Prompt of Discovering Bugs

Here are the permission descriptions for each state in the Base model and Divergent model.
[Base model state semantic]
[Divergent model state semantic]
Based on the analysis of the two models, the next task is: Analyze security risks (Base model vs. Divergent model). Attention: if Divergent model is not provided, just analyse Base model.
1. Model Understanding:
* Base model (S0-S[X]): Understand normal business logic (designer intent). Note: The Base model itself may contain undesirable behaviors.
* Divergent model (remaining states): Understand the attack paths an attacker attempts to construct. Note: Attackers can only perform one attack operation—currently [Hookable Action]. Focus exclusively on whether this operation introduces security risks.
2. Comparative Analysis:
* Compare the Divergent model with the Base model.
* Identify state transitions unique to the Divergent model. Not all unique transitions indicate issues.
* Evaluate whether operations and responses on unique transitions in the Divergent model pose security risks under the current state semantics (permission context), assessed from CIA and CWE vulnerability perspectives.
* Focus on User2's knowledge accumulation rule:
* Knowledge Accumulation Phase: When an attacker performs actions to acquire parameters while possessing legitimate permissions, this is normal behavior and not considered a vulnerability.
* Vulnerability Exploitation Phase: If an attacker is able to successfully execute the attack using the parameters from the historical records without proper authorization, it indicates the presence of a vulnerability.
* Pay special attention to information leakage:
* Principle 1 - Direct Leakage: Attackers must not obtain unauthorized user identity information or device attributes from responses.
* Principle 2 - Differential Inference: Attackers must not infer unauthorized system state changes or user/device states through response variations from the same operation across different states.
3. Key Considerations:
* An attack request might be rejected (error code returned) with no actual attack effect. If no information leakage occurs, there is no security risk.
* Permission Check Rules. When analyzing whether User2's actions pose security threats, follow these rules. Actions are not considered vulnerabilities if User2 has legitimate permissions:
1) Family Member Priority: If User2 is User1's family member, they automatically retain control rights to all User1's devices (including removed/re-added devices) without re-invitation. Permissions remain valid until revoked.
2) Direct Sharing Permission: If User2 is directly shared only specific devices, permissions are limited to those instances. Permissions expire upon device removal/re-addition and require re-sharing by User1.
3) Threat Determination: Actions are potential security threats only when User2 is neither a family member nor directly shared the device.
4) Permission timeliness: Permission validity follows current state semantics. If permissions are invalid per state description, subsequent operations are considered unauthorized.
* When analyzing information leakage:
1) Feedback about the attacker's own permission status is not leakage. Only exposure of other entities' sensitive information/states constitutes a threat.
2) For Principle 2 (Differential Inference), thoroughly analyze whether responses contain exploitable differences. Especially when the symbols (CLS_number) in response description (before <Operation result> and <Evidence>) are different, there is a high possibility of causing leakage. No differences/Same symbol → no leakage.
3) Avoid misclassifying harmless textual variations as information leakage.
* Double-check if issues are found:
1) No issue is normal: The Divergent model might have no problems; vulnerability hunting is not mandatory.
2) When user2 executes the attack operation in a scenario where it is legally possible to do so, it is regarded as the Knowledge Accumulation Phase, and there are no vulnerabilities. For example, User2 accepting User1's re-sharing request is not a vulnerability. Because at this state, user2 received the invitation and was able to accept it normally. Even though the attack operation that user2 attempted currently was to accept the invitation illegally.
Input Format: State transition dictionary - {"Current State": ["State Semantic Description", {"Target State": "Action/Response"}]}
Output Format:
Base model

* [If no issues:] No issues found.
* [If issues exist:]
* **Issue Description:** xxx
* **Problematic State(s):**
* ‘sx‘: Performed **xx action**, received **xx response**, transitioned to **State X**, causing **xx impact**.
Divergent model
* [If no issues:] No issues found.
* [If issues exist, report as vulnerabilities:]
* **Vulnerability 1: XXX** (Briefly describe the nature of the vulnerability)
* **Impact:** Specifically describe the harm caused
* **Problematic State(s):**
* ‘sx‘: Performed **xx action**, received **xx response**, transitioned to **State X**, causing **xx impact**.
Input: