ChatGPT

디지털자산 수탁 보안 설계 – 사례 기반 패턴 학습

본 문서는 백엔드 개발자 대상의 수탁형 지갑 설계 교육을 위한 배경 자료이다. 수탁형 지갑의 목표는 **"잘못된 송금이 구조적으로 불가능한 통제 설계"**를 구현하는 것이다. 키를 숨기는 것만으로는 충분하지 않으며, 정책(rule)과 절차가 지갑 시스템에 깊게 녹아 있어야 한다. 아래의 4개 패턴은 글로벌 수탁업체들의 설계에서 도출된 핵심 교훈으로, 각 패턴마다 실제 업체 사례→기술적 이유→설계 규칙을 정리했다. 마지막에는 발표자가 기억해야 할 **1페이지 보안통제 메모**를 제공한다.

1. 패턴 1 – "Allowlist + Hold: 실수와 침해를 구조적으로 느리게 만든다"

사례 및 근거

- **Coinbase** – 암호자산을 인출할 때 "주소록(allowlist)" 기능을 제공한다. 주소록에 없는 주소로는 출금할 수 없으며, 새 주소를 추가하면 **48 시간 보류(hold) 기간**이 적용된다. 이 보류기간 동안 주소는 미활성 상태로 남아 있어, 해커가 계정을 침해해도 즉시 송금할 수 없다 [1]. 이러한 주소 화이트리스트 + 보류 정책은 **인증되지 않은 주소로의 송금을 지연하여 인간의 복사-붙여넣기 오류나 악성 소프트웨어가 주소를 바꿔치기하는 사고를 완화**하고, 계정 침해가 발생해도 대응할 시간을 벌어 준다 [1].
- **다른 거래소**도 유사한 구조를 적용한다. 예를 들어 2025년 Aura 보안 블로그는 Coinbase 주소록 기능을 소개하면서 "허용 리스트에 주소를 추가하거나 제거하면 **48시간 지연**이 적용된다"고 언급하며, Coinbase "Vault" 계좌는 여러 승인자를 요구하고 출금 시 **48시간 지연**이 적용된다고 설명한다 [2]. Reddit 사용자들도 **허용리스트를 끄거나 변경하면 48시간 동안 거래가 지연**된다는 사실을 공유했다 [3].

왜 그렇게 했나? (기술적 이유)

- **실수와 침해를 느리게**: 주소를 미리 등록하도록 강제하고, 등록 후 일정 기간 동안 사용 불가능하게 만들어 사용자의 착오 및 악성 주소 대체 공격에 대비한다.
- **대응 시간 확보**: 계정이 탈취되더라도 48시간 내에 이상 징후(이메일 알림, SMS 등)를 확인하고 주소를 차단할 수 있다.
- **규정 준수 및 감사 용이성**: 주소 추가와 활성화 과정에서 승인·로그를 남기면 사후 감사와 규제 준수가 용이하다.

설계 규칙 (Best Practice)

1. **화이트리스트는 지연장치까지 포함해야 한다**. 단순히 승인된 주소 목록을 유지하는 것에 그치지 말고, **'등록 → 승인 → 지연(hold) → 활성화'**라는 상태 머신을 구현해라.
2. **사용자 경험과 보안을 균형 있게 설계**: 48시간 hold를 기본값으로 두되, 기업 고객의 경우 관리자 서명이나 다른 통제 하에 지연 시간을 조절할 수 있다.
3. **시스템 내 통제 모듈 배치**: 이 기능은 **정책 엔진**에서 규칙으로 관리하고, 주소 등록 승인은 별도의 **Approval** 모듈에서 수행한다. 모든 상태 변화는 **Ledger/Audit**에 기록해 추적 가능하도록 한다.
4. **알림 및 UI UX**: 새 주소 등록·활성화·거절 시 사용자에게 명확하게 안내하고, 의심스러운 변경 요청은 관리자에게 경보를 보낸다.

2. 패턴 2 – "정책 엔진 무결성: 정책을 바꾸는 것이 사고다"

사례 및 근거

- **Fireblocks** – 디지털 자산 인프라 기업인 Fireblocks는 모든 민감한 연산을 **하드웨어 격리 환경(TEE)**에서 실행한다. Fireblocks가 발표한 보안 백서에 따르면, **정책 엔진은 보안 격리 환경(예: Intel SGX) 안에서 실행되며 외부 공격자와 내부 악의적 직원이 정책 규칙을 수정하지 못하도록 한다** [4].
- Fireblocks는 정책 엔진을 **버전 관리되는 규칙 세트**로 취급하고, 변경 시 관리자 quorum 승인(메이커-체커)을 요구한다 [5]. 정책 변경 자체를 고위험 사건으로 분류하여 감사 로그를 남기며, 정책이 적용될 때뿐만 아니라 **"정책 변경 이벤트"**가 중요한 보안 포인트임을 강조한다 [5].
- Fireblocks의 다른 문건은 정책 엔진과 키 쉐어를 모두 **Intel SGX 엔클레이브** 안에서 운영해, 해커나 내부자가 역할/규칙을 변경하지 못하도록 한다고 설명한다 [6].

왜 그렇게 했나? (기술적 이유)

- **정책 변경은 곧 사고**: 주소당 제한, 금액 한도 등 policy rule 자체가 자산 보호의 마지막 방어선이다. 정책을 변경하면 즉시 다른 규칙이 적용되므로, 공격자는 정책을 바꿔 출금을 쉽게 만들려 한다.
- **코드와 데이터의 무결성 보호**: 정책 엔진을 **TEE (신뢰 실행 환경)** 안에 넣으면, 엔진 자체와 규칙 파일이 암호화된 메모리에서만 실행되고, 승인된 관리자 서명 없이는 수정할 수 없다.
- **감사 가능성**: 정책 변경 이벤트는 별도로 기록되고, 변경 전후 버전을 비교할 수 있어 감사인이 위험 변경을 추적할 수 있다.
- **쓸모 있는 관제 포인트**: 많은 침해는 정책을 무력화하여 송금 제한을 풀어버리는 것으로 시작한다. 정책 변경을 감지하면 즉시 경보를 발생시켜 사고를 예방한다.

설계 규칙 (Best Practice)

1. **정책 = 코드 if문이 아니라 버전 관리되는 규칙 세트**: 정책 엔진은 환경설정 파일처럼 저장되고 버전 관리가 돼야 한다. 변경 전후 diff 비교가 가능해야 하며, 롤백도 지원해야 한다.
2. **정책 변경은 별도 승인 프로세스**: 트랜잭션 승인과 분리하여, 정책 변경은 고위험 작업으로 처리한다. 관리자 quorum 승인, 다중 서명 또는 이중 확인을 요구한다 [5].
3. **정책 엔진을 신뢰 경계로 승격**: SGX 등 TEE나 클라우드 HSM과 결합해 정책 엔진을 하드웨어 수준에서 보호한다 [4].
4. **모니터링 및 감사**: 모든 정책 변경 이벤트를 **Ledger/Audit**에 기록하고, 변경 요청 → 승인 → 지연 → 적용의 상태 머신을 유지한다.

3. 패턴 3 – "Signer 경계 안으로 정책을 넣는다" (policy-in-signer)

사례 및 근거

- **Anchorage Digital** – Anchorage는 FTX 사태 이후 수탁 위험을 완화하기 위한 글에서 HSM 모델을 설명한다. Anchorage는 **개인 키를 에어갭된 HSM 안에 보관하면서 동시에 HSM 내부에 정책 엔진을 구현하여 서명 지시를 검증**한다고 밝힌다. 즉, **트랜잭션 생성과 서명이 같은 시스템에서 수행되며, 서명 시에 정책을 다시 확인한다** [7].
- Anchorage의 기술 문서는 HSM 내부에 **펌웨어 정책 엔진**을 내장해, 각 거래 지시를 조직의 정책에 따라 독립적으로 검증한다고 설명한다. HSM은 키와 정책을 모두 보호하며, **Anchor Digital만의 설계는 정책을 검증할 때 "서명해야 하는지 여부뿐 아니라 '무엇을 서명하는지'도 확인**한다** [8] [9].
- Anchorage는 이 구조를 통해 **키·서명·정책을 동일한 경계 안에 두고**, 오케스트레이션 API가 허용했다는 이유만으로 서명을 수행하지 않는다. 각 HSM이 독립적으로 검증하는 구조는 **무단 변경이나 API 우회 공격**을 차단한다 [10].

왜 그렇게 했나? (기술적 이유)

- **API 우회 공격 대응**: API 계층이나 오케스트레이터가 해킹되면 거래가 허용될 수 있다. 서명 직전에 다시 정책을 검증하면, API 가 허락해도 HSM이 거부할 수 있다.
- **최종 경계에서의 통제**: 서명은 자산 이동의 마지막 단계이므로, 여기서 정책을 체크하면 실패한 검증을 즉시 차단할 수 있다.
- **추적 가능성**: HSM 안에서 정책 검증과 서명 과정의 로그를 남기면, 각 거래가 정책을 만족했는지 명확하게 증명할 수 있다.
- **단일 실패점 제거**: 정책 엔진과 서명 기능이 동일한 HSM 환경에 있으므로, 중간 시스템이 악용돼도 HSM이 마지막 방어선을 제공한다.

설계 규칙 (Best Practice)

1. **서명 직전에 정책을 재검증**: 정책 엔진은 오케스트레이션 계층뿐 아니라 **Signer (HSM 또는 MPC 서명 모듈)** 안에서도 정책을 검증해야 한다. API가 거래를 허용해도 HSM이 다시 체크한 후 서명한다.
2. **정책-서명 통합 엔진 사용**: 가능하다면 HSM 또는 SGX 내에 정책 엔진을 내장해, 키와 정책을 동일 경계에서 관리한다 [7].
3. **중간 계층 신뢰 최소화**: 오케스트레이터는 단순한 전송·알림 역할만 수행하며, 최종 서명 권한은 Signer 경계에 둔다.
4. **로그와 감사**: 서명 모듈은 정책 검증 결과와 서명 이벤트를 기록하여 사후 분석이 가능하도록 한다.

4. 패턴 4 – “SoD (분리 적합)를 키 구조로 강제한다”

사례 및 근거

- **BitGo** – BitGo는 모든 온체인 지갑에서 **3개의 키(사용자 key, 백업 key, BitGo key)**를 사용한다. 개발자 문서는 “BitGo Bank & Trust는 이 키들을 서로 다른 소유주에게 분산해 **어떤 사람도 하나 이상의 키를 제어할 수 없게 한다**”고 설명한다 [11]. 또 BitGo의 멀티시그 또는 MPC 지갑은 **항상 2-of-3 키로 서명**하며, 멀티시그 구조에서 **사용자와 BitGo가 함께 서명해 출금이 완성**된다고 명시한다 [12].
- BitGo의 2025년 블로그는 “**2-of-3 구성에서 세 명의 키 소유자 중 두 명이 승인해야만 거래가 실행되며, 한 사람이 자금을 이동시킬 수 없다**”는 점을 강조한다 [13]. 이 구조는 내부자 사고나 키 분실에도 자금이 보호되며, 더 큰 조직은 3-of-5나 4-of-7 등으로 서명 요구수를 높여 보안을 강화할 수 있다고 설명한다 [14].
- 해당 구조를 통해 BitGo는 **분리된 역할과 책임(SoD)**를 시스템 수준에서 강제한다. 예를 들어, **BitGo가 서명 참여자로 있지만, 사용자 키 없이는 출금할 수 없기 때문에 BitGo 단독으로 자금을 이동할 수 없고**, 반대로 사용자가 두 개의 키(사용자 + 백업)를 보유하는 셀프커스터디 옵션에서는 BitGo가 코사인할 때만 출금이 가능하다 [15].

왜 그렇게 했나? (기술적 이유)

- **내부자 위험 감소**: 한 사람(또는 하나의 시스템)이 두 개 이상의 키를 갖지 못하도록 해 내부자가 자금을 탈취하는 것을 방지한다.
- **키 분실 대비**: 2-of-3 구조는 하나의 키가 손실돼도 나머지 두 키로 복구할 수 있다.
- **정책 준수 강제**: 정책 엔진에서 “어떤 역할이 어떤 조건에서 서명할 수 있는지”를 정의하고, 그 규칙에 맞게 서명 요청을 분배함으로써 조직도의 직무 분리뿐 아니라 시스템 레벨의 강제력을 부여한다.
- **확장 가능성**: 조직 규모에 따라 3-of-5, 4-of-7 등으로 유연하게 확장해 더 높은 보안을 구현할 수 있다.

설계 규칙 (Best Practice)

1. **키 구조로 분리 적합 강제**: **승인자/요청자/서명자**를 서로 다른 키나 서명 역할에 매핑한다. **Signer 키와 Policy 엔진, Approval 역할을 한 시스템이나 사람이 동시에 맡지 않도록 한다**.
2. **2-of-3 기본, 필요시 3-of-5 이상**: 작은 팀은 2-of-3으로 균형을 맞추고, 대형 조직이나 고액 자금은 3-of-5 이상으로 서명 요구수를 높인다 [14].

3. **키 분배와 물리적 보관 분리**: 키는 서로 다른 위치, 다른 관리자에게 분배하며, 백업 키는 오프사이트나 다른 기관이 보관한다 [11].
4. **정책 엔진과 Signer 분리**: 서명 모듈은 정책 엔진의 승인을 받아야 하지만, 정책 엔진과 서명 장치가 동일한 키나 인프라를 공유하지 않도록 분리하여 내부자 위험을 줄인다.

Security Controls – 사례 기반 1페이지 메모

아래 메모는 발표자가 세션에서 강조할 **핵심 통제 규칙**을 요약한 것이다. 발표 때 **사례→교훈→규칙** 순으로 서술하자.

규칙	교훈/사례	설계 포인트
Allowlist = 지연장치 포함	Coinbase의 주소록은 새 주소를 등록하면 **48시간 hold**를 적용하여 복사-붙여넣기 오류와 계정 침해에 대응한다 [1].	주소등록-승인-지연-활성화 상태 머신을 구현하고, 지연기간 동안 알림과 취소 옵션 제공.
정책 엔진 변경은 고위험	Fireblocks는 정책 엔진을 SGX 엔클레이브에 넣고, **정책 변경은 관리자 quorum 승인 및 감사 로그를 필요**로 한다 [4] [5].	정책을 코드로 관리하고 버전화하며, 변경 프로세스를 별도로 승인·지연·롤백할 수 있도록 설계.
Signer 내부 정책 검증	Anchorage는 HSM 내에 정책 엔진을 내장해 **서명 직전에 거래 지시를 검증**하며, 키와 정책을 하나의 보안 경계 안에서 보호한다 [7] [10].	최종 서명 장치에서 정책을 재검증하고, API 오케스트레이션을 신뢰하지 않는 구조를 채택한다.
SoD는 키 구조로 강제	BitGo는 모든 지갑에 **사용자 key · 백업 key · BitGo key**를 사용해 **2-of-3 서명** 구조를 적용하고, 어떤 사람도 하나 이상의 키를 보유하지 못한다 [11] [12].	승인자·요청자·서명자를 키와 역할에 분리하고, 필요시 3-of-5 이상으로 확장한다.

추가 배경 지식 – 예상 질문 대비

HSM, SGX 및 TEE는 무엇인가?

- **HSM(Hardware Security Module)**: 암호화 키를 안전하게 생성·저장·사용하기 위해 설계된 장치다. 키는 HSM 내부에서만 사용되고 외부 메모리로 노출되지 않는다. Anchorage의 HSM 모델은 **에어갭된 하드웨어**에 정책 엔진을 내장해 서명 지시를 검증한다 [7].
- **SGX(Intel Software Guard Extensions)**: CPU 내부에 **신뢰 실행 환경(TEE)**을 제공하는 기술로, 프로그램과 데이터를 암호화된 메모리(enclave)에 넣어 운영체제나 관리자 권한을 가진 공격자도 내부 상태를 볼 수 없게 한다. Fireblocks는 SGX 안에서 정책 엔진과 키 쉐어를 실행하여 외부 공격자·내부자 모두로부터 보호한다 [4].
- **TEE**: 신뢰 실행 환경의 일반적인 용어이며, SGX와 AWS Nitro Enclaves 등 여러 구현이 있다. TEE 내부에 정책 엔진을 배치하면 코드와 데이터의 무결성을 보장하고, 원격 attestation(원격 검증)을 통해 실행 중인 코드가 신뢰할 수 있는 버전임을 검증할 수 있다 [16].

멀티시그 vs MPC 지갑의 차이

- **멀티시그**: 각 키가 독립적인 전체 키이며, 블록체인 네이티브 기능이나 스마트컨트랙트로 다중 서명을 요구하는 지갑이다. BitGo의 멀티시그 지갑은 **2-of-3 서명**이 필요하며 투명한 온체인 감사 기록을 제공한다 [14].
- **MPC(Multi-Party Computation)**: 하나의 개인 키를 여러 조각(key shares)으로 나누고, 각 파티가 자신의 조각만 보유하여 서명을 공동 계산하는 방식이다. BitGo는 MPC 지갑에서도 **2-of-3 key shares**를 사용하며, 각 파티는 자신의 key share를 보유하므로 아무도 전체 키를 보지 못한다 [17]. MPC는 키가 온체인에 노출되지 않고 여러 체인을 지원하는 장점이 있지만, 일부 네트워크에서 전통적 멀티시그가 더 간단하다.

- 두 방식 모두 **분리 적합(SoD)**를 달성하며, 조직 상황과 자산 특성에 따라 선택한다.

키 복구와 관리

- **키 분실 시 복구**: 2-of-3 구조에서는 하나의 키를 잃어도 나머지 두 키로 자금을 이동할 수 있다. 백업 키는 안전한 장소(예: 금고, 별도 기관)에서 오프라인으로 보관한다.
- **정기적 키 회전**: 시간이 지나면 키를 교체하는 '키 회전(key rotation)'을 실시해 장기간 노출로 인한 위험을 줄인다 [18]. 회전은 새로운 지갑을 만들어 소액 테스트 트랜잭션으로 검증한 뒤 본격적으로 이전한다.
- **사망·퇴사 등 비상 상황**: 조직은 백업 서명자와 구체적인 비상 대응 절차를 문서화해야 하며, 필요한 법적 권한과 준수를 확인한다.

정책 설계 시 고려해야 할 질문

1. **어떤 조건에서 거래가 승인되나?** – 금액 한도, 자산 종류, 승인자 수 등.
2. **주소 등록·삭제는 누가 할 수 있고, 어떤 절차를 거치나?**
3. **정책 변경은 얼마나 자주, 어떤 절차로 발생하나?**
4. **비상 대응 계획은?** – 키 분실, 침해, 내부자 악용 등 시나리오를 가정하여 정책과 절차를 마련한다.
5. **사용자 경험(UX)과 보안의 균형** – 과도한 지연이나 승인 요구는 업무 효율을 저해할 수 있으므로, 고객 유형에 따라 정책을 세분화한다.

세션 스토리라인 예시 (1시간 내외)

1. **도입 (0–5분)** – 수탁형 지갑의 목적과 세션의 핵심 메시지: **"보안은 키를 숨기는 기술이 아니라, 잘못된 송금이 구조적으로 불가능하도록 설계하는 것이다."**
2. **패턴 1 설명 (5–20분)** – Coinbase 사례 소개 → 주소 화이트리스트와 48시간 hold의 효과 설명 → 등록·승인·지연·활성화 상태 머신 시각화 → 우리 시스템 설계 규칙 정리.
3. **패턴 2 설명 (20–35분)** – Fireblocks 사례 소개 → SGX 엔클레이브, 정책 변경 quorums 설명 → TEE 데모 또는 코드 버전 관리 사례 → 정책 무결성 설계 규칙 도출.
4. **패턴 3 설명 (35–50분)** – Anchorage 사례 소개 → HSM 내부 정책 엔진, API 우회 공격 방어 설명 → 서명 경계에서의 재검증 데모 → 설계 규칙 정리.
5. **패턴 4 설명 (50–60분)** – BitGo 2-of-3 구조와 분리 적합 설명 → 다중 서명 vs MPC 비교 → 키 분배·회전·비상 계획 논의 → 설계 규칙 정리.
6. **마무리 및 질의응답** – 보안 통제 1페이지 메모를 다시 보여주며 요점 강조. 청중이 멱등성/재시도 등 신뢰성 질문을 하더라도 이번 세션에서 다루지 않는다고 명확히 전달하고, 해당 내용은 별도 세션(Tracker/Reliability)에 포함될 것이라고 안내한다.

---

[1] Coinbase Transfer to Wallet: The Ultimate Guide
https://www.walletfinder.ai/blog/coinbase-transfer-to-wallet

[2] Is Coinbase Safe? How To Protect Your Cryptocurrency
https://www.aura.com/learn/is-coinbase-safe

[3] Coinbase's "Allow List" Hold Is Misleading and Support Is No Help. : r/Coinbase
https://www.reddit.com/r/Coinbase/comments/1kdifyk/coinbases_allow_list_hold_is_misleading_and/

[4] Fireblocks Governance and Policies Engine
https://www.fireblocks.com/platforms/governance-and-policies

[5] [16] Securing Digital Assets in an Evolving Threat Landscape: The Fireblocks Defense-in-Depth Approach to Security | Fireblocks
https://www.fireblocks.com/report/the-fireblocks-defense-in-depth-approach-to-security

6 Fireblocks - Submission in response to: Crypto asset secondary service providers: Licensing and custody requirements
https://treasury.gov.au/sites/default/files/2022-12/c2022-259046-fireblocks.pdf

7 Mitigating custody risks in the wake of FTX
https://www.anchorage.com/insights/mitigating-custody-risks-in-the-wake-of-ftx

8 9 10 Anchorage Digital_Finding End-To-End Security In Crypto Custody_Guide
https://learn.anchorage.com/Finding-End-to-End-Security-in-Crypto-Custody.pdf

11 12 17 BitGo Wallet Types
https://developers.bitgo.com/docs/wallet-types

13 14 18 What are Multi-Signature Wallets? Crypto Wallet Security | BitGo
https://www.bitgo.com/resources/blog/what-is-a-multi-signature-wallet/

15 SEC Issues Investor Bulletin on Best Practices for Crypto Asset Custody | BitGo
https://www.bitgo.com/resources/blog/sec-issues-investor-bulletin-on-best-practices-for-crypto-asset-custody/