VeriOSS: using the Blockchain to Foster Bug Bounty Programs

Gabriele Costa, Andrea Canidio, Letterio Galletta
name.surname@imtlucca.it

DLT@Ancona 2020

The problem

- Open source software (OSS) is ubiquitous
  - Web browsers
  - Operating Systems
  - Libraries
- Many security-critical utilities
  - OpenSSL, KeePass, GnuPG, ...
- Vulnerabilities in OSS may spread out to many systems
  - Also closed source and proprietary software
- Security analysts can inspect the code
  - But vulnerability detection is hard and requires workforce and money

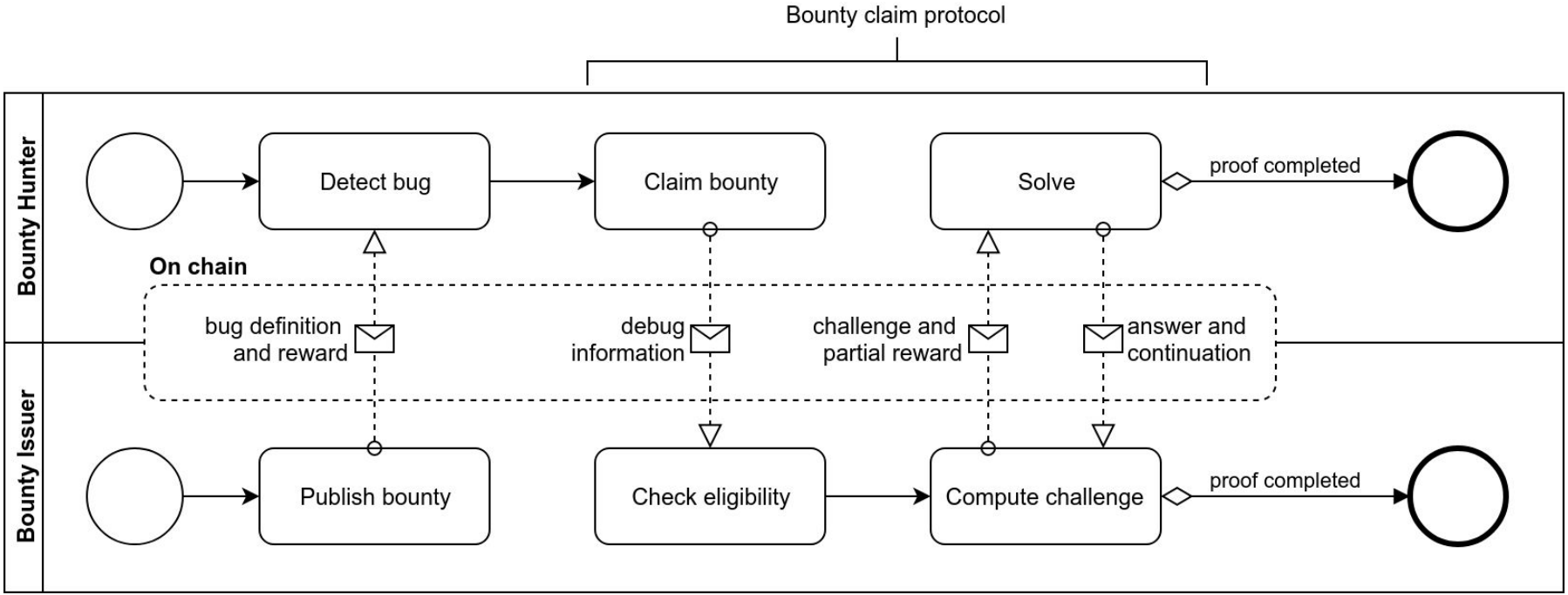
Bug bounty

- Bounty issuers offer rewards for each bug found
  - Rewards depend on type and impact
- Bounty hunters report their bugs and apply for a bounty
  - E.g., they disclose a vulnerability by filling a report
- Hunters can federate in large organizations of ethical hackers
  - E.g., **HackeOne**
- For OSS, bug bounties can be offered by third parties
  - E.g., public institutions (see **EU-FOSSA** and **EU-FOSSA 2**)

Market performance

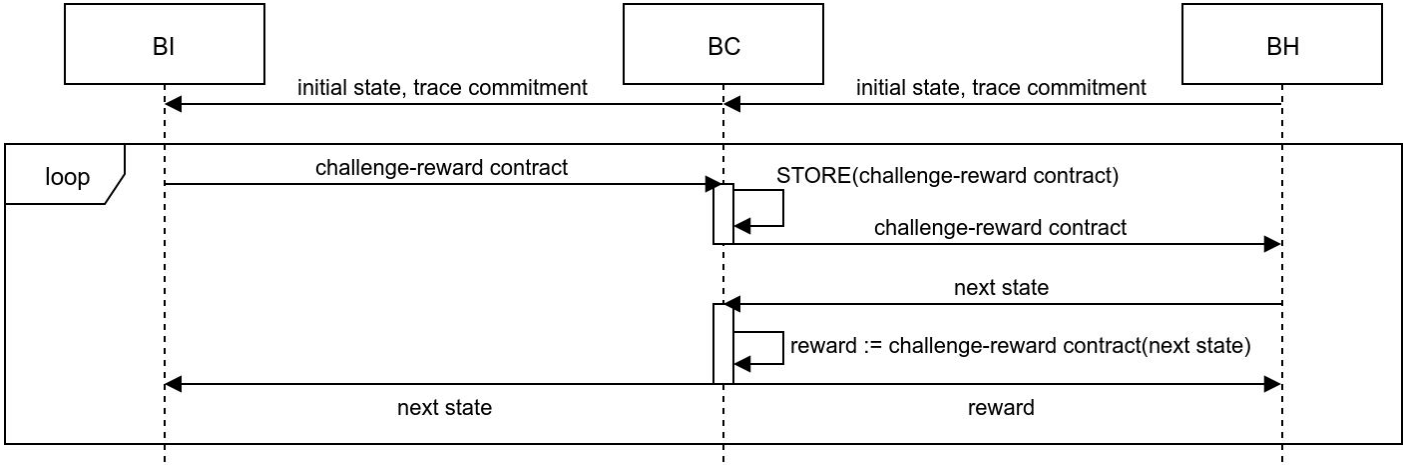
- Two markets: bug bounties vs. 0-day exploits
  - 0-day exploits can be sold on gray/black markets
  - E.g., zerodium
- If a bug can be turned into an exploit, the value increases significantly
  - **Google Chrome: from 5000$ to 300000$**
  - source: https://tinyurl.com/vlroo69
- Many bounty issuers also require an **evidence** of the bug
  - Typically an exploit
- Bug reporting requires extra effort
  - E.g., providing a remediation plan
- Bug eligibility is decided by the bounty issuer **after** disclosure
  - Limited bargaining power for the bounty hunter

VeriOSS


Bounty claim protocol
Bounty Hunter
Detect bug
Claim bounty
Solve
proof completed
On chain
bug definition and reward
debug information
challenge and partial reward
answer and continuation
Bounty Issuer
Publish bounty
Check eligibility
Compute challenge
proof completed

Bounty claim protocol

- On chain (BC), fair exchange protocol between issuer (BI) and hunter (BH)
  - Pay-per-Knowledge (P2K)
- BH initially claims to know an execution trace hitting a bug and makes a commitment
- BI checks the bug eligibility but cannot reproduce the trace



Challenge-response

- The loop in the bounty claim protocol is a **proof-of-knowledge** (PoK)
  - BH proves she knows the next segment of the execution trace
- BI publishes a smart contract with a **challenge**
  - The contract pays its balance to BH only if she can solve the challenge
- The challenge is a NP-hard problem if BH does not know the trace
  - E.g., providing a model for a satisfiable SMT formula
- **Backward symbolic execution** can support it!
  - BI and BH run a remote, backward, symbolic debug session

Conclusion and future work

- VeriOSS aim to support a fair, reliable and open market for bug bounty programs
- Solidity contracts are under development

NEXT STEPS

- Equilibria for partial rewards
- Alternative challenge-response implementations
- Formal verification of the protocols

Thank you

Extra: Implementation

- Challenge takes a state as a byte vector
  - Decommits and solve the challenge
  - Transfer in case of success
- Solve returns true only if the provided state is a valid model for the symbolic constraints
- Decommit checks the state hash
- BI can revoke the contract after a while

```
contract PartialReward {

  address public hunter = /* ... */;
  uint    public reward = /* ... */;
  uint    public expire = /* ... */;

  function challenge(bytes4[] state) public {
    if(decommit(state) && solve(state))
      hunter.transfer(reward);
  }

  function solve(bytes4[] state) private
  returns (bool)
  {
    if(state[0] <= 255) // not (a > 255)
      return false;
    return true;
  }

  function decommit(bytes4[] state) private
    returns (bool) { /* check state hash */ }

  function timeout() public {
    require(now >= expire);
    selfdestruct(this);
  }
}
```