, isolated case of mad cow disease in one Alberta cow on May 20th. The farmer…knew nothing about cattle ranching…any self-respecting rancher would have shot, shovelled and shut up, but he didn’t do that.” — Ralph Klein, Western Governors Association meeting in Big Sky, Mont. September 2004.

Disclosure

Technical **BAD** example:

A network-wide system update by Starbucks in April 2015 resulted in 60% of stores closing.

The register malfunction occurred because of an "internal failure during a daily system refresh" according to Starbucks. The software failure left thousands of stores across North America unable to proceed with their business as the cash registers were unable to process orders and take payment. Starbucks **refused** to give any details.

Point of Sale Terminal = networked computer running customized software

Bottom line

Hiding an issue is NOT an option, but you need to be strategic in how you admit to it and provide a solution for your customers.

Disclosure

- Situation: You know there are issues with your product that *will* affect your customer base, and it affects your certified crypto module.
- Initial technical steps:
  1. Figure out exactly how it happens (down to the incorrect bit or flag set by accident)
  2. Develop a solution, test it out as many ways as needed.
  3. Ping the lab that certified your crypto module, get their advice on how to address the issue with CMVP, have the lab test your solution ASAP and confirm independently that your fix (or workaround in advance of the fix) solves the issue, then get the ball rolling with CMVP.

Disclosure

- Customer-facing steps:
  1. Let your customers know, tailoring the message to the knowledge-level of the individuals in each customer organization.
  2. Assuming you have a website for instructions on fixing or tweaking your product(s), that's where you put the patch and instructions for updating the crypto module component in your product.
- Good news (apart from happy customers):
  - The optional flaw remediation component in CC actually requires this, and ISO 19790 looks favourably on this.
  - Your proactive disclosure will be appreciated, CMVP will work with you, and your reputation will be preserved.

In this Presentation we explored

- selecting a module or DIY: pros and cons of each
- combinations
- entropy
- keeping up to date
- maintaining disclosure

Questions?