```
/*
1、这部分内容是注释内容
2、从实用的角度来说这个文件目前作为一种指南，因为我目前的需求场景还没有涉及到这种级别的安全
3、这里提到的一些关于安全的建议却是可以进行借鉴吸收的
*/
```

Security Policy(安全政策)

在计算机系统领域中，“Security Policy”通常被翻译为“安全策略”。这个翻译比较准确地反映了其在信息安全管理中的作用和意义。

解释：
- **安全策略**（Security Policy）是指定和维护一个组织内部用以保护其信息和技术资产的一系列准则和规定。
- 这些策略为组织的信息安全活动提供指导，确保信息系统的安全性、完整性和可用性。

安全策略涵盖的内容可能包括：
1. **用户认证**：如何验证用户身份的政策，比如密码政策、多因素认证等。
2. **数据保护**：如何保护存储和传输中的数据不被非授权访问、修改或删除。
3. **访问控制**：谁可以访问哪些资源，以及在什么条件下可以访问。
4. **应急响应**：在信息安全事件发生时的应对措施和程序。
5. **网络安全**：如何保护网络不受攻击，如防火墙、入侵检测系统的配置。

通过这样的翻译和解释，可以确保对“Security Policy”在计算机和信息安全上下文中的理解更加深入和精确。

Using llama.cpp securely（安全的使用llama.cpp）

Untrusted models（不受信任的模型）

Be careful when running untrusted models. This classification includes models created by unknown developers or utilizing data obtained from unknown sources.（当运行不受信任的models的时候需要小心。此分类包括由未知开发人员创建的模型或利用从未知来源获得的数据的模型。）

Always execute untrusted models within a secure, isolated environment such as a sandbox (e.g., containers, virtual machines). This helps protect your system from potentially malicious code.

总是在一个安全的、隔离的环境例如sandbox中执行不受信任models（例如：容器、虚拟机）这将会帮助保护你的系统不会收到潜在的致命的危险代码。

> [!NOTE] The trustworthiness of a model is not binary. You must always determine the proper level of caution depending on the specific model and how it matches your use case and risk tolerance.（模型的可信度不是二元的。您必须始终根据具体模型以及它如何匹配您的使用情况和风险承受能力来确定适当的谨慎级别。）

Untrusted inputs（不受信任的输入）

Some models accept various input formats (text, images, audio, etc.). The libraries converting these inputs have varying security levels, so it's crucial to isolate the model and carefully pre-process inputs to mitigate script injection risks.（一些models接受不同的输入格式（文本、图片、语音等等）。转换这些输入的库具有不同的安全级别，因此隔离模型并仔细预处理输入以减轻脚本注入风险至关重要。）

For maximum security when handling untrusted inputs, you may need to employ the following（为了在处理不受信任的输入时获得最大的安全性，您可能需要采用以下方法）:

- Sandboxing: Isolate the environment where the inference happens.（sandboxing:当进行inference的时候隔离环境）
- Pre-analysis: Check how the model performs by default when exposed to prompt injection (e.g. using fuzzing for prompt injection). This will give you leads on how hard you will have to work on the next topics.（预分析：检查模型在暴露于即时注入时的默认表现，这将为你提供线索，让你知道你需要付出多大的努力来完成下一个主题）
- Updates: Keep both LLaMA C++ and your libraries updated with the latest security patches.（更新：使用最新的安全补丁更新 LLaMA C++ 和你的库）
- Input Sanitation: Before feeding data to the model, sanitize inputs rigorously. This involves techniques such as（输入净化：在将数据输入模型之前，严格净化输入。这涉及以下技术）:
    - Validation: Enforce strict rules on allowed characters and data types.（验证：对允许的字符和数据类型实施严格的规则。）
    - Filtering: Remove potentially malicious scripts or code fragments.（过滤：删除潜在的恶意脚本或代码片段）
    - Encoding: Convert special characters into safe representations.（编码：将特别的字符转化成安全的表达）
    - Verification: Run tooling that identifies potential script injections (e.g. models that detect prompt injection attempts).（验证：运行能够识别潜在脚本注入的工具）

Data privacy（数据隐私）

To protect sensitive data from potential leaks or unauthorized access, it is crucial to sandbox the model execution. This means running the model in a secure, isolated environment, which helps mitigate many attack vectors.

为了能够保护敏感数据，以免被潜在的泄漏或者不受信任的访问，将模型放在sandbox中运行是至关重要的。这意味着在一个安全的、隔离的环境中运行model，这种方式将有助于缓解许多攻击媒介。

Untrusted environments or networks（不受信任的环境或者网络）

If you can't run your models in a secure and isolated environment or if it must be exposed to an untrusted network, make sure to take the following security precautions（如果你不能在一个安全的、隔离的环境中运行模型，或者说这个模型必须暴露在一个不受信任的网络中，确保采取以下安全预防措施）:

- Confirm the hash of any downloaded artifact (e.g. pre-trained model weights) matches a known-good value（确认任何下载的工件（例如，预先训练的模型权重）的哈希值与已知的良好值匹配）
- Encrypt your data if sending it over the network.（如果model是通过网络，那么加密你的数据。）

Multi-Tenant environments（多租户环境）

If you intend to run multiple models in parallel with shared memory, it is your responsibility to ensure the models do not interact or access each other's data. The primary areas of concern are tenant isolation, resource allocation, model sharing and hardware attacks.（如果你想要在一个共享的内存中并行的运行多个models，确保模型之间不能交互或者访问彼此的数据是你的职责。主要关注的领域是租户隔离、资源分配、模型共享和硬件攻击。）

1. Tenant Isolation: Models should run separately with strong isolation methods to prevent unwanted data access. Separating networks is crucial for isolation, as it prevents unauthorized access to data or models and malicious users from sending graphs to execute under another tenant's identity.（租户隔离：模型应使用强大的隔离方法单独运行，以防止不必要的数据访问。分离网络对于隔离至关重要，因为它可以防止未经授权访问数据或模型，以及恶意用户发送图表以另一个租户的身份执行。）

2. Resource Allocation: A denial of service caused by one model can impact the overall system health. Implement safeguards like rate limits, access controls, and health monitoring.（资源分配：一个模型导致的拒绝服务可能会影响整个系统的健康。实施速率限制、访问控制和健康监控等保障措施。）

3. Model Sharing: In a multitenant model sharing design, tenants and users must understand the security risks of running code provided by others. Since there are no reliable methods to detect malicious models, sandboxing the model execution is the recommended approach to mitigate the risk.（模型共享：在多租户模型共享设计中，租户和用户必须了解运行他人提供的代码的安全风险。由于没有可靠的方法来检测恶意模型，因此建议使用沙盒化模型执行来降低风险。）

4. Hardware Attacks: GPUs or TPUs can also be attacked. Researches has shown that side channel attacks on GPUs are possible, which can make data leak from other models or processes running on the same system at the same time.（硬件攻击：GPU 或 TPU 也可能受到攻击。研究 表明，GPU 可能遭受侧信道攻击，这可能导致同一系统上同时运行的其他模型或进程的数据泄露。）

Reporting a vulnerability（报告漏洞）

Beware that none of the topics under Using llama.cpp securely are considered vulnerabilities of LLaMA C++.（请注意，安全地使用 llama.cpp 下的任何主题均不被视为 LLaMA C++ 的漏洞。）

However, If you have discovered a security vulnerability in this project, please report it privately. **Do not disclose it as a public issue.** This gives us time to work with you to fix the issue before public exposure, reducing the chance that the exploit will be used before a patch is released.（但是，如果你已经在这个项目中发现了一个安全漏洞，请私下报告，**不要作为一个公开问题暴露出来**，在将这个漏洞公开出来，这会给我们时间和你一起协作修复这个问题。在补丁发放之前，减少暴露的概率）

Please disclose it as a private security advisory.（请将其披露为私人）

A team of volunteers on a reasonable-effort basis maintains this project. As such, please give us at least 90 days to work on a fix before public exposure.（一个志愿者团队以合理的努力维护此项目。因此，请给我们至少 90 天的时间进行修复，然后再公开发布。）