Adaptive Code Assisted Security Assessments

Matt Jones, elttam

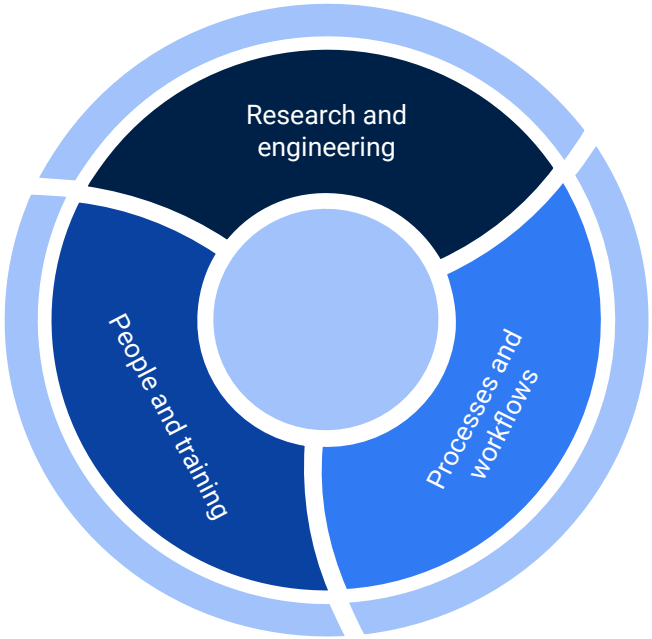
Being adaptable is more essential than ever

Over the past decade we've seen:

- **Practices steadily maturing:** The Appsec and DevSecOps space has steadily matured
- **The need for testing firms to evolve:** Firms need to invest in engineering to be efficient and provide genuine value on projects

This presentation shares some learnings and will look at how we've used engineering to:

- Support processes and workflows
- Building and leveraging tools
- Pros/cons of different options



elttam

A few upfront points about engineering

Decisions and considerations

- Build vs. buy vs. leverage OSS
- Security and data sovereignty
- Measuring ROI

Dodging pitfalls

- Shaving the yak
- Rabbit holing

Goals and outcomes

- Flexibility - the right tool for the job
- Collaboration - supporting teamwork
- Capability Uplift - improve effectiveness
- Efficiency
  - Automation
  - Reduce context switching

elttam

Consultants see a big range of tech.

- The range of technologies, programming languages, frameworks, and industries seen on consulting projects is huge
- Specialisations lean much deeper into technologies and languages, rather than just the test type (e.g. “web” or “cloud”)
- Project collaboration is increasingly important, particularly on projects with many components and/or intricacies
- Finding ways to become more efficient and effective is key to stay ahead

python™ GO Java node JS Rust C++ kubernetes aws Azure

*recent/current

elttam

Processes

Workflows

General:

- Each consultancy has different technology stacks for managing workflows and projects
- Documentation/wikis, Tasks, and Project Management may be managed by different tools entirely
- Data sovereignty and access control may be a critical requirement for customers

Our solution:

- Everything Git - documentation, tasks, wiki, engineering tools, research projects, and customer projects
- Asciidoctor was chosen for our document model
- Building bespoke tools and CICD workflows that automate phases in process, including secure archival

2015

O365

Started with the dreaded Microsoft Word.

2016-2018

Experimentation

Experimented with several self-hosted tools to assist in document generation, findings tracking, etc.

2019

Migration to Git

Chose to separate everything out as repos to standardise documentation, tasks, and projects in one place

2024

Mature Workflows

Fully dockerized, incremental improvements from a backlog

elttam

Reporting

General:

- There are many commercial and open-source reporting tools available for security testing
- Cloud hosted introduces concerns, and many self-hosted solutions had limitations

Our solution:

- Built our own leveraging Asciidoctor so projects are run end to end within a Git repository
- Features have been built steadily to support all use cases, including automated linters through to secure archival, encrypted delivery, and data exports.
- CI/CD is used for builds - everything is tagged (reproducible) and Dockerised

```
[source,typescript]
----
async function checkRecoveryCode(id: string, code: string): Promise<ShowCode> {
  const user = await dbService.getUser({ id: id })
  if (!user) {
    throw new NotFound(Errors.UserNotFound)
  }
  const { recoveryCode, recoveryTag, recoveryIV } = user.mfa
  const decryptedCode = decrypt(recoveryCode, recoveryTag, recoveryIV)

  const success = code === decryptedCode <1>

  [...]

}
----
<1> A time safe comparison check is not being performed between `code` and `decryptedCode`.
```

The following shows the `checkRecoveryCode` not performing a time safe comparison check:

```
async function checkRecoveryCode(id: string, code: string): Promise<ShowCode> {
  const user = await dbService.getUser({ id: id })
  if (!user) {
    throw new NotFound(Errors.UserNotFound)
  }
  const { recoveryCode, recoveryTag, recoveryIV } = user.mfa
  const decryptedCode = decrypt(recoveryCode, recoveryTag, recoveryIV)

  const success = code === decryptedCode ①

  [...]

}
```

① A time safe comparison check is not being performed between `code` and `decryptedCode`.

elttam

Environments

General:

- The ability to reproduce specific stacks helps with speed and consistency with consulting and research projects
- Manually creating environments is time consuming and prone to inconsistencies
- Blindly trusting 3P images and environments can be a big concern

Our approach:

- When a certain environment is in demand and setup is time consuming, we invest time to create a repeatable environment for testing and research

Terraform

docker

ANSIBLE

elttam

Processes - Recap

01	Workflows - Git	• Pros: Quick to get running, logical structure, version control, CICD automation is a big plus • Cons: Occasional bugs and refinements needed, not overly friendly for back office staff
02	Reporting - Bespoke tool	• Pros: Extends OSS, clean design, reasonable flexibility, supports CICD • Cons: Moderate upfront investment, regular maintenance, some limitations (e.g. Word)
03	Environments - Adhoc	• Pros: Useful knowledge gains, very convenient when running well. • Cons: Requires maintenance, incredibly prone to rabbit holing and over engineering

elttam

Tools and Extensions

Tools and Extensions - Static analysis

General:

- Many larger commercial options are not viable due to licensing or reliant on cloud. Semgrep and CodeQL have become great options for consultancies
- Creating ad-hoc rules during assessments to improve efficiency is key, particularly for enormous code-bases
- Requires a solution that is well maintained and refined

Our approach:

- Semgrep has become our goto
- Private rule-sets with internal guides, rules from engagements, and patterns to aid auditing.
- Public rule-sets - constructs we've found missing in public rules and outputs from research projects.
- Investment into upstream bug fixes and enhancements

2015-2017

Leverage OSS Static Analysis

Technology/language specific analysis tools .

2019~

Semgrep Adoption

Internal guides, customer rule-set development, and upstream bug pushes

2022

Public rule-pack release

Open sourced our own public semgrep-rules.

2024

BAU

New rules created from retrospectives or part of research projects

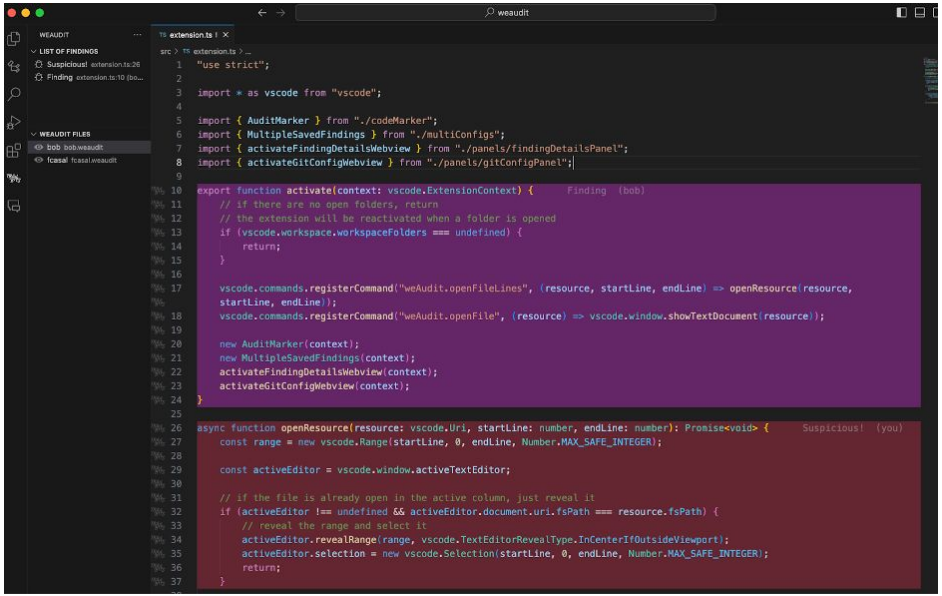
elttam

Tools and Extensions - IDEs and Extensions

- The right tool for the job - VSCode has many features and cross-platform. However, other specific IDE's tools can be better, such as IntelliJ and Source Insight.

- VSCode extensions can be useful to allow collaboration, reduce context switching, and overlay useful data.

- Useful VSCode extensions include:
  - Security Notes
  - weAudit
  - Sarif Explorer
  - GitLens



elttam

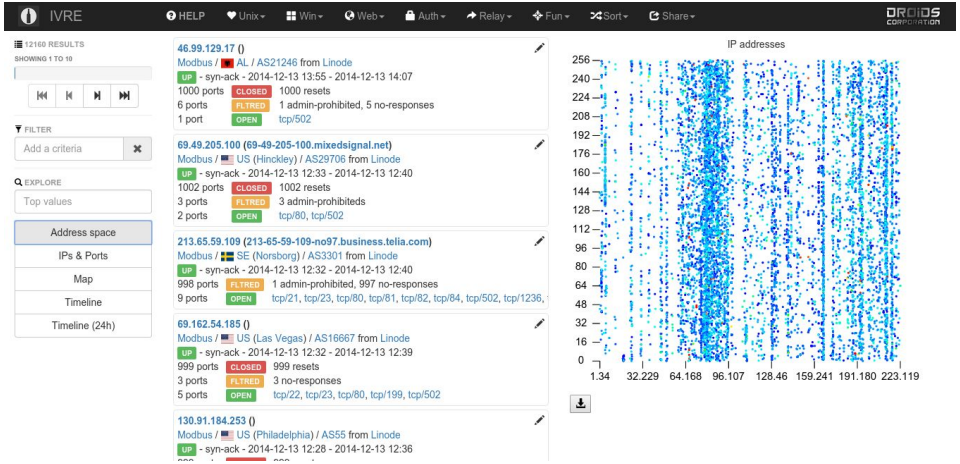
Tools and Extensions - Perimeter Assessment

General:

- Many product assessments require a network component - dangling infrastructure, misconfigurations, scaling tests.
- A huge selection of commercial and open-source options available, with both steadily maturing.

Our approach:

- Started simple with basic manual network reconnaissance tools as part of assessment methodologies.
- Leveraged commercial offerings to assist and make recon and scanning more efficient.
- Invested time into building our own tool (on top of Ivre) to aggregate multiple data sources and run further analysis.
- After realising we rabbit holed, took a step back and simplified our guides to either use commercial or OSS.



elttam

Tools and Extensions - Recap

01	Static analysis	• Pros: Regularly improving, versatile, extendable, can be normalised easily. • Cons: Initial learning curve, requires refining, shortcomings/bugs that can be a time sink.
02	IDE's and 3P Extensions	• Pros: Flexible, extendable, and reduces context switching drastically • Cons: Licenses of some IDEs, occasional bugs, 3P extensions require review
03	Perimeter Assessments	• Pros: Fun problem! Tools always maturing and well maintained. • Cons: Highly susceptible to rabbit holing, licensing costs can be expensive.

elttam

Resource Library

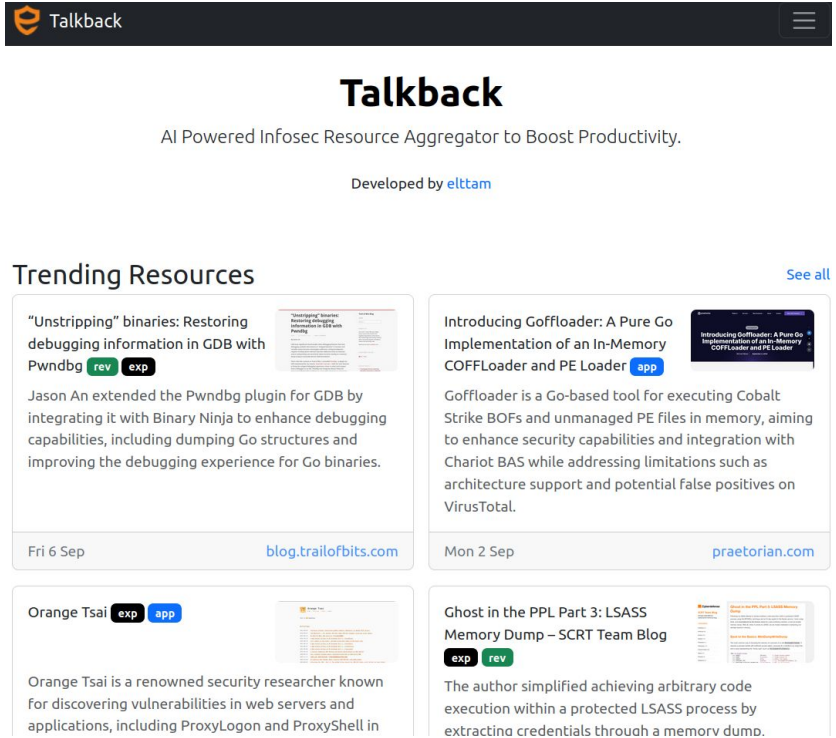
Resource Library

General:

- Infosec is fast moving with lots of noise
- Various mediums exist - Podcasts, social media, newsletters, intelligence feeds, etc.
- Getting up to date information on specific technologies, topics and attack trends is challenging.

Our approach:

- Invested time to build our own aggregator tool with a web UI, mobile friendly, and API
- Fully autonomous tapping into dozens of mediums and hundreds of RSS feeds
- Many features, including TLDR summaries of content, resource-ranking, resource classification, topic extraction, …



elttam

Filtering and searching

Talkback Home Chronicles Resources Vulnerabilities Categories Topics Feeds About graphql

Talkback / Resources

Showing 1 to 20 of 257

« Prev 1 2 3 4 ... Next »

A Confused Deputy Vulnerability in AWS AppSync cloud app

1 year ago - securitylabs.datadoghq.com

C2 and the Docker Dance: Mythic 3.0's Marvelous Microservice Moves

1 year ago - posts.specterops.io

Listen to the whispers: web timing attacks that actually work app

1 month ago - portswigger.net/research

CI/CD secrets extraction, tips and tricks app cloud

1 year ago - synacktiv.com

Testing GraphQL APIs | Web Security Academy app net

1 year ago - portswigger.net

Nemesis 1.0.0

Filters Clear Filters

URL

URL

Home

Tag

c[vw]e-

Type

Category

appsec

Topic

Featured

Vendor

Medium

Show Only

Saved resources

Read resources

Reset Apply

elttam

Resource Details

Home / Resource Details

“Unstripping” binaries: Restoring debugging information in GDB with Pwndbg

— *blog.trailofbits.com*

By Jason An GDB loses significant functionality when debugging binaries that lack debugging symbols (also known as “stripped binaries”). Function and variable names become meaningless addresses; setting breakpoints requires tracking down relevant function addresses from an external source; and printing out structured values involves staring at a memory dump trying to manually discern field boundaries....

5 min read

Key Details

URL	https://blog.trailofbits.com/2024/09/06/unstripping-binaries-restoring-debugging-information-in-gdb-with-pwndbg/
Date	2024-09-06 - 2 days ago
Categories	reverse exploit

OpenAI Summary

Debugging stripped binaries in GDB can be challenging due to the loss of functionality like meaningful function and variable names.

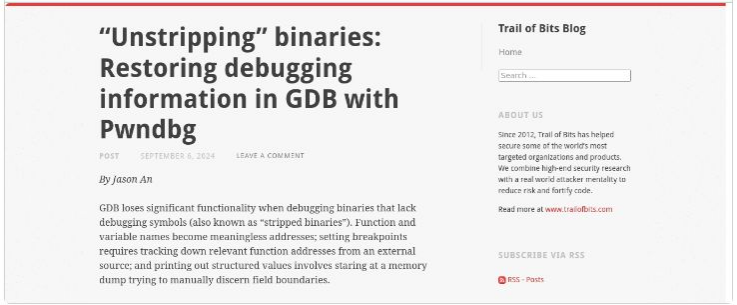
Jason An extended the Pwndbg plugin for GDB by integrating it with Binary Ninja to enhance debugging intelligence and enable dumping Go structures for improved Go binary debugging.

The Binary Ninja integration allows Pwndbg to access Binary Ninja’s analysis database for syncing symbols, function signatures, and stack variable offsets.

An added feature displays the current program counter as an arrow in Binary Ninja and allows setting breakpoints from within Binary Ninja.

Jason also created the go-dump command for dumping Go values and developed a parser to extract type information for arbitrarily nested types, improving the debugging experience for Go binaries.

Screenshot



Wordcloud

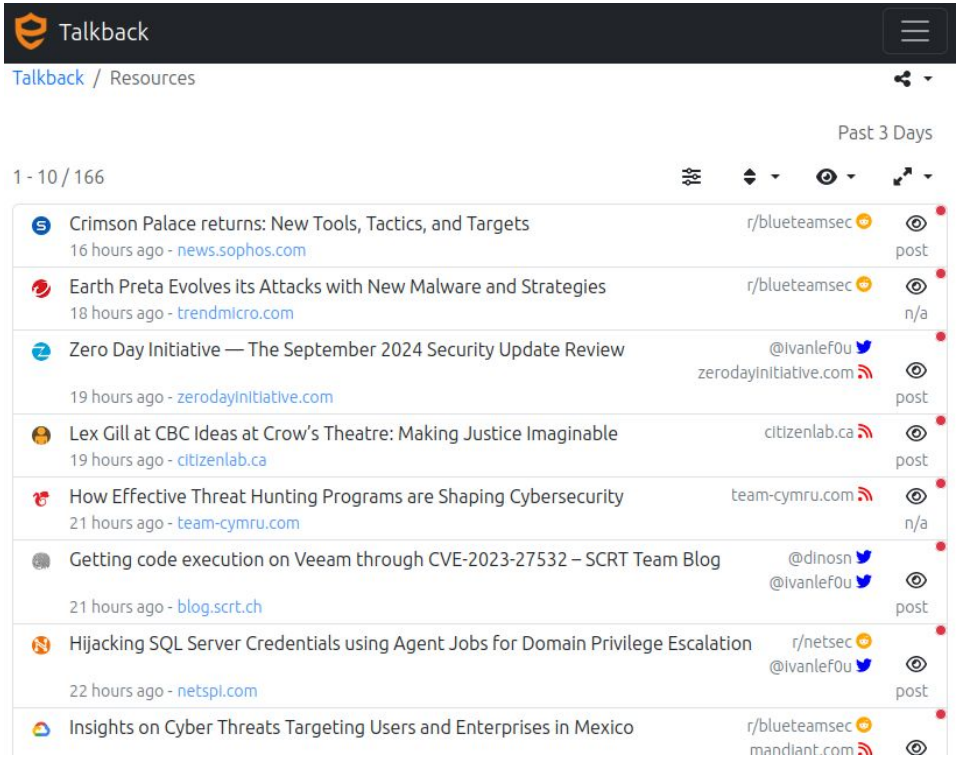
Hosting Information

blog.trailofbits.com

First resource	2012-10-29	**ISP**	Automattic, Inc
Last resource	2024-09-06	**Location**	United States

Resource Library - Pros

- Allows for keeping up with latest publications more efficiently
- Allows for integrations into our tools and processes
- Gets a lot of positive feedback from users and great brand awareness
- Provides us with a valuable data-set to work with



elttam

Resource classification and cross referencing

Categories

Reverse Engineering 90%

The process of analyzing software to understand how it functions, often involving tasks like debugging, disassembling, and decompiling to gain insights into the inner workings of a program.

Exploit Development 70%

The process of creating and refining exploits to take advantage of vulnerabilities in software or systems, often involving reverse engineering and understanding the target application's behavior.

Topics

Pwndbg plugin for GDB pwndbg

The Pwndbg plugin for GDB was extended by Jason An to integrate it with Binary Ninja, enhancing debugging capabilities for stripped binaries.

Debugger plugin 60%

Binary Ninja Binary Ninja

Binary Ninja's integration with Pwndbg allows for improved debugging intelligence by syncing symbols, function signatures, and stack variable offsets.

Binary analysis platform 40%

References

GitHub - pwndbg/pwndbg: Exploit Development and Reverse Engineering with GDB Made Easy rev exp

1 year ago - github.com/pwndbg oss

Tobasco VSCode Extension via API (internal)

audit (Workspace) [Dev Container: SecureBoot research: ESP32]

TOBASCO

Search

"esp-idf"

- GitHub - m3y54m/Embedded-Engineering-Roadmap: A comprehensive roadmap for aspiring Embedded Systems Engineers, featuring a curated list of learning resources.
  01/09/2024 - github.com/m3y54m
- Product Security Review Methodology for Traeger Grill Hack
  03/07/2024 - bishopfox.com
- Unveiling secrets of the ESP32: creating an open-source MAC Layer
  06/05/2024 - zeus.ugent.be
- GDBFuzz - Fuzzing Embedded Systems Using Hardware Breakpoints
  08/04/2024 - kitploit.com
- GitHub - risinek/esp32-wifi-penetration-tool: Exploring possibilities of ESP32 platform to attack on nearby Wi-Fi networks.
  31/03/2024 - github.com/risinek
- Bluetooth Low Energy CTF - Write up
  30/03/2024 - blog.tclaverie.eu
- Say Friend and Enter: Digitally lockpicking an advanced smart lock (Part 1: functional analysis)
  21/02/2024 - alephsecurity.com
- Hacking a Smart Home Device
  05/02/2024 - jmswmr.com
- Breaking the Flash Encryption Feature of Espressif's Parts
  11/01/2024 - courk.cc
- Keeping up with the Pwnses
  10/01/2024 - elttam.com
- Fault Injection Attacks against the ESP32-C3 and ESP32-C6
  09/01/2024 - courk.cc

esp_image_format.c 1 | sdkconfig | Settings | bootloader_flash.c

ESP-IDF: Search Error Hint

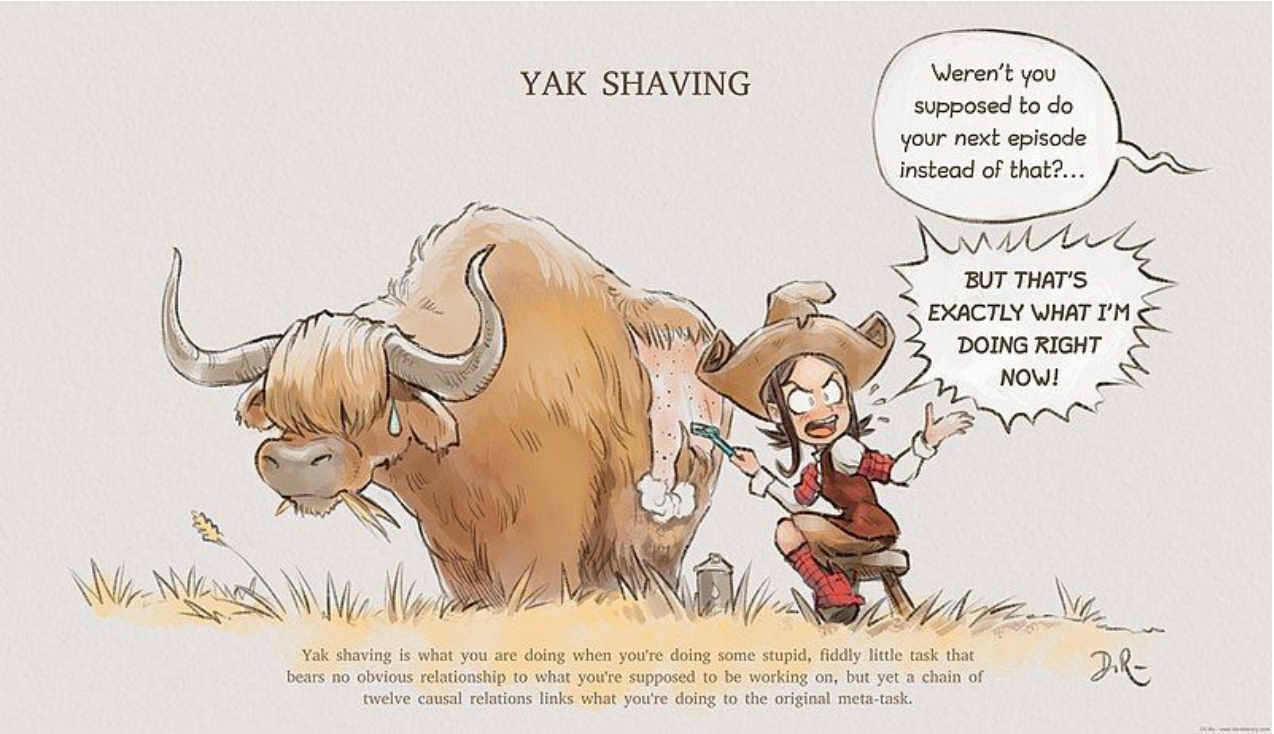
components > bootloader_support > src > esp_image_format.c > image_load

```c
/*
 * SPDX-FileCopyrightText: 2015-2024 Espressif Systems (Shanghai) CO LTD
 *
 * SPDX-License-Identifier: Apache-2.0
 */
#include <string.h>
#include <sys/param.h>
#include <esp_cpu.h>
#include <bootloader_utility.h>
#include <bootloader_signature.h>
#include <esp_secure_boot.h>
#include <esp_fault.h>
#include <esp_log.h>
#include <esp_attr.h>
#include <spi_flash_mmap.h>
#include <bootloader_flash_priv.h>
#include <bootloader_random.h>
#include <bootloader_sha.h>
#include "bootloader_util.h"
#include "bootloader_common.h"
#include "esp_rom_sys.h"
#include "esp_efuse.h"
#include "esp_app_desc.h"
#include "bootloader_memory_utils.h"
#include "soc/soc_caps.h"
#if CONFIG_IDF_TARGET_ESP32
#include "esp32/rom/secure_boot.h"
#elif CONFIG_IDF_TARGET_ESP32S2
#include "esp32s2/rom/secure_boot.h"
#elif CONFIG_IDF_TARGET_ESP32S3
#include "esp32s3/rom/secure_boot.h"
#elif CONFIG_IDF_TARGET_ESP32C3
```

Dev Container: SecureBoot research: ESP3... | ESP-IDF v4.x | /dev/ttyUSB0 | esp32s2 | 0 1 | 0 | clangd: idle | [ESP-IDF: OpenOCD Server] | Ln 159, Col 1 | Spaces: 4 | UTF-8 | LF | C | Search Resources

elttam

Resource Library - Cons



elttam

Wrapping up

Conclusions

- **Being adaptable is critical:** The field is fast moving and maturing steadily and it's key for teams to keep on their toes.
- **Engineering comes with risks:** All eng. investments requires due diligence and periodic review.
- **Retrospectives are key:** Having light retrospective processes help to constantly refine, challenge, and improve things.
- **Context switching is lethal:** Identifying and improving ways to give a consistent experience helps productivity.
- **Bespoke tools have a place:** In moderation, your own tooling creates fun challenges and opportunities for a team and also a sense of pride.
- **Consider how to give back:** Open source tools give back to the community and help others get to know about you.

elttam

References

Elttam Resources

- Talkback: https://talkback.sh
- semgrep-rules: github.com/elttam/semgrep-rules
- Blog: elttam.com/blog

Supporting tools:

- Asciidoctor: https://asciidoctor.org/
- Source Insight: https://www.sourceinsight.com

Other Resources:

- WeAudit: https://marketplace.visualstudio.com/items?itemName=trailofbits.weaudit
- Sarif Explorer: https://marketplace.visualstudio.com/items?itemName=trailofbits.sarif-explorer
- Security Notes: https://marketplace.visualstudio.com/items?itemName=refactor-security.security-notes
- GitLens: https://www.gitkraken.com/gitlens

elttam

Thank you

Any questions?

elttam