

# Reverse Engineering dan Industri Security

Yohanes Nugroho

# Saya akan membahas...

Perkenalan apa itu reverse engineering

Berbagai profesi yang berhubungan dengan Reverse Engineering

Contoh melakukan reverse engineering aplikasi mobile

Masa depan Reverse Engineering di era AI

# Sebelum membahas masa depan RE

- perlu memahami apa itu RE
- perlu memahami pentingnya RE dalam security
- pelu paham tantangan apa yang ada saat ini

# Reverse Engineering Pokémon GO Plus

November 21, 2018

hardware, reverse-engineering, writeup

7 Comments

TL;DR; You can clone a Pokemon GO Plus device that you own. I have managed to get the certification algorithm. However, there is a per device blob used (specific to a Bluetooth Mac Address) for key generation. I have not figured out how you can generate your own blob and key. Using other's people blob may be blacklisted in the future (or Niantic may ban your account).

[Pokemon GO Plus](#), (which I will refer from now on as PGP) is a wearable Bluetooth Low Energy (BLE) device to be used with the Pokemon GO game for Android or iOS. There have been many attempts to clone this device, but only Datel seems to figure out the algorithm, while the other clones are cloning the exact hardware and firmware.

# Bug Mandiri e-Money Isi Ulang (November 2015)



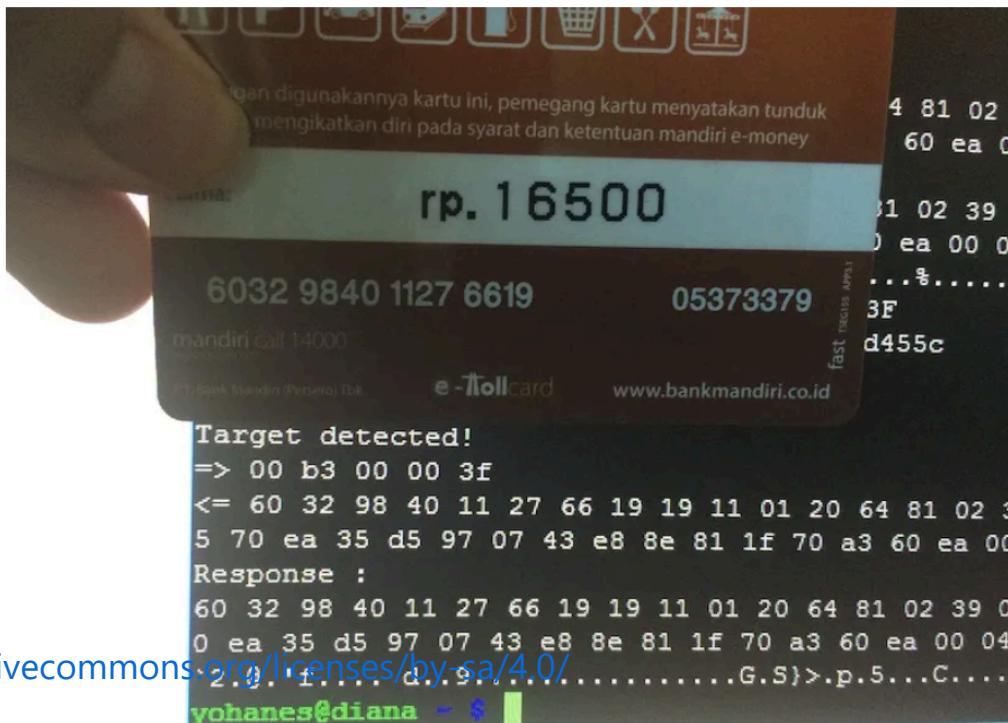
yohanes

02/01/2016

elektronik, reverse-engineering, security, Technology

Jika Anda belum membaca, sebaiknya baca dulu pengantar seri ini: [Mencari dan melaporkan bug security](#). Perlu dicatat: bug ini sudah dilaporkan (akhir November 2015), sudah (sebagian besar) diperbaiki. Posting ini hanya untuk pelajaran bersama.

Kartu [mandiri e-money](#) adalah stored value card dengan teknologi NFC. Tadinya proses isi ulang tidak bisa dilakukan dari ponsel, tapi sejak sekitar Februari 2015, aplikasi isi ulang diluncurkan untuk ponsel Android dengan NFC.



CARI KONTEN

ENHANCED BY Google

SEARCH

Blog ini [bernenaga surya](#)

PAGES

- [About Us](#)
- [Alkitab](#)
- [Arsip](#)
  - [Arsip Tulisan Risna](#)
  - [Arsip Tulisan Yohanes](#)
- [Tanya Jawab Reverse Engineering](#)

Untuk menghubungi kami, gunakan fitur kirim pesan di page Yohanes & Risna. Kami memakai Email, Telegram, WhatsApp, Line, Skype dan berbagai aplikasi lain tapi hanya untuk yang sudah kami kenal, untuk pengunjung blog, silakan gunakan fitur ini untuk mengirim pesan pada kami.

[Yohanes & Risna](#)

SITE KAMI YANG LAIN

- [Geneus DNA Indonesia](#)
- [Google Aja](#)
- [Yohan.es](#)
- [Gitar Pemula](#)

# Membedah e-KTP



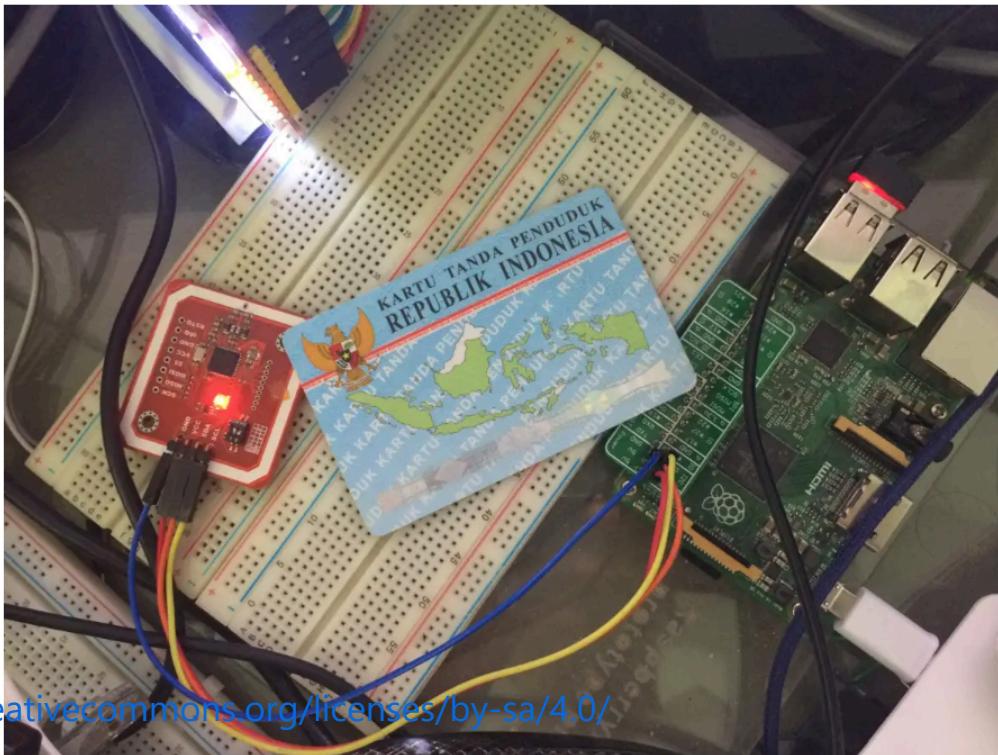
yohanes

10/06/2018

elektronik, oprekan,  
Programming, reverse-  
engineering, security

Posting ini sekedar membahas tentang kartu tanda penduduk elektronik (e-KTP). Sampai saat ini saya belum pulang ke Indonesia untuk mengurus e-KTP karena KTP lama masih berlaku. Waktu orang tua saya datang ke sini tahun lalu saya sudah sempat ngoprek e-KTP mereka sedikit, dan sekarang selagi mereka berkunjung saya teruskan dan tuliskan hasil eksplorasi saya.

Sebagian isi tulisan ini didapat dari reverse engineering, dan sebagian lagi dari berbagai informasi yang tersebar di Internet. Ada juga bagian yang merupakan spekulasi saya dari informasi yang ada.



## CARI KONTEN

ENHANCED BY Google

SEARCH

Blog ini [bertenaga surya](#)

## PAGES

- [About Us](#)
- [Alkitab](#)
- [Arsip](#)
  - [Arsip Tulisan Risna](#)
  - [Arsip Tulisan Yohanes](#)
- [Tanya Jawab Reverse Engineering](#)

Untuk menghubungi kami, gunakan fitur kirim pesan di page Yohanes & Risna. Kami memakai Email, Telegram, WhatsApp, Line, Skype dan berbagai aplikasi lain tapi hanya untuk yang sudah kami kenal, untuk pengunjung blog, silakan gunakan fitur ini untuk mengirim pesan pada kami.

*Yohanes & Risna*

## SITE KAMI YANG LAIN

- [Geneus DNA Indonesia](#)
- [Google Aja](#)
- [Yohanes](#)

# Reverse Engineering Model Handwritten Digits Recognition Aplikasi Mobile SIREKAP



yohanes

23/02/2024

ai, android, reverse-engineering

3 Komentar

Di tulisan ini saya akan melakukan reverse engineering aplikasi SIREKAP 2024, mengekstrak model TensorFlow Lite untuk inference, lalu membuat aplikasi Android untuk mengetes model tersebut. Saya bandingkan juga dengan model sederhana yang jadi contoh tutorial tensorflow lite.

## Handwritten Digits Recognition

Persoalan pengenalan digit tulisan tangan merupakan hal paling dasar (semacam *Hello World*) di pelajaran AI modern. Ini merupakan subset dari [OCR](#) (optical character recognition), di mana scopenya hanya digit saja.

# Apa itu reverse engineering?

Memahami suatu hal dengan membongkar hal tersebut:

- Reverse engineering resep masakan
- Reverse engineering design mesin
- Reverse engineering layout PCB
- Reverse engineering kode program

Presentasi ini hanya membahas membongkar kode program

# Untuk apa?

- Membongkar malware (untuk berbagai tujuan: forensik, pembuatan anti malware, dsb)
- Mencari bug security
- Memahami algoritma tertentu atau protokol tertentu
- Modifikasi program/game (cheat)
- Mengetahui keamanan sebuah program

# RE sangat penting

- Tanpa RE tidak bisa menemukan bug low level
- Tanpa RE tidak bisa membuat exploit low level
- Tanpa RE tidak bisa menganalisis virus, membuat anti malware, decryptor, dsb

# RE untuk kejahatan

- Konten WebRIP hasil dari reverse engineering Widevine DRM
- Game dan software bajakan
- Game mod dan cheat (catatan: ada banyak modifikasi yang legal)
- Modifikasi aplikasi misalnya Gojek

# Apakah RE software illegal?

Tergantung hukum negara tempat melakukan RE

Biasanya dibolehkan untuk tujuan positif, misalnya interoperabilitas

RE malware merupakan hal yang legal

# Pekerjaan RE

- *Saat ini* sulit mencari pekerjaan *legal* yang hanya berfokus pada RE di Indonesia
- Pekerjaan pentest dan forensik kadang berurusan dengan RE, tapi bukan fokus utama
- Pekerjaan RE banyak tersedia di luar negeri (paling dekat: Singapura)
  - Reversing malware
  - Mencari bug di software/hardware IOT

# Source code dan Machine code

Source code program dalam bahasa tertentu bisa dijalankan langsung (contoh: JavaScript, Python, dsb)

Untuk efisiensi, biasanya kode program dikompilasi menjadi kode mesin

Walau tujuan utamanya untuk efisiensi, ini mempersulit reverse engineering

# Tool

Tool bergantung pada: apa yang ingin direverse engineer

- Aplikasi mobile: apktool, jadx
- Aplikasi desktop: IDA, Binary Ninja, Ghidra
- Device IOT: IDA, binary ninja, ghidra, logic analyzer

Bergantung juga pada proteksi yang ada: butuh debugger atau tool lain

# Blackbox RE

Jika kode program tidak tersedia (misalnya ada di remote device, atau di hardware yang diproteksi), maka blackbox RE bisa dilakukan:

- Sekedar mengamati program ketika dijalankan
- Mengubah input dan melihat perubahan output
- Kelemahan: Sulit menemukan hal tersembunyi, misalnya jika program berubah perilakunya para Jumat Kliwon tanggal 13, tidak akan terdeteksi dengan hanya mencoba-coba

# Whitebox RE

Jika kode program tersedia, maka cara ini yang dilakukan:

- Memahami program dengan membaca kodennya
- Kadang source code tersedia dan bisa dibaca (contoh: kode Python, PHP) tapi dibuat menjadi rumit (obfuscated)
- Sering kali source code sudah diterjemahkan menjadi bahasa mesin

# Mendapatkan program

Kode program desktop mudah dicopy (cukup cari filenya).

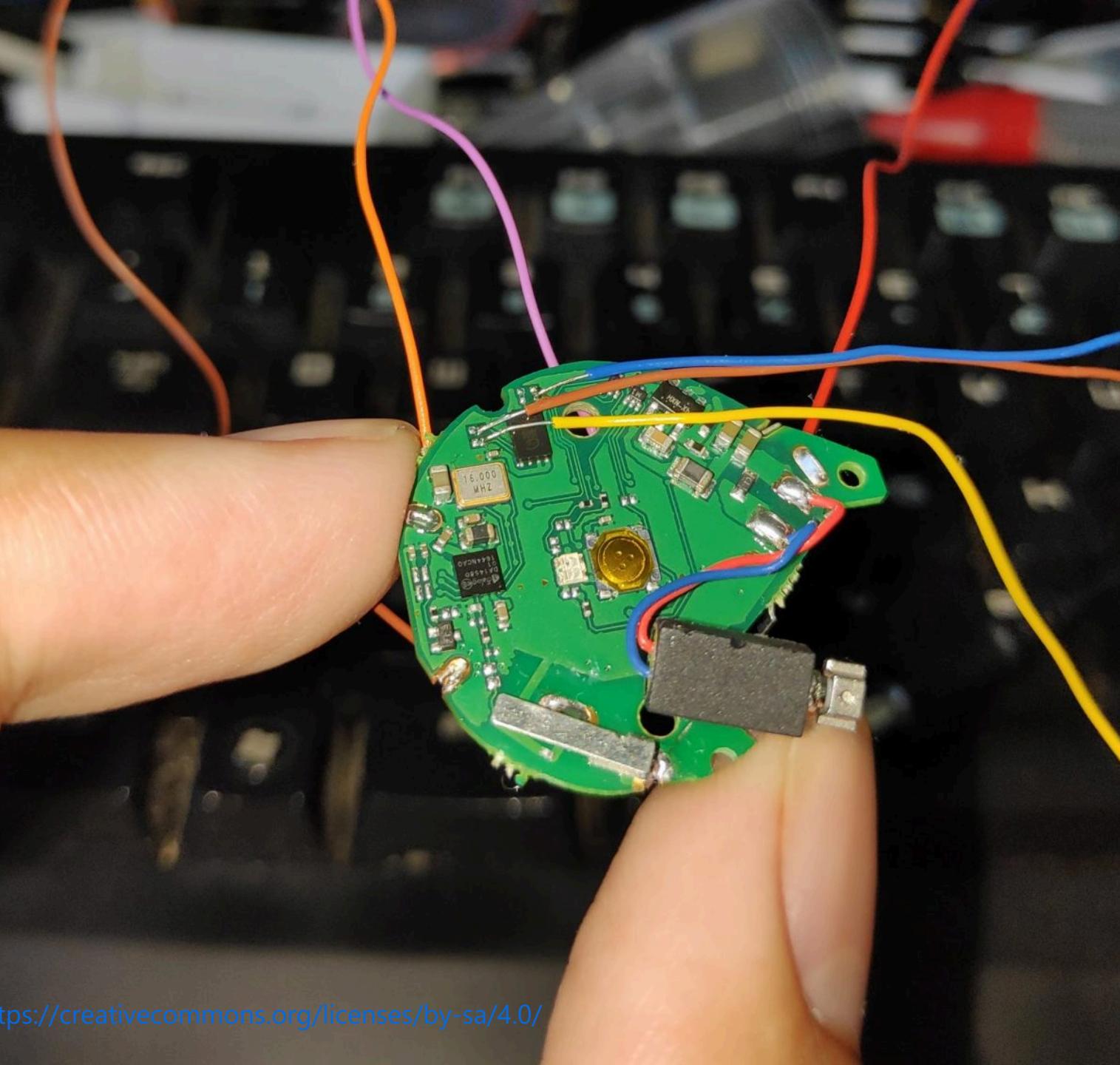
Kode program mobile bisa diekstrak dengan mudah untuk versi android, tapi untuk iOS butuh device yang dijailbreak.

Kode malware kadang sifatnya fileless, hanya ada di memori (perlu diekstrak dari memori).

Kode program dalam chip kadang perlu diekstrak dengan hardware tertentu.

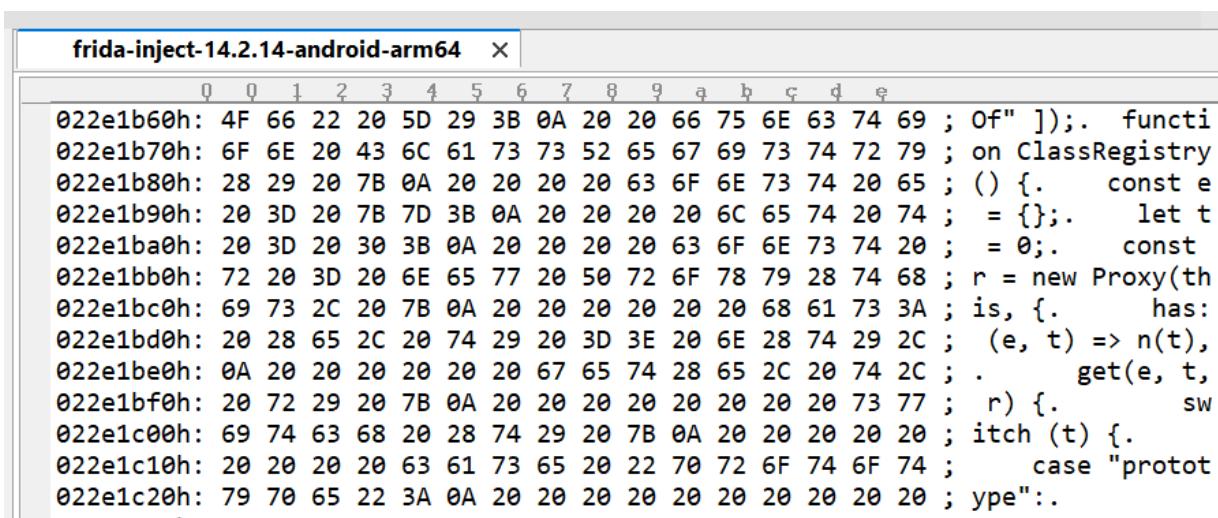
# Reverse engineering embedded system

- Mengekstrak kode program (kadang sangat mudah, kadang sangat sulit)
- Mengenal dan mendapatkan dokumentasi berbagai chip yang ada di hardware
- Pada dasarnya kembali ke reversing software (firmware yang berjalan pada chip)



# Hex Editor

- Untuk Aplikasi yang sangat sederhana, sudah cukup untuk membongkar program
- Kita melihat representasi heksadesimal dan string-string yang terbaca
- Apa yang dicari: string yang tidak biasa (kemungkinan password), URL, email



The screenshot shows a hex editor window titled "frida-inject-14.2.14-android-arm64". The window displays a memory dump with columns for address, hex value, and ASCII representation. The ASCII column contains readable text, including strings like "function", "on ClassRegistry", "const e", "let t", "const", "new Proxy(th", "is, {. has:", "(e, t) => n(t)", ". get(e, t", "itch (t) {", "case "protot", and "ype":". Above the ASCII column, there are additional columns labeled with characters (q, Q, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f) which likely correspond to different character encodings or analysis tools.

	0	Q	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
022e1b60h:	4F	66	22	20	5D	29	3B	0A	20	20	66	75	6E	63	74	69	; Of" ]);. functi
022e1b70h:	6F	6E	20	43	6C	61	73	73	52	65	67	69	73	74	72	79	; on ClassRegistry
022e1b80h:	28	29	20	7B	0A	20	20	20	63	6F	6E	73	74	20	65	; () {. const e	
022e1b90h:	20	3D	20	7B	7D	3B	0A	20	20	20	20	6C	65	74	20	74	; = {});. let t
022e1ba0h:	20	3D	20	30	3B	0A	20	20	20	20	63	6F	6E	73	74	20	; = 0);. const
022e1bb0h:	72	20	3D	20	6E	65	77	20	50	72	6F	78	79	28	74	68	; r = new Proxy(th
022e1bc0h:	69	73	2C	20	7B	0A	20	20	20	20	20	68	61	73	3A	; is, {. has:	
022e1bd0h:	20	28	65	2C	20	74	29	20	3D	3E	20	6E	28	74	29	2C	; (e, t) => n(t),
022e1be0h:	0A	20	20	20	20	20	67	65	74	28	65	2C	20	74	2C	; . get(e, t,	
022e1bf0h:	20	72	29	20	7B	0A	20	20	20	20	20	20	20	73	77	; r) {. sw	
022e1c00h:	69	74	63	68	20	28	74	29	20	7B	0A	20	20	20	20	; itch (t) {.	
022e1c10h:	20	20	20	20	63	61	73	65	20	22	70	72	6F	74	6F	74	; case "protot
022e1c20h:	79	70	65	22	3A	0A	20	20	20	20	20	20	20	20	20	; ype":.	

# Disassembler

Membaca kode biner dan menampilkan sebagai assembly

Assembly tergantung arsitektur CPU (Intel, Arm, RISC-V, PowerPC, MIPS, Loongson, dsb)

Kode sederhana dan kecil bisa dibaca, tapi kode kompleks akan sulit dibaca

# Disassembler

```
000401190      push   rbp
000401191      mov    rbp, rsp
000401194      sub    rsp, 20h
000401198      mov    [rbp+var_8], rdi
00040119C      mov    [rbp+var_C], esi
00040119F      mov    rdi, offset aBubbleSortMeth ; "Bubble sort method\n"
0004011A9      mov    al, 0
0004011AB      call   _printf
0004011B0      mov    [rbp+var_10], 0
0004011B7
0004011B7 loc_4011B7:           ; CODE XREF: sub_401190+B7↓j
0004011B7      |      mov    eax, [rbp+var_10]
0004011BA      cmp    eax, [rbp+var_C]
0004011BD      jge   loc_40124C
0004011C3      mov    eax, [rbp+var_10]
0004011C6      add    eax, 1
0004011C9      mov    [rbp+var_14], eax
-----
```

# Decompiler

Mengembalikan kode sebisa mungkin ke high-level language.

Dalam kode yang dirilis, nama fungsi, variabel, dsb tidak semua bisa direkonstruksi.

Perlu campur tangan manual agar benar-benar terbaca.

Agar terbayang, akan saya berikan contoh seperti apa RE sebuah fungsi sangat sederhana.

# Source asli

```
static void bubble_sort(int *numbers, int count)
{
    printf("Bubble sort method\n");
    int i,j;
    for (i = 0; i < count; i++) {
        for (j=i+1; j < count; j++) {
            if (numbers[i] > numbers[j]) {
                int tmp = numbers[i];
                numbers[i] = numbers[j];
                numbers[j] = tmp;
            }
        }
    }
}
```

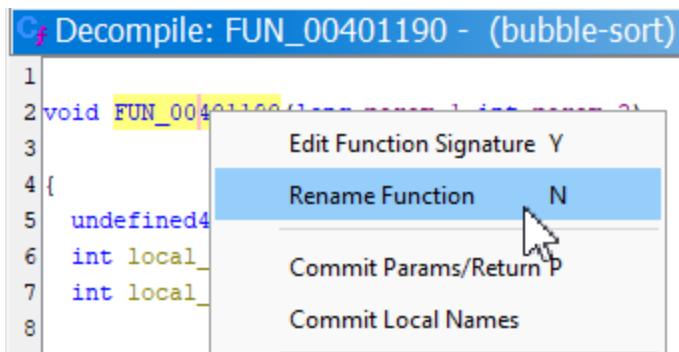
# Hasil dekompilasi (mentah)

```
void FUN_00401190(long param_1,int param_2)
{
    undefined4 uVar1;
    int local_1c;
    int local_18;

    printf("Bubble sort method\n");
    for (local_18 = 0; local_1c = local_18, local_18 < param_2; local_18 = local_18 + 1) {
        while (local_1c = local_1c + 1, local_1c < param_2) {
            if (*(_int *)param_1 + (long)local_1c * 4) < *(_int *)param_1 + (long)local_18 * 4)) {
                uVar1 = *(_undefined4 *)param_1 + (long)local_18 * 4);
                *(_undefined4 *)param_1 + (long)local_18 * 4) =
                    *(_undefined4 *)param_1 + (long)local_1c * 4);
                *(_undefined4 *)param_1 + (long)local_1c * 4) = uVar1;
            }
        }
    }
    return;
}
```

# Rename function

Kita tahu dari string `Bubble sort method`, bahwa ini adalah bubble sort



The screenshot shows the OllyDbg debugger interface. A context menu is open over a function call instruction at address `00401190`. The menu options are:

- Edit Function Signature Y
- Rename Function N** (The option is highlighted with a blue background)
- Commit Params/Return P
- Commit Local Names

The assembly code visible in the window is:

```
1
2 void FUN_00401190()
3 {
4     undefined4 local_
5     int local_
6     int local_
7 }
```

# Retype Variable

Berdasarkan pengetahuan bagaimana pointer di C bekerja, operasi yang dilakukan adalah array dengan elemen berukuran 4 byte (jadi kita gunakan `int*` ).

A screenshot of a code editor showing a context menu for variable retype. The menu items are:

- Edit Function Signature Y
- Rename Variable N
- Auto Create Structure Shift+Ope
- Retype Variable Y** (highlighted in blue)

```
1
2 void bubble_sort(long param_1 int param_2)
3
4 {
5     undefined4 uVar1;
6     int local_lc;
7     int local_18;
8 }
```

# Sekarang fungsi mulai terbaca

```
void bubble_sort(int *param_1,int param_2)
{
    int iVar1;
    int local_1c;
    int local_18;

    printf("Bubble sort method\n");
    for (local_18 = 0; local_1c = local_18, local_18 < param_2; local_18 = local_18 + 1) {
        while (local_1c = local_1c + 1, local_1c < param_2) {
            if (param_1[local_1c] < param_1[local_18]) {
                iVar1 = param_1[local_18];
                param_1[local_18] = param_1[local_1c];
                param_1[local_1c] = iVar1;
            }
        }
    }
    return;
}
```

## Rename variabel-variabel lain

- indeks array adalah `i` dan `j`
- temporary tipenya adalah `int`
- perhatikan bahwa loop `for` yang kedua tetap dikenali sebagai `while` (dan ekivalen dengan source aslinya)

# Hasil akhir

```
void bubble_sort(int *array,int count)
{
    int j;
    int i;
    int temp;

    printf("Bubble sort method\n");
    for (i = 0; j = i, i < count; i = i + 1) {
        while (j = j + 1, j < count) {
            if (array[j] < array[i]) {
                temp = array[i];
                array[i] = array[j];
                array[j] = temp;
            }
        }
    }
    return;
}
```

# Source code

- Mendapatkan source code hanya satu langkah saja, memahami kode adalah langkah yang lebih sulit
- Sudah banyak software di github, tapi berapa banyak programmer yang bisa paham?
- Contoh lain: Kernel Linux juga sudah terbuka source codenya bahkan ada banyak buku yang menjelaskan tiap komponennya, apakah bisa dimengerti kebanyakan programmer?
- memahami kode bisa butuh waktu berjam-jam, berhari-hari, dan bahkan berbulan-bulan

# Bahasa lain

Berbagai teknologi punya decompilernya masing-masing

Contoh: Android dengan JADX-GUI

.NET memiliki beberapa decompiler

Issues: 11 warnings

PowderActivity R SDLAudioManager

```
        throw new UnsupportedOperationException("Method not decompiled: org.libsdl.app(SDLAudioManager).open(boolean, int, int, int, int):int[]");  
    }  
  
    public static int[] audioOpen(int i, int i2, int i3, int i4) {  
        return open(false, i, i2, i3, i4);  
    }  
  
    public static void audioWriteFloatBuffer(float[] fArr) {  
        if (mAudioTrack == null) {  
            Log.e(TAG, "Attempted to make audio call with uninitialized audio!");  
            return;  
        }  
        int i = 0;  
        while (i < fArr.length) {  
            int write = mAudioTrack.write(fArr, i, fArr.length - i, 0);  
            if (write > 0) {  
                i += write;  
            } else if (write == 0) {  
                try {  
                    Thread.sleep(1L);  
                } catch (InterruptedException unused) {  
                }  
            } else {  
                Log.w(TAG, "SDL audio: error return from write(float)");  
                return;  
            }  
        }  
    }  
  
    public static void audioWriteShortBuffer(short[] sArr) {  
        if (mAudioTrack == null) {  
            Log.e(TAG, "Attempted to make audio call with uninitialized audio!");  
            return;  
        }  
        int i = 0;  
        while (i < sArr.length) {  
            int write = mAudioTrack.write(sArr, i, sArr.length - i);  
            if (write < 0) {  
                i = 0;  
            } else if (write == 0) {  
                try {  
                    Thread.sleep(1L);  
                } catch (InterruptedException unused) {  
                }  
            }  
        }  
    }  
}
```

## Tidak selalu bisa ideal

Dalam contoh di atas, fungsi hampir kembali 99% seperti semula, namun ini jarang terjadi:

- Tidak selalu ada string yang bisa membantu kita menebak nama fungsi
- Optimasi compiler bisa menggabung banyak fungsi menjadi satu
- Programmer membuat kode yang aneh/tidak efisien sehingga sulit dimengerti
- Tool obfuscator digunakan untuk membuat fungsi sulit dipahami

# Obfuscated

```
22
23     while( true ) {
24         while( true ) {
25             while( true ) {
26                 while( true ) {|
27                     while( true ) {
28                         while( true ) {
29                             while( true ) {
30                                 while( true ) {
31                                     while( true ) {
32                                         while( local_2c == -0x4d98a826) {
33                                             local_24 = local_20 + 1;
34                                             local_2c = 0x629e5c5d;
35                                             if ((DAT_0040407c * (DAT_0040407c + -1) & 1U) == 0 ||
36                                                 DAT_00404088 < 10) {
37                                                 local_2c = 0xd4536596;
38                                             }
39                                         }
40                                         if (local_2c != -0x4589fc89) break;
41                                         local_2c = 0x5b785d81;
42                                         if ((local_9 & 1U) != 0) {
43                                             local_2c = 0xdcc1b8f9;
44                                         }
45                                     }
```

# WHEN YOU ENCOUNTER A 20MB OBFUSCATED VB SCRIPT

@tank.sinatra



## Ransomware dan RE

Versi awal ransomware banyak memiliki bug, dengan RE bisa dicari keynya (atau kelemahan ketika menghasilkan keynya).

Saat ini ransomware semakin bagus, kebanyakan tidak memiliki kelemahan lagi

Ada ransomware pemula yang masih salah dalam implementasi enkripsi, sehingga bisa ditangani dengan melakukan RE .

# Malware dan RE

Dengan RE, kita bisa mengetahui:

- Eksplot apa yang digunakan oleh malware (jika ada)
- Kemungkinan asal malware (jika pembuatnya tidak teliti)
- Bagaimana melakukan aksi defensif (misalnya dengan mengetahui bagaimana malware manargetkan suatu sistem, kita bisa memblok aksesnya)

Contoh kasus: Stuxnet yang menargetkan PLC (programmable logic controller)

```
109  
110  <!--Featured Content Section-->  
111  <div class="container">  
112    <div class="row">  
113      <div class="col-md-4"></div>  
114      <div class="col-md-4"> FEATURED CONTENT <h2> FEATURED CONTENT </h2> <br class="small-->  
115      <div class="col-md-4"></div>  
116    </div>
```

# How I reversed a NodeJS malware and found the author



The Devops Guy

[Follow](#)



Jan 30 · 4 min read



To give a bit of context, I am a Discord admin on a small server about development, and we recently got a report from one of our users that someone was trying to get him to download an EXE file.

## Contoh kisah RE

Ini tentang Aplikasi pembaca buku/majalah Indonesia untuk iOS (saat ini aplikasinya sudah tidak ada lagi)

- Banyak orang tidak mau membeli atau sharing account
- Banyak yang ingin membajak/mengcopy PDF majalahnya

Mari kita lihat perkembangan aplikasi dan securitynya, supaya bisa melihat peran RE dalam membongkar (dan juga mengamankan) sebuah aplikasi.

## Contoh kisah RE: Level 0

Dulu ketika dilaunch, aplikasi memakai In App Purchase (IAP) tapi caranya tidak aman (tidak mengecek server).

Di device yang dijailbreak, bisa mendownload semua majalah gratis dengan menginstall program yang akan memberitahu bahwa "pembelian sukses".

Di level ini sekedar coba-coba melihat behavior program.

Ini sekedar blackbox RE, atau bahkan bisa dianggap tidak perlu RE.

# Contoh kisah RE: Level 1

Aplikasi diperbaiki: Pengecekan IAP di server (tidak bisa download jika tidak membeli).

- Format file majalah adalah PDF yang dienkripsi (enkripsi standard PDF)
- PDF didownload aplikasi dengan password statik, sama untuk semua majalah/ebook.
- Hex Editor cukup untuk membukanya (terlihat string passwordnya), tool RE di sini hanya hex editor.

Sekarang yang dilakukan orang: satu orang membeli majalah, lalu PDF-nya disebarluaskan setelah didekripsi.

Semua orang bisa mendekripsi dengan mudah karena passwordnya sama.

## Contoh kisah RE: Level 2

Perbaikan versi berikutnya:

- Password berbeda untuk tiap majalah/buku
- Password disertakan (plaintext) dalam request REST/JSON untuk konten
- Tidak memakai SSL Pinning
- Cukup memakai intercepting proxy untuk melihat JSON yang berisi password

Semakin sedikit user yang bisa mendownload dan menyebarkan majalah digitalnya.

## Contoh kisah RE: Level 3

- Dalam JSON disertakan password terenkripsi, perlu didekrip oleh aplikasi
- Sekedar intercept tidak bisa (tidak tahu apa algoritma dan keynya)
- Harus melakukan RE untuk mencari algoritma dan keynya
- Kode dekrip memakai library standard, mudah diintercept dengan Frida

Di level ini, yang mau melakukan prosesnya semakin sedikit

# Contoh kisah RE: Level 4

Versi terakhir:

- Kode dekrip password diobfuscate, tidak mudah dibaca
- Memakai algoritma custom, sulit mencari algoritmanya
- Kode program harus dibaca seksama
- Mulai ada pembajak komersial yang menawarkan PDF berbayar dengan harga murah

Pada akhirnya aplikasi ini tutup.

# Pelajaran

- perlu secure software development lifecycle
- sebelum rilis, ditest menggunakan RE untuk tahu seberapa rentan aplikasinya
- produk RASP (Runtime Application Self-Protection) bisa membantu menambahkan kesulitan

Sebagai catatan: segala macam bentuk DRM, pasti bisa dikalahkan, misalnya user bisa memfoto layar HP-nya.

# Mau belajar RE?

- Belajar dasar programming supaya tahu algoritma dasar
- Bahasa/teknologi apa yang perlu dipelajari?
  - Tergantung pada apa yang mau direverse engineer
  - Tidak bisa memilih (misalnya: tidak bisa meminta malware developer untuk tidak memakai Go)

# Latihan



# Tool vs Skill

- Mengajari mengetik dengan Word bisa dilakukan dalam hitungan jam
- Tapi mengajari orang jadi penulis buku/novel butuh waktu bertahun-tahun

# Buku RE

Kebanyakan buku cepat ketinggalan jaman, tapi ilmu dasarnya tetap terpakai

- Practical Reverse Engineering: X86, X64, ARM, Windows Kernel, Reversing Tools, and Obfuscation
- Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software
- The Ghidra Book: The Definitive Guide / The IDA Pro Book, 2nd Edition

# Online Course

<https://www.udemy.com/topic/reverse-engineering/>

<https://www.sans.org/cyber-security-courses/reverse-engineering-malware-malware-analysis-tools-techniques/>

# CTF

Banyak CTF membutuhkan RE, tantangan kategori binary exploitation butuh reverse engineering

CTF Flare-ON berfokus hanya pada RE (setahun sekali, arsip tahun-tahun sebelumnya bisa diakses)

# Masa depan Reverse Engineering di era AI

AI akan membantu reverse engineering, tapi masih cukup panjang jalannya untuk bisa otomatis memahami proteksi yang kompleks

AI juga perlu direverse engineer

## AI membantu RE

Saat ini sudah ada yang memanfaatkan AI/LLM (Large Language Model) untuk membantu menjelaskan kode hasil RE.

Masih belum bisa digunakan untuk hal yang kompleks, karena keterbatasan arsitektur LLM saat ini.

Perkembangannya untuk RE masih cukup lambat, karena contoh data untuk RE juga masih terbatas.

## Keterbatasan LLM saat ini

Berbagai model berbasis LLM seperti ChatGPT, Gemini, Claude, dsb tidak memiliki kemampuan komputasi (akan salah jika ditanya pertanyaan matematika).

Saat ini ChatGPT dan yang lain membuat lalu mengeksekusi kode.

Reverse engineering saat ini penuh dengan persoalan komputasi (perhitungan address, aritmatika pada data, dsb)

# Melakukan reverse engineering terhadap Aplikasi berbasis AI

Banyak aplikasi akan memanfaatkan AI (AI sebagai komponennya)

Kita bisa melakukan reverse engineering untuk mengetahui apakah modelnya memang akurat, bisa mencuri modelnya, atau mencari kelemahannya (adversarial AI)

Contoh kasus: reverse engineering pengenalan digit SIREKAP

# Melakukan reverse engineering terhadap model AI

Saat ini banyak riset yang berusaha membongkar model AI, misalnya: mengetahui parameter model AI, bagaimana mengekstraksi data dari model AI.

Riset mengenai ini masih di tahap awal.

# Penutup

RE Sangat penting untuk security

RE tidak mudah, tapi sangat menarik dan penuh tantangan

Masa depan RE masih cerah

# Group Reversing.ID

Group Telegram (banyak member walau agak kurang aktif)

<https://t.me/ReversingID>

# Blog

Blog pribadi saya: <https://tinyhack.com/> dan  
<https://blog.compactbyte.com/category/reverse-engineering/>

Blog hacking hardware: <https://hackaday.com/blog/>

Blog Ida pro: <https://hex-rays.com/blog/>

Security: <https://revers.engineering/>

Berbagai tulisan di Medium: <https://medium.com/tag/reverse-engineering>