REAL WORLD CTF

HACK THE REAL

RealWorldCTF/

国际〈CTF〉

网络安全

大赛/

A Novel Journey of Blockchain Security

Who am I

- Zhiniang Peng
- PhD. In Cryptography
- Cryptographer and Security Researcher
  @360 Core Security
- Interested in:
  - Date Security
  - Software Security

About the Topic

- Blockchain security
  - hot topic, emergency
- Attack surfaces of public blockchain:
  - Smart contract virtual machine
  - Consensus mechanism
  - P2P protocol
  - Smart contract
- Real world examples

Smart contract virtual machine

Some contract:

- Turing complete programming language.
- Run on every full node of the public chain.
- Virtual Machine.

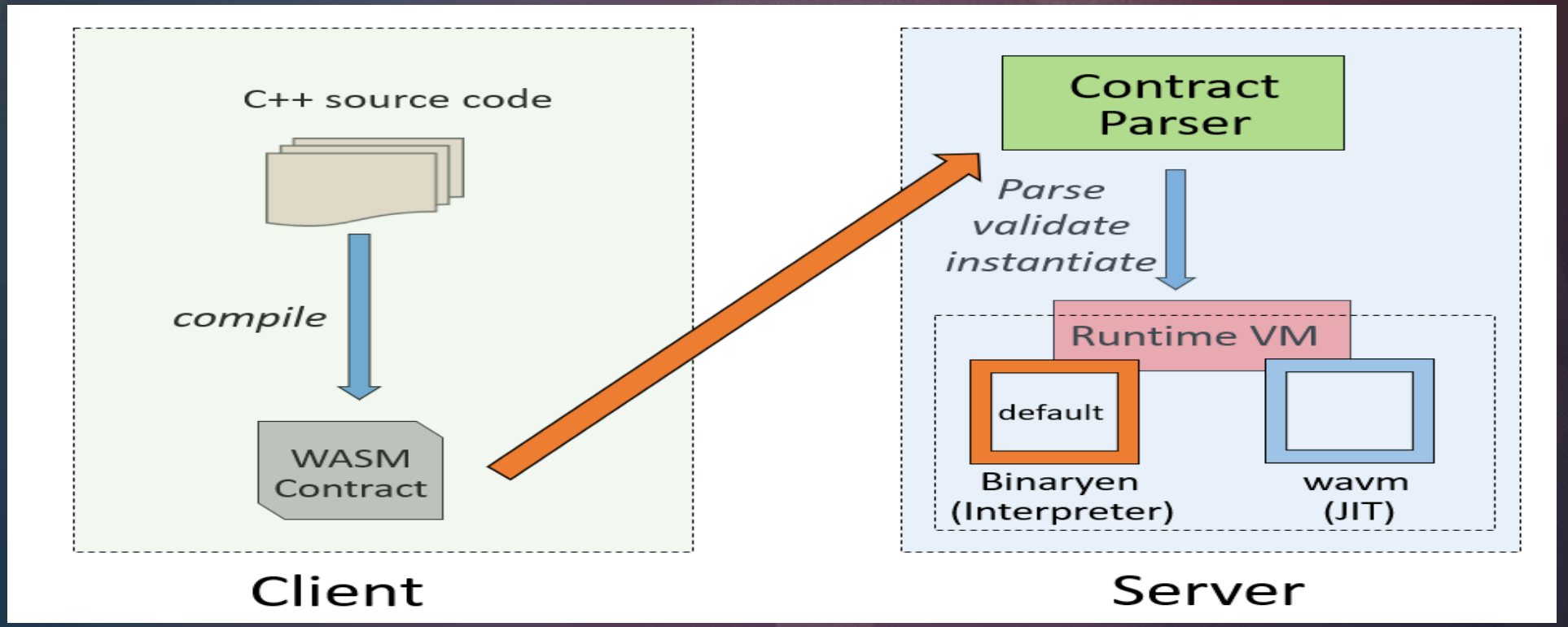
Security risk in nature:

- Suppose you can run your JavaScript on everyone’ s computer.

The ideal place to achieve every hackers’ dream:

- One bug to rule the world, or crash the world.

EOS VM RCE

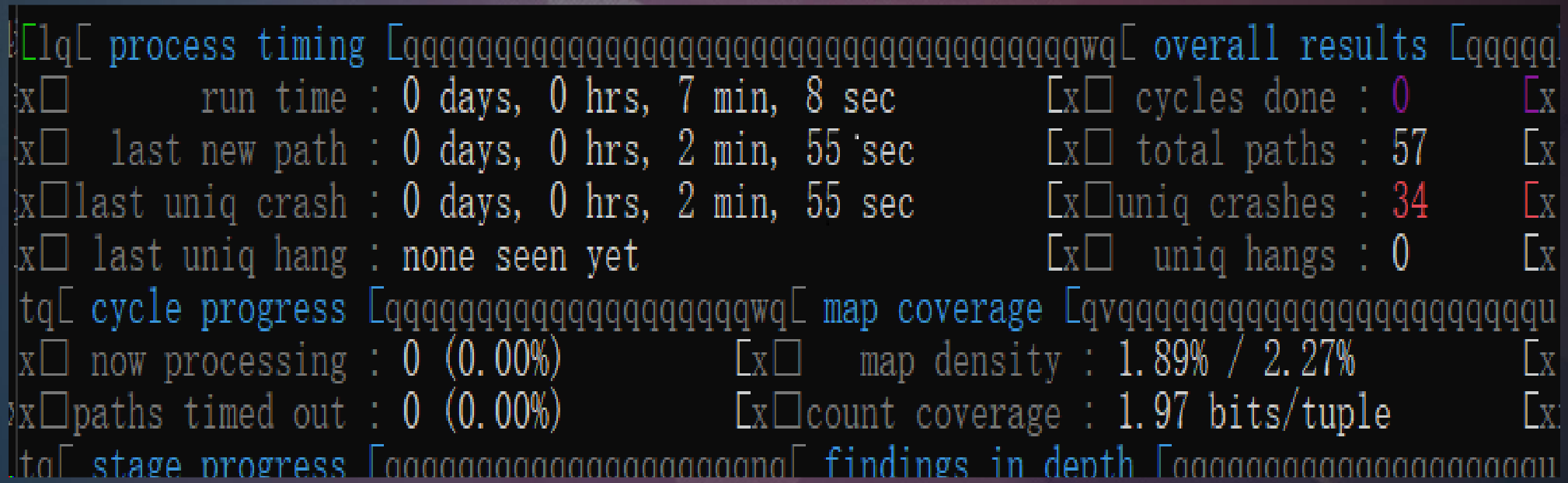


Out-of-Bounds Write + memory uninitialization➔RCE

http://blogs.360.cn/post/eos-node-remote-code-execution-vulnerability.html

Fuzzing EOS VM with AFL

34 unique crashes in 7 mins:



Results in:

Crash

Info leak

Fork

NEO VM DoS

Written in C# (memory safety)

Try catch everything

```
try
{
    ExecuteOp(opcode, CurrentContext);
}
catch
{
    State |= VMState.FAULT;
}
```

NEO VM DoS

```
private void SerializeStackItem(StackItem item, BinaryWriter writer)
{
    switch (item)
    {
        case ByteArray _:
            writer.Write((byte)StackItemType.ByteArray);
            writer.WriteVarBytes(item.GetByteArray());
            break;
        case VMArray array:
            writer.WriteVarInt(array.Count);
            foreach (StackItem subitem in array)
                SerializeStackItem(subitem, writer, serialized);
            break;
```

Attack: **Serialize(a[a])**

Stack-overflow, cannot be caught.

http://blogs.360.cn/post/neo-runtime_serialize-dos.html

Consensus mechanism

Crucial for a blockchain

Make sure everyone agree with the same blockchain.

May be insecure by design:

All PoS is vulnerable to long range attack.

PoW, may not secure as you think.

May have bugs in implementation:

Software bugs.

Fork in NEO dBFT

dBFT consensus mechanism:

Byzantine Fault Tolerance

POS+pBFT:

Choose a small number of committees by voting.

Use pBFT algorithm to reach consensus among committees.

Only guarantees a consensus between honest consensus nodes.

http://blogs.360.cn/post/NEO_dBFT_en.html

VRF bypassed in ONT vBFT

```
ECVRF_prove(y, x, alpha)
Input:
    y - public key, an EC point
    x - private key, an integer
    alpha - VRF input, an octet string
Output:
    pi - VRF proof, octet string of length m+3n
Steps:
1.  h = ECVRF_hash_to_curve(y, alpha)
2.  gamma = h^x
3.  choose a random integer nonce k from [0, q-1]
4.  c = ECVRF_hash_points(g, h, y, gamma, g^k, h^k)
5.  s = k - c*x mod q (where * denotes integer multiplication)
6.  pi = EC2OSP(gamma) || I2OSP(c, n) || I2OSP(s, 2n)
7.  Output pi
```

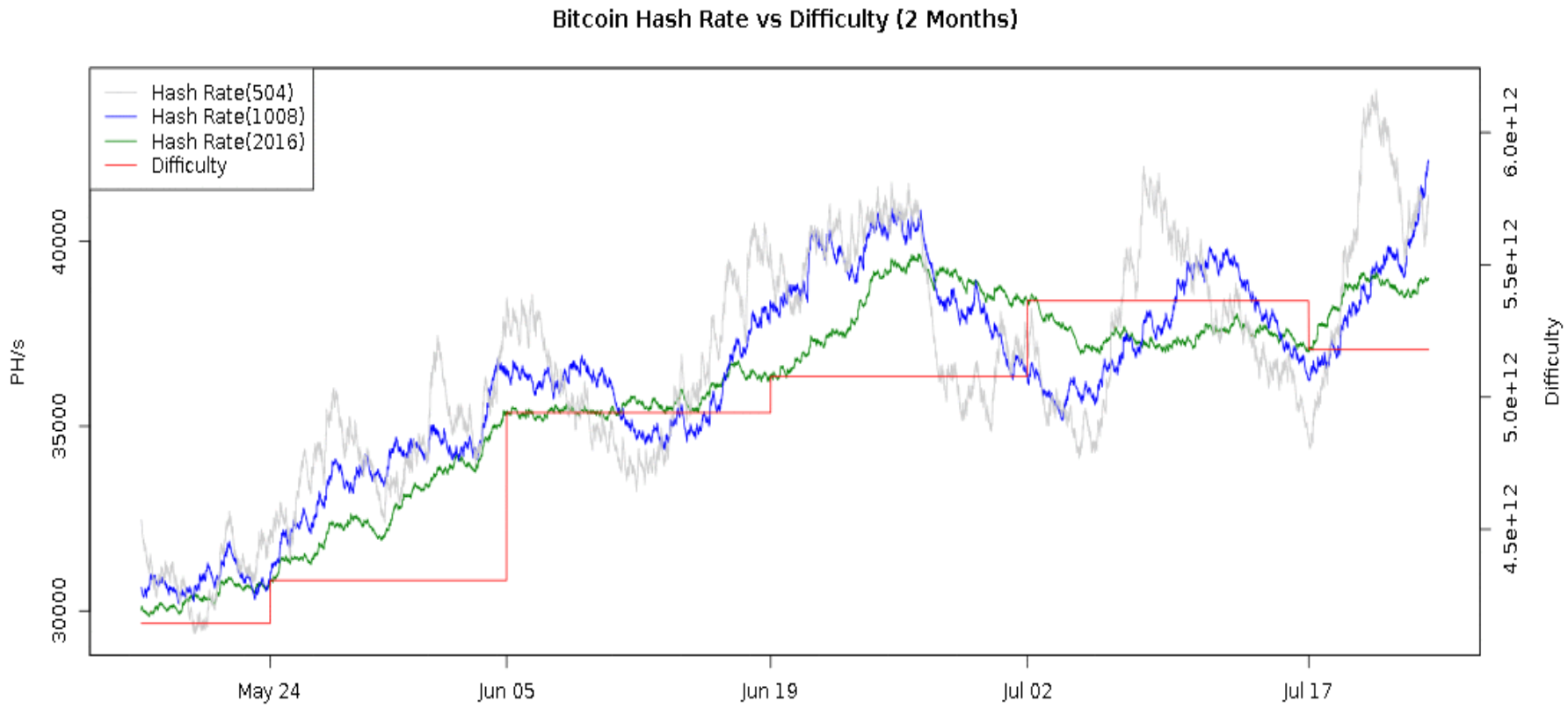
Implementing crypto primitive is easy to make mistake.

PoW doesn't mean secure

Difficulty adjustment algorithm:

Every M blocks ($M = 2016$ for Bitcoin) the difficulty is recalculated as

$$D_{i+1} = D_i \cdot \frac{M \cdot |\Delta|}{S_m}$$



Coin hopping attack

625191	Jul 16, 2018 7:47:38 AM
625190	Jul 16, 2018 7:45:39 AM
625189	Jul 16, 2018 7:45:38 AM
625188	Jul 16, 2018 7:45:37 AM
625187	Jul 16, 2018 7:45:36 AM
625186	Jul 16, 2018 7:45:35 AM
625185	Jul 16, 2018 7:45:34 AM
625184	Jul 16, 2018 7:43:31 AM
625183	Jul 16, 2018 7:43:30 AM
625182	Jul 16, 2018 7:43:29 AM
625181	Jul 16, 2018 7:43:28 AM
625180	Jul 16, 2018 7:41:29 AM
625179	Jul 16, 2018 7:39:26 AM
625178	Jul 16, 2018 7:37:24 AM

This attack can not be eliminate.

DAA simulator: https://github.com/edwardz246003/DAA_simulator

P2P protocol

Peer to peer network:

Every node is both a server and a client.

Use Peer discovery mechanism to find all the peers in the network.

Again, one bug to kill them all.

RPC protocol:

Execute a specified procedure with supplied parameters.

May have some dangerous procedures.

Should not be accessed by untrusted users.

Json Parse in EOS, NEO

```
-           internal new static JArray Parse(TextReader reader)
+           internal new static JArray Parse(TextReader reader, int max_nest)
            {
+               if (max_nest < 0) throw new FormatException();
                SkipSpace(reader);
                if (reader.Read() != '[') throw new FormatException();
                SkipSpace(reader);
                JArray array = new JArray();
                while (reader.Peek() != ']')
                {
                    if (reader.Peek() == ',') reader.Read();
-                   JObject obj = JObject.Parse(reader);
+                   JObject obj = JObject.Parse(reader, max_nest - 1);
                    array.items.Add(obj);
                    SkipSpace(reader);
                }
```

data = '{"' + '}' * 0x10100 + '":' + '{"x":' * 0x10000 + '"}'

RPC security of NEO

Supported RPC function:

- Dump private key
- transfer money
- ……

Monitor NEO network with a crawler. (Aug. 2018)

3931 full nodes in total.

287 nodes open RPC without authentication.

Smart contract

Popular in CTF games:

Ethereum smart contract vulnerability is Popular in CTF games.

Integer overflow, random number vulnerability, logical bugs ……

Many security accidents in real world:

Ethereum and EOS smart contract

Gambling games , Tokens, financial scam …….

Cryptocurrency make bank robbery great again.

EOS asset multiplication integer overflow

```
asset& operator*=( int64_t a ) {
  eosio_assert( a == 0 || (amount * a) / a == amount,
  eosio_assert( -max_amount <= amount, "multiplication
  eosio_assert( amount <= max_amount, "multiplication
  amount *= a;
  return *this;
}
```

Not a single contract, but a official template for issuing token.

3 bugs in 5 lines: http://blogs.360.cn/post/eos-asset-multiplication-integer-overflow-vulnerability.html

Exploited by attacker after we reported to EOS.

Dice2Win fairness vulnerabilities

Commit-and-reveal is popular in gambling contract:

It doesn't guarantee fairness.

Selective-abort attack applies to all those contract.

Hard to generate random number in smart contract.

Communication models in blockchain is different:

Fork and rollback exist.

Secure Multi-parity Computation cannot be directly applied to smart contract.

Details: http://blogs.360.cn/post/Fairness_Analysis_of_Dice2win_EN.html

THANKS