

# AI Enhanced Reviews for Linux Networking

Presented at Linux NetDEV Conference 0x18 (2024)

Jesse Brandeburg - Principal Software Engineer, Intel NEX  
Kamel Ayari - Principal Engineer, Intel NEX

# Agenda

Introduction

Problem Statement

Proposed Solution

Solution Benefits

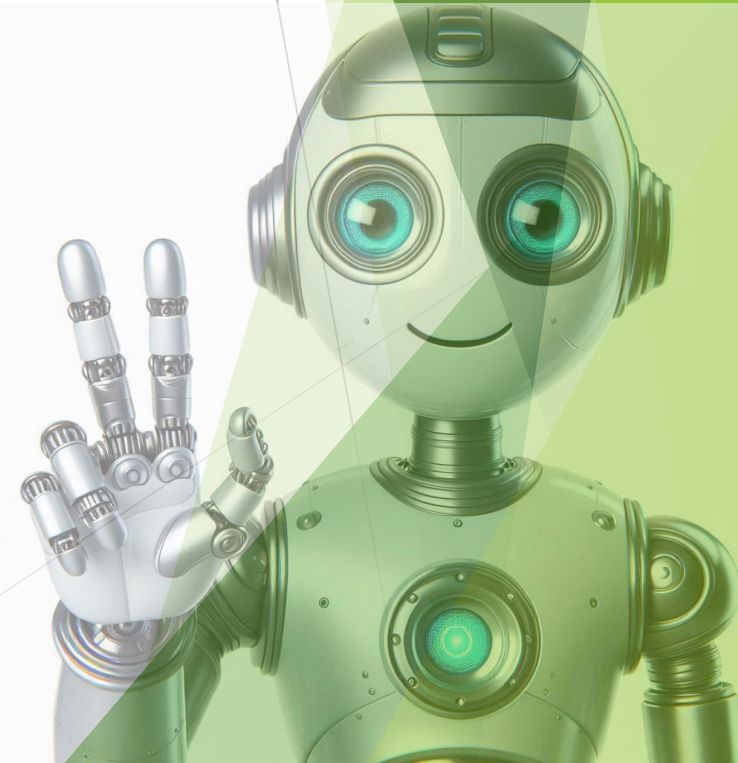
Experimental Results

Potential Future Work

# Who are these people?

- ▶ Jesse Brandeburg, 31 years at Intel
- ▶ Job Focus
  - ▶ Enable customer vision using open-source code
- ▶ Passions
  - ▶ Linux evangelism
  - ▶ Developing technical staff
  - ▶ Performance enhancements
- ▶ Fun
  - ▶ Drive cars/karts
  - ▶ Fix said cars
  - ▶ Raise two girls

- ▶ Kamel Ayari
- ▶ Job focus
  - ▶ Enhancing software quality through innovative technologies and AI
- ▶ Passions
  - ▶ streamlining SDLC through efficient scalable methods to produce high-quality software
- ▶ Fun
  - ▶ Aquatic sports and diving with sharks





Why use AI  
to review?



# Problem Statement



Open-source projects rely on community contributions



High volume of patches overwhelms reviewers



Reviewer's time wasted on minor issues (e.g., whitespace, comment formats)



Legacy tools (e.g., checkpatch) lack adaptability and context understanding

# Proposed Solution

- ▶ Provide a Large Language Model (we used ChatGPT 4o) the rules for netdev emails containing patches
- ▶ Provide the user a way of applying the “rules” to a particular email and generate an email response that can be reviewed before sending
- ▶ User can then review code at deeper level, ignoring the simple rule violations that were already commented upon



# Solution Benefits



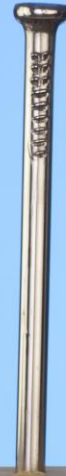
Increases developer efficiency



Reduce tedious work (reviewers)



Standardized review results in increased consistency for submitters





# Experimental Results

Lets look at some comparisons between human reviews and AI reviews



# AI vs Legacy Automation

'Understands' the commit guidelines

\* [PATCH resubmit 3] net: fec: Fix FEC\_ECR\_EN1588 being cleared on link-down



human-like error feedback vs. rigid automation messages.

## 1. Subject Line Issues:

- The subject line includes "resubmit 3", which is not a standard prefix.



[PATCH net-next] net: fec: Fix FEC\_ECR\_EN1588 being cleared on link-down

Able to suggest resolution

This place is fetching the stats, so u64\_stats\_fetch\_begin and u64\_stats\_fetch\_retry should be used

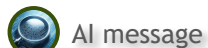


Helps enhance the message quality

- The explanation is too brief and lacks details on the why and how of the change.
- The commit message should provide a self-contained, technical description of the change.
- The text before the diff should wrap at 72 characters.
- The commit message should use the imperative verb mood.



Ensure proper synchronization when fetching statistics by using u64\_stats\_fetch\_begin and u64\_stats\_fetch\_retry. Replace the usage of u64\_stats\_update\_begin and u64\_stats\_update\_end to avoid potential race conditions.



AI message



Author Message

# AI Support to Human Review



Consistently catches trivial errors without fatigue

- The prefix `[PATCH]` should include the target tree, e.g., `[PATCH net]`.
- There is a typo in "Becausetxctrl->pi" which should be "Because txctrl->pi".

Enabled Human reviewer to focus on content and context

'index' is u32, the first condition is not needed please use netdev\_err() instead.

2024-06-16 20:01

[PATCH] net: dwc-xlgmac: fix missing MODULE\_DESCRIPTION() warning

Available within minutes

- It does not mention the target tree.
- The "Fixes:" tag is missing.


2024-06-17 11:04

Looks okay. Missing "Fixes" tag though. Please add it and send v2. Also, please make obvious what tree you target using "[PATCH net]" prefix.

Human Review

Even when review is redundant, the faster AI Feedback (CI Speed) saves manual review effort



 AI message

 Author Message

 Human Reviewer message

# Review Comments: AI vs Human



## AI Comments:

Readability, formatting, and clarity

Adherence to guidelines and conventions

Provide detailed feedback and suggestions

Weak (hallucinations?) when more context is required



## Humans Comments:

More extensive coverage on technical specifics and potential issues

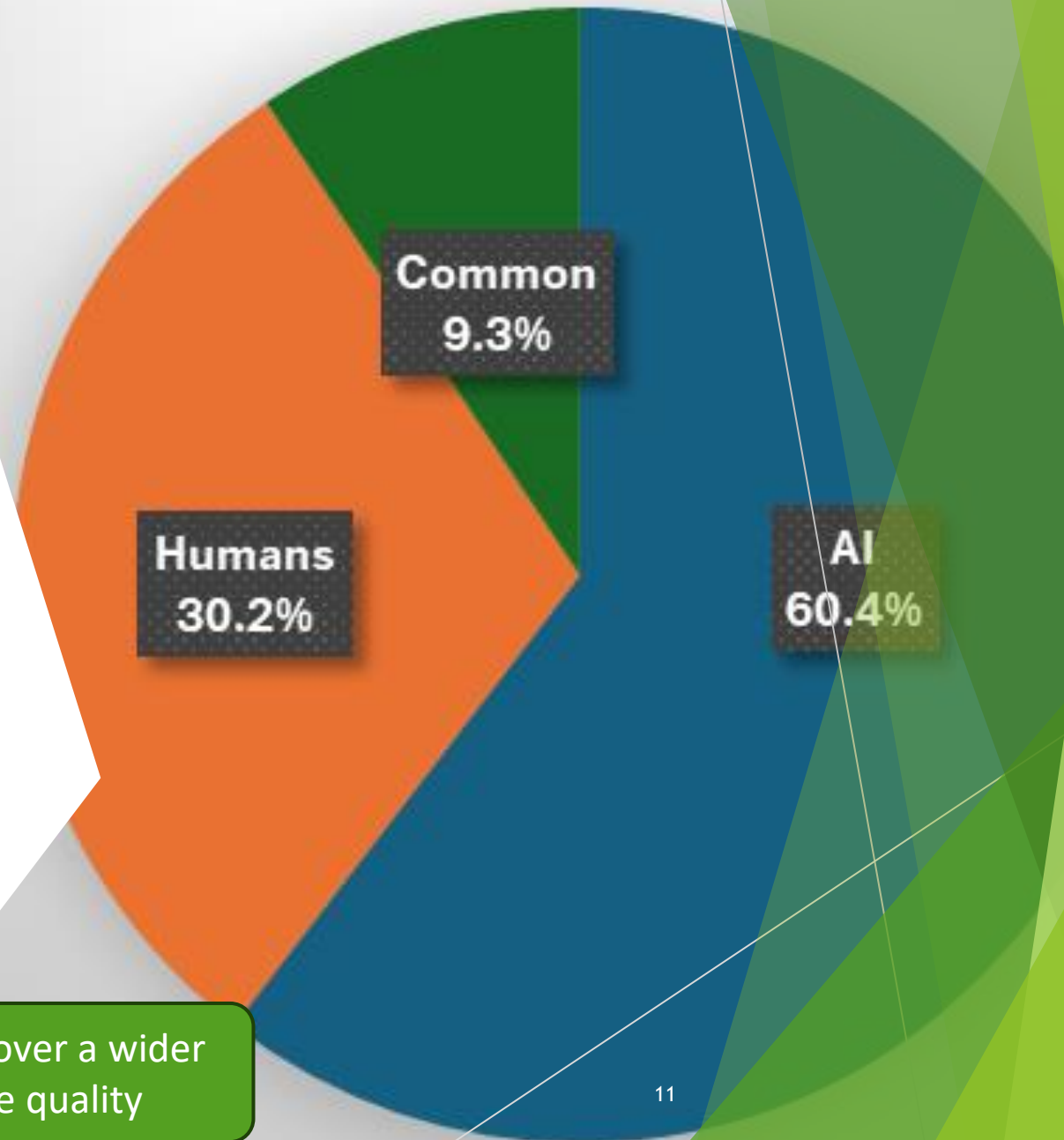
Detailed analysis of code functionality and impact

**Excellent at providing context aware feedback**



## Common Comments:

Guidelines and message clarity



Using both human and AI reviewers can cover a wider range of issues, enhancing overall code quality

# Potential Future Work

- ▶ Open Source “free” LLM licenses for mailing lists
- ▶ Include this in a tool like [b4](#)
- ▶ Make the process into a python library
- ▶ Code review based on context
- ▶ (Github actions style) Patch and commit message editing with suggestions
- ▶ CI that applies the patches and proposed changes from the LLM?
- ▶ If the community likes it, fully automated replies or inclusion into the zero-day bot



# Conclusion



AI-enhanced reviews improve efficiency, adaptability, and consistency



Future work aims to integrate and enhance the process



AI complements human reviewers by handling repetitive tasks

Jesse Brandeburg <jesse.brandeburg@intel.com>

Kamel Ayari <kamel.ayari@intel.com>

[www.intel.com/go/ethernet](http://www.intel.com/go/ethernet)

Thank you

# FAQ

- ▶ Is this a good idea?
  - ▶ Yes
- ▶ Can we start small?
  - ▶ Yes
- ▶ Don't LLMs hallucinate?
  - ▶ Yes, they do sometimes, but we haven't seen much evidence of this in the context of reviewing commit messages and simple patch when providing tight rule sets and doing natural language analysis

# FAQ - more

- ▶ Will this replace me?
  - ▶ No, the idea here is simply add a new tool for the developer
  - ▶ AI is good at repetitive tasks and doesn't get tired
  - ▶ Let people provide the curiosity, inventiveness, and instant adaptation



Questions	Answers
What is the problem addressed in the document?	The document addresses the issue of the volume of patches in open-source projects overwhelming reviewers, causing them to spend considerable time on mundane tasks such as checking adherence to guidelines and conventions.
What are the limitations of legacy automation tools?	Legacy automation tools lack the flexibility to adapt to new practices or understand the context. They typically concentrate on syntactic correctness but fall short when it comes to semantic subtleties and are unable to provide meaningful feedback.
What is the proposed solution to the problem?	The proposed solution is to leverage the advancement in generative Artificial Intelligence and Large Language Models (LLM) to provide an AI-based patch review. The solution will ingest patch emails, then submit the raw email as context attached to the review request prompt.
What are the benefits of the proposed solution?	The benefits include increased efficiency, adaptability, semantic understanding, faster review process, scalability, and consistency.
Will the AI-based approach replace human reviews?	No, the AI-based approach will not replace human reviews. It will, however, significantly reduce the time spent by human reviewers on mundane comments.
What are the future plans for this approach?	Future plans include integrating this AI-based review process into the zero-day bot, or enabling direct replies on the mailing list (with maintainers' permission) to enhance its effectiveness.
Who originated the idea of applying AI to streamline repetitive code review tasks?	The idea of applying AI to streamline repetitive code review tasks originated with Jesse Brandeburg. Kamel Ayari then transformed this concept into a practical set of rules for an AI code reviewer.
What is the role of the AI assistant?	The AI assistant is designed to meticulously review developer-submitted messages for Linux patch submissions. It evaluates messages against specific rules and provides feedback to the author about the message's quality.

# Proposed Solution



Leverage generative AI and Large Language Models (LLMs).



AI-based patch review ingests patch emails.



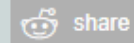
LLM generates review comments as email replies.



Focus on commit message rules and mailing list best practices.

# LKML: Tao Chen: [PATCH] samples: bpf: Add valid info for VMLINUX\_BTF

From Tao Chen <>  
Subject [PATCH] samples: bpf: Add valid info for VMLINUX\_BTF  
Date Mon, 29 Apr 2024 00:10:32 +0800



When i use the command 'make M=samples/bpf' to compile samples/bpf code in ubuntu 22.04, the error info occurred:  
Cannot find a vmlinux for VMLINUX\_BTF at any of " /home/ubuntu/code/linux/vmlinux", build the kernel or set VMLINUX\_BTF or VMLINUX\_H variable

Others often encounter this kind of issue, new kernel has the vmlinux, so we can set the path in error info which seems more intuitive, like:  
Cannot find a vmlinux for VMLINUX\_BTF at any of " /home/ubuntu/code/linux/vmlinux", build the kernel or set VMLINUX\_BTF like "VMLINUX\_BTF=/sys/kernel/btf/vmlinux" or VMLINUX\_H variable

Signed-off-by: Tao Chen <chen.dylane@gmail.com>

---

```
samples/bpf/Makefile | 2 +-  
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
diff --git a/samples/bpf/Makefile b/samples/bpf/Makefile  
index 4ccf4236031c..6f9e9345eb6a 100644
```

```
--- a/samples/bpf/Makefile
```

```
+++ b/samples/bpf/Makefile
```

```
@@ -326,7 +326,7 @@ $(obj)/vmlinux.h: $(VMLINUX_BTF) $(BPFTOOL)
```

```
ifeq ($(VMLINUX_H),)
```

```
ifeq ($(VMLINUX_BTF),)
```

```
$(error Cannot find a vmlinux for VMLINUX_BTF at any of "$(VMLINUX_BTF_PATHS)",\  
build the kernel or set VMLINUX_BTF or VMLINUX_H variable)
```

```
- build the kernel or set VMLINUX_BTF or VMLINUX_H variable)
```

```
+ build the kernel or set VMLINUX_BTF like "VMLINUX_BTF=/sys/kernel/btf/vmlinux" or VMLINUX_H variable)
```

```
endif
```

```
$(Q)$(BPFTOOL) btf dump file $(VMLINUX_BTF) format c > $@
```

```
else
```

```
--
```

```
2.34.1
```



Review Link

[PATCH] hippi: fix possible buffer overflow caused by bad DMA value in rr\_start\_xmit()

## Manual Review

- ▶ “LGTM”

## AI Review

▶ <https://chatgpt.com/share/f5a21fd4-ef5c-4570-958c-f2d9500d2c12>

▶ **Subject Line:**

- ▶ The subject line exceeds the 50-character limit.
- ▶ It does not start with the appropriate prefix indicating the subsystem or file modified.

▶ **Commit Message:**

- ▶ Minor grammatical error: "Becausetxctrl->pi" should be "Because txctrl->pi".



[PATCH net v2] ionic: fix use after netif\_napi\_del()

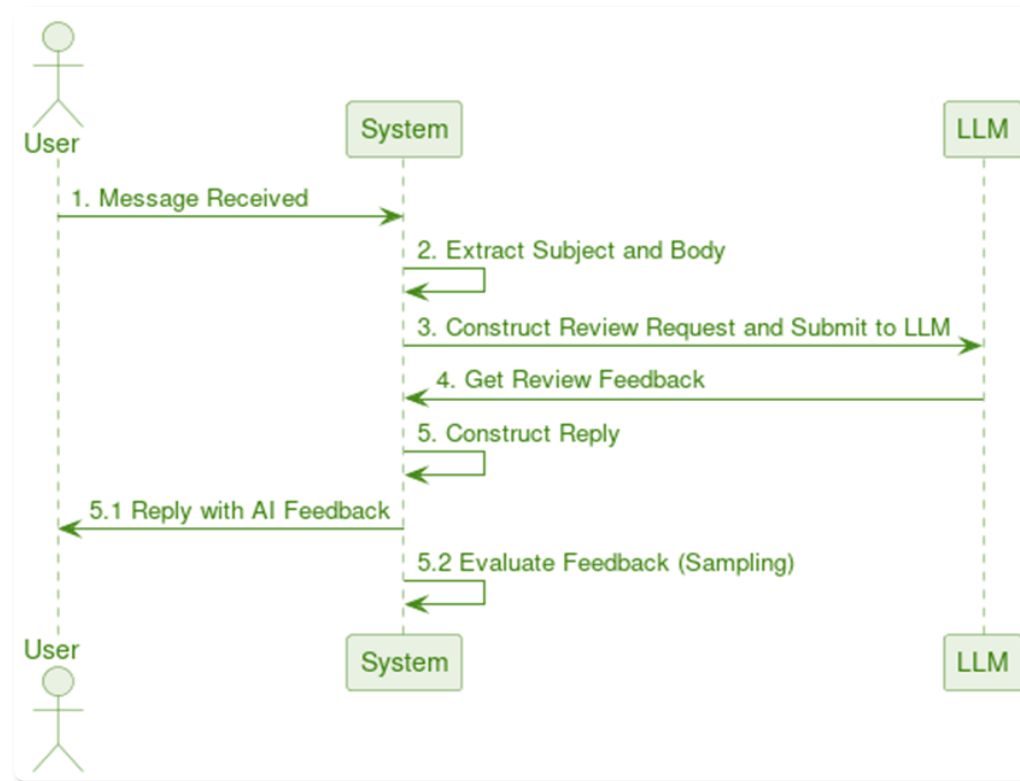
## Manual Review

- ▶ R1: “LGTM. Thanks for finding/fixing this!”
- ▶ R2: “Yes, thanks.”

## AI Review

- ▶ <https://chatgpt.com/share/f90568b0-2aa8-4db8-abbb-61bd5aac44c1>

# AI Review Pipeline



# AI Supporting Human Review

Flags suspected code errors

```
@_ 000,0 1000,7 @_ enam {  
    RELEASE_ALL_PAGES_MASK = 0x4000,  
};  
  
+#define MAX_RECLAIM_NPAGES -50000  
static int req_pages_handler(struct notifier_block *nb,  
                             unsigned long type, void *data)  
{
```



### 3. Technical Issues in Code:

- The defined macro `MAX_RECLAIM_NPAGES` should not have a negative value. It should be `#define MAX_RECLAIM_NPAGES 50000`.

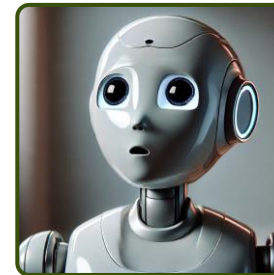


Can return false positives as well

**[PATCH] hippo: fix possible buffer overflow caused by bad DMA value in rr\_start\_xmit()**



- ✗ The subject line should not end with a period.
- ✓ The subject line does not start with a capital letter after the prefix.
- ✓ The prefix `[PATCH]` should include the target tree, e.g., `[PATCH net]`.



AI message



Author Message