심볼릭 실행 기반 LLM 에이전트를 통한 허니팟 스마트 컨트랙트 탐지와 온체인 인텔리전스

이동하 · 김수민 · 한보윤

Concept

악성 허니팟을 탐지하고,
피해 근원지를 추적한다

연구 과정

주제 제안서

Web3 허니팟 + CTI 구상

주제 발표

문제 정의 · 기존 연구 조사

Stealth 구축

우회 허니팟 v1→v2→v3 설계

비교 실험

HoneyBadger 재현 · 정량 평가

학술대회 논문

탐지 방법론 주제로 논문 게재

Dashboard

온체인 CTI 도구 개발 → 최종

Contents

01

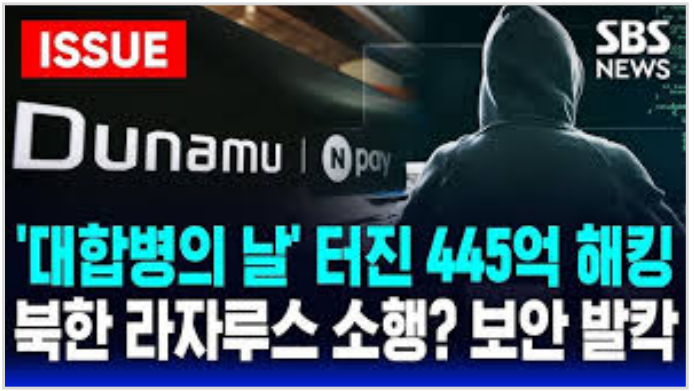
SECTION

1
연구 배경 및 필요성

- 왜 Web3인가
- 허니팟 피해자 유형
- CTI — Web3의 침해지표(IoC)
- Honeypot Contract
- 기존 연구의 한계

왜 Web3인가

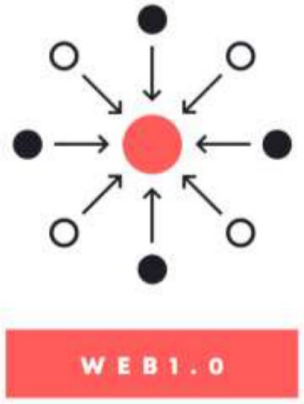
공격자는 Web3로 공격을 진행중이다 — **탈취 자금의 출처·경로** 추적이 핵심 과제



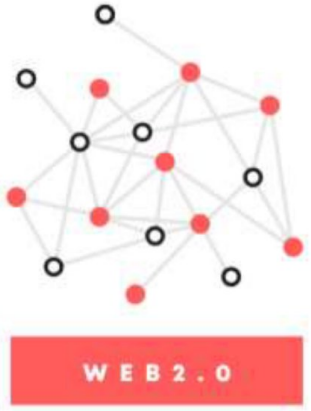
북한 Lazarus

Web3 위협은 탈중앙·익명 구조로 **추적이 어렵다**

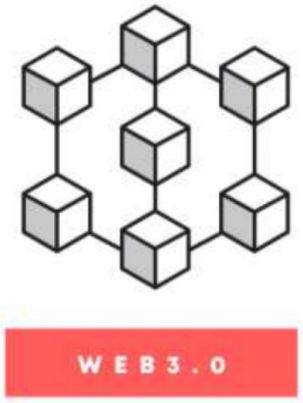
CTI - Web3의 침해지표(IoC)



- IP Address
- Domain
- URL



- File Hash
- Email
- C2 Server



- Wallet Address
- Smart Contract Address
- Transaction Hash
- Token Contract
- DeFi Protocol Interaction

Honeypot Contract

겉보기엔 일반 컨트랙트, 실제로는 피해자만 자금을 잃는 함정 컨트랙트

피해자가
ETH 입금

→

숨겨진 조건·상태로
출금 차단

→

공격자만 회수
피해자 손실 확정

허니팟 피해자 유형

코드형 허니팟

타겟 피해자: 코드를 분석할 줄 아는 준전문가

유인 방식:
Etherscan에 고의로 코드를 배포하여 취약점으로 위장

피해 원리: 피해자가 취약점을 공격해 돈을 빼내려 ETH를 입금하는 순간, 함정이 발동해 출금 불가능화

투자스캠 광고형 허니팟

타겟 피해자: 일반 Web3 사용자

유인 방식:
SNS, 텔레그램 등을 통한 고수익 광고 및 피싱 사이트

피해 원리: 화려한 UI에 속아 내용도 모른 채 지갑 연동 및 서명 승인

입금·매수는 되는데 출금·매도는 주인만. 본 연구의 탐지·추적 대상은 코드형

기존 연구의 한계

HoneyBadger

- 심볼릭 실행 단독 + 휴리스틱 8종
- 경로 2^n 폭발 → 탐색 불완전

SCH-Hunter

- Hybrid Fuzzing
- 탐지 한정 — CTI·추적 미연계
- 깊은 조건 탐색 한계

Our Gap

- 탐지 결과를 온체인 CTI와 미연결
- 탐지에서 멈춤 — 추적·시각화 부재

→ Sw3et 메우는 지점

02

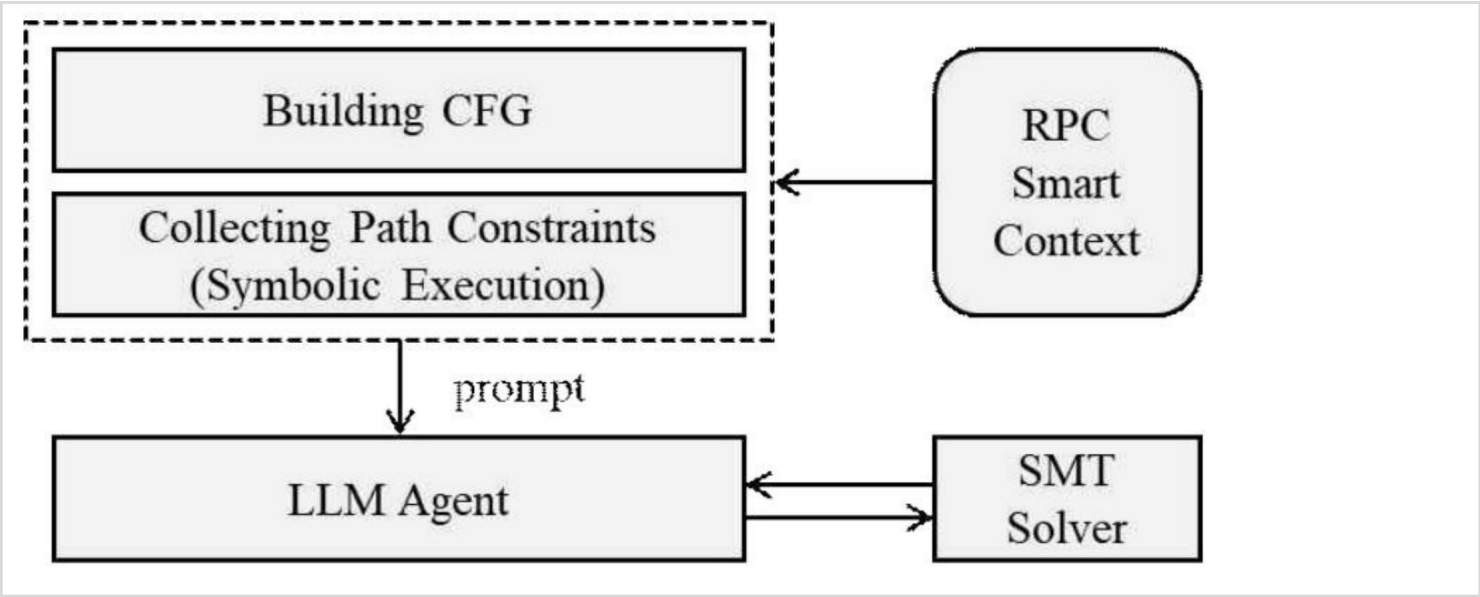
SECTION

2
제안 방법

- 탐지 파이프라인 (CFG→Symbolic→LLM↔SMT)
- 왜 LLM + SMT 인가
- 실험 설계 (RQ · 통제 환경)

제안 방법

CFG → 심볼릭 실행 → LLM(ReAct) ↔ SMT 하나의 탐지 파이프라인



① CFG 구성

② 경로 제약 수집 심볼릭 실행

③ LLM Agent ReAct 추론

④ SMT Solver (Z3) SAT / UNSAT

▲ 학술대회 논문 Overview — 공식 방법론 도식

왜 LLM + SMT 인가?

LLM Agent

- 경로 조건을 자연어로 요약
- 허니팟 시나리오 후보 도출
- SMT 검증 쿼리 생성

SMT Solver (Z3)

- 경로 조건 충족 가능성 검사
- SAT = 허니팟 확정
- UNSAT = 후보 제거

실험 설계

RQ1 코드 구조가 변형된 허니팟에 HoneyBadger는 얼마나 강건한가?

RQ2 LLM 기반 의미 분석은 변형 허니팟을 탐지할 수 있는가?

RQ3 LLM이 SMT를 도구로 쓰는 구조는 단순 심볼릭과 무엇이 다른가?

항목	HoneyBadger (기존)	제안 (Sw3et)
분석 입력	바이트코드(컴파일된 기계어)	Solidity 소스 (Etherscan)
핵심 엔진	심볼릭 실행 + 정해진 규칙 8종	LLM(GPT-5.3-Codex) + Z3
검증 방식(SMT)	식 1개당 1초 · 전체 50초	의심 패턴 Z3 검증
탐색 방식	반복문 10 · 깊이 50 · 가스 400만	의심 패턴 ReAct 반복 추론

03

SECTION

3
실험 및 결과

- 우회 허니팟 제작 (v1→v2→v3)
- 우회 효과 검증
- Case Study
- 실험 결과

우회 허니팟 제작

용도 — 탐지기 성능을 검증하기 위한 평가용 데이터셋 (Red Teaming)

v1 코드 패딩

- 가짜 변수, 로직을 채움
- 이벤트·로직 삽입
- 탐지 영향 거의 없음

v2 구조 변형

- inline assembly
- sstore / call
- 4단계 상속 shadowing

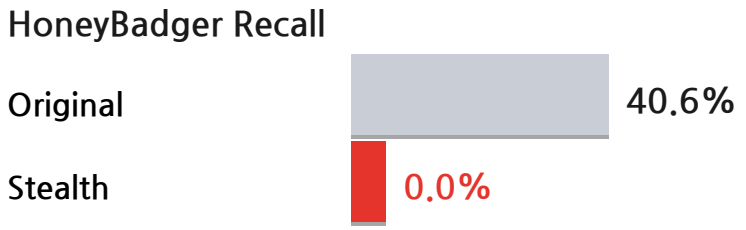
v3 프록시 위임

- 핵심 로직 delegatecall 위임
- 동적 주소 sload
- 교차 컨트랙트 은닉

우회 효과 검증

단순 패딩 (v1)
기존 탐지 여전히 탐지 → 난독화만으론 부족

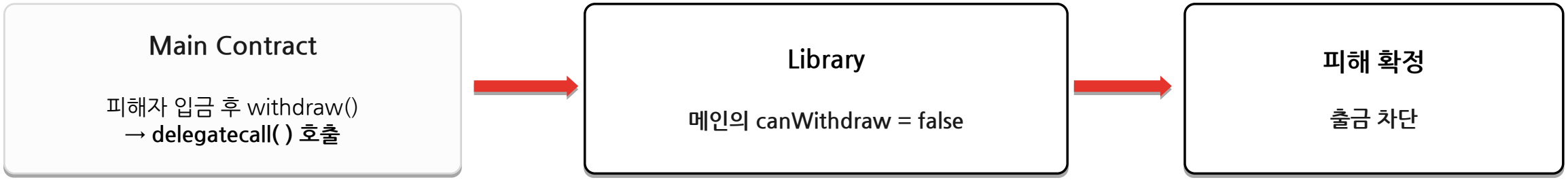
구조 변형 (v2 · v3)
기존 탐지율 → **0%로 하락**



실험	N	방법	Precision	Recall	F1	Acc
Original	32	기존	100%	40.6%	57.8%	40.6%
		제안	100%	84.4%	91.5%	84.4%
Stealth	32	기존	-	0.0%	0.0%	0.0%
		제안	100%	90.6%	95.1%	90.6%

Case Study

delegatecall로 숨긴 Hidden State Update



기존 심볼릭 단독
delegatecall 경계에서 상태 추적이
끊겨 숨은 변경을 미탐

제안 (LLM + SMT)
LLM이 교차 컨트랙트 시나리오 가설
→ SMT가 도달성 검증 → 탐지

실험 결과

Original 32 · Stealth 32 | LLM gpt-5.3-codex · SMT z3-solver | Precision = 오탐률 역지표 · Recall = 미탐률 역지표

데이터셋	방법	Precision	Recall	F1	Accuracy
Original	HoneyBadger (기존)	100%	40.6%	57.8%	40.6%
Original	**제안 방법 (Sw3et)**	**100%**	**84.4%**	**91.5%**	**84.4%**
Stealth	HoneyBadger (기존)	-	0.0%	0.0%	0.0%
Stealth	**제안 방법 (Sw3et)**	**100%**	**90.6%**	**95.1%**	**90.6%**

기존 도구는 레드티밍 목적으로 제작한 Stealth 우회 허니팟에 대한 탐지율 0% (F1 0%) → 제안 F1 95.1%

04

SECTION

4
구현 — Sw3et Dashboard

- 탐지에서 추적으로 (논문 → 도구)
- N-Hop BFS 트랜잭션 탐색
- Sw3et Dashboard — 자금 흐름 3D 추적
- 분석대상 허니팟 코드
- 허니팟 유형별 온체인 구조차이

탐지에서 추적으로

논문(탐지 방법론) → Sw3et Dashboard(온체인 CTI)

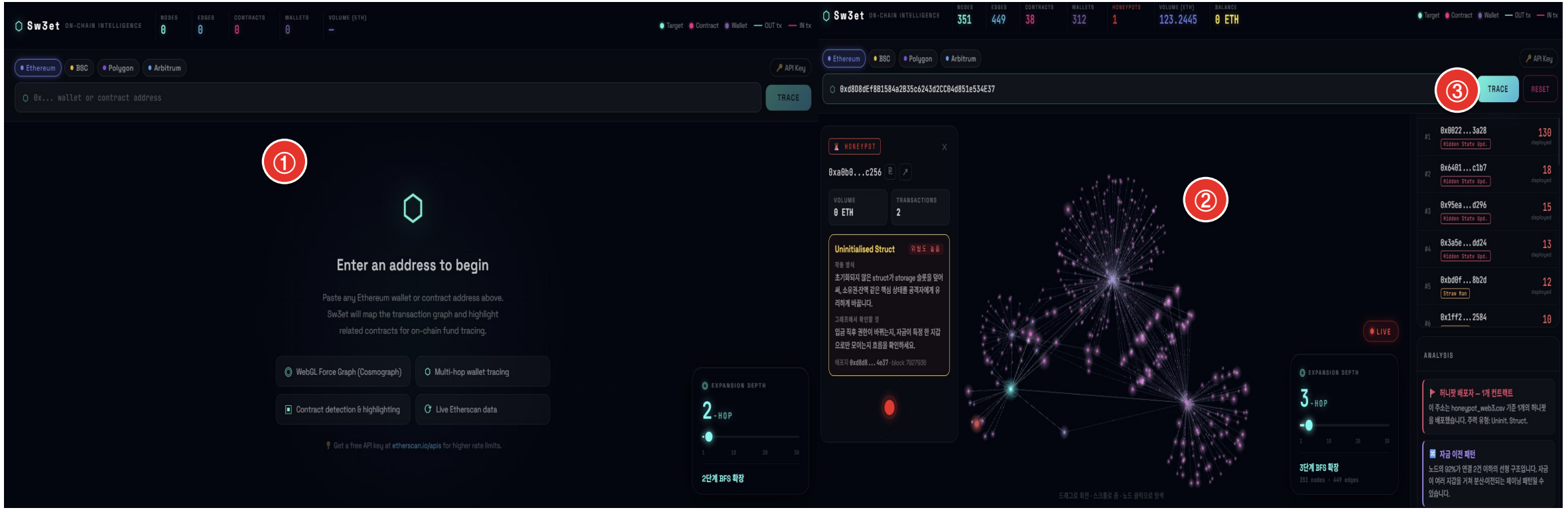
N-Hop BFS 트랜잭션 탐색

$$frontierSize(hop, val) = max(2,\ round(val \times 1.5 \div 1.8^{(hop-1)}))$$

- hop이 깊어질수록 확장 노드 수를 1.8배씩 감소 → 그래프 폭발 억제
- batchFetch( ) — 4개 요청씩 Promise.all 병렬 처리
- hash + from + to Set → O(1) 중복 제거

Sw3et Dashboard

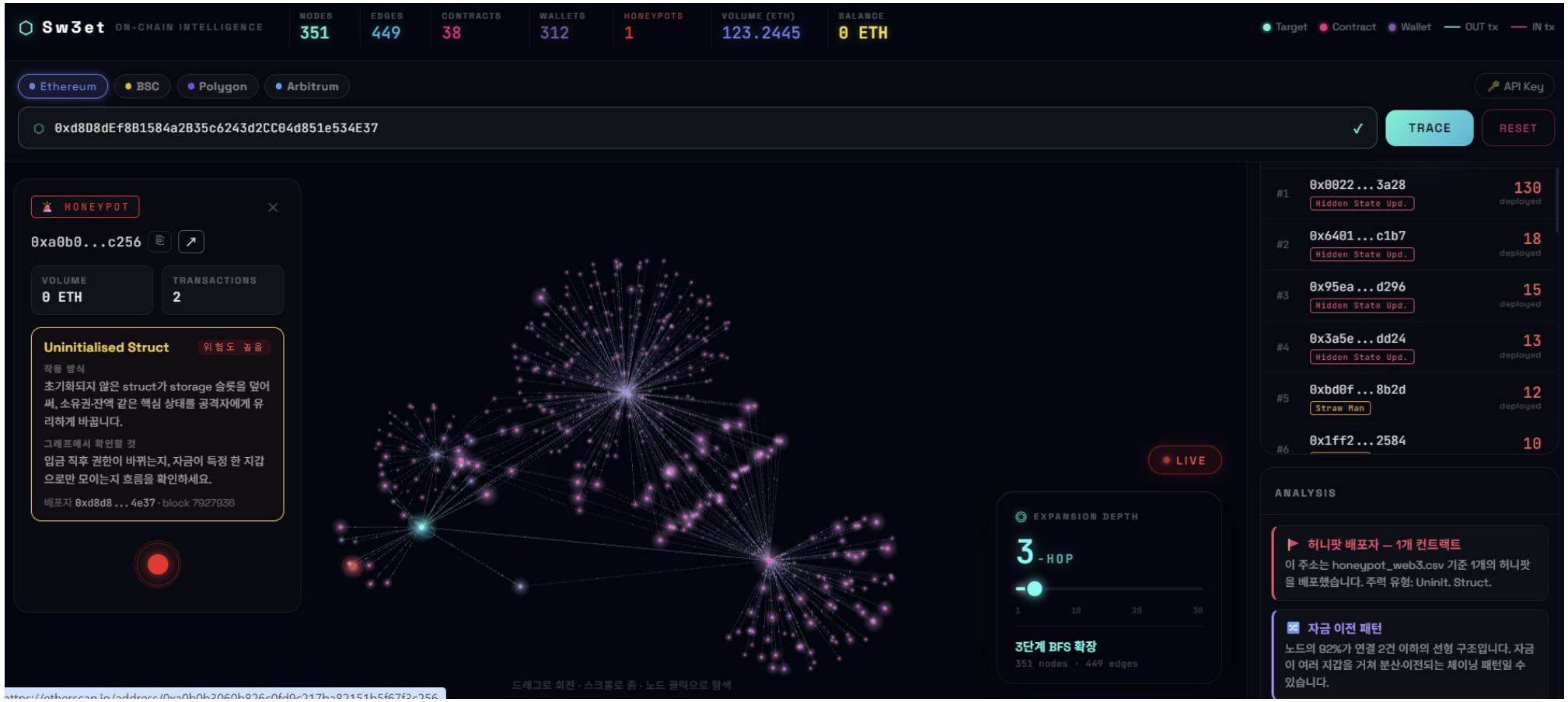
Etherscan 주소 입력 → 피해 경로를 3D 그래프로 추적



① 주소 입력 ② 3D 자금 흐름 그래프 ③ 노드 패널

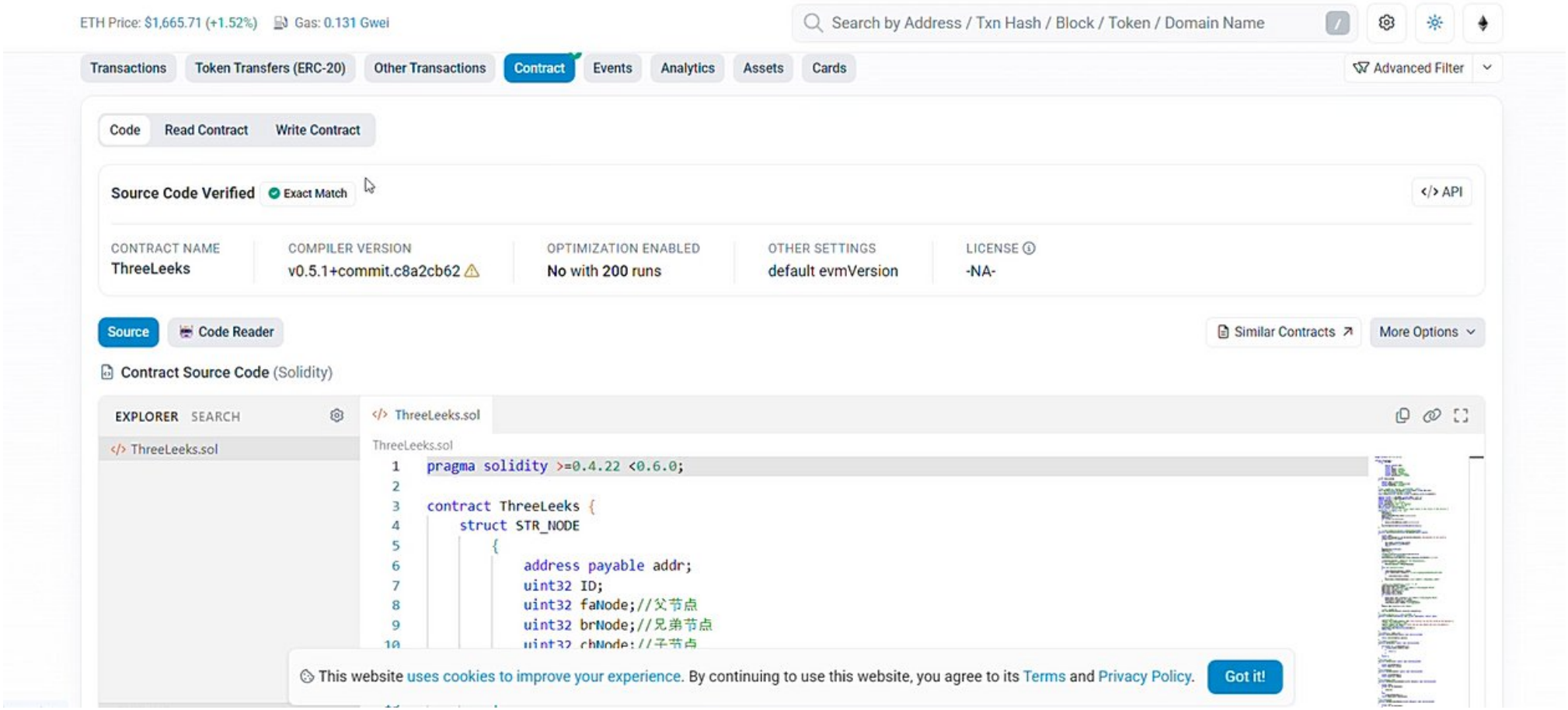
Sw3et Dashboard

확인된 허니팟 → Etherscan Contract 코드 확인



Sw3et Dashboard

확인된 허니팟 → Etherscan Contract 코드 확인



분석 대상 허니팟 코드 — STR_NODE

```
// STR_NODE  (0xa0b0...c256)  -  Ponzi-type honeypot
//   * excerpt, simplified for readability *
contract STR_NODE {
    struct Node { address addr; uint income; }
    Node[] public nodes;
    address owner;

    function join() public payable {     // anyone deposits
        nodes.push(Node(msg.sender, 0));
    }

    function distribute() public {
        require(msg.sender == owner);              // (1)
        for (uint i; i < nodes.length; i++)
            nodes[i].addr.transfer(profit*35/100);
        owner.transfer(address(this).balance);    // (2)
    }
    // (3) no withdraw() for participants
}
```

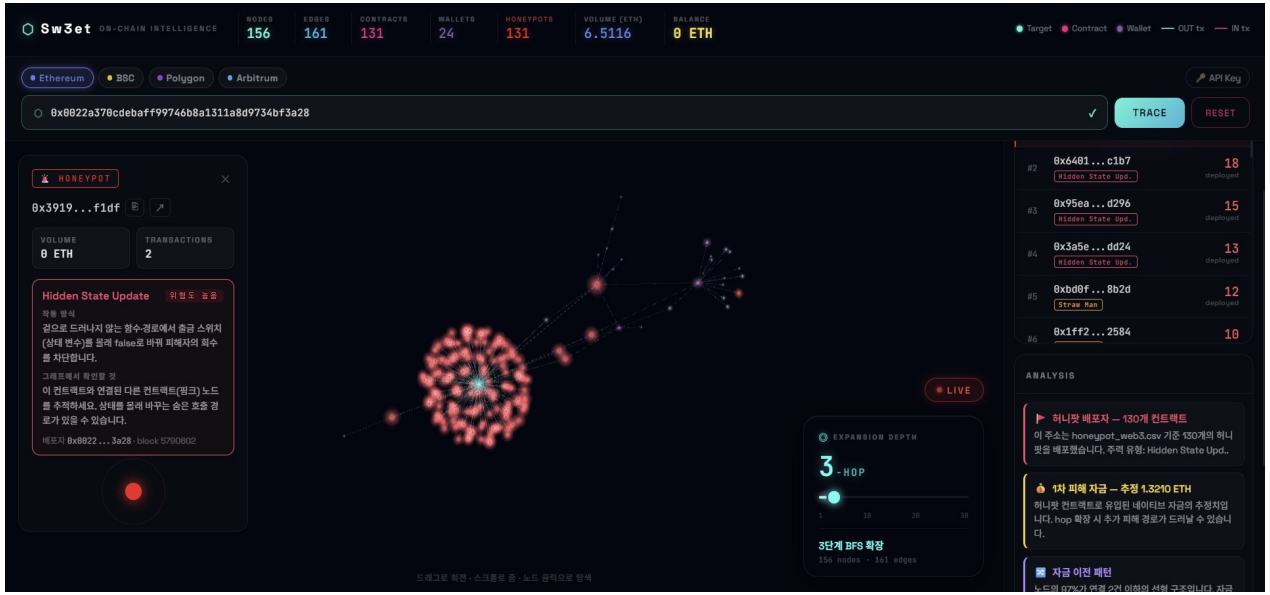
겉보기엔 '수익 분배' 폰지처럼 보이지만, 참여자는 입금만 가능하고 회수 경로가 없어 자금이 owner로 집중된다.

① 분배(distribute)는 owner만 호출 — 피해자는 트리거 불가

② 컨트랙트 잔액 전액이 owner로 — 자금 독식

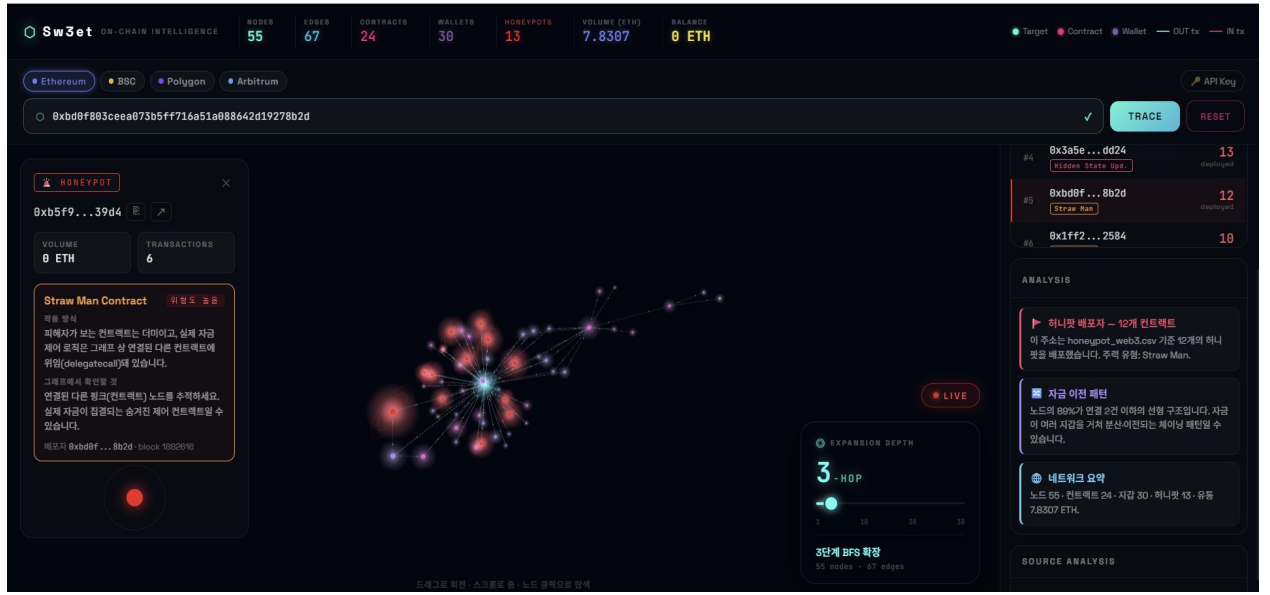
③ 사용자용 withdraw() 부재 — 입금하면 회수 불가

허니팟 유형별 온체인 구조 차이



Hidden State Update

· 붉은 허니팟 군집



Straw Man Contract

· 일반 컨트랙트·지갑이 섞인 분산 구조

05

SECTION

5
결론

- 계획 대비 결과
- 연구 결과물 · 기여 & 향후 연구

계획 대비 결과

제안서 목표	최종 구현	상태
Web3 허니팟 + CTI(위협 인텔리전스)	허니팟 탐지 + 온체인 추적 Dashboard	✓
능동형 공격 유인 · 행위 수집	실 APT 유인 제약 → 탐지 방법론으로 전환	△
온 · 오프체인 데이터 융합	온체인 코드 · 트랜잭션 분석 (오프체인 향후)	△
악성 주소 네트워크 식별	N단계 연결 추적(BFS) · 배포자 인프라 군집화	✓
기존 기법 대비 정량 입증	Stealth F1 95.1% · 오탐 0 · HoneyBadger 비교	✓

연구 결과물 & 향후

결과물

1. 심볼릭 + LLM(ReAct) + SMT 통합 파이프라인
   → 견고한 탐지 방법론 제시
2. 우회 허니팟(Stealth) 데이터셋 설계
3. 탐지를 추적 가능한 온체인 CTI로 연계한 Dashboard
4. 학술대회 논문 게재

향후 연구

- 오프체인 융합
- 데이터셋 확장
- 실시간 IoC 피드 (다크웹 / Telegram 등) / 다중 체인 확대

참고문헌

관련 연구

[1] Ferreira Torres, Steichen & State (2019). The Art of the Scam: Demystifying Honeypots in Ethereum Smart Contracts. USENIX Security. (arXiv:1902.06976)

[2] Zhang, Wang, Fu & Shi (2025). SCH-Hunter: A Taint-Based Hybrid Fuzzing Framework for Smart Contract Honeypots. Information, 16(5), 405.

[3] Luu, Chu, Olickel, Saxena & Hobor (2016). Making Smart Contracts Smarter. ACM CCS. (Oyente)

방법론 · 도구

[4] Yao et al. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. ICLR. (arXiv:2210.03629)

[5] de Moura & Bjørner (2008). Z3: An Efficient SMT Solver. TACAS, LNCS 4963.

[6] pyevmasm — Trail of Bits (crytic). github.com/crytic/pyevmasm

[7] OpenAI. GPT-5.3-Codex (실험 사용 LLM).

[8] Etherscan — 블록 익스플로러 · API. etherscan.io

위협 동향 · 통계

[9] Chainalysis (2025). 2025 Crypto Crime Report — 2024년 북한 연계 약 13.4억 달러 탈취(전체 61%).

[10] Chainalysis (2025). Record-Breaking Bybit Theft — 2025.2 약 15억 달러(401K ETH), DPRK 연계.

[11] FBI PSA (2025). Bybit 해킹을 Lazarus(TraderTraitor·APT38) 소행으로 확인.

[12] 뉴스 보도 — MBC뉴스 · 연합뉴스 · SBS뉴스 (이미지 인용).

제안서 단계 참고자료

[13] Sonatype. How North Korea-Backed Lazarus Group Is Weaponizing Open Source.

[14] ClearSky (2021). Attributing CryptoCore Attacks Against Crypto Exchanges to Lazarus.

[15] Binance. Honeypot Contract Risk Warning on Ethereum Smart Contracts.

감사합니다

이동하 · 김수민 · 한보윤 · Sw3et

[Q&A] 허니팟 8종 유형

1. Balance Disorder
2. Hidden State Update
3. Inheritance Disorder
4. Uninitialised Struct
5. Hidden Transfer
6. Straw Man Contract
7. Type Deduction Overflow
8. Skip Empty String Literal

[Q&A] 스마트 컨트랙트 동작

작성 → 배포 → 호출 → 검증 → 상태 기록