Audit+ 청서

목차

1) 개요
2) 본 문서 읽기
3) 커뮤니티에 현존하는 솔루션
   a) Klevoya Inspect
   b) 소프트웨어 개발 라이브러리
4) 커뮤니티에서 요구하는 솔루션
   a) 안전한 스마트 컨트랙트 개발을 위한 소프트웨어 라이브러리
   b) 버그 바운티
   c) 컨트랙트 업그레이드 승인 DAO
5) 다른 생태계들에서 수행된 동등한 작업
   a) dApp용 보안 레지스트리
   b) 오픈소스 보안 도구
   c) 안전한 스마트 컨트랙트 개발을 위한 소프트웨어 라이브러리
   d) 스마트 컨트랙트 작성 시 일반적인 보안 문제들
   e) 버그 바운티
   f) AML 플랫폼
6) EOS 생태계를 개선할 수 있는 이니셔티브 목록
   a) 오픈소스 보안감사 API 및 플랫폼
   b) 컨트랙트 업그레이드 승인 DAO
   c) AML 플랫폼 구축
   d) 안전한 스마트 컨트랙트 개발을 위한 소프트웨어 라이브러리
   e) 버그 바운티
   f) 자동화된 보안감사 도구 구축
   g) EOSIO 스마트 컨트랙트 작성 시 일반적인 보안 문제 목록
7) 재단에 권고하는 최선의 행동 방침 목록
8) 부록

개요

본 문서에서 저희는 EOSIO 블록체인의 전반적인 보안을 개선하기 위해 저희의 리서치 및 이니셔티브를 제공합니다.

저희의 연구에 따르면 EOSIO에는 보안에 초점을 맞춘 솔루션이 거의 존재하지 않습니다. EOSIO 코어 시스템은 보안 지향적인 접근 방식을 허용하도록 아름답게 설계되었지만, 다른 블록체인들이 제공하는 것과 동일한 보안 표준을 제공하려면 아직 많은 작업을 필요로 합니다.

본 문서에 존재하는 핵심 영역은 지식입니다. 적절한 도구와 문서를 제공해야만 개발자 커뮤니티 내에서 스스로 영속적인 지식을 얻을 수 있습니다.

해커 커뮤니티의 방대한 보안 관련 지식과 그들의 호기심 많은 특성은 저희가 EOSIO를 지속적으로 보호하기 위해 활용해야 하는 부분입니다.

EOSIO dApp 과의 상호 작용이 안전하고 안심할 수 있다는 사용자들의 인식을 제고하는 것은 저희가 기본 서비스와 API를 제공하는 한 EOSIO의 고유한 구조로 인해 달성 가능합니다.

앞서 언급한 것처럼 EOSIO의 전반적인 핵심 디자인은 보안 지향적인 접근 방식을 취합니다. 저희는 기본에 집중하고 다른 블록체인과 차별화되는 데 도움이 될 몇 가지 EOSIO의 주요 요소들을 활용함으로써 한 단계 더 나아갈 수 있습니다.

본 문서 읽기

제공된 지침에 따라 본 문서는 다음과 같이 구분됩니다.

1) 현존하는 솔루션: 먼저 EOSIO에서 현재 사용 가능한 솔루션들이 무엇인지 설명합니다.
2) 커뮤니티에 의해 요구된 솔루션**:** 이어서 저희의 리서치를 기반으로 커뮤니티에서 요구하는 솔루션들에 대해 다룹니다.
3) 다른 커뮤니티들에 현존하는 솔루션**:** 이후에 저희는 다른 커뮤니티들에 EOSIO에 적용 가능한 어떤 솔루션들이 존재하는지를 살펴봅니다.
4) 우리의 이니셔티브**:** 본 문서의 후반부에서는 전반적인 EOSIO 보안을 개선하기 위한 모든 이니셔티브들을 전달합니다.
5) 본 문서의 마지막 부분에는 보안 기반 형성을 위한 출발점으로 가장 적합하다고 생각되는 4가지 이니셔티브들이 나열되어 있습니다.
6) 부록에서는 독자에게 생소할 수 있는 용어들에 대한 설명이 제공됩니다.

1. 커뮤니티에 현존하는 솔루션

EOSIO 커뮤니티에는 사용할 수 있는 몇 가지 솔루션들이 존재하지만, 전반적으로 기본적인 보안에 관한 솔루션들이 부족한 경향을 보입니다.

아래에서는 주목할 만하고 현재 사용되고 있는 몇 가지 솔루션들에 대해 설명합니다.

1) Klevoya Inspect

A) 설명

Klevoya Inspect는 개발자가 컴파일된 WASM 코드를 업로드하고 Klevoya 팀이 개발한 고유한 패턴을 기반으로 취약점을 검사하는 도구입니다. [1]

검사에는 정적 분석이라는 기술이 사용됩니다. 정적 분석은 실제로 실행하지 않고 컴퓨터 프로그램의 소스 코드 또는 일부 중간 표현(IR, intermediate representation)에 대한 추론에 의해 수행됩니다.[1]

내부적으로 업로드된 WASM 코드와 ABI는 데이터베이스에 로드되는 독점적 중간 표현(intermediate representation)으로 업로드됩니다. 그런 다음 알려진 취약점을 찾는 데이터베이스의 엔티티에 대해 패턴 매쳐(Patterns matcher)를 실행합니다. [2]

해당 제품은 오픈 소스가 아니며 유료 서비스입니다.

B) 특징

- 업로드된 Compiled WASM 코드의 자동 스캔
- 내부 설계 패턴을 기반으로 WASM 코드의 취약점을 탐지하는 스캐닝 엔진

C) EOSIO에 대한 이점

비록 이 기능은 유료 서비스이며 수동 보안 감사와 같은 깊이 있는 분석을 보장하지는 않지만, 정적 분석 퍼징(Static analysis fuzzing, fuzz)은 매우 강력한 도구이며 개발자들이 보안 문제를 스스로 진단하고, 해결할 수 있는 고유한 기회를 제공할 수 있습니다. 이는 시간이 지남에 따라 이들이 보안적으로 더 우수한 스마트 컨트랙트를 작성할 수 있도록 할 것입니다.

D) 참고 자료

- [1] ***https://klevoya.com/inspect/index.html***
- [2] ***https://klevoya.medium.com/?p=242400e10cdc***
- [3] ***https://www.fuzzingbook.org/html/ConcolicFuzzer.html#SMT-Solvers***

2) 소프트웨어 개발 라이브러리

A) 설명

EOSIO는 Bios, System, msig, Wrap 및 토큰 컨트랙트를 포함하는 eosio.contracts warehouse를 제공합니다. MIT 라이선스 컨트랙트를 통해 개발자는 코드를 거의 무제한으로 사용할 수 있으며 거버넌스 매개변수와 경제 모델을 포함한 이러한 컨트랙트들을 학습함으로써 EOS 메인넷을 마스터할 수도 있습니다.[1]

커뮤니티에서 사용할 수 있는 몇 가지 범용 SDK들 또한 존재합니다.:

1) SX 조직이 좋은 품질의 코딩 표준이 담긴 9건의 DeFi 유형 스마트 컨트랙트를 Github에 공개했고, 그 중 일부가 보안 감사를 통과했지만, 이러한 라이브러리에는 따라야 할 저작권 협약을 명시하지 않았다는 점에 유의해야 합니다.[2]
2) Blockmatic은 112개의 커뮤니티 오픈 소스 EOS 스마트 컨트랙트 소스 코드를 수집하고 해당 목록을 공개했습니다. 해당 코드에는 게임, DeFi, 결제, DAO, 스테이블코인, NFT, ICO 및 기타 유형에 대한 컨트랙트 예시, 그리고 Rust를 개발 언어로 활용하는 프로젝트까지 포함되어 있어, 저희는 여기서 EOS 개발에 대한 해당 개발자의 열정을 엿볼 수 있었습니다.[3]

B) 특징

현재 커뮤니티에는 오픈 소스 코딩 예시 목록들이 많이 존재하지만, 보안 감사를 거친 코드는 상대적으로 적고, 심지어 그 양은 이더리움보다 몇 배나 적습니다. 가장 놀라운 점은 대부분의 코드 베이스들이 유지 관리를 위해 일시 중단된 상태라는 것입니다.

C) EOSIO에 대한 이점

블록체인 개발의 핵심 역할 중 하나는 개발자입니다. EOSIO 생태계에 더 많은 개발자를 유치하는 방법은 EOS 커뮤니티가 생각해야 하는 중요한 문제입니다. 고품질의 오픈 소스 코드

베이스를 개발하고 공유함으로써 개발의 어려움을 줄일 수 있고, 동시에 경험이 없는 개발자는 안전한 스마트 컨트랙트를 빠르게 개발할 수 있어, EOS의 채택을 늘릴 수 있습니다.

D) 참고 자료

- *[1]https://github.com/EOSIO/eosio.contracts*
- *[2]https://github.com/stableex*
- *[3]https://github.com/blockmatic/eosio-contracts-list*

2. 커뮤니티에서 요구하는 솔루션

저희는 저희의 연구와 EOSIO 커뮤니티 내 이해관계자들과의 대화를 기반으로 EOSIO 생태계의 전반적인 보안을 개선하는 데 도움이 될 수 있는, 커뮤니티에서 요구하는 몇 가지 핵심 부분에 대해 기술합니다.

1) 안전한 스마트컨트랙트 개발을 위한 소프트웨어 라이브러리

A) 설명

보안이 강화된 소프트웨어 개발 라이브러리(SDK)를 통해 소프트웨어 개발자는 테스트를 거친 컨트랙트 템플릿과 라이브러리를 활용하여 스마트 컨트랙트를 개발할 때 위험을 최소화할 수 있습니다.

표준 SDK와 비슷하지만 여기에는 최상의 보안 사례를 기반으로 구축된 특정 작업을 수행하는 사전 구축된 예제가 포함되어 있으므로 특정 작업을 수행하는 함수를 직접 생성하는 대신 보안 문제에 관한 테스트를 거친 이미 존재하는 방법들을 활용할 수 있습니다.

B) 추론

이러한 라이브러리를 사용하면 개발자들이 보안 관점에서 이미 면밀히 검토된 기존 방법을 활용하는 데 도움이 될 뿐만 아니라, 유닛 테스트에도 보안 테스트를 포함할 수 있으므로 스마트 컨트랙트 작성 당시 위험을 최소화할 수 있습니다.

EOSIO의 모든 개발자가 EOSIO의 최신 보안 문제를 해결하기 위해 지속적으로 업데이트되는 동일한 SDK 목록을 사용하기 시작하면, 개발자가 쉽게 피할 수 있는 실수들을 하지 않도록 방지하는 데 도움이 되기 때문에 스마트 컨트랙트의 전반적인 보안을 향상시킬 수 있습니다.

C) 필요 당사자

EOSIO 소프트웨어 개발자는 더 안전한 스마트 컨트랙트들을 생성하길 원하지만, 우리는 전반적인 도구가 부족합니다.

2) 버그 바운티

A) 설명

버그 바운티는 애플리케이션이나 소프트웨어의 취약점을 성공적으로 발견한 윤리적인 해커들에게 주어지는 금전적 보상입니다.

이는 모든 체인에 걸쳐 광범위하게 적용되며, 특히 코드 베이스가 지속적으로 변경되거나 여러 종류가 존재하는 경우, 블록체인 코드를 지속적으로 면밀히 조사하는 데 매우 중요한 역할을 합니다.

B) 추론

버그 바운티 프로그램을 원활하게 운영하면 EOSIO의 전반적인 보안을 향상시킬 수 있는 최고의 해커 커뮤니티를 끌어들일 수 있습니다. 아직은 이러한 프로그램의 부재로 인해 해커 커뮤니티는 EOSIO 코드 베이스를 조사하기 위해 시간과 노력을 투자하지 않습니다.

C) 필요 당사자

윤리적인 해커들은 EOSIO에 참여하기를 원하지만, 우리가 제대로 실행되는 버그 바운티 프로그램을 정의하고 시작한 후에야 그들은 시간과 노력을 투자할 것입니다.

3) 컨트랙트 업그레이드 승인 **DAO**

A) 설명

EOSIO에서 스마트 컨트랙트들은 활성 계정에 배포되며 일반적으로 개발자는 향후 스마트 컨트랙트를 업그레이드할 수 있는 모든 권한을 갖습니다. 이는 신뢰 문제를 야기하는데, 해당 스마트 컨트랙트의 사용자들은 본질적으로 스마트 컨트랙트의 보안 감사 상태에 관계없이 '해당 계정 소유자가 악의를 저지르지 않을 것'이라고 신뢰해야 하기 때문입니다.

물론 사용자와 스마트 컨트랙트 소유자 간의 신뢰를 형성하기 위해 누구도 해당 스마트 컨트랙트 를 업데이트하지 못하도록 할 수는 있지만, 이는 향후 해결이 요구되는 버그나 보안 취약성이 식별될 당시에 필요한 스마트 컨트랙트 업데이트에 대한 방해요소가 될 수 있습니다.

따라서 사용자와 스마트 컨트랙트 소유자 간의 신뢰를 촉진하고 스마트 컨트랙트 소유자가 필요할 때 적시에 컨트랙트 업데이트를 수행할 수 있는 일종의 다중 서명 시나리오가 필요합니다.

B) 추론

앞서 언급한 바와 같이, 사용자와 스마트 컨트랙트 소유자 간의 더 큰 신뢰를 촉진하는 동시에 스마트 컨트랙트 소유자의 소프트웨어 업데이트를 가능하게 하기 위한 더 나은 컨트랙트 권한 관리 방법이 필요합니다.

C) 필요 당사자

커뮤니티와 스마트 컨트랙트 소유자 모두 이러한 유형의 기능을 필요로 합니다. dApp 소유자는 그들의 사용자 기반이 자신을 신뢰함과 동시에 필요할 경우 dApp을 업데이트할 수 있는 유연성도 갖기를 원합니다. 커뮤니티가 사용하는 dApp이 해킹당할 위험이 없다거나 소유자가 선의 또는 악의가 있다고 완전하게 약속할 수는 없지만, 이러한 추가 신뢰 레이어를 제공하면 커뮤니티와 dApp 소유자 사이에 더 많은 신뢰를 구축하는 데 도움이 됩니다.

3. 다른 생태계들에서 수행된 동등한 작업

저희는 다른 블록체인들의 조사를 통해, 일반적으로 상위 블록체인 프로토콜들은 항상 사용할 수 있는 보안 관련 콘텐츠 및 제품 목록을 제공했으며, 전반적으로 보안 접근 방식에 대한 일반적인 주제는 모든 블록체인들 사이에서 대부분 유사하다는 것을 발견했습니다.

저희는 또한 본 문서의 후반부에서 이더리움에서 결실을 맺기 시작한 상당히 새로운 아이디어를 저희의 핵심 이니셔티브 중 하나로 다룹니다.

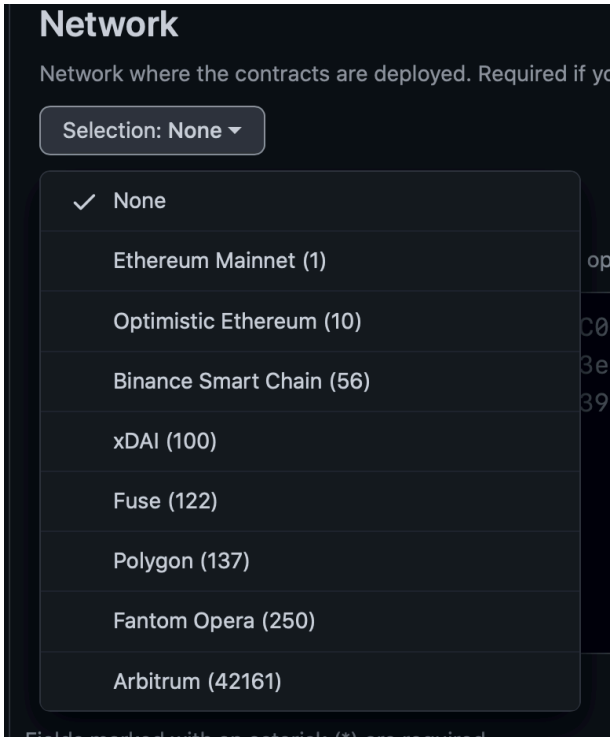
1) dApp용 보안 레지스트리

A) 설명

저희는 이더리움에서 체인, 주소 및 그들이 속한 관련 프로젝트를 고려하여 현재의 모든 스마트 컨트랙트를 나열하려고 시도하는 깃허브 REPO가 존재한다는 것을 발견했습니다.
2021년 9월에 시작된 REPO 내역을 보면 ETH의 상당히 새로운 시도인 것으로 보입니다.[1]

"주어진 컨트랙트 주소에서 취약성이 발견될 경우 알려진 프로젝트의 보안 연락처 추적은 물론, 사용자가 특정 주소와 상호 작용할 때 해당 주소에 대한 정보를 제공할 수 있도록 하는 것이 목표입니다."

현재 지원되는 ETH EMV 체인 목록은 다음과 같습니다.



B) 특징

1. 프로젝트 리스팅: 프로젝트를 리스팅 하기 위해서는 이름, Github 주소, 연락처 이메일(보안 문제 관련), 체인 및 컨트랙트 주소 등 프로젝트 세부 정보를 포함하는 리스팅 양식을 작성합니다.
2. 비상 연락처: 프로젝트를 리스팅 할 때 비상 보안 연락처 이메일이 요구됩니다. 여기에는 컨트랙트에서 심각한 취약점이 확인되면 커뮤니티가 프로젝트 측에 이를 고지할 수 있도록 하는 아이디어가 내포되어 있습니다.
3. 자동화된 보안 연락처 추적: 스마트 컨트랙트 생성 시 보안 연락처 세부 정보를 자동으로 등록할 수 있도록 하기 위해 개발자가 해당 정보를 코드 코멘트로 제공 가능한 특정 형식이 채택되어 있습니다.

C) 지양해야 할 주요 문제

현재 솔루션은 불필요한 단계를 많이 추가하는 매우 수동 프로세스인 것으로 보이며, 이는 EOSIO를 사용하여 더욱 자동화할 수 있습니다.

D) 참고 자료

- [1]https://github.com/ethereum-lists/contracts

2) 오픈소스 보안 도구

A) 설명

오픈 소스 보안 도구는 개발자가 스마트 컨트랙트의 전반적인 보안을 자체 평가할 수 있는 매우 쉬운 방법입니다. 흔하지는 않지만, 이러한 도구들이 존재하는데, 특히 상위 프로토콜들 내에 주로 존재합니다. 저희는 이 분야의 적극적인 경쟁을 위해 노력하고 있습니다. 이러한 도구들은 자가 진단을 촉진하고 피해야 할 일반적인 보안 문제에 대한 개발자 커뮤니티의 지식을 향상시킵니다. [1]

B) 특징

이러한 도구들의 일반적이면서도 훌륭한 기능들은 다음과 같습니다.

- 오류, 오류 유형, 심각도 및 소스 코드 상의 발생 위치를 식별합니다.
- 지속적인 통합 및 SDK에 쉽게 통합됩니다.
- 개발자가 맞춤형 분석을 작성할 수 있는 API의 활용이 가능합니다.
- 실행 속도는 매우 빠릅니다.

- 코드는 대부분 오픈소스이며 보안 및 개발자 커뮤니티에 의해 유지 관리됩니다.

C) 지양해야 할 문제

모호한 언어로 이러한 도구를 개발하고 소스 코드를 닫힌 상태로 유지하는 것을 피해야 합니다. 대부분의 이러한 도구들은 오픈 소스이며 Python과 같은 분석에 매우 적합한 프로그래밍 언어로 개발되었습니다. 그러나 Solana 같은 예 또한 존재하는데, 이는 사용할 수 있는 유일한 도구가 비공개 소스이며, 이로 인해 커뮤니티 참여가 제한되고 도구의 성장을 억제할 수 있습니다.

D) 참고 자료

- [1]https://github.com/crytic/slither
- [1]https://github.com/ConsenSys/mythril
- [1]Soteria (Solana) - https://medium.com/coinmonks/soteria-a-vulnerability-scanner-for-solana-smart-contracts-cc202cf17c99

3) 안전한 스마트 컨트랙트 개발을 위한 소프트웨어 라이브러리

A) 설명

보안에 중점을 둔 이러한 소프트웨어 개발 라이브러리는 스마트 컨트랙트의 개발에 가이드를 제공하고, 개발의 어려움을 줄이며, 코드 품질과 보안을 향상시키기 위해 사용됩니다.

이들은 여러분들의 표준 SDK와 비슷하지만 개발자가 더 안전한 코드를 작성하도록 안내하는 추가 기능이 함께 제공됩니다.

이는 일반적으로 공식 개발 담당자가 주도하거나 잘 알려진 프로젝트 당사자가 개발합니다. 개발 라이브러리는 일반적으로 토큰, NFT, 랜딩, 스왑, 거버넌스 등과 같은 애플리케이션을 그 예로 들 수 있습니다. 일부는 오버플로우를 방지하는 산술 계산을 위한 기능을 제공하는 safemath 라이브러리와 같이 컨트랙트 개발에 일반적으로 사용되는 기능을 제공합니다.

B) 특징

이러한 보안 개발 라이브러리는 우수한 코드 품질을 가지며 다음과 같은 구체적인 특징이 존재합니다.

- 각 유형의 애플리케이션에 완전한 기능 구현 존재
- 단순하고 모듈화된 코드;

- 명확한 의미와 일관된 스타일의 함수 및 변수 명명 방법
- 변수 및 함수 기능 주석 설명;
- 포괄적인 유닛 테스팅;
- npm, cargo와 같은 일반 패키지 관리 도구 지원
- 보다 포괄적인 보안 감사 진행 이후

C) 지양해야 할 문제

일회성 개발 아웃소싱은 피해야 하며, 개발 라이브러리는 보안을 위해 지속적으로 업데이트되어야 함과 동시에 커뮤니티에 최대한의 기술 지원을 제공해야 합니다.

D) 참고 자료

- https://github.com/OpenZeppelin/openzeppelin-contracts
- https://github.com/open-web3-stack/open-runtime-module-library
- https://github.com/solana-labs/solana-program-library

4) 스마트 컨트랙트 작성 시 일반적인 보안 문제들

A) 설명

일반적인 보안 문제(Common security pitfalls)는 특정 블록체인에서 개발자가 저지른 과거 보안 실수를 나열하는 심층적인 최신 문서들입니다. 이 문서들은 미래의 개발자가 같은 실수를 반복하지 않도록 돕기 위해 만들어졌습니다. [1]

따라야 할 규칙의 목록이라고 생각할 수 있습니다. 이 기본 규칙을 따르면 스마트 컨트랙트는 일반적으로 해커로부터 안전합니다.

B) 특징

이들은 공격 유형에 대한 설명, 오류가 발생한 위치, 이러한 오류로 인해 발생할 수 있는 취약점, 개발자가 자신의 코드에서 동일한 실수를 반복하지 않도록 하는 예방 기술을 보여주는 몇 가지 예제 소스 코드가 포함된 일반적인 보안 문제 목록들입니다. .[2]

때때로 해커가 특정 코드 조각을 악용하는 이전 실제 실수 사례도 제공되어 개발자에게 위험에 대해 더 자세히 교육하는 데 도움이 됩니다.[3]

C) 지양해야 할 문제

저희는 리서치 도중 이러한 유형의 문서들 중 일부가 매우 분산되어 있고 제대로 업데이트되지 않은 것을 발견했습니다. 우리는 이 정보가 너무 많은 장소에 분산되는 것을 피해야 하며, 이상적으로 이러한 유형의 정보는 Git 유형의 시스템에 남아 있어야 하며, 모든 참조는 동일한 위치를 가리켜야 합니다.

D) 참고 자료

- https://medium.com/coinmonks/soteria-a-vulnerability-scanner-for-solana-smart-contracts-cc202cf17c99
- [1]**https://blog.neodyme.io/posts/solana_common_pitfalls**
- [2]https://consensys.github.io/smart-contract-best-practices/known_attacks/
- [3]https://github.com/sigp/solidity-security-blog

5) 버그 바운티

A) 설명

대부분의 상위 블록체인은 자체 버그 바운티 프로그램[1]을 운영하거나 hackerone[2], bugcrowd[3] 및 Immunefi[4]와 같은 버그 바운티를 촉진하는 외부 플랫폼을 통한 프로그램들을 진행합니다.

버그 바운티 프로그램은 많은 웹사이트, 조직 및 소프트웨어 개발자가 제공하는 것으로, 개인이 특히 보안 악용 및 취약성과 관련된 버그 보고에 대해 인정 및 보상을 받을 수 있습니다.

B) 특징

- 관련 바운티들과 함께 잘 정의된 취약성 분류 및 심각도 범위
- 버그를 구성하는 것에 대해 잘 정의된 규칙
- 버그 제출에 관련하여 잘 정의된 절차
- 특정 버그 바운티에 대한 상위 해커를 보여주는 리더보드
- 좋은 커뮤니케이션 및 비용 지불
- 다른 모든 EOSIO 체인에 대한 취약점 공유 및 그러한 취약점의 공개 발표에 대한 좋은 커뮤니케이션

C) 지양해야 할 문제

ETH와 마찬가지로 EOSIO는 멀티체인이 되었기 때문에 취약성이 있는 경우 다른 모든 체인과 통신하는 표준화된 방법이 마련되어 있어야 합니다. 일단 해당 취약점이 공개되면, 해커가 이를 활용하여 다른 EOSIO 체인을 공격할 수 없어야 합니다.

우리는 모든 체인이 새로 배포된 보안 패치 및 업그레이드를 평가할 기회를 갖기 전에 취약점을 공개적으로 알리는 것을 피해야 합니다. 따라서 EOSIO 보안 문제를 주요 사람들에게 효과적으로 전달하기 위한 방법은 중요 요구 사항입니다.

D) 참고 자료

- [1]https://ethereum.org/en/eth2/get-involved/bug-bounty/
- [2]https://hackerone.com/cardano-foundation?type=team
- [3]https://bugcrowd.com/vulnerability-rating-taxonomy
- [4]https://immunefi.com

6) AML 플랫폼

A) 설명

EOS는 DAO 시스템으로, 스마트 컨트랙트를 공격한 것으로 의심되는 해커 계정을 동결할지 여부 등 많은 의사결정이 체인 상의 정보 분석에 의존하고 있습니다. 이때 이러한 행동을 분석할 수 있는 데이터 플랫폼이 필요합니다. AML 플랫폼을 구축하면 해킹 사고 후 자금 흐름을 추적할 수 있는 충분한 수단을 확보할 수 있습니다. 잘 알려진 이더리움 블록 탐색기 Etherscan은 이러한 플랫폼을 구축했습니다. 해킹 사건이 공개되면 플랫폼은 공격자의 계정을 식별하여, 해당 계정을 착취자로 표시하며, 거래소와 같은 관련 이해관계자는 플랫폼에 쿼리하여 이 계정과 주고받는 송수신을 차단할 수 있습니다.

B) 특징

태깅(Tagging) 플랫폼은 계정 태깅의 정확성을 보장할 수 있는 충분한 데이터가 있어야 하며, 새로운 이벤트를 적시에 처리할 수 있도록 생태계에 충분한 참여가 있어야 합니다.

C) 지양해야 할 문제

계정 표시가 적고, 해당 표시가 정확하지 않으며 커뮤니티 사건에 대한 느린 대응이 다소 존재합니다.

D) 이러한 솔루션을 통해 얻을 수 있는 비즈니스 기회

일부 유료 블록체인 분석 서비스를 제공할 수 있습니다.

E) 참고 자료

- Etherscan Label Word Cloud: https://etherscan.io/labelcloud
- https://eosdetective.io/network

4. EOS 생태계를 개선할 수 있는 이니셔티브 목록

저희는 현재 커뮤니티에 존재하는 모든 솔루션, 커뮤니티가 요구하는 사항, 다른 블록체인 프로토콜에 존재하는 사항 등을 토대로 EOSIO가 보안 측면에서 앞서나가는 데 도움이 될 이니셔티브 목록을 도출했습니다.

다른 블록체인과 동일한 수준에 도달하기 위해서는 몇 가지 작업이 필요하지만, 우리의 기준선이 구축되면 EOSIO만의 고유한 디자인을 통해 다른 블록체인을 한 단계 뛰어넘을 수 있습니다.

1. 오픈소스 보안감사 API 및 플랫폼

워킹 그룹	Audit+, Wallet+, Core+
목표 대중	거래소, 커뮤니티 및 기타 dApps
이니셔티브 유형	개발, 디자인
MVP 제안 여부	Yes
MVP 비용	$95,500

A) 문제

현재로서는 커뮤니티가 상호작용하고 있는 스마트 컨트랙트 코드에 대한 감사(Audit) 및 안전성 여부를 확인할 수 있는 서비스가 존재하지 않으며, 감사자가 감사 결과 및 관련 정보를 게시할 수 있는 중앙화된 장소도 존재하지 않습니다.

B) 목표

커뮤니티가 상호작용 하고 있는 스마트 컨트랙트의 보안성을 쉽게 검증할 수 있도록 하는 컨트랙트 소스 코드 검증 플랫폼[3]과 감사 정보 공개 플랫폼[1]을 구축합니다. 커뮤니티는 사용자 및 월렛, 거래소와 같은 다른 애플리케이션들이 포함됩니다.

C) EOSIO에 대한 이점

1. 거래소들은 온체인 상의 해시가 마지막으로 감사를 수행한 해시와 일치하는지를 검증함으로써 스마트 컨트랙트들의 보안을 확인할 수 있습니다.
2. 월렛은 상호 작용 중에 사용되는 스마트 컨트랙트들에 대한 승인 스탬프를 제공하기 위해 잘 유지된 표준들을 사용하여 API와 통합할 수 있습니다.
3. 사용자는 월렛이나 어떠한 애플리케이션, 관련 스마트 컨트랙트 혹은 오딧 상태를 보기 쉽게 도와주기 위해 제공된 프론트엔드를 통해 애플리케이션들의 보안을 간편하게 확인할 수 있습니다.

D) 산출물

1. 신규 감사 푸시를 위한 온 체인 컨트랙트: 보안 감사자가 재단 또는 MSIG를 통해 완료된 새로운 감사를 체인 상에 푸시 할 수 있습니다. 각 감사에는 다음이 포함되어야 합니다.
   a. 컨트랙트가 로드되는 계정 주소
   b. 감사가 수행된 컴파일드 WASM의 SHA256 해시
   c. 코드 repo URL.
   d. 감사 날짜
   e. 감사자명
   f. 감사자 이메일 주소
   g. (해당되는 경우) 인증서 링크
   h. (공용인 경우) 보고서 링크
   i. 결과: 통과 또는 실패 여부

2. 새로운 스마트 컨트랙트를 자동으로 추적하는 **ABI** 표준: 새로운 스마트 컨트랙트를 자동으로 등록하려면, 개발자는 그들의 ABI에 그들의 컨트랙트에 대한 기본 정보를 등록해야 합니다. (**"__comment" : ""** 필드를 사용할 수 있습니다.)[4]
   필요 정보:
      a) 배포 시 프로젝트를 위한 보안 연락처 정보
      b) 프로젝트 이름
      c) 코드 Repo 링크

3. **API** 표준 **:** Github 상에 등록 및 유지되는 API 표준 생성[5]

[5]API JSON 예시

```
{
  "account": "compcontract",
  "auditStatus":0 , // 0=not audited,1=auditing, 2=audited
  "timestamp": "timestamp of latest audit",
  "org": {
    "company_name": "dApp company",
    "website": "https://www.company.io",
    "email": "info@company.io",
    "code_repo":{
       "code_location": "CODE URL"
    },
```

```
    "social": {
      "steemit": "",
      "twitter": "company01",
      "facebook": "",
      "github": "company01",
      "keybase": "company01",
      "telegram": "company01"
    }
  },
  "audits": [{
    "timestamp": "timestamp of audit",
    "organisation": {
      "name": "Sentnl",
      "country": "GB",
      "website": "https://www.sentnl.io",
      "email": "charles@sentnl.io",
    },
    "name": "name of smart contract",
    "hash": "sha256",
    "report": "link to report",
    "certificate": "link to certificate",
    "result": "passed"

  },
  "timestamp": "timestamp of audit",
  "organisation": {
      "name": "Slowmist",
      "country": "CH",
      "website": "https://www.slowmistio",
      "email": "charles@slowmist.io",
    },
    "name": "name of smart contract",
    "hash": "sha256",
    "report": "link to report",
    "certificate": "link to certificate",
    "result": "passed"
  },
]
}
```

4. 프론트엔드**:** API 표준에 설명된 바와 같이, dApp 리스트 및 감사 정보를 보여주는 커뮤니티용 웹 프론트엔드 **+** 모바일 친화적 플랫폼

E) 노력 수준

이 이니셔티브는 '온체인 데이터', 'API' 및 '배포된 스마트 컨트랙트를 모두 나열하는 프런트엔드'의 3가지 개별 작업 부분으로 구성됩니다.

- 사전 요구 사항 **&** 추가 리서치:
    - 감사 정보를 체인으로 푸시 하는 것은 MSIG에 의해 통제되어야 하며, 이는 감사인이 제공한 정보를 체인으로 푸시 하기 전에 해당 정보를 검증할 수 있도록 해야 합니다. 작업 MVP를 제공하기 위해 이것이 필수는 아니지만, 저희는 이 주제를 연구하는 워킹 그룹을 제안합니다.
    - API 표준에 대한 연구 및 정의 확립
    - ABI 표준에 대한 연구 및 정의 확립
    - 이 정보를 커뮤니티에 제공하는 것과 관련된 법적인 의미를 조사해야 합니다. 만약 애플리케이션들이 API 소비자에 의해 안전한 것으로 표시되었지만, 실제로는 해당 애플리케이션이 손상된 경우, API 소비자에 대해서는 법적으로 어떻게 해석되는지 확인해 볼 필요가 있습니다. 여러 번 감사를 진행하더라도 컨트랙트는 결코 완벽하게 안전하다고 간주될 수 없으므로, 항상 사용자들이 손해를 볼 위험이 존재한다는 것을 유의하는 것은 매우 중요합니다.

- 예상 소요 시간: 1,280 시간
- 예상 소요 비용: 95,500 달러
- 필요 전문 인력:
    - 스마트 컨트랙트 개발자 - 280 시간 소요 @ 시간당 $100 ($28,000)
    - 프론트 엔드 디자이너 - 350 시간 소요 @ 시간당 $75 ($26,250)
    - 풀스택 개발자 - 350 시간 소요 @ 시간당 $75 ($26,250)
    - 프로젝트 매니저- 300 시간 소요 @ 시간당 $50($15,000)

F) 최소 실행 가능 제품 **(MVP)**

여기서 제안된 MVP는 매우 작은 스마트 컨트랙트 세트를 기반으로 합니다. 이를 제품에서 사용되도록 하기 위해 우리는 우선 체인으로부터 모든 스마트컨트랙트 ABI 및 현재 감사 내역들을 불러오기 위한 신뢰할 수 있는 방법을 구축할 필요가 있습니다.

1. **API** 표준 생성**: (**풀스택 개발자 **- 20** 시간**)**
    a. github Repo 만들기
    b. JSON 형식으로 API 표준 정의[5]
2. **ABI** 표준 생성**: (**스마트 컨트랙트 개발자 **- 20** 시간**)**
    a. 프로젝트 정보를 제공하기 위해 개발자가 따라야 할 ABI 표준 정의
3. **ABI** 형식을 사용하여 테스트 컨트랙트 작성 및 체인상 배포**: (**스마트 컨트랙트 개발자 **- 25**시간**)**

   a. ABI 표준을 사용하여 3가지 더미 스마트 컨트랙트 생성 후 체인상 배포
4. 체인 감사 테이블 생성 및 일부 테스트 데이터 게시 **(**스마트 컨트랙트 개발자 **- 25** 시간**)**
   a. 감사인이 결과를 게시할 수 있는 계정 및 온체인 테이블 생성
   b. 일부 감사 결과들을 게시하기 위해 보안 엔지니어들과 협력
5. **API** 개발**: (**풀스택 개발자 **- 50** 시간**)**
   a. DB 설정 및 DB 레이아웃 정의
   b. API 백엔드 설정
   c. API 경로 생성
6. **Hyperion**에서 데이터를 불러오기 위한 **MVP**용 **Write** 함수**(**풀스택 개발자 **- 100** 시간**)**
   a. 관련된 더미 감사 내역 및 DB에 대한 포스트를 포함하는 더미 컨트랙트들을 위해 Hyperion ABI 상세정보를 가져오는 Write 함수
7. 프론트엔드 개발 및 디자인**: (**프론트엔드 디자이너 **- 250**시간**,** 풀스택 개발자 **- 100** 시간**)**
   a. API로부터 데이터를 검색하는 최소 실행 프런트엔드 구성
   b. 스마트 컨트랙트 및 사용 가능한 모든 API 데이터를 표시
8. 도커 제작**: (**풀스택 개발자 **- 50** 시간**)**
   a. 간편한 배포 및 테스팅을 위한 프론트엔드, 백엔드 및 DB 인스턴스 도커라이즈

F) 최소 실행 가능 제품**+**

최소 실행 가능 제품의 모든 MPV 부품을 포함하며, 이 제품을 생산 단계로 전환하는 데 필요한 단계 또한 포함할 것입니다.

1. 히스토리 솔루션**:**
   a. 완료된 보안감사 컨트랙트의 컨트랙트 델타 및 DB에 저장될 수 있는 스마트 컨트랙트 ABI를 하나로 묶는 솔루션
   b. 이 솔루션은 어떠한 컨트랙트 ABI도 누락되지 않도록 프로세스가 마지막으로 멈춘 곳에서 시작하고 멈출 수 있어야 합니다.

G) 향후 목표

1. EOSIO 컨트랙트의 감사인(Auditor)들에게 이러한 유형의 정보 제공에 대한 인센티브가 주어져야 하며, 스마트 컨트랙트 개발자들 또한 컨트랙트의 일부로서 해당 정보를 요청해야 하므로 이러한 이니셔티브에 대한 교육 및 마케팅이 요구될 수 있습니다.
2. 히스토리 솔루션 사용 시 DB 손실이 발생한 경우, 감사 스마트 컨트랙트 테이블 및 컨트랙트 ABI 내의 온체인 상에 나열된 내용으로 API 데이터베이스를 재작성/동기화하는 방법

H) 참고 자료

- [1]Certik has a list of all their audits completed - https://www.certik.com
- [2]https://github.com/ethereum-lists/contracts

- [3]https://sourcify.dev
- [4]https://developers.eos.io/welcome/v2.1/smart-contract-guides/understanding-ABI-files
- [5]https://github.com/eosrio/bp-info-standard

2. 컨트랙트 업그레이드 승인 **DAO**

워킹 그룹	Audit+
목표 대중	EOSIO 커뮤니티
이니셔티브 유형	개발, 디자인, 리서치

A) 문제

우리 모두가 알고 있듯이 EOS 컨트랙트는 일반 EOS 계정들에 배포될 수 있으며 기본적으로 소유자는 원하는 대로 컨트랙트를 변경할 수 있는 모든 권한을 가지고 있습니다. 이로 인해 스마트 컨트랙트 소유자와 제품 사용자 간의 신뢰 문제가 발생합니다. 사용자들은 어떻게 하면 소유주가 갑자기 컨트랙트를 검증되지도 않고 감사되지도 않은 버전으로 변경하거나 최악의 경우, 컨트랙트에 잠겨있는 자금을 모두 가져가지 않을 것이라는 것을 알 수 있을까요? 우리는 개발자들이 컨트랙트를 업그레이드할 수 있는 선택권을 허용하는 동시에 이러한 신뢰를 만들 수 있는 방법을 찾아야 합니다.
스마트 컨트랙트의 업그레이드 권한을 관리하기 위해서는 일종의 다중 서명 DAO가 필요합니다.

B) 목표

EOS 컨트랙트 업그레이드 처리를 담당하는 하나 이상의 컨트랙트 권한 관리 DAO 구축 :

dApp 계정의 소유자 및 활성 키 권한을 관리하기 위한 DAO 스마트 컨트랙트 구축.

반드시 단일 주체가 이를 처리해야 하는 것은 아니며, 이러한 유형의 서비스는 감사 회사가 그들이 수행하는 철저한 감사들을 통해 시간이 지남에 따라 개발자들의 신뢰를 얻는 것과 마찬가지로, 이러한 비즈니스의 종류들 또한 이들이 수행하는 스마트 컨트랙트 검증 및 업그레이드 관련 작업들을 기반으로 시간이 지남에 따른 신뢰를 확보해야 합니다.

이 이니셔티브에 관련된 이해관계자의 설명으로 인해 이 주제에 대해 더 많은 연구가 필요하지만 이러한 시스템이 작동하는 방법의 예는 아래에 나열되어 있습니다.

DAO의 예**:**

Audit+ 청서

DAO XYZ는 스마트 컨트랙트 업그레이드 기능을 제공합니다. DAO는 5 x top21 BP, 보안 감사인 및 잘 알려진 EOSIO 트위터 사용자로 구성되어 있습니다.

Jolly Defi는 EOS 메인넷에 가입하여 DAO XYZ의 서비스를 사용하기로 결정하고, 다른 외부 보안 감사관이 감사한 초기 컨트랙트를 설치하고 체인상에서 이를 실행한 후, 그들의 소유자와 활성 키를 6/8 MSIG로 업데이트합니다.

MSIG 설정

1. Top21 BP
2. Top21 BP
3. Top21 BP
4. Top21 BP
5. Top21 BP
6. Security Auditor
7. EOSIO twitter user
8. Jolly Defi

DAO XYZ 회사는 체인 활동을 모니터링하고 Jolly Defi가 그들을 MSIG로서 추가 및 DAO XYZ에 필요한 정보를 보낸 사실을 인지합니다.

a) 컨트랙트 세부 정보 **-** 컨트랙트, 코드 웨어하우스, 컴파일러 버전 및 보안 감사 링크.
b) 클라이언트 세부정보 **-** 애플리케이션 웹사이트, 코드 웨어하우스, 링크드인 프로파일 및 기타 사항 (필요한 경우)

DAO XYZ는 보안 감사의 해시가 체인에 로드된 컨트랙트와 일치하는지 확인하고 모든 것이 확인되면 승인된 애플리케이션 목록에 Jolly Defi를 추가합니다.

현재 Jolly Defi를 사용하는 모든 사용자는 먼저 DAO XYZ를 방문하여 Jolly Defi가 실제로 DAO XYZ에 의해 관리되는 것이 맞는지 여부 및 모든 사항을 확인할 수 있습니다. 이러한 사항이 블록 탐색기에 통합된다면, 이는 매우 이상적일 수 있습니다.

3개월 후 Jolly Defi는 자체 성능 향상을 위해 컨트랙트를 업그레이드하고자 합니다. 이후 그들은 재확인을 위해 모든 관련 정보를 DAO XYZ에 보내고 MSIG를 시행하여 컨트랙트를 업그레이드합니다.

DAO XYZ가 확인 작업을 수행한 후, 서명 및 MSIG를 실행하게 되면 새 컨트랙트가 체인상에 업로드됩니다.

C) EOSIO에 대한 이점

Audit+ 청서

표준화된 방법을 통해, 스마트 컨트랙트의 다중 서명은 개발자가 다중 서명을 사용하는 임곗값을 줄이고 스마트 컨트랙트에 대한 EOS 사용자의 신뢰를 향상시키는 관례적인 방법이 되었습니다.

D) 산출물

EOSIO 이해관계자가 동의하는 DAO MSIG 프로토콜을 설정합니다. 이러한 유형의 서비스를 제공하려는 모든 DAO는 비즈니스를 설정하기 위해 이를 따를 수 있습니다.

E) 노력 수준

- 예상 시간: 400 시간
- 예상 비용: 40,000 달러
- 필요 전문 인력:
    - 연구원 - 400시간 @ 시간당 $100($40,000)

1. 이 주제에 대한 심층 리서치 및 이러한 유형의 서비스에 대한 DAO 프로토콜 구축 (연구원 - 350 시간)
    a. 여러 체인에 걸쳐 EOSIO 이해관계자와 함께 본 주제에 대한 리서치 시행

2. DAO MSIG 프로토콜 관련 문서 제작 (연구원 - 50 시간)

F) 비즈니스 기회

이러한 서비스는 유료여야 하고 서비스 제공자들은 서로 경쟁해야 합니다.

미래)

향후 DAO는 다음과 같은 추가 검사를 포함하도록 확장하여 dApp에 등급 시스템을 추가할 수 있습니다.

해당 컨트랙트는 책임을 결정하기 위한 Ricardian 컨트랙트들을 제공
컨트랙트가 오픈 소스이고 커뮤니티에 의해 검증/감사가 가능한지 여부

G) 최소 실행 가능 제품

컨트랙트 소스 코드 확인 및 컨트랙트 업그레이드 승인을 담당하는 DAO 구축합니다.

H) 참고 자료

없음.

3. AML 플랫폼 구축

A) 문제

신기술 플랫폼으로서의 블록체인은 개방성과 익명성이라는 특징을 가지고 있습니다. 동시에 블록체인에는 다수의 가치 있는 자산이 유통되고 있기에 해커들의 주요 공격 타깃 중 하나가 되고 있습니다. SlowMist 해킹 기록에 따르면 현재까지 547건의 해킹 사고로 인해 총 20,779,621,384 달러 상당의 손실이 발생한 것으로 알려졌으며, 이 중 120건은 EOS 공격으로 인해 다수의 EOS 자산이 도난당한 사건들입니다. 해커들이 자금을 빼돌린 뒤 거래소로 이체해 매각하거나 다른 자산으로 교환하는 경우가 많이 존재합니다. 다만 거래소는 관련 공격 정보를 제때 얻지 못하고 해커의 거래를 막지 못해, 거래소가 해커의 돈 세탁 공범이 되는 결과를 초래하기도 했습니다. 사실, 이는 단순히 거래소에만 해당하는 문제는 아닙니다. EOS 생태계의 모든 월렛, 브라우저 및 디앱은 국제적인 반부패 활동을 준수함과 동시에 악성 행위를 줄이고, 돈 세탁의 공범이 되는 것을 방지하기 위해 악성 행위와 거래 위험을 평가해야만 할 수도 있습니다. 자금 세탁 및 대테러 자금조달 방지에 관한 규정을 생각해 볼 수 있습니다.
EOS는 TPS가 높기 때문에 과거 거래 기록이 엄청나게 큰 저장 공간을 차지하는데, AML 시스템의 핵심 기술 중 하나가 과거 거래 내역을 분석하는 것이기 때문에, 이 같은 시스템 구축에는 고비용 하드웨어 장비에 대한 투자가 요구됩니다. 이는 또한 AML 시스템의 발전을 방해하는 요소입니다.

B) 목표

EOS에 기원한 자금 세탁 방지 시스템을 구축하고, 정리 및 통합을 위해 다양한 데이터 소스(다크 웹에서부터 전 세계 수백 개의 거래소까지 이르는 모든 콘텐츠, EOS 커뮤니티에서 제공된 지식 포함)로부터 데이터를 수집하고, 여기에 인공지능 기술을 결합하여 대용량 데이터로부터 정확한 데이터들을 축출하고, 계정에 신원 특성을 표시하고, 실시간으로 자산 이체 내역을 모니터링합니다. 사용자 또는 파트너는 인터페이스를 통해 계정 또는 트랜잭션에 악의적인 행위가 포함되어 있는지를 확인할 수 있습니다.

C) EOSIO에 대한 이점

1. 해킹당한 프로젝트 당사자가 자금을 회수할 수 있도록 지원하는 등 체인상 악성 행위가 줄어들 수 있도록 EOS 거버넌스를 위한 데이터 제공
2. EOS 거래소 또는 월렛이 자금 이체를 감시하고 불법 자금을 적시에 동결할 수 있도록 지원
3. 규제 기관에 법 집행의 근거 제공, 자금 세탁 행위 분석 및 EOS의 규정 준수를 위한 발전 촉진

D) 산출물

1. 자산 이전 경로를 보여주는 시각적 차트를 제공하는 웹사이트;
사용자는 차트상의 자산 이전 상태를 보기 위해 프론트엔드의 검색 상자에 계정 번호를 입력
2. 계정 태그 쿼리를 위한 API 인터페이스
API JSON 형태 예시:

```
{
        "lable": "eos project exploit",
        "transactions": [
                "0x123456789...",
                "0x123456789...",
                "0x123456789...",
        ],
        "accounts": [
                "eoshacker...",
                "eoshacker...",
        ]
}
```

E) 노력 수준

전제 조건:
1. 프로젝트 당사자는 특정 블록체인 인텔리전스 데이터를 가지고 있어야 합니다
2. EOS 기록 노드 데이터를 빠르게 다운로드해야 합니다.

예상 시간**: 1200** 시간
예상 비용**: $114,000**
필요 전문 인력**:**

- 프론트엔드 디자이너 - 240 시간 @ 시간당 $75($18,000)
- 풀 스택 엔지니어 - 480 시간 @ 시간당 $100($48,000)
- 알고리즘 엔지니어 - 480 시간 @ 시간당 $100($48,000)

F) 최소 실행 가능 제품

1. 트랜잭션 데이터 처리(풀 스택 엔지니어 - 240 시간)
    a. EOS 전체 히스토리 노드 구축
    b. 모든 전송 정보를 관계형 데이터베이스에 기록

2. 정보 수집 및 처리(전체 스택 엔지니어 - 240 시간)
    a. 해커 공격 이력과 관련된 계정 수집
    b. dApp 및 거래소 계정 같은 가능한 한 많은 식별 가능한 계정 수집

3. 계정 연관 알고리즘 개발(알고리즘 엔지니어 - 480 시간)
    a. 대상 계정의 연관 계정 및 자금 관계 정렬
    b. 데이터 쿼리 인터페이스 생성

4. 디자인 웹 UI(프론트 엔드 디자이너 - 240 시간)
    a. 계정 표시 정보 표시

G) 향후 목표

1. AML 프로젝트 당사자는 악성 행위 식별의 정확성을 지속적으로 개선하고 시스템의 신뢰성과 사용성을 향상시켜야 합니다;
2. EOS 월렛, 브라우저, 트랜잭션 및 기타 플랫폼은 AML 시스템을 사용하여 체인 상의 악성 행위를 줄이도록 권장되어야 합니다.

H) 참고 자료

- **Etherscan Label Word Cloud:** https://etherscan.io/labelcloud
- **SlowMist AML:** https://aml.slowmist.com/
- **Chain Analysis:** https://www.chainalysis.com/

4. 안전한 스마트컨트랙트 개발을 위한 소프트웨어 라이브러리

워킹 그룹	Audit+
목표 대중	개발자
이니셔티브 유형	개발, 문서화

A) 문제

디파이 프로토콜이 수십억 달러의 자산을 관리하는 상황에서 블록체인 커뮤니티는 스마트 컨트랙트 구축 시 보안을 더 이상 간과할 수 없습니다. 공격의 위험이 너무 높습니다. 보안 취약점의 위험을 줄이는 데 감사와 보안 운영이 핵심이긴 하지만, 그것만으로는 충분하지 않습니다. 안전한 스마트 컨트랙트 시스템을 개발하고 출시할 때 누락되는 기본 구성요소가 존재합니다: 개발 내 보안 통합

B) 목표

경험이 풍부한 개발자를 고용하고, 일반적으로 사용되는 몇몇 스마트 컨트랙트 템플릿을 개발하여 커뮤니티의 개발자들에게 공개합니다.

C) EOSIO에 대한 이점

경험이 부족한 개발자라도 개발 사례 활용 혹은 학습을 통해 안전한 스마트 컨트랙트를 개발할 수 있게 되어, 해커들의 공격이 줄어듭니다.

D) 결과물

개발 템플릿 및 널리 사용되는 6개 비즈니스 구현 코드를 오픈 소스 형태로 구축하고, 2곳 이상의 유명 보안 회사에 대한 보안 감사를 통과합니다.

E) 노력 수준

- 전제 조건**:**

- 다른 블록체인 생태계의 스마트 컨트랙트 구현 사례를 조사합니다.

예상 시간**: 660** 시간
예상 비용**: $170,000**

- 스마트 컨트랙트 엔지니어 - 1000 시간 @시간당 $100($100,000)
- 스마트 컨트랙트 보안 엔지니어 - 350 시간 @시간당 $200($70,000)

F) 최소 실행 가능 제품

1. 일반적인 애플리케이션 유형에 대한 개발 템플릿 생성**(**스마트 컨트랙트 엔지니어 **- 1000** 시간 **)**
    a. 스왑 프로토콜 컨트랙트
    b. Oracle 프로토콜 컨트랙트
    c. NFT 표준 컨트랙트
    d. Flashloan 프로토콜 컨트랙트
    e. 스테이블 코인 프로토콜 컨트랙트
    f. 랜딩 프로토콜 컨트랙트
2. 컨트랙트에 대한 보안 감사 수행
    a. 모든 개발 템플릿에 대해 두 개의 독립적인 감사가 완료되어야 함(스마트 컨트랙트 보안 엔지니어 - 350 시간)

G) 향후 목표

- 에코 시스템의 지속적인 성장을 위한 더 많은 개발 템플릿 생성

H) 참고 자료

- https://github.com/OpenZeppelin/openzeppelin-contracts
- https://github.com/open-web3-stack/open-runtime-module-library
- https://github.com/solana-labs/solana-program-library

5. 버그 바운티

워킹 그룹	Audit+
목표 대중	해커 커뮤니티
이니셔티브 유형	개발, 문서화

A) 문제

현재 EOSIO에는 해커 커뮤니티가 EOSIO 코드 베이스 분석을 위한 시간을 투자하도록 유도하는 잘 정의된 버그 바운티 프로그램이 없습니다.

거의 모든 최고급 블록체인들은 버그 바운티 프로그램을 운영중이며, 이들은 모두 보상 측면에서 해커 커뮤니티로부터 최고의 관심을 끌기 위한 경쟁력이 여전히 존재합니다.[1]

B) 목표

해커 커뮤니티로부터 최고의 인재들을 끌어들이기 위해 매력적인 보상이 적절하게 설정된 잘 관리되고 정의된 버그 바운티 프로그램을 마련합니다. 이는 최고의 인재를 유치할 뿐만 아니라 프로그램이 장기간 유지될 수 있도록, 잘 정의되고 관리되어야 합니다.

우리는 또한 버그가 공개되기 전에 적시에 업데이트가 진행될 수 있도록 보장하기 위해, 모든 EOSIO 기반 체인들에 높은 취약성 버그들을 전달하는 방법이 필요합니다. [2]

C) EOSIO에 대한 이점

버그 바운티 플랫폼이 잘 관리될 경우, EOSIO는 최고의 해커들이 EOSIO 코드 베이스를 지속적으로 평가하도록 끌어들일 수 있습니다. 이를 통해 전체 EOSIO 블록체인들을 더 안전하게 보호할 수 있을 뿐만 아니라, 그 위에서 실행되는 스마트 컨트랙트들 또한 보호할 수 있는 전반적으로 더 안전한 코드 케이스가 보장됩니다. 해커가 발견한 EOSIO 취약점이 포함된 라이브러리를 이용하는 다른 소프트웨어에 해당 취약점이 영향을 줄 경우, 이 서비스와 연계된 마케팅 요소 또한 존재합니다. [3]

D) 결과물

1. 잘 정의된 버그 바운티
    a. 바운티와 관련하여 잘 정의된 취약성 분류 및 심각도 범위
    b. 무엇이 버그를 구성하는지에 대해 잘 정의된 규칙
    c. 버그를 제출하는 방법에 대해 잘 정의된 절차
    d. 재단과 해커/보안 커뮤니티 간의 커뮤니케이션을 적시에 보장할 수 있는 잘 정의된 커뮤니케이션 방법론
2. 버그 바운티 프론트엔드
    a. 가능한 EOSIO 버그 바운티, 리더보드, 바운티 정의 및 버그를 제출하는 방법을 보여주는 해커/보안 커뮤니티용 웹 프론트 엔드 + 모바일 친화적인 플랫폼
3. 취약성이 높은 버그를 전달하는 방법
    a. 다른 EOSIO 체인들과 함께 취약성이 높은 버그를 가장 잘 관리하는 방법에 대한 조사

E) 노력 수준

이를 실현하기 위한 노력은 매력적인 웹사이트 및 UX와 함께 매우 잘 정의된 버그 바운티를 통해 구성될 것입니다. 보안 커뮤니티를 위한 간단한 버그 제출 방법을 구현하고자 합니다. 추가 연구가 필요한 또 다른 중요한 측면은 다른 모든 EOSIO 체인에 취약성이 높은 버그를 전달하는 방법입니다.

- 필수 조건**:**
    - 재단 및 기타 EOSIO 체인은 각 레벨에 해당하는 심각도를 포함한 버그 바운티 레벨 및 관련 바운티를 결정해야 합니다. 이 이니셔티브는 모든 체인에 적용되기 때문에, 이 서비스의 운영 비용은 함께 부담해야 합니다.
    - 버그 바운티 사이트에는 EOSIO의 전담 핵심 개발팀이 있어야 제출된 보고서들을 평가하고, 버그 바운티 수준을 결정하고, 필요한 경우 보안 업그레이드에 대한 구성이 가능합니다.
    - 핵심 EOSIO 개발자들은 버그 바운티 프로그램을 관리하기 위한 동의가 필요할 것입니다.

- 버그 바운티 기금**:**
    - MVP 상품 외에도 보안 커뮤니티에 의해 발견된 어떠한 버그들 또한 보상이 따를 수 있습니다. 지급될 보상금에 대한 예산을 쉽게 예측할 수 있는 방법이 존재하지 않습니다..
- 예상 시간**:** 260 시간
- 예상 비용**:** $16,625 + 버그 바운티 기금
- 필요 전문 인력**:**
    - 풀스텍 개발자 - 105 시간 @ 시간당 $75($,7875)
    - 프로젝트 매니저 - 5 시간 @ 시간당 $50($250)
    - 콘텐츠 작가 - 50 시간 @ 시간당 $15($750)
    - 스마트 컨트랙트 보안 엔지니어 20 시간 @ 시간당 $200($4,000)
    - 프론트엔드 디자이너 - 50 시간 @ 시간당 $75($3,750)

F) 최소 실행 가능 제품

1. 디자인 프론트엔드**(**프론트엔드 디자이너 **- 50** 시간**)**
    a. 디자인 프론트엔드 UX

2. 버그 바운티 정의**(**콘텐츠 작가 **- 50** 시간**,** 스마트 컨트랙트 보안 엔지니어 **20** 시간**)**

    a. 각 수준에서 버그를 구성하는 예를 포함하여 버그 바운티 수준 및 관련 바운티 정의
    b. 각 수준에 해당하는 심각도 정의
3. 높은 취약점 버그 전달 방법**(**프로젝트 매니저 **- 5** 시간 **)**
    a. 다른 EOSIO 체인과의 미팅을 구성하여, 높은 취약성의 EOSIO 버그 관리 방법에 대한 연구 시작
4. 프론트엔드 개발**(**풀스택 개발자 **- 75** 시간**)**
    a. 버그 바운티와 관련된 모든 정보를 나열하는 단일 페이지 제작
    b. 최고의 버그 바운티 헌터들을 나열하는 리더보드 페이지 제작
5. 도커**: (**풀스택 개발자 **- 20** 시간**)**
    a. 쉬운 배포 및 테스팅을 위한 프론트엔드 도커나이즈
6. 버그 제출 양식**(**풀스택 개발자 **- 10**시간**)**
    a. 버그를 제출하기 위한 간단한 Google 양식

G) 향후 목표

- 버그 바운티 프로그램을 확장하여 EOSIO의 모든 디앱이 자체 미니 버그 바운티를 생성할 수 있도록 합니다. 이를 통해 보안 연구원들은 발견된 취약점을 적극적으로 보완할 수 있습니다.

H) 참고 자료

- [1]*https://ethereum.org/en/eth2/get-involved/bug-bounty/#rules*
- *https://guidovranken.com/notable-vulnerabilities-found-in-high-profile-software/*
- [2]*https://twitter.com/etherchain_org/status/1431256423489495045?s=20*
- [3]*https://www.telos.net/news/telos-evm-uncovers-ethereum-vulnerability*

6. 자동화된 보안감사 도구 구축

워킹 그룹	Audit+
목표 대중	개발자
이니셔티브 유형	개발, 문서화

A) 문제

현재 EOSIO 커뮤니티에는 개발자가 자유롭게 사용할 수 있는 스마트 컨트랙트의 기본 보안 평가 도구가 없습니다. 일부 도구가 존재하지만 오픈 소스가 아닐뿐더러, 무료도 아닙니다.[1]

B) 목표

누구든지 EOSIO 용으로 작성된 오픈 소스 스마트 컨트랙트에 대해 도구를 실행할 수 있고, 현재 잘 알려진 취약점에 대해 컨트랙트가 어떻게 진행되는지에 대한 보고서를 제공받을 수 있는, 적극적으로 유지관리되는 오픈 소스 도구를 만드는 것을 제안합니다.[2][3]

도구는 무료여야 하며 설치가 쉽고 결과를 신속하게 제공할 수 있어야 합니다.

제공된 결과는 명확해야 하며 개발자가 이러한 문제를 해결하는 방법을 이해하는 데 도움이 되는 참조 자료가 제공되어야 합니다. 이상적으로는 EOSIO 스마트 컨트랙트 작성 시 일반적인 보안 문제 목록이라는 또 다른 이니셔티브와 다시 연결되어야 합니다. 이는 문서가 흩어지는 것을 방지하기 위해 공통 지식 기반을 보장할 것입니다.

C) EOSIO에 대한 이점

이러한 도구를 사용할 수 있게 되면 자가 진단을 촉진하고 방지해야 할 일반적인 보안 문제에 대한 개발자 커뮤니티의 지식을 향상시킬 수 있습니다. 장기적으로 이를 통해 개발자 및 보안 커뮤니티 간에 EOSIO 보안에 대한 지속적인 자가발전을 이룩할 수 있습니다.

D) 결과물

1. 자동화된 보안 감사 도구**.**
    a. 용이한 기여를 위해 python 형태의 모듈식 보안 감사 도구를 구축해야 합니다.
    b. 도구는 문제 해결 방법에 대한 보고 및 참조 지식 기반 제공해야 합니다.
    c. 도구는 설치가 쉬워야 하며 설치 과정이 문서화되어야 합니다.

d. 도구는 스마트 컨트랙트를 분석하고 일반적으로 알려진 취약점에 기초하여 컨트랙트의 이용 가능 영역을 식별해야 합니다.

E) 노력 수준

이 툴에 대한 최선의 접근 방식을 위한 사전 계획이 요구되므로, 이 툴과 관련된 노력의 수준은 다른 이니셔티브보다 상당히 높습니다.

- 필수 조건**:**
  - 외부 기여와 지속적인 개발을 위해 모듈식 기본 도구를 만드는 최선의 방법에 대한 추가 연구가 이루어져야 합니다.
  - 이상적으로는 이 프로젝트를 시도하기 전에 ***EOSIO*** 스마트 컨트랙트 작성 시 일반적인 보안 문제 목록이라는 이니셔티브가 완료되어야 합니다.
- 예상 시간**:** ~1,550 시간
- 예상 비용**:** $152,500
- 필요 전문 인력:
  - Python 개발자 - 1,050 시간 @시간당 $50 ($52,500)
  - 스마트 컨트랙트 보안 엔지니어 - 500 시간 @ 시간당 $200 ($100,000)

F) 최소 실행 가능 제품

1. 자동화된 보안 감사 도구. (Python 개발자 - 1,000 시간, 스마트 컨트랙트 보안 엔지니어 500 시간)

   a. Python을 이용하여 기본 보안 감사 도구를 구축합니다.
   b. 도구는 추가 보안 검사를 추가하기 위해 외부 기여가 가능한 모듈식이어야 합니다.
   c. 설치 지침은 쉽게 따라 할 수 있어야 합니다.
   d. 도구는 반드시 Git에서 설정해야 합니다

2. 보고 **(Python** 개발자 **- 50** 시간**)**
   a. 도구는 기본적인 보고를 제공해야 하며 일반적인 보안 문제 목록을 참조하는 링크를 포함해야 합니다.

H) 참고 자료

- [1]***https://klevoya.com/inspect/index.html***
- [2]https://github.com/crytic/slither
- [3]https://github.com/ConsenSys/mythril

7. EOSIO 스마트 컨트랙트 작성 시 일반적인 보안 문제 목록

워킹 그룹	Audit+
목표 대중	개발자
이니셔티브 유형	문서화

A) 문제

스마트 컨트랙트 작성 시 공유 지식 기반에 대한 적절한 액세스를 통해 가장 일반적인 보안 문제들을 가장 쉽게 방지할 수 있습니다. 이러한 유형의 실수들에 대한 실제 문서는 존재하지 않습니다.

이는 다른 주요 블록체인의 매우 일반적인 기능으로, 우리는 이러한 요구 사항을 충족시켜야 한다고 생각합니다.[1]

이에 대한 몇 가지 예는 EOSIO에서 확인할 수 있습니다.

https://github.com/slowmist/eos-smart-contract-security-best-practices/blob/master/README_EN.md

https://cc32d9.medium.com/eosio-contract-security-cookbook-20210527-69797efe9c96

B) 목표

개발자가 스마트 컨트랙트를 작성할 때 능동적으로 유지된 일반적인 보안 문제 목록을 참조합니다. [2]

C) EOSIO에 대한 이점

이러한 유형의 문서는 개발자가 스마트 컨트랙트를 개발할 당시, 그들에게 EOSIO 상의 최신 위협에 대해 다시 한번 상기시킬 수 있다는 이점을 가지고 있습니다.

D) 결과물

1. EOSIO 스마트 컨트랙트 작성 시 일반적으로 발생하는 보안 문제 목록

E) 노력 수준

이 이니셔티브는 EOSIO 스마트 컨트랙트 작성 시 가장 일반적인 보안 문제를 나열하고 버전 추적이 가능한, 능동적으로 유지 보수될 수 있는 문서 또는 위키를 제공하는 것입니다.

- 전제 조건**:**
  - 해당 없음
- 예상 시간**:** ~455 시간
- 예상 비용: $54,000
- 필요 전문 지식**:**
  - 스마트 컨트랙트 보안 엔지니어 - 255 시간 @ 시간당 $200($51,000)
  - 콘텐츠 작가 - 200 시간 @ 시간당 $15($3,000)

F) 최소 실행 가능 제품

1. 일반적인 보안 문제**: (**스마트 컨트랙트 보안 엔지니어 **- 250** 시간**,** 콘텐츠 작가 **- 150** 시간**)**
   a. 다음을 포함한 알려진 10가지 일반적인 보안 문제 목록 작성
      i. 공격 유형에 대한 설명
      ii. 실수가 발생한 예제 소스 코드
      iii. 이로 인해 발생할 수 있는 취약점
      iv. (가능한 경우)실제 사례

2. **GIT**과 같은 분산 버전 제어 시스템 상의 구축 **(** 스마트 컨트랙트 보안 엔지니어 **- 5** 시간**,** 콘텐츠 작가 **- 50** 시간 **)**
   a. 버전 관리를 위한 Git Repo 설정
   b. Wiki와 같은 역할을 하는 Gitbook 설정

G) 참고 자료

- [1]**https://blog.neodyme.io/posts/solana_common_pitfalls**
- [2]https://github.com/sigp/solidity-security-blog
- https://cc32d9.medium.com/eosio-contract-security-cookbook-20210527-69797efe9c96
- https://github.com/slowmist/eos-smart-contract-security-best-practices/blob/master/README_EN.md

5. 재단에 권고하는 최선의 행동 방침 목록

여기서 제안하는 4가지 이니셔티브는 EOSIO가 요구하는 것뿐만 아니라 블록체인 영역의 다른 커뮤니티들이 요구하는 것을 중심으로 한 저희의 모든 연구를 바탕으로 합니다. 권고안 1-3은 모든 새로운 블록체인이 보안 관점에서 구축할 수 있는 건전한 기반을 촉진하기 위해 채택하려고 시도하는 기본 요구 사항을 구성합니다.

권고안 4는 우리가 이더리움에서 발견한 비교적 새로운 제안입니다. 이러한 접근 방식을 채택하는 것에 있어, EOSIO는 매우 유리할 것입니다. EOSIO 핵심 기술은 이러한 유형의 솔루션을 쉽게 적용할 수 있도록 하는 몇 가지 뚜렷한 장점을 가지고 있습니다. 큰 사업이기는 하지만 모든 워킹 그룹의 여러 이해관계자들이 시스템의 기본 신뢰에 큰 영향을 미칠 최상의 접근 방식을 결정해야 합니다.

1. EOSIO 스마트 컨트랙트 작성 시 일반적인 보안 문제 목록

당위성 **:**

이는 매우 쉽게 성취할 수 있으며, 개발자 커뮤니티가 강력하고 안전한 스마트 컨트랙트를 만들도록 돕는 좋은 시작점이 될 수 있습니다. 또한 자동화된 보안 도구 세트와 같은 향후의 도구화를 위한 기반 역할을 합니다.

비용 예측**:**

MVP 비용**:**

- 비용: 초기 MVP는 22,500 달러 상당의 비용이 소요됩니다.

지속적인 비용**:**

- 비용: 해당 목록이 최신 상태로 유지되기 위해 연간 22,500 달러 상당의 비용이 소요됩니다.

해당 목록이 계속 업데이트되도록 하기 위해 지속적인 비용이 발생할 것입니다. 리스트가 오픈소스이기 때문에 일부 작업은 개발자 커뮤니티에 의해 진행될 수 있지만, 재단은 연 1~2회, 전담 전문가를 통해 리스트가 확실히 업데이트되도록 해야 합니다.

2. 안전한 스마트 컨트랙트 개발을 위한 소프트웨어 라이브러리

당위성 **:**

이는 개발자 커뮤니티의 필수적인 부분이며 일반적인 보안 문제 목록 이니셔티브와 유사합니다. 안전한 스마트 컨트랙트 작성과 관련된 모범 사례를 구축하는 데도 도움이 됩니다.

비용 예측**:**

MVP 비용**:**

- 비용: 초기 MVP는 152,500 달러 상당의 비용이 소요됩니다.

지속적인 비용**:**

- 비용: 도구를 최신 상태로 유지하기 위해 연간 $50,000 상당의 비용이 소요됩니다.

개발자 커뮤니티에 의해 도구가 최신 상태로 유지되는 것이 이상적이지만, 재단은 도구를 최신 상태로 유지하기 위해 연 1~2회, 전담 전문가를 통해 도구의 최신화 작업을 수행해야 합니다.

3. 버그 바운티

당위성 **:**

보안 관점에서 기본 EOSIO 핵심 소프트웨어가 견고하게 유지되도록 외부 보안 커뮤니티가 EOSIO 코드 베이스를 지속적으로 테스트하도록 유도해야 합니다.

비용 예측**:**

MVP 비용**:**

- 비용: 초기 MVP는 16,625 달러 상당의 비용이 소요됩니다.

지속적인 비용**:**

- 비용: 알 수 없음

버그 바운티 관리는 B1과 재단과는 논외인 핵심 EOSIO 개발자들의 동의가 필요하기 때문에 플랫폼의 지속적인 유지보수를 위한 예상 비용은 제공할 수 없습니다.

또한, 발견된 보안 버그에 대한 보상 비용도 발생하는데, 이는 다시 예측할 수 없는 비용이기 때문에, 여기서의 발생 비용은 알 수 없는 상태로 남아 있습니다.

4. 오픈 소스 보안 감사 API 및 플랫폼

당위성 **:**

대부분의 블록체인들이 보유하고 있지 않은 강력한 보안 플랫폼을 구축하여 사용자들이 디앱들과 상호작용할 때, EOSIO가 그들이 마음에 안도감을 줄 수 있는 최고의 플랫폼이 될 수 있도록 합니다.

비용 예측**:**

MVP 비용**:**

- 비용 : 초기 MVP는 54,750 달러 상당의 비용이 소요됩니다.

지속적인 비용**:**

- 비용: 플랫폼을 최신 상태로 유지하기 위해 매년 50,000 달러 상당의 비용이 소요됩니다.

플랫폼이 구축되면 대부분의 작업이 완료되겠지만, EOSIO 생태계의 요구사항에 맞게 다양한 플랫폼과 표준들을 업데이트해야 합니다.

6. 부록

Application Binary Interface (ABI): 스마트 컨트랙트의 두 프로그램 모듈 간의 인터페이스입니다.

Application Programming Interface(API): 프로그램이 서로 통신할 수 있도록 하는 소프트웨어 매개체입니다.

버그 바운티**(Bug Bounty) :** 버그 바운티 프로그램은 많은 웹사이트, 기관 및 소프트웨어 개발자가 제공하는 거래로서 개인이 버그, 특히 보안 착취 및 취약점과 관련된 버그를 보고함으로써 인정과 보상을 받을 수 있습니다.

DAO: 분산형 자율 기업(DAC)이라고도 불리는 분산형 자율 조직(DAO)은 투명하고, 조직 구성원들이 제어하며, 중앙 정부의 영향을 받지 않는 컴퓨터 프로그램으로 인코딩된 규칙에 의해 대표되는 조직입니다.

dAppS: 탈중앙 화 애플리케이션(dApps)은 EOSIO 블록체인들 상에 존재 및 구동되는 디지털 애플리케이션이 혹은 프로그램입니다.

DeFi: 탈중앙 화 금융 기반의 애플리케이션

퍼징**(Fuzzing):** Fuzzing 혹은 fuzz 테스팅은 유효하지 않거나, 예상치 못했거나, 혹은 임의의 데이터를 dApp이나 컴퓨터 프로그램에 입력값으로 제공하는 자동화된 소프트웨어 테스팅 기술입니다. dApp 혹은 프로그램은 충돌, 내장 코드 어설션 실패 또는 잠재적인 메모리 누수와 같은 예외사항들을 모니터링 합니다.

SDK(Software Development Kit): 프로그램 개발을 용이하게 하는 소프트웨어 개발 도구들의 모음입니다.

WebAssembly (WASM): 웹 애플리케이션들의 효율적인 작동을 위한 이식 가능한 바이너리 코드 형식