YLChen-007 / **ISSUE-Github-REPORT-Moderation-Filter-Bypass.md** Secret

Last active last month

Code Revisions 2

Content Moderation Bypass in Arena Side-by-Side Views (Incomplete Fix for 34eca62)

ISSUE-Github-REPORT-Moderation-Filter-Bypass.md

Advisory Details

Title: Content Moderation Bypass in Arena Side-by-Side Views (Incomplete Fix for 34eca62)

Description:

Summary

In FastChat's Chatbot Arena side-by-side mode, the content moderation filter fails to properly check conversation history before each turn. In two files, only Model A's history is checked due to a wrong array index (`states[0]` instead of `states[1]`). In a third file — the Vision Anonymous Arena — `moderate_input()` receives only the current user input instead of the conversation history, making **all prior conversation content completely invisible to moderation**.

The root cause was fixed in commit 34eca62 for `gradio_block_arena_named.py`, but three other files were missed — two with the same wrong-index bug and one with a more severe missing-history variant.

Details

Commit `34eca62` fixed the moderation text construction in `gradio_block_arena_named.py` by correcting the array index for the right-side model's conversation state:

```
# gradio_block_arena_named.py - FIXED in 34eca62:
all_conv_text_left = states[0].conv.get_prompt()
```

```
all_conv_text_right = states[1].conv.get_prompt()  # Correctly reads Model B
```

However, the same copy-paste bug remains unfixed in two other arena modules, and a third module has an even more severe variant. Two files use `states[0]` for the right-side model, meaning `states[1]` (Model B) is never read. The third file skips conversation history entirely.

Occurrence 1 — `gradio_block_arena_anony.py` `add_text()` (line 310)

```
# Lines 309-314
model_list = [states[i].model_name for i in range(num_sides)]
# turn on moderation in battle mode
all_conv_text_left = states[0].conv.get_prompt()
all_conv_text_right = states[0].conv.get_prompt()  # ← BUG: should be states[
all_conv_text = (
    all_conv_text_left[-1000:] + all_conv_text_right[-1000:] + "\nuser: " + t
)
flagged = moderation_filter(all_conv_text, model_list, do_moderation=True)
```

This is the **Anonymous Arena** mode — the most popular mode on lmarena.ai. Note that `do_moderation=True` is explicitly hardcoded here, meaning moderation is **always active** regardless of model type. This makes this variant more impactful than the original patched bug in the named arena, where moderation only triggers for specific model keywords (claude, gpt, gemini, etc.).

Occurrence 2 — `gradio_block_arena_vision_named.py` `add_text()` (line 245)

```
# Lines 244-248
all_conv_text_left = states[0].conv.get_prompt()
all_conv_text_right = states[0].conv.get_prompt()  # ← BUG: should be states[
all_conv_text = (
    all_conv_text_left[-1000:] + all_conv_text_right[-1000:] + "\nuser: " + t
)
```

This is the **Vision Named Arena** mode that supports image uploads. The constructed `all_conv_text` is then passed to `moderate_input()` which calls `moderation_filter()` — again with Model B's history completely missing.

Occurrence 3 — `gradio_block_arena_vision_anony.py` `add_text()` (line 314-315) [MOST SEVERE]

```
# Lines 310-316
model_list = [states[i].model_name for i in range(num_sides)]
```

```
images = convert_images_to_conversation_format(images)

text, image_flagged, csam_flag = moderate_input(
    state0, text, text, model_list, images, ip
)
#              ^^^^  ^^^^
# 2nd param: text (current user input)
# 3rd param: text (current user input) — should be all_conv_text!
```

This is the **Vision Anonymous Arena** mode. Unlike the other two occurrences which at least build `all_conv_text` (albeit incorrectly), this file **never constructs `all_conv_text` at all**. The variable doesn't exist in this function — `grep 'all_conv_text' fastchat/serve/gradio_block_arena_vision_anony.py` returns nothing.

The `moderate_input()` function signature is `moderate_input(state, text, all_conv_text, model_list, images, ip)`. Its 3rd parameter `all_conv_text` is supposed to contain the full conversation history for moderation. But here, the caller passes `text` (just the current user message, e.g. "tell me more") as the 3rd parameter, so `moderation_filter()` only sees the current short input — **neither Model A's nor Model B's conversation history is checked**.

Compare with the correct single-model implementation in `gradio_block_arena_vision.py:254-260`:

```
# CORRECT (single-model reference):
all_conv_text = state.conv.get_prompt()
all_conv_text = all_conv_text[-2000:] + "\nuser: " + text
moderate_input(state, text, all_conv_text, ...)  # ← history included
```

Why this matters:

In side-by-side arena mode, users chat with two models simultaneously. The moderation filter is supposed to check the full conversation context from **both** models before allowing each new turn.

- Occurrences 1 & 2: `states[0]` is read twice and `states[1]` is never read — Model B's history is missing
- Occurrence 3: Neither `states[0]` nor `states[1]` history is read — **ALL conversation history is missing**

The fixes:

```
# Fix for occurrences 1 & 2: change states[0] to states[1] for the right-side
all_conv_text_right = states[1].conv.get_prompt()

# Fix for occurrence 3: add the missing all_conv_text construction and pass i
all_conv_text_left = states[0].conv.get_prompt()
all_conv_text_right = states[1].conv.get_prompt()
all_conv_text = (
    all_conv_text_left[-1000:] + all_conv_text_right[-1000:] + "\nuser: " + t
)
moderate_input(state0, text, all_conv_text, model_list, images, ip)
```

PoC

This vulnerability was identified through code audit by comparing the fixed file (`gradio_block_arena_named.py`, commit `34eca62`) against its unpatched counterparts. The bug is directly observable in the source code:

```
# Show the VULNERABLE lines — both use states[0]:
$ grep -n "all_conv_text_right = states\[0\]" fastchat/serve/gradio_block_are
310:    all_conv_text_right = states[0].conv.get_prompt()

$ grep -n "all_conv_text_right = states\[0\]" fastchat/serve/gradio_block_are
245:    all_conv_text_right = states[0].conv.get_prompt()

# Show the FIXED line — correctly uses states[1]:
$ grep -n "all_conv_text_right = states\[1\]" fastchat/serve/gradio_block_are
182:    all_conv_text_right = states[1].conv.get_prompt()
```

Three files, same function (`add_text()`), same code block. One was fixed, two were missed. The pattern is clear: `all_conv_text_right` should reference `states[1]` (Model B), not `states[0]` (Model A).

Multi-turn attack scenario:

1. User opens the Anonymous Arena (Battle) tab
2. User sends a message — two anonymous models are assigned and both respond
3. Model B generates a response containing content above the moderation threshold (e.g., sexual content)
4. User sends a follow-up message
5. The moderation filter constructs `all_conv_text` from `states[0]` + `states[0]` — Model B's violating response in `states[1]` is never included

6. `moderation_filter()` does not flag the conversation, and the violating content persists

Impact

This is a **content moderation bypass** with varying severity across the three occurrences:

- **Occurrences 1 & 2** (wrong index): Model B's conversation history is never moderated. Model A's history is checked (duplicated). New user input is still moderated.
- **Occurrence 3** (missing history): **ALL** conversation history from **both** models is invisible to moderation. Only the current short user input (e.g., "tell me more") is checked. This is the most severe variant.

The Vision Anonymous Arena variant (Occurrence 3) is particularly concerning because:

- **Complete bypass**: In multi-turn conversations, no prior model response is ever checked
- A user can send an innocuous follow-up ("continue", "tell me more") and the moderation system will only see that short string
- Both models' potentially violating responses persist un-moderated indefinitely

The Anonymous Arena variant (Occurrence 1) is also critical because:

- It is the primary mode used on the public lmarena.ai platform
- Moderation is **always-on** (`do_moderation=True`), yet partially ineffective
- Users have no control over which models are assigned

Affected products

- **Ecosystem**: pip
- **Package name**: fschat
- **Affected versions**: <= 0.2.36 (latest at time of report, commit `587d5cf`)
- **Patched versions**:

Severity

- **Severity**: Medium-High
- **Vector string**: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

Weaknesses

- **CWE**: CWE-670: Always-Incorrect Control Flow Implementation

Occurrences

Permalink
https://github.com/lm-sys/FastChat/blob/587d5cfa1609a43d192cedb8441cac3c17db105d/fastchat/serve/gradic
https://github.com/lm-sys/FastChat/blob/587d5cfa1609a43d192cedb8441cac3c17db105d/fastchat/serve/gradic L248
https://github.com/lm-sys/FastChat/blob/587d5cfa1609a43d192cedb8441cac3c17db105d/fastchat/serve/gradic L316
https://github.com/lm-sys/FastChat/blob/587d5cfa1609a43d192cedb8441cac3c17db105d/fastchat/serve/gradic