

Analysis Report on Korean Blockchain Projects

Part 1 ICON Project



Disclaimer

본 자료는 오직 정보 제공을 목적으로 제공되었습니다.

ADEVT Research Team은 보고서를 작성하기 전에 그 내용을 신중하게 검토하고, 발행하기 위한 합리적인 노력을 기울입니다. 하지만, ADEVT는 본 자료의 정보가 정확하거나 혹은 완전하다고 보장하지 않습니다. 따라서 본 자료에서 제공하는 정보의 이용 또는 비이용이나 사용 또는 미사용으로 인한 피해, 또는 부정확하거나 불완전한 정보 이용으로 인한 피해에 대한 책임을 지지 않습니다.

ADEVT Research Team은 본 자료를 작성하는 과정에 있어서 어떠한 외부의 부당한 압력이나 간섭을 받지 않았습니다.

Copyright

본 자료는 당사 고객의 기술적 판단을 돕기 위한 자료로 제작되었습니다. 본 자료에 포함되는 모든 정보, 데이터 및 기타 콘텐츠는 상표권, 저작권 또는 기타 지적 재산권은 법에 의해 보호를 받습니다. 본 자료에서 제공되는 정보는 어떠한 경우에도 당사의 허가 없이 복사, 유포, 수정, 재배포 될 수 없습니다.

Table of Contents

SECTION 1 Analysis Overview

우리는 왜 블록체인 프로젝트를 분석하게 되었는가?

분석 방법론 : 코드가 모든 것을 말해 줄 것이다.

ICON Project 분석 결과 요약

SECTION 2 Structural Analysis

Loopchain 구조 개요

Node : RadioStation, Peer, Citizen

Channel : 체인 구성의 단위

SCORE : ICON의 스마트 컨트랙트 환경

SECTION 3 Security Analysis

OS Command Injection : CWE-78

Missing Authentication for Critical Function : CWE-306

Bad SCORE Sandboxing

공격 시나리오 1 : 오염된 Radio Station

공격 시나리오 2 : Peer의 Radio Station 조작

공격 시나리오 3 : Peer의 오염

공격 시나리오 4 : 상호 인증이 미적용된 Peer

SECTION 4 Conclusion

SECTION 1

Analysis Overview**우리는 왜 블록체인 프로젝트를 분석하게 되었는가?**

2017년 블록체인에 대한 관심이 증가하면서, 전세계적으로 블록체인 프로젝트가 활발히 시작되고 진행되었다. 국내에서도 이러한 추세를 따라, ICON, HYCON, HDAC 등 여러 블록체인 프로젝트들이 생겨나고 ICO를 통하여 많은 액수의 투자를 받았다.

ICO를 진행할 당시에는, 블록체인에 대한 개발 자료와 분석을 할 만한 전문가가 거의 없었기 때문에 블록체인 프로젝트들은 그들의 기술에 대해서 거의 도전을 받지 않았고, 대부분의 투자자들은 기술의 실재 여부나 가치보다는 자신들이 투자한 코인이 어느 거래소에 상장하는 지를 중요하게 여겼으며, 따라서 프로젝트 운영자들은 실제 구현체를 공개하지 않아도 그들의 투자자들을 설득 할 수 있었다.

프로젝트명	프로젝트 시작	ICO 모금액
BOSCoin	2017년 5월	170억원
ICON	2017년 8월	1000억원
HYCON	2017년 9월	800억원
HDAC	2017년 3월	2800억원

[Fig 1] 초기 진행된 국내 블록체인 프로젝트의 ICO 모금액

여러 매체들과 사람들이 국내 블록체인 산업의 도덕적 해이를 지적하는 상황에서 우리는 이 문제에 대해서 기술적으로 접근을 하고자 한다. 우리는 국내의 블록체인 플랫폼을 분석하여 수많은 투자자들의 돈이 들어간 그들의 작품이 정말로 신세계로의 길이었는지, 아니면 황금빛 허상에 불과했는지 알아보려고 한다.

분석 방법론 : 코드가 모든 것을 말해 줄 것이다.

블록체인 프로젝트가 본질적으로는 소프트웨어를 만드는 프로젝트이므로, 우리는 프로젝트 진행자들이 제공하는 기술 문서와 Github에 등재된 코드를 기초로 하여 분석을 진행했다. 필요에 따라, 소스코드가 공개되어 있지 않은 지갑 어플리케이션을 역공학¹ 기법을 통해 분석을 진행했다.

우리는 크게 두 가지 방향에서 프로젝트 분석을 진행했는데, 첫 번째는 백서 일치성이다. 백서 상의 구조와 일치하는지, 그들이 제시한 일정을 착실히 달성하였는지, 프로젝트가 기술적으로 지속가능한지 여부를 Github Commit 로그와 소스코드 분석을 통해서 확인한다.

두 번째는 보안성이다. 소스코드 분석과 실제 테스트넷에 대한 공격을 진행하여 블록체인 프로젝트의 핵심이라고 할 수 있는 보안성이 제대로 보장되고 있는지를 확인하며, 구조적 혹은 구현 상의 취약점이 발견된 경우, 공격 시나리오와 실제 코드를 통한 컨셉 증명 (Proof-of-Concept)를 수행한다.

이번 분석은 대한민국 1세대 블록체인 프로젝트라고 불리는 ICON Project를 대상으로 이루어졌다. 이번 분석에서는 ICON Project의 코어 엔진인 Loopchain에 대한 분석이 집중적으로 이루어졌다. 분석 대상 저장소는 다음과 같다.

저장소 명	저장소 내용
Loopchain	ICON 노드 엔진
ICON-Service	SCORE (ICON Smart Contract) 엔진
ICON-RPC-Server	엔진 간 중계 (RPC 프로토콜)
Earlgrey	엔진 간 중계 (RPC 서버)

[Fig 2] 분석 대상 저장소

분석 과정 중에 몇몇 수정 사항이 있었으나, 실질적으로 우리의 분석 결과에 영향을 주는 수정 사항은 없었다.

¹ Reverse Engineering. 완성된 어플리케이션 혹은 서비스의 구조를 파악하는 공학 기법

ICON Project 분석 결과 요약

위에서 언급했듯이, 우리는 Loopchain의 분석에 집중하였다. ICON Project의 핵심 엔진인 Loopchain은 신뢰 기반의 프라이빗 블록체인 엔진이다.

백서에 특별히 기술적인 구조에 대한 명기가 없어 확실히 어떠한 것을 만들고자 하였는지 확인하기는 어려우나, 신뢰받는 노드들(Peer), C-Rep과 P-Rep들에 의해서 네트워크가 유지되는 프라이빗 네트워크의 구현은 완료된 것으로 보인다.

실제로 테스트 네트워크를 기동하였을 때, 크게 문제없이 작동하였고, 코드 상으로도 대부분의 명시된 기능에 대한 구현이 그 코드의 질적 완성도에 상관없이, 개발 완료되어 있었다.

하지만, 구현 상의 심각한 취약점이 존재하여, 네트워크에 치명적인 장애를 일으키는 것으로 확인되었다. 노드 확인을 위해 이용하는 서드-파티 라이브러리에 원격으로 컴퓨터에 명령어를 실행하여 대상 컴퓨터를 공격할 수 있게 되는 Remote Command Injection 취약점이 존재하여, 일반 노드가 리더 노드 혹은 네트워크를 통제하는 관리 노드 (Radio Station)을 오염시킬 수 있다. 따라서 악의적인 의도를 가진 사용자가 노드의 접근 권한을 가지게 될 경우, 모든 노드를 오염시켜 악성 거래를 성사시킬 수 있게 되는 것이다. 이는 합의 프로토콜 전체를 흔들 수 있는 취약점이다.

또한, Loopchain의 Smart Contract인 SCORE의 경우에는 시스템에 영향을 줄 수 있는 Python 프로그래밍 언어를 사용함에도 불구하고 제대로 된 샌드박스²이 이루어지지 않아, 악성 코드가 시스템 자원에 직접적인 위협에 있도록 하였다.

사용하고 있는 서드-파티 라이브러리가 블록체인 엔진의 핵심 부분에 사용되고 있으므로, 이를 제거하고 대체 코드를 작성하는 데에 어느 정도의 시간이 소요될 것으로 보인다.

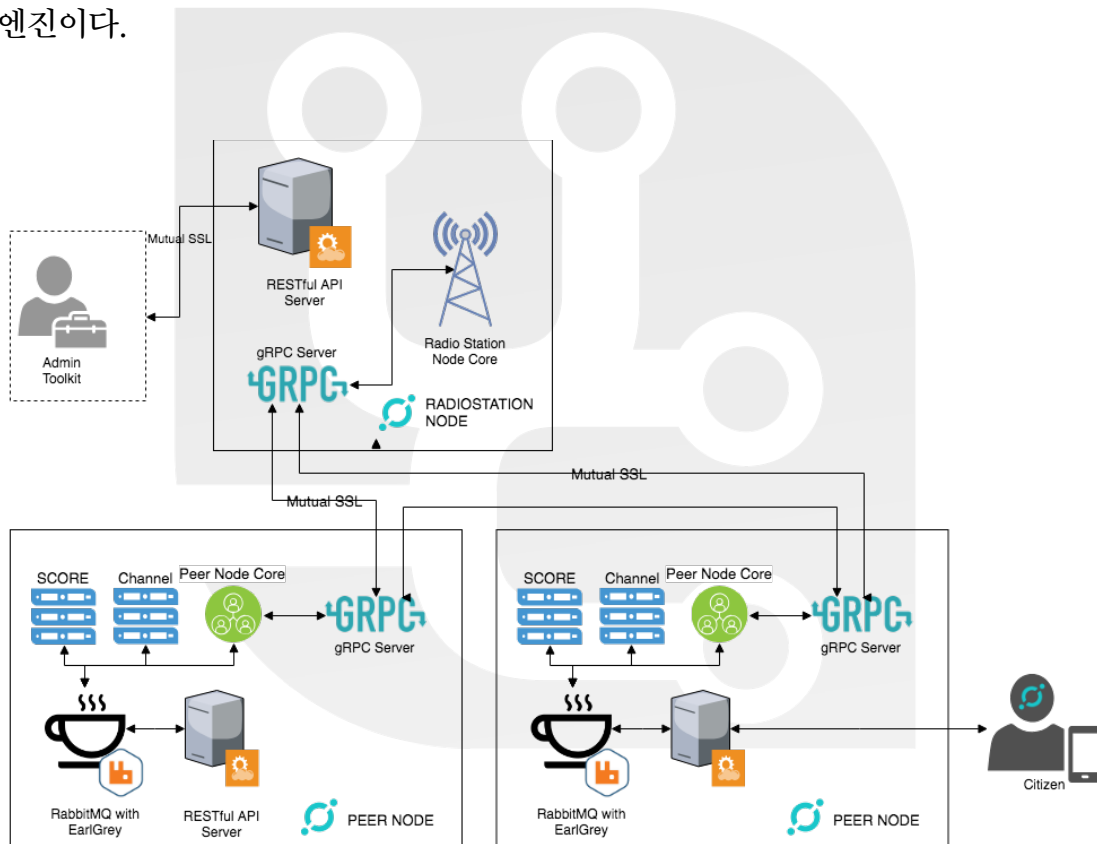
² Sandboxing. 가상 컴퓨터와 같은 분리된 환경에서 프로그램을 작동시켜 악성 행위나 에러가 직접적으로 물리 시스템에 영향을 미치지 못하게 하는 것.

SECTION 2

Structural Analysis

Loopchain 구조 개요

Loopchain은 스마트 계약을 지원하는 고성능 엔터프라이즈 블록체인을 목표 개발된 블록체인 엔진이며, ICON Project에서 C-Rep 노드들을 이어주는 블록체인 엔진이다.



[Fig 3] Loopchain Network 구조 예시

위 그림은 Loopchain의 구조도를 간략하게 나타낸 도식이다. 예시 네트워크에서는 노드가 2개 밖에 없어서 실제로는 합의가 이루어질 수 없고, 유의미한 합의를 위해서는 4개 이상의 Peer 노드가 필요하다.

Peer 노드는 Radio Station 노드에 의해서 관리되며, 각 노드 간의 통신은 gRPC 를 통해서 이루어 진다. Citizen 노드는 Peer 노드처럼 등록이나 인증이 필요없는 대신에, Loopchain에 실질적인 관여를 할 수는 없으며, 단지 Peer 노드에 REST API를 통해 요청을 넣는 것 정도 밖에 할 수 없다.

Loopchain은 기본적으로 멀티-체인을 지원하는 데, 원장을 별도로 관리하기 위한 단위로 Channel이라는 것을 채택했다. 서로 다른 Channel의 합의와 블록 생성에는 다른 노드 집합이 사용이 되며, 원장이 분리 되었으므로, SCORE와 트랜잭션 정보도 분리된다.

SCORE는 중앙화된 Git을 통하여 버전을 관리하며, 블록에 저장된 주소를 통하여 Python 코드를 내려받아 실행하는 식으로 동작한다. SCORE의 코드는 발행을 위해서는 정해진 노드의 보안 감사를 받아야 하며, 이는 과정은 모두 사람에게 의해 수동으로 이루어진다.

구조적 분석을 수행한 결과, 채널을 통한 멀티-체인 및 데이터 접근 제어, 신뢰에 기반한 LFT 합의 프로토콜, 튜링-완전한 스마트 컨트랙트 언어 등의 향후 구현 계획을 제외한 백서 상 명시되고 있는 대부분의 기능은, 그 질적 완성도에 상관 없이, 개발 완료되어 있음을 확인하였다.

Node : RadioStation, Peer, Citizen

ICON Project에서는 거버넌스(Governance)적인 문제를 해결하기 위하여 C-Rep, P-Rep, C-Node 등 여러 참여자를 제시하고 있지만, 구조적으로는 Loopchain에는 RadioStation, Peer, Citizen의 3가지 종류의 노드가 존재한다.

Peer 노드는 RadioStation 노드에 의해 등록되고, 인증되며 관리되는 주체이며, Loopchain에서 트랜잭션이 포함된 블록들을 만들고, SCORE을 배치하고 처리하기 위한 합의를 진행하는 주체이다. Peer 노드 중에서 Leader Peer 노드가 선출되며, Leader Peer 노드가 블록을 실질적으로 생성하며, 나머지 노드들은 블록의 검증과 투표를 수행한다.

RadioStation 노드는 Peer 노드를 사전에 등록하고 관리하며, Peer 노드들에게 다른 Peer 노드들이 어디있는지 알려주는 역할을 한다. 노드와 채널에 대한 접근 제어를 수행하므로, 일종의 멤버십 서비스를 제공한다고 볼 수 있다.

Citizen 노드는 블록의 생성이나 검증에는 관여하지 않는다. Peer 노드의 REST API와 통신하며, ICX 거래나 SCORE 함수 호출과 같은 요청을 Peer 노드에게 할 수 있다. 일반 사용자가 ICX 지갑을 이용하거나 DEX를 이용할 때 다음과 같은 권한으로 Loopchain 네트워크에 접근하게 된다.

Channel : 체인 구성의 단위

ICON Project에서는 Channel이라고 하는 체인 구성의 단위를 만들어 거래와 SCORE를 분리하는 할 수 있도록 한다. 이는 멀티-체인을 이용하여 원장을 별도로 관리 할 수 있게 하는 것이다. 현재 외부에 공개되어 있는 채널은 메인넷과 2개의 테스트넷이다. Channel 별로 별도의 프로세스를 할당하여 운영한다.

SCORE : ICON의 스마트 컨트랙트 환경

ICON Project에서는 SCORE이라고 하는 스마트 컨트랙트 환경을 제공한다. 현재는 Python 구현체 만을 받고 있으며, SCORE 배치를 위해서는 인증된 노드의 검수를 받아야 한다. SCORE 코드 자체는 블록 외부에 저장하며, 사전에 정의된 Git 서버를 이용하여 버전 관리를 한다.

테스트넷에서 공개 되어 있는 SCORE 코드는 Governance.py (ICON Governance 스마트 컨트랙트), link_coin.py (LINK Coin 스마트 컨트랙트), multisig_wallet.py (멀티시그니처 월렛 스마트 컨트랙트)가 있다.

SECTION 3

Security Analysis

OS Command Injection : CWE-78

Loopchain을 분석하는 과정에서, 우리는 Loopchain의 개발자들이 데이터의 송수신에 pickle이라는 서드-파티 라이브러리를 사용한다는 사실을 발견했다. pickle의 개발자 문서를 보면, 비신뢰 환경에서는 절대 사용하지 말라는 경고를 확인할 수 있다.

pickle — Python object serialization

Source code: [Lib/pickle.py](#)

The `pickle` module implements binary protocols for serializing and de-serializing a Python object structure. “Pickling” is the process whereby a Python object hierarchy is converted into a byte stream, and “unpickling” is the inverse operation, whereby a byte stream (from a [binary file](#) or [bytes-like object](#)) is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as “serialization”, “marshalling,” [1] or “flattening”; however, to avoid confusion, the terms used here are “pickling” and “unpickling”.

Warning: The `pickle` module is not secure against erroneous or maliciously constructed data. Never unpickle data received from an untrusted or unauthenticated source.

[Fig 4] pickle 라이브러리 개발자 문서

pickle 라이브러리의 OS Command Injection 취약점은 CVE-2005-2875에서 최초로 보고 되었다. 최초 발견후 14년이 지난 지금까지도 여러 제품에서 빈번히 발견되고 있다.

OS Command Injection 취약점이란 공격자가 목표로하는 운영체제에 대하여 악성 명령어를 실행 할 수 있도록 하는 취약점이다. 이 취약점은 공격자로 하여금 단순히 목표 시스템의 파괴나 서비스의 정지가 아니라, 목표 시스템의 작동 권한을 완전히 장악하는 것 또한 가능하여 매우 치명적이다.

우리는 ICX 거래 내역 확인 로직, 합의 알고리즘 로직, Peer 로직 등 블록체인 네트워크 운영에 있어서 핵심적인 기능을 하는 컴포넌트들에 pickle 라이브러리가 포함되어 있는 것을 확인하였다.

Missing Authentication for Critical Function : CWE-306

원격 통신을 하는 프로그램에 있어서는 권한 제어는 매우 중요한 이슈이다. 잘못된 권한 제어 정책이나 권한 제어의 미비는 곧 공격자의 권한 상승으로 이어지기 때문이다.

우리는 Radio Station의 RESTful API³ 로직과 RPC 루틴에 적절한 권한제어 조치가 이루어지지 않는다는 사실을 발견하였다. 이는 네트워크 관리자 권한을 공개적으로 내어주고 있는 것과 같으며, 네트워크에 치명적인 타격을 줄 수도 있다.

Bad SCORE Sandboxing

최근에 virtualenv라고 하는 Python 샌드박스 라이브러리의 취약점이 발견되었다. (CVE-2018-17793) Loopchain의 서비스들은 서비스 전반에 걸쳐 가동 시에 virtualenv를 샌드박스로 사용하기를 권장하는데, 이는 최근에 발견된 취약점에 의해서 쉽게 돌파될 수 있다. 이 경우, 악성코드가 직접적으로 시스템에 영향을 줄 수 있게 되므로 시스템이 위협에 노출되게 된다.

또한, 샌드박싱의 범위를 잘못 설정하였는데, SCORE의 실행 환경과 Peer 노드 엔진의 실행환경을 같게 할 경우, SCORE에 악성 코드가 삽입되어 있을 경우 노드 인증서의 탈취 혹은 노드 시스템의 파괴 등이 이루어질 수 있다.

³ 웹에 존재하는 모든 자원(이미지, 동영상, DB 자원)에 고유한 URI를 부여해 활용할 수 있도록 하는 도구

공격 시나리오 1 : 오염된 Radio Station

우리는 위에서 기술한 취약점을 바탕으로 몇가지 공격 시나리오를 구성해 보았다. 먼저 공격자가 Radio Station의 권한을 가지게 되었을 경우의 공격 시나리오이다. 시나리오는 다음과 같다.

1. 오염된 Radio Station이 Peer List 요청에 대한 응답으로 제어권 탈취를 위한 악성 코드를 전송
2. 정상 Peer의 악성 코드 실행 (Remote Command Injection : CWE-78)
3. Peer 제어권 확보 (Sandbox Escaping : CWE-254)
4. 오염된 Peer가 정상 Peer에게 제어권 탈취를 위한 악성 코드를 포함한 Block 검증 요청을 전송
5. 정상 Peer의 악성 코드 실행 (Remote Command Injection : CWE-78)
6. 모든 Peer의 제어권 확보
7. Peer가 속한 Private Channel의 SCORE 정보 탈취 / Peer 내부 정보 탈취



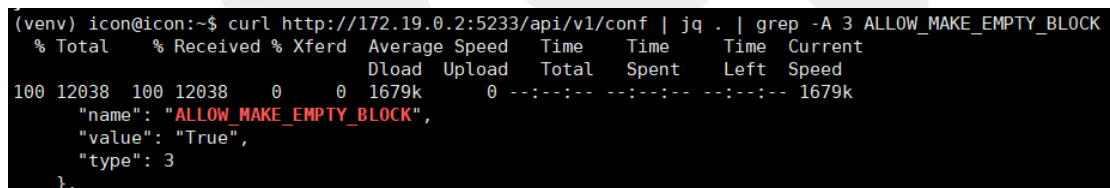
[Fig 5] 공격 시나리오 1 증명

위의 스크린샷은 오염된 Radio Station을 통하여 토끼 문자열을 Peer에게 표시하도록 하는 코드를 실행한 것이다. PoC를 통하여 우리는 시나리오 1이 실현 가능함을 증명하였다.

공격 시나리오 2 : Peer의 Radio Station 조작

악의적인 의도를 가진 Peer가 Radio Station의 설정을 조작하여 네트워크의 권한을 탈취하는 시나리오이다. 시나리오는 다음과 같다.

1. 악의를 가진 Peer가 RESTful API로 Radio Station 관리 명령어를 호출
(Missing Authentication : CWE-306)
 - Case A
 - 가. 악의를 가진 Peer가 자신에게 Private Channel에 대한 접근 권한을 부여
 - 나. Private Channel의 SCORE 정보 탈취
 - Case B
 - 가. 악의를 가진 Peer가 다른 Peer들의 Private Channel에 대한 접근 권한을 삭제
 - 나. Private Channel 장악
 - 다. 단독 합의를 통한 Citizen 교란



```
(venv) icon@icon:~$ curl http://172.19.0.2:5233/api/v1/conf | jq . | grep -A 3 ALLOW_MAKE_EMPTY_BLOCK
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload  Total      Spent    Left     Speed
100 12038  100 12038    0     0  1679k      0  --:--:-- --:--:-- --:--:-- 1679k
{"name": "ALLOW_MAKE_EMPTY_BLOCK",
 "value": "True",
 "type": 3
}
```

[Fig 6] 공격 시나리오 2 증명

위의 스크린샷은 악의적인 Peer 노드가 정상적인 Radio Station에게 설정 변경을 요청하는 API 요청을 만들어 요청하여 Radio Station의 설정을 임의로 수정하는 스크린샷이다. PoC를 통하여 우리는 시나리오 2가 실행 가능함을 증명하였다.

공격 시나리오 3 : Peer의 오염

악의적인 의도를 가진 Peer가 다른 Peer들을 감염시켜 네트워크의 권한을 탈취하는 시나리오이다. Peer가 Leader Peer일 경우와 그렇지 않을 경우로 시나리오가 나뉜다. 시나리오는 다음과 같다.

1. Leader Peer의 오염

1. 오염된 Leader Peer가 정상 Peer에게 제어권 탈취를 위한 악성 코드를 포함한 Block 검증 요청을 전송
2. 정상 Peer의 악성 코드 실행 (Remote Command Injection : CWE-78)
3. 모든 Peer의 제어권 확보
4. 51% 공격을 통한 블록 조작 / Peer가 속한 Private Channel의 SCORE 정보 탈취 / Peer 내부 정보 탈취

2. 일반 Peer의 오염

1. 오염된 Peer가 정상 Peer에게 제어권 탈취를 위한 악성 코드를 포함한 Block 정보를 전송
2. 정상 Peer의 악성 코드 실행 (Remote Command Injection : CWE-78)
3. 모든 일반 Peer의 제어권 확보
4. 51% 공격을 통해 오염된 Peer로 Leader Peer 변경

```
File "/usr/local/lib/python3.6/asyncio/events.py", line 145, in _run
    self._callback(*self._args)
...
0118 07:08:53,311 716 139700763301632 hxbf2834 ... Leader Peer Attack
peer_id(198a7e35ff85bad741f7d67d6ed86c2ccc749c0)
```

[Fig 7] 공격 시나리오 3-1 증명

```
0121 11:55:02,893 717 140559030904576 hx7effe5 loopchain_default ERROR _init_.py(535) Another Peer Attack
0121 11:55:02,894 717 140559030904576 hx7effe5 loopchain_default DEBUG channel_inner_service.py(302) #block
peer_id(5f1633dcc83b67f4e093bcd7f6fdef6c97d61026)
height(10000027)
hash(b'\xe8\x953\x89\xc47\xb6\x80\xb6\xe6o\x0b\xa2b\xdf\x5\x85\xf6 \xe4\xb0\xc4:\x7f\x1q\xa5Y\x1e\xf2\xe2\x8d')
0121 11:55:02,895 717 140559030904576 hx7effe5 loopchain_default INFO block_manager.py(250) unconfirmed_block 10000027, False
0121 11:55:02,896 717 140559030904576 hx7effe5 loopchain_default DEBUG core.py(246) Initiating transition from state Vote to state Vote...
0121 11:55:02,897 717 140559030904576 hx7effe5 loopchain_default DEBUG core.py(125) Exiting state Vote. Processing callbacks...
0121 11:55:02,897 717 140559030904576 hx7effe5 loopchain_default SPAM channel_statemachine.py(141)
```

[Fig 8] 공격 시나리오 3-2 증명

위의 스크린샷들은 악의적인 Peer 노드가 정상적인 Peer에게 임의의 문자열 (Leader Peer Attack, Another Peer Attack)을 Peer에게 표시하도록 하는 코드를 실행한 것이다. PoC를 통하여 우리는 시나리오 3가 실행 가능함을 증명하였다.

공격 시나리오 4 : 상호 인증이 미적용된 Peer

악의적인 의도를 가진 Citizen이 관리자의 실수로 gRPC 상호인증이 걸려있지 않은 Peer를 공격하는 시나리오이다. 시나리오는 다음과 같다.

1. 악의적 Citizen이 Peer로 위장하여 설정이 잘못된 Peer에게 제어권 탈취를 위한 악성 코드를 포함한 Block 검증 요청을 전송
2. 정상 Peer의 악성 코드 실행 (Remote Command Injection : CWE-78)
3. 일반 Peer의 제어권 확보
4. 오염된 Peer를 이용한 2차 공격 수행

시나리오 4의 증명은 테스트넷을 이용하여 별도의 클라이언트 수정을 하지 않고 Peer 배치 시의 설정파일에서 gRPC 상호인증 옵션을 끄고 임의의 단말로 Peer로 접근을 하면 접근이 되는 것을 확인하였다. 실행이 일반적인 Peer의 작동과 다를 바가 없어 스크린샷으로 확인이 불가하므로, 스크린샷을 첨부하지 아니하였다.

SECTION 4

Conclusion

이번 분석에서는 국내 블록체인 프로젝트 분석의 첫 번째로 ICON Project를 핵심 엔진인 Loopchain을 중점으로 분석하였다. Loopchain은 신뢰 기반의 프라이빗 블록체인 엔진으로, ICON Republic의 연결 기반이다.

ICON Project는 그 코드의 질적 수준과는 별개로, 현재 작동하는 네트워크를 운영하고 있으며, 자체적인 로드맵을 충실하게 이행하였다.

다만, 현재 블록체인을 운영하는 데에 있어 핵심적인 컴포넌트들에 사용 중인 서드-파티 라이브러리에 중대한 취약점이 존재한다. 이 취약점은 네트워크의 신뢰 기반 자체를 공격할 수 있으며, 이는 필히 해결되어야 할 문제이다.

현재 ICON Foundation에서 P-Rep 노드를 선발하기 위한 과정을 진행 중인데, 이러한 중대한 취약점이 존재하는 상태에서 임의의 집단에게 Peer 노드 권한을 부여하는 것은 보안상 중대한 위협을 초래할 수 있다.

위의 취약점을 해결하기 위해서는 pickle 라이브러리를 제거하고, SCORE의 실행환경을 분리환경으로 만들어야 할 필요성이 있다. 하지만 pickle을 json 라이브러리로 대체할 경우 데이터 평균 처리 시간이 약 2배 정도로 증가하는데, 해당 라이브러리가 합의 과정에 사용되는 것을 감안하면 Loopchain의 속도가 감소할 것으로 보인다.

상기한 취약점을 해결할 경우에만, Loopchain은 블록체인 산업과 실생활에서 실험적으로 사용될 수 있을 것이다.