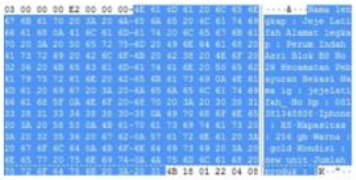
Vol. 4 No. 2 (2021): Edisi November 2021

Cyber Security dan Forensik Digital

Published: 19-04-2022

Articles



IDENTIFIKASI BUKTI DIGITAL INSTAGRAM WEB DENGAN LIVE FORENSIC PADA KASUS PENIPUAN ONLINE SHOP

erly ariyanti 58-62

PDF DOI : https://doi.org/10.14421/csecurity.2021.4.2.2436

Abstract Viewed = 434 times | PDF downloaded = 532 times

ANALISIS STATIK KEAMANAN APLIKASI VIDEO STREAMING BERBASIS ANDROID MENGGUNAKAN MOBILE SECURITY FRAMEWORK (MOBSF)

Fitri Nurindahsari, Bita Parga Zen 63-80

PDF DOI : https://doi.org/10.14421/csecurity.2021.4.2.3373

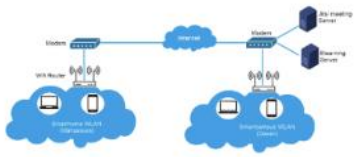
Abstract Viewed = 710 times | PDF downloaded = 764 times

ANALISIS TINGKAT KEMIRIPAN SUARA SEBAGAI BUKTI DIGITAL DENGAN MENGGUNAKAN TEKNIK AUDIO FORENSIK

Mia Nuur Aini 81-86

PDF DOI : https://doi.org/10.14421/csecurity.2021.4.2.2445

Abstract Viewed = 142 times | PDF downloaded = 181 times

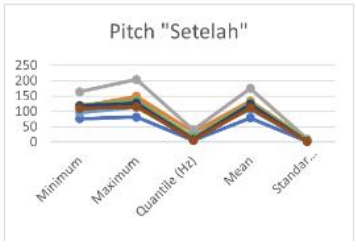


UJI KERENTANAN SMART HOME MENGGUNAKAN METODE SQUARE UNTUK MENDUKUNG SMART CAMPUS

Arini Arini, Nurul Faizah Rozy, Iik Muhammad Malik Matin 87-95

PDF DOI : https://doi.org/10.14421/csecurity.2021.4.2.3355

Abstract Viewed = 302 times | PDF downloaded = 343 times



ANALISIS PENEMUAN BARANG BUKTI DIGITAL MELALUI REKAMAN SUARA MENGGUNAKAN PRAAT DENGAN METODE AUDIO FORENSIK

Hafiz Pratama S Nawawi 96-103

PDF DOI : https://doi.org/10.14421/csecurity.2021.4.2.2638

English Version DOI : https://doi.org/10.14421/csecurity.2021.4.2.2638

CyberSecurity dan Forensik Digital e-ISSN: 2615-8442
Vol. 4, No. 2, November 2021, hlm. 63-80

ANALISIS STATIK KEAMANAN APLIKASI *VIDEO STREAMING* BERBASIS ANDROID MENGGUNAKAN *MOBILE SECURITY FRAMEWORK* (MOBSF)

Fitri Nurindahsari[1], Bita Parga Zen[2]

[1,2]Program Studi Informatika
Institut Teknologi Telkom Purwokerto
Email: [1]fitrinurindahsari26@gmail.com, [2]bita@ittelkom-pwt.ac.id

Penulis korespondensi : bita@ittelkom-pwt.ac.id

Abstrak

Video streaming yaitu layanan transmisi video dan audio melalui internet. Layanan ini dibroadcast kepada banyak pengguna yang mengakses suatu situs *video streaming*. Pengguna aplikasi *video streaming* di Indonesia mengalami peningkatan dari tahun 2018 sampai 2021. Kondisi pada tahun 2021 sampai 2022 yang masih dalam masa pandemi semakin meningkatkan keinginan masyarakat untuk menggunakan aplikasi *video streaming*, terkhusus aplikasi *video streaming* berbasis android. Pengguna aplikasi *video streaming* tidak bisa mengabaikan keamanan dari aplikasi tersebut. Dengan banyaknya akses ke aplikasi *video streaming* berbasis android dapat menjadi target utama oleh *Cracker* dalam melakukan tindak kejahatan. Apalagi terdapat data pribadi pengguna aplikasi *video streaming* seperti nama pengguna, kata sandi dan nomor handphone, jika hal ini sampai dapat diakses oleh oknum yang tidak bijak maka dapat disalahgunakan. Tujuan penelitian ini untuk menemukan celah keamanan yang terdapat pada aplikasi *video streaming* berbasis android. Peneliti menggunakan *mobile security framework* (MobSF) untuk menganalisis statik keamanan dengan parameter *dangerous permissions*, *weak crypto*, *root detection*, SSL *bypass* dan *domain malware check* pada aplikasi *video streaming* berbasis android. Hasil yang diperoleh dari penelitian ini yaitu aplikasi Y, N dan V memiliki *dangerous permissions*, *weak crypto*, dan SSL *bypass*, dan *domain malware check* dengan status *good*. Hanya aplikasi Y yang tidak memiliki *root detection*.

Kata kunci: *analisis statik, android, keamanan*, MobSF, *video streaming*.

SECURITY STATIC ANALYSIS OF ANDROID-BASED VIDEO STREAMING APPLICATION USING MOBILE SECURITY FRAMEWORK (MOBSF)

Abstract

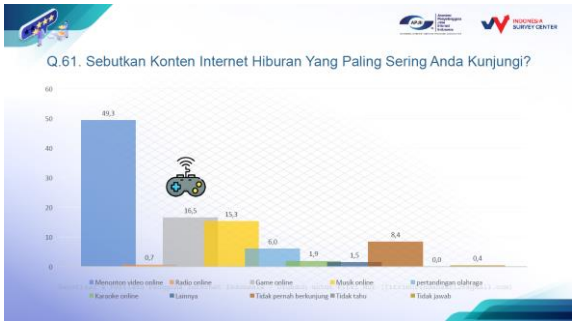
Video streaming is an internet-based video and audio transmission service. This service is transmitted to a large number of people that visit an online video website. Video streaming application users in Indonesia have increased from 2018 to 2021. The current situation, which is still in a pandemic phase, is increasing the public's desire to use video streaming programs, particularly those based on Android. Users of streaming video applications cannot overlook the app's security. Because so many people have access to Android-based video streaming apps, crackers may use them to conduct crimes. Furthermore, there is personal data of users of video streaming programs, such as usernames, passwords, and phone numbers, which could be accessed by unscrupulous individuals. The goal of this research is to identify security flaws in an Android-based video streaming software. On android-based video streaming apps, researchers employed the mobile security framework (MobSF) to examine static security with parameters such as dangerous permissions, weak crypto, root detection, SSL bypass and domain malware check. Applications Y, N, and V have dangerous permissions, weak crypto, SSL bypass and domain malware check with a good status, according to the findings of this study. Only application Y is not able to root detection.

Keywords*: android, MobSF, security, static analysis, video streaming.*

1. PENDAHULUAN

Kemajuan teknologi informasi dan sistem pertahanan siber saat ini berkembang begitu pesat (BP Zen, 2020) Seiring kemajuan teknologi pada 2021 dan 2022 yang semakin berkembang, pemakaian internet pun semakin meningkat. Dalam kehidupan masyarakat, internet mencakup berbagai sektor termasuk sektor hiburan. Kondisi pada tahun 2021 sampai 2022 yang masih dalam masa pandemi semakin meningkatkan keinginan masyarakat untuk menggunakan aplikasi *video streaming*, terkhusus aplikasi *video streaming* berbasis android.

Asosiasi Penyedia Jasa Internet Indonesia (APJII) menyebutkan hingga kuartal II tahun 2020 di sektor hiburan, video daring menjadi akses hiburan terbesar dengan 49,3%, disusul game daring 16,5%, dan musik daring 15,3% (Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020). Gambar 1. menunjukkan hasil survei konten internet hiburan yang paling sering dikunjungi.



(Sumber: Laporan Survei Internet APJII 2019 – 2020 (Q2))
Gambar 1. (Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020)

We Are Social mengungkapkan laporan "*Digital 2021: The Latest Insights Inti The State of Digital*" yang diterbitkan pada 11 Februari 2021, alasan masyarakat menggunakan internet untuk menonton video online, acara tv dan film sebesar 51,7% (Simon Kemp, 2021). Dapat disimpulkan jika pengaksesan *video streaming* di Indonesia mengalami peningkatan selama 4 tahun terakhir. Tabel 1. menunjukkan data akses aplikasi *video streaming*.

Tabel 1. Data Akses Aplikasi *Video streaming* (Dify Virginia Rizaldy, 2021)(Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020)(Simon Kemp, 2021)

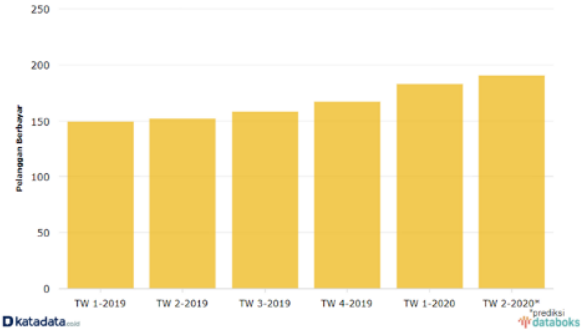
No.	Tahun	Presentase Akses Aplikasi *Video streaming*
1.	2018	19%
2.	2019	49,3%
3.	2020	49,3%
4.	2021	51,7 %

Data dari dari ComScore menunjukkan, ada lebih dari 93 juta penonton unik di Indonesia (berusia di atas 18 tahun) yang menonton video di Y setiap bulannya selama 2021. Jumlah itu tercatat meningkat hingga 10 juta dibanding tahun 2020 (Tesalonica, 2020). Table 2. menunjukkan data pelanggan berbayar N di Indonesia.

Tabel 2. Data Pelanggan Berbayar N di Indonesia (Pusparisa, 2020)

No.	Tahun	Banyak Pelanggan Berbayar
1.	TW 2-2019	151,6 Juta
2.	TW 3-2019	158,3 Juta
3.	TW 4-2019	167,1 Juta
4.	TW 1-2020	182,9 Juta

Pada gambar 2. menunjukkan data pelanggan N. V mencatat peningkatan signifikan jumlah pengguna aplikasi aktif dan pengunduhan aplikasi selama masa pembatasan sosial berskala besar (PSBB). Pengguna aktif layanan *streaming video on demand* itu mencapai 62 juta pada April 2020. Peningkatan mulai terlihat sejak kehadiran program "V Bebas Nonton" (Asrianti, 2020).



(Sumber: https://databoks.katadata.co.id)
Gambar 2. Data Pelanggan N (Pusparisa, 2020)

Tabel 3. menunjukkan data pengguna aplikasi *video streaming* pada tahun 2020.

Tabel 3. Data Pengguna Aplikasi *Video streaming* pada tahun 2020, (Tesalonica, 2020)(Pusparisa, 2020)(Asrianti, 2020)

No.	Nama Aplikasi *Video streaming*	Pengguna Aplikasi *Video streaming*
1.	Y	93 Juta
2.	N	182 Juta
3.	V	62 Juta

Situs web *streaming* membuat pemirsa tertarik untuk menonton acara *online*, baik acara olahraga, pesta, saluran TV, atau lainnya. Terlepas dari risiko menonton *streaming* langsung gratis, banyak pemirsa mungkin tidak menyadari bahaya sebenarnya yang mengintai. Di antara skema termasuk menempatkan iklan *overlay* dengan tombol "tutup" palsu ke *video streaming* yang dapat mengelabui pemirsa agar mengunduh *malware* berbahaya (Tanjung, 2016).

Dibalik kemudahan penggunaan dan keuntungan yang diberikan oleh aplikasi *video streaming*, pengguna juga tidak bisa mengabaikan keamanan dari aplikasi *video streaming* tersebut. Dengan semakin banyaknya akses ke aplikasi *video streaming* berbasis android dapat menjadi target utama oleh *Cracker* dalam melakukan tidak kejahatan. Dari banyaknya aplikasi android yang beredar tidak semuanya sudah menerapkan pengujian keamanan dengan baik sesuai ISO/27001 (Putra, Nurhayati dan Windasari, 2016). ISO/27001 merupakan suatu standar Internasional dalam menerapkan sistem manajemen kemanan informasi atau lebih dikenal dengan *Information Security Management Systems* (ISMS) (ISOCENTER INDONESIA, 2022). Apalagi terdapat data pribadi pengguna aplikasi *video streaming* seperti email dan nomor handphone, jika hal ini sampai dapat diakses

oleh oknum yang tidak bijak maka dapat disalahgunakan.

Situs *video streaming* CAM4 pada Bulan Maret menjadi *website* yang terkena serangan data *breach*. Hal ini diungkapkan oleh Anurag Sen dari Safety Detectives. Kebocoran data ini muncul karena adanya kesalahan konfigurasi *server ElasticSearch* yang berakibat pada *database* pengguna dan anggota *platform* menjadi tidak aman. Akibatnya, sebesar 10,88 miliar data pribadi dapat terakses (NKD, 2020).

Data pelanggan sebanyak 10.000 dicuri dari situs *video streaming* MyFreeCams.com dalam pelanggaran data pada tahun 2021. Penjahat dunia maya telah menjual data pelanggan yang dicuri seharga 1.500 dolar AS secara *online* meliputi nama pengguna, kata sandi tidak terenkripsi, alamat email, saldo MyFreeTokens pelanggan untuk setiap akun. Data dieksfiltrasi pada Desember 2020 melalui serangan injeksi SQL (Dvorak, 2021).

Pada tahun 2020 ada sekumpulan laporan tentang data akun yang muncul di forum kejahatan *dark web*. Dari audit *dark web* menunjukkan ada 15 miliar *login* yang dicuri dari 100.000 pelanggaran, hingga peretas yang memberikan 386 juta catatan curian secara gratis. Basis data yang tidak aman dengan cepat menjadi masalah perlindungan data yang sangat besar, sehingga meninggalkan data profil pribadi hampir 235 juta pengguna Instagram, TikTok, dan Y untuk diperebutkan (Winder, 2020).

Pada penelitian sebelumnya, peneliti pertama meneliti tentang analisis dan deteksi *malware* dengan metode *hybrid analysis* menggunakan *framework* MobSF yang memiliki masalah pada mencari hal-hal yang janggal pada *malware* yang menginfeksi perangkat android dan mendapatkan beberapa karakteristik dari *malware Bouncing Golf* dan *Riltok*. Peneliti kedua meneliti tentang analisis statis menggunakan *mobile security framework* untuk pengujian keamanan aplikasi *mobile e-commerce* berbasis android yang memiliki masalah pada mencari celah-celah keamanan pada aplikasi *mobile e-commerce* dan mendapatkan lima besar *mobile e-commerce* di Indonesia memiliki beberapa celah keamanan masih ada dengan tingkat kemanan yang relatif sama. Peneliti ketiga meneliti tentang *a comprehensive analysis of the android permissions system* yang memiliki masalah pada mencari masalah keamanan dan cara aplikasi menangani informasi dan sumber daya berisiko tinggi dari sistem izin android dan mendapatkan beberapa ancaman keamanan yang dapat membuka OS untuk berbagai serangan.

Pada penelitian kali ini, penulis menggunakan *mobile security framework* untuk melakukan analisis statik pada aplikasi *video streaming* dengan subyek yang digunakan yaitu situs *video streaming* di Indonesia inisial Y, N dan V, dengan harapan penelitian ini dapat mendukung analisis sistem keamanan pada situs *video streaming* yang dapat menjadi rujukan untuk menilai tingkat keamanannya.

Langkah yang akan dilakukan peneliti yaitu parameter analisis *dangerous permissions*, *weak crypto*, *root detection*, SSL *bypass* dan *domain malware check*, sehingga dapat ditemukan celah keamanan pada aplikasi *video streaming* jika memang ada. Kemudian peneliti menganalisis statik dan merekomendasikan untuk celah keamanan yang ditemukan.

Berdasarkan latar belakang yang sudah dijabarkan, peneliti memilih topik tersebut untuk diteliti dengan judul Analisis Statik Keamanan Aplikasi *Video Streaming* Berbasis Android Menggunakan *Mobile Security Framework* (MobSF) dengan tujuan adanya penelitian ini dapat berguna untuk memberikan kesadaran terhadap masyarakat maupun pengguna aplikasi, memberikan masukan kepada pihak pengembang aplikasi untuk terus meningkatkan aspek keamanan dan dari pengguna aplikasi menyadari akan risiko keamanan terhadap setiap aplikasi *video streaming*.

2. TINJAUAN PUSTAKA

2.A. Aplikasi Video Streaming

Streaming adalah suatu teknologi untuk menjalankan file audio atau video secara langsung menggunakan jaringan internet maupun lokal (Suroso, Ciksadan dan Sholihatun, 2020).

File audio atau video yang terletak pada sebuah *server* dapat secara langsung dijalankan pada komputer *client* sesaat setelah ada permintaan dari pengguna sehingga proses *download* file tersebut yang membutuhkan waktu cukup lama dapat dihindari. Saat file tersebut diputar maka akan terbentuk sebuah *buffer* di komputer *client* dan data audio atau video tersebut akan mulai di-*download* ke dalam *buffer* yang telah terbentuk pada mesin *client*. Setelah *buffer* terisi dalam waktu beberapa detik, maka secara otomatis file video ataupun audio akan di jalankan oleh sistem. Sistem akan membaca informasi dari *buffer* sambil tetap melakukan proses *download* file sehingga proses *streaming* tetap berlangsung ke mesin *client* (Arsam, 2017).

2.B. Mobile Security Framework (MobSF)

Mobile Security Framework (MobSF) adalah *framework* yang digunakan untuk pengujian penetrasi terhadap aplikasi seluler (android/iOS/windows) otomatis yang mampu melakukan analisis statis, dinamis, dan *malware*. MobSF digunakan untuk analisis keamanan yang efektif dan cepat dari aplikasi seluler dan mendukung kedua binari (*Android Package Kit* (APK), *iPhone Application* (IPA) & APPX yang merupakan format file *windows store*) dan kode sumber zip. MobSF dapat melakukan pengujian aplikasi dinamis saat *runtime* untuk aplikasi Android dan memiliki kemampuan *fuzzing* API Web yang didukung oleh CapFuzz, pemindai keamanan khusus Web API. *Fuzzing* merupakan suatu metode mencari kesalahan piranti lunak dengan menyediakan input yang tidak diduga lalu mengamati hasilnya (Tugas dan El, 2016). MobSF dirancang untuk membuat integrasi *Continuous Integration*/

Continuous Delivery (CI/CD) atau *Development, Security, and Operations* (DevSecOps) secara bagus (Abraham, 2021). CI/CD merupakan metode untuk mengirimkan aplikasi ke pelanggan secara rutin dengan memperkenalkan otomatisasi ke dalam tahapan pengembangan aplikasi (NDK, 2020). DevSecOps adalah kerangka kerja kolaborasi yang memperluas dampak *Development and Operations* (DevOps) dengan menambahkan praktik keamanan ke proses pengembangan dan pengiriman perangkat lunak (Dynatrace, 2021).

Mobile Security Framework adalah kerangka kerja gabungan yang melakukan analisis statis dan dinamis dari sebuah APK. MobSF dikembangkan berdasarkan Python. Untuk menghasilkan laporan MobSF menggunakan html. Ini menjalankan server lokal melalui baris perintah di komputer *host* (Shahriar, Arabin Talukder dan Saiful Islam, 2019). Semua alat analisis statis dan dinamis yang ada melakukan analisis menggunakan *DroidMon-Dalvik Monitoring Framework* (Idan, 2016) dan *Xposed Module Repository*. Xposed adalah kerangka kerja untuk modul yang dapat mengubah perilaku sistem dan aplikasi tanpa menyentuh APK apa pun. Itu bagus karena itu berarti modul dapat bekerja untuk versi yang berbeda dan bahkan ROM tanpa perubahan apa pun (selama kode aslinya tidak terlalu banyak diubah) (Tansen dan Nurdiarto, 2020).

2.C. Analisis Statik

Metode statis analis dilakukan tanpa benar-benar menjalankan programnya dan lebih seperti menyelidiki apa yang terjadi pada *source code* dengan tujuan utama yaitu untuk mengetahui kode berbahaya seperti apa yang tertanam dalam aplikasi tersebut (Skylot, 2020).

Analisis statis mengacu pada dekompilasi APK aplikasi ke file Java dan XML yang sesuai. Untuk melakukan analisis statis, penganalisis statis harus memiliki fitur untuk memeriksa XML dengan benar dan presisi. File Java dapat diekstraksi dengan dekompiler DEXtoJar (Rovo, 2019). Alat analisis Statis mendekompilasi kode APK ke format yang dapat dibaca manusia. Sehingga penganalisa dapat membaca kode dan mengidentifikasi kerentanan yang mungkin dimiliki aplikasi (Shahriar, Arabin Talukder dan Saiful Islam, 2019).

Gambar 3. Parameter Analisis Statik

Dari hasil analisis statis maka didapat hasil daftar *malwords* yang sering muncul sebagai daftar string berbahaya dengan tingkat resiko masing-masing, setelah mendapatkan daftar malwords, peneliti akan menganalisisnya (Anwar et al., 2020). Gambar 3. menunjukkan parameter pada analisis statik.

2.D. Weak Crypto

Analisis *weak crypto* dilakukan dengan acuan ada atau tidaknya implementasi algoritma kriptografi yang lemah atau penggunaan algoritma kriptografi yang sudah usang atau sudah dianggap tidak layak.

2.E. SSL Bypass

Analisis SSL *bypass* dilakukan dengan melakukan cek ada atau tidaknya *service* yang melibatkan protokol http yang tidak mewajibkan penggunaan SSL sebagai persyaratan keamanan transaksi dalam protokol http menggunakan SSL seperti mengizinkan http di manifest, atau terdapat string berkonten http:// yaitu *weak implementation*.

2.F. Dangerous Permissions

Aplikasi Android dibangun untuk melakukan serangkaian tindakan, beberapa di antaranya memerlukan izin dari pengguna. Analisis terhadap *permission* dilakukan dengan melihat pada seberapa banyak *dangerous permissions* yang digunakan oleh aplikasi.

2.G. Root Detection

Analisis *root detection* dilakukan dengan melakukan cek ada atau tidaknya aplikasi mempunyai fungsi untuk melakukan deteksi akses root terhadap perangkat android yang digunakan. Dimana akses memungkinkan untuk akses langsung ke dalam sistem termasuk data-data yang dimiliki aplikasi.

2.H. Domain Malware Check

Analisis *domain malware check* dilakukan dengan melakukan cek ada atau tidaknya domain-domain yang terdapat dalam aplikasi terindikasi dalam kategori domain yang mengandung malware atau tidak (Hanifurohman dan Hutagalung, 2020).

3. METODE PENELITIAN

Metode penelitian yang digunakan dalam melaksanakan penelitian ini adalah dengan menggunakan metode kuantitatif dengan uji analisis *video streaming* menggunakan *Mobile Security Framework* (MobSF) dan obyek penelitian yaitu keamanan aplikasi *video streaming* di Indonesia yaitu Y, N dan V, dengan tujuan penelitian ini mendukung sistem keamanan pada situs *video streaming* tersebut. Dari 3 situs *video streaming*, kriteria yang akan dijadikan sebagai parameter penelitian adalah *dangerous permissions, weak crypto, root detection, SSL bypass dan domain malware check*.

3.A. Diagram Alir Penelitian

Diagram alir penelitian pada ini dengan mengidentifikasi suatu permasalahan atau fenomena yang dianggap perlu untuk diteliti, yaitu penelitian tentang analisis statik keamanan aplikasi *video*

streaming berbasis android menggunakan *mobile security framework*. Permasalahan terjadi pada banyaknya kasus pencurian data. Pada masa pandemik tahun 2020 sampai 2021 banyak masyarakat Indonesia yang menggunakan aplikasi *video streaming* berbasis android. Terdapat 3 aplikasi *video streaming* berbasis android yang memiliki pengguna banyak yaitu Y, N dan V.

Gambar 4. Diagram Alir Penelitian

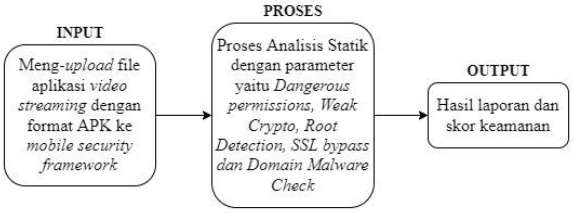
Peneliti menggunakan *mobile security framework* untuk mengetahui celah keamanan yang terdapat pada aplikasi *video streaming* berbasis android, kemudian menganalisis statik dan merekomendasikan untuk celah keamanan yang ditemukan sehingga masyarakat dapat mengetahui tingkat keamanan dari aplikasi *video streaming* berbasis android yang telah diteliti.

3.B. Melakukan Pengujian dan Pengambilan Hasil Data Uji

Tahap ini untuk melakukan pengujian, kemudian melihat hasil pengujian menggunakan *mobile security framework* terhadap aplikasi *video streaming* berbasis android dengan format APK.

1. Cara Pengujian

   Gambar 5. menunjukkan blok diagram analisis statik menggunakan *Mobile Security Framework* (MobSF).



Gambar 5. Blok Diagram Analisis Statik Menggunakan *Mobile Security Framework* (MobSF)

Input: meng-*upload* file aplikasi *video streaming* dengan format APK ke *mobile security framework*, tunggu hingga proses *upload* selesai.
Proses: *mobile security framework* akan melakukan proses analisis statik dengan parameter yaitu *dangerous permissions, weak crypto, root detection, ssl bypass dan domain malware check.*
Output: saat proses telah selesai, maka akan muncul hasil pengujian file tersebut yang berupa laporan dan skor keamanan (Kartono, Sularsa dan Ismail, 2019).

2. Parameter yang Diuji

   Dangerous permissions, weak crypto, root detection, SSL *bypass* dan *domain malware check.*

3. Data yang Diambil

   Tingkat dan ceklah keamanan dari 3 aplikasi *video streaming* berbasis android yaitu Y, N dan V

3.C. Hasil Analisis dan Pembahasan

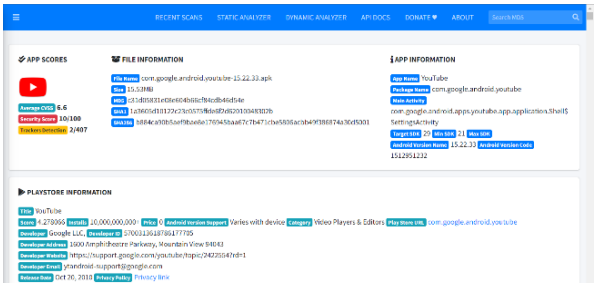
Tahap ini untuk membahas hasil pengujian keamanan dari 3 aplikasi *video streaming* yang dilakukan menggunakan *mobile security framework.* Peneliti akan menganalisis celah keamanan yang ditemukan pada aplikasi *video streaming* dari hasil pengujian sebelumnya. Tahap terakhir adalah berisi kesimpulan dari penelitian yang dilakukan sesuai dengan ruang lingkup pengujian. Bagian ini juga berisi saran untuk penelitian kedepannya.

4. HASIL DAN PEMBAHASAN

4.A. Hasil Analisis

File aplikasi *video streaming* dengan format APK diunduh terlebih dahulu sebagai tahap awal. Analisis dilakukan dengan cara meng-*upload* file aplikasi *video streaming* dengan format APK ke *mobile security framework*. Saat proses telah selesai, maka akan muncul hasil analisis file tersebut.

Pada gambar 6. menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF).



Gambar 6. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF)

Pada aplikasi Y bagian *app score* terdapat:

1. *The Common* Vulnerability *Scoring System* (CVSS) yang memberikan representasi numerik 0 sampai 10 dari tingkat keparahan kerentanan keamanan informasi, aplikasi Y mendapat skor 6.6 yang artinya memiliki peringkat kualitatif sedang,
2. Security score mendapatkan skor 10/100 yang artinya memiliki risiko yang kritis.
3. Trackers detection mendapatkan skor 2/407 yang artinya memiliki 2 pelacak.

Pada file information terdeteksi: Nama file yaitu com.google.android.Y-15.22.33.apk dengan ukuran 15.53 MB. Pada *app information* terdeteksi:

1. Nama app yaitu Y dengan package name yaitu com.google .android.Y.
2. Aktifitas utama pada com.google.android .apps.Y.app.application.Shell$SettingsActivity.
3. Software Development Kit (SDK) adalah sebuah koleksi dari alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal Target SDK adalah versi yang ditargetkan untuk dijalankan oleh aplikasi, tercantum 29, min SDK adalah versi minimal yang dapat dijalankan oleh aplikasi, tercantum 21 dan tidak ada max SDK.
4. Nama versi android 15.22.33 dan kode versi android 1512951232.

Pada *playstore information* terdeteksi:

1. Judul yaitu Y dengan skor 4.278066 yaitu rating pada playstore.
2. Diinstal sebanyak 10,000,000,000+ kali.
3. Dengan harga aplikasi 0.
4. Dukungan versi android bervariasi menurut perangkat.
5. Kategori aplikasi yaitu pemutar video & editor.
6. URL play store yaitu com.google.android.Y.
7. Pengembang aplikasi yaitu Google LLC.
8. ID pengembang yaitu 5700313618786177705
9. Alamat pengembang yaitu 1600 Amphitheatre Parkway, Mountain View 94043.
10. Website pengembang yaitu https://support.google.com/Y/topic/2422554?rd=1
11. Email pengembang yaitu ytandroid-support@google.com.
12. Tanggal rilis aplikasi yaitu 20 Oktober 2010.
13. Privacy policy yaitu link privasi.

Pada gambar 7. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF).

Gambar 7. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF)

Pada aplikasi N bagian *app score* terdapat:

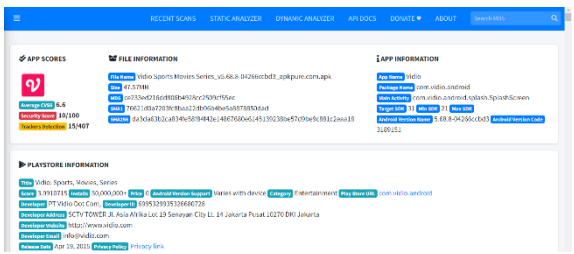
1. The Common Vulnerability Scoring System (CVSS) yang memberikan representasi numerik 0 sampai 10 dari tingkat keparahan kerentanan keamanan informasi, aplikasi N mendapatkan skor 6.2 yang artinya memiliki peringkat kualitatif sedang.
2. Security score mendapatkan skor 35/100 yang artinya memiliki risiko keamanan yang tinggi.
3. Trackers detection mendapatkan skor 2/407 yang artinya memiliki 2 pelacak.

Pada file information terdeteksi, nama file yaitu: com.N.ninja_8.3.1_build_45464546_minAPI22(armeabi-v7a)(nodpi)_apkmirror.com.apk. Dengan ukuran file sebesar 79.37 MB. Pada *app information* terdeteksi:

1. Nama app yaitu N dengan package name yaitu com.N.ninja.
2. Aktifitas utama pada .MainActivity.
3. Software Development Kit (SDK) adalah sebuah koleksi dari alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal. Target SDK adalah versi yang ditargetkan untuk dijalankan oleh aplikasi, tercantum 31, min SDK adalah versi minimal yang dapat dijalankan oleh aplikasi, tercantum 22 dan tidak ada max SDK.
4. Nama versi android 8.3.1 build 4546 dan kode versi android 4546.

Pada *playstore information* terdeteksi:

1. Judul yaitu N dengan skor 2.28 yaitu rating pada playstore.
2. Diinstal sebanyak 100,000,000+ kali.
3. Dengan harga aplikasi 0.
4. Dukungan versi android bervariasi menurut perangkat.
5. Kategori aplikasi yaitu hiburan.
6. URL play store yaitu com.N.ninja.
7. Pengembang aplikasi yaitu N, Inc.
8. ID pengembang yaitu N,+Inc.
9. Alamat pengembang yaitu 100 Winchester Circle Los Gatos, CA 95032-1815 USA.
10. Website pengembang yaitu http://www.N.com.
11. Email pengembang yaitu playstore@N.com.
12. Tanggal rilis aplikasi yaitu 27 Juni 2014.
13. Privacy policy yaitu link privasi.
14. Pada gambar 8. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF).



Gambar 8. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF)

Pada aplikasi V bagian *app score* terdapat:

1. The Common Vulnerability Scoring System (CVSS) yang memberikan representasi numerik 0

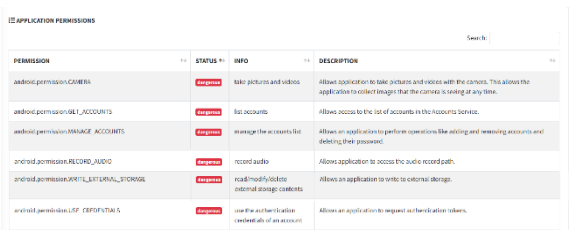
sampai 10 dari tingkat keparahan kerentanan keamanan informasi, aplikasi V mendapatkan skor 6.6 yang artinya memiliki peringkat kualitatif sedang.
2. Security score mendapatkan skor 10/100 yang artinya memiliki risiko keamanan yang kritis.
3. Trackers detection mendapatkan skor 15/407 yang artinya memiliki 15 pelacak.

Pada *file information* terdeteksi nama file yaitu V Sports Movies Series_v5.68.8-04266ccbd3_apkpure.com .apk dengan ukuran file sebesar 47.57 MB. Pada *app information* terdeteksi:
1. Nama app yaitu V dengan package name yaitu com.V.android.
2. Aktifitas utama pada com.V.android.splash .SplashScreen.
3. Software Development Kit (SDK) adalah sebuah koleksi dari alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal. Target SDK adalah versi yang ditargetkan untuk dijalankan oleh aplikasi, tercantum 31, min SDK adalah versi minimal yang dapat dijalankan oleh aplikasi, tercantum 21 dan tidak ada max SDK.
4. Nama versi android 5.68.8-04266ccbd3 dan kode versi android 3189151.

Pada *playstore information* terdeteksi:
1. Judul yaitu V: Sports, Movies, Series dengan skor 3.9910715 yaitu rating pada playstore.
2. Diinstal sebanyak 50,000,000+ kali.
3. Dengan harga aplikasi 0.
4. Dukungan versi android bervariasi menurut perangkat.
5. Kategori aplikasi yaitu hiburan.
6. URL play store yaitu com.V.android.
7. Pengembang aplikasi yaitu PT V Dot Com.
8. ID pengembang yaitu 6995329935326680728.
9. Alamat pengembang yaitu SCTV TOWER Jl. Asia Afrika Lot 19 Senayan City Lt. 14 Jakarta Pusat 10270 DKI Jakarta.
10. Website pengembang yaitu http://www.V.com.
11. Email pengembang yaitu info@V.com.
12. Tanggal rilis aplikasi yaitu 19 April 2015.
13. Privacy policy yaitu link privasi.

4.A.i. Dangerous Permissions

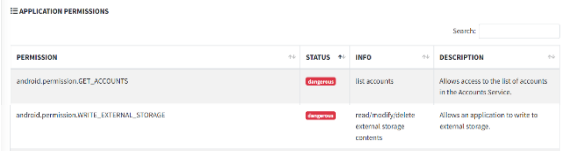
Pada gambar 9. yang menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian *application permissions*.



Gambar 9. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian *Application Permissions*

Pada aplikasi Y terdapat 6 *dangerous permissions* yaitu:
1. android.permission.CAMERA yang mengizinkan akses perangkat kamera sehingga aplikasi dapat mengambil gambar atau video melalui perangkat kamera kapan saja.
2. android.permission.GET_ACCOUNTS yang mengizinkan akses ke daftar akun di layanan akun sehingga aplikasi dapat memperoleh informasi tentang akun.
3. android.permission.MANAGE_ACCOUNTS yang mengizinkan aplikasi melakukan operasi seperti menambah dan menghapus akun serta menghapus sandinya sehingga aplikasi bisa mengambil alih akun.
4. android.permission.RECORD_AUDIO yang mengizinkan aplikasi mengakses jalur rekaman audio atau mikrofon sehingga aplikasi dapat merekam audio kapan saja.
5. android.permission.WRITE_EXTERNAL_STORAGE yang mengizinkan aplikasi untuk membaca/memodifikasi/menghapus konten penyimpanan eksternal sehingga aplikasi dapat menyalahgunakan konten yang ada pada penyimpanan eksternal.
6. android.permission.USE_CREDENTIALS yang mengizinkan aplikasi meminta token autentikasi atau menggunakan kredensial otentikasi akun.

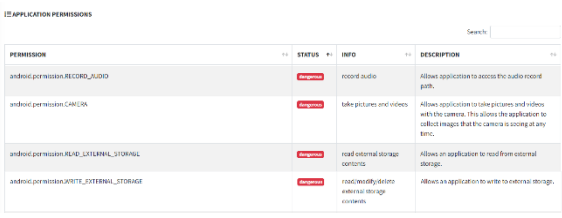
Pada gambar 10. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian *application permissions*.



Gambar 10. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian *Application Permissions*

Pada aplikasi N terdapat 2 *dangerous permissions* yaitu:
1. android.permission.GET_ACCOUNTS yang mengizinkan akses ke daftar akun di layanan akun sehingga aplikasi dapat memperoleh informasi tentang akun.
2. android.permission.WRITE_EXTERNAL_STORAGE yang mengizinkan aplikasi untuk membaca/memodifikasi/menghapus konten penyimpanan eksternal sehingga aplikasi dapat menyalahgunakan konten yang ada pada penyimpanan eksternal.

Pada gambar 11. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *application permissions*.



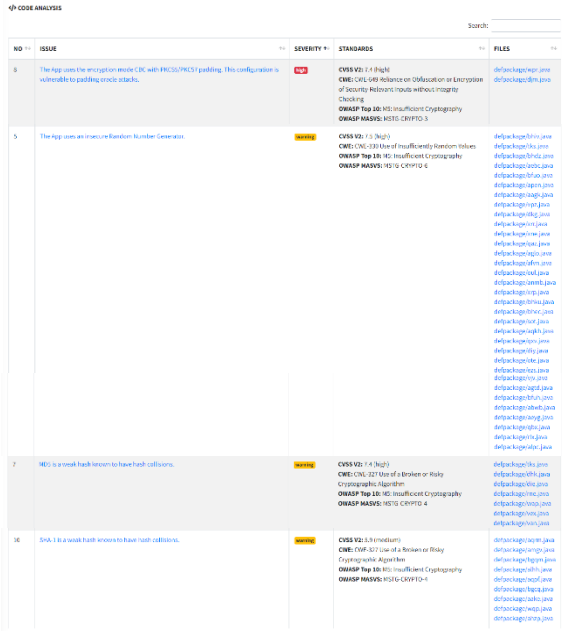
Gambar 11. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Application Permissions*

Pada aplikasi V terdapat 4 *dangerous permissions* yaitu:

1. android.permission.RECORD_ AUDIO yang mengizinkan aplikasi mengakses jalur rekaman audio atau mikrofon sehingga aplikasi dapat merekam audio kapan saja.
2. android.permission.CAMERA yang mengizinkan akses perangkat kamera sehingga aplikasi dapat mengambil gambar atau video melalui perangkat kamera kapan saja.
3. android.permission.READ_EXTERNAL_STORAGE yang mengizinkan aplikasi membaca dari penyimpanan eksternal sehingga aplikasi dapat memperoleh informasi tentang konten yang ada pada penyimpanan eksternal.
4. android.permission.WRITE_EXTERNAL_STORAGE yang mengizinkan aplikasi untuk membaca/memodifikasi/menghapus konten penyimpanan eksternal sehingga aplikasi dapat menyalahgunakan konten yang ada pada penyimpanan eksternal.

4.A.ii. Weak Crypto

Pada gambar 12. menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian *code analysis.*



Gambar 12. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada aplikasi Y terdapat 4 *weak crypto* yang terdiri dari 1 *high severity* yaitu aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding*. Serangan *oracle padding* adalah serangan yang menggunakan validasi *padding* dari pesan kriptografi untuk mendekripsi ciphertext (Ichi.pro, 2021). Dengan standar:

1. Common Vulnerability Scoring System (CVSS) adalah suatu nilai pengukuran yang dipakai untuk menilai suatu kerentanan terhadap sistem, pada CVSS V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
2. Common Weakness Enumeration (CWE) adalah daftar yang menampilkan keberadaan bug pada software atau hardware yang berbahaya, terdeteksi CWE-649:ketergantungan pada kebingungan atau enkripsi input keamanan yang relevan tanpa pemeriksaan integritas.
3. Open Web Application Security Project (OWASP) Top 10 adalah sebuah panduan bagi para developers dan security team tentang kelemahan-kelemahan pada web apps yang mudah diserang dan harus segera disiasati, terdeteksi M5:kriptografi tidak memadai.
4. OWASP The Mobile Application Security Verification Standard (MASVS) adalah standar untuk keamanan aplikasi seluler, terdeteksi MSTG-CRYPTO-3 artinya aplikasi menggunakan primitif kriptografi yang sesuai untuk kasus penggunaan tertentu, serta dikonfigurasi dengan parameter yang mematuhi praktik terbaik industri.

Lokasi file yaitu defpackage/wpr.java dan defpackage/djm.java. Gambar 13. menunjukkan kode pada defpackage/wpr.java.

```
93.     public final Cipher e() {
94.         try {
95.             Cipher instance = Cipher.getInstance("AES/CBC/PKCS7Padding");
96.             instance.init(1, (SecretKey) this.d.getKey("YouTubeFingerprintKey", null));
97.             return instance;
```

Gambar 13. Kode pada defpackage/wpr.java

Kode tersebut artinya membuat cipherinstance dengan memanggil getInstance() metodenya dengan parameter yang diisi dengan jenis algoritma enkripsi yang digunakan yaitu AES dengan mode operasi CBC dan padding PKCS7. Terlebih dahulu masukkan kelas java yaitu javax.crypto.Cipher untuk membuat cipher. *Advanced Encryption Standard* (AES) merupakan algoritma yang menggunakan kunci yang sama (*private key*) untuk enkripsi dan dekripsi. Mode *Cipher Block Chaining* (CBC) adalah suatu mode dimana blok yang satu dengan blok lain saling terkait (*chained*), enkripsi dan dekripsi suatu blok data selalu melibatkan ciphertext (hasil enkripsi) blok sebelumnya. *Public Key Cryptographic Standard* (PKCS) adalah penambahan data pada awal, tengah, atau akhir sebelum enkripsi, nilai tiap bit yang ditambahkan adalah banyak bit yang ditambahkan. Gambar 14. menunjukkan kode pada defpackage/djm.java.

```
98.     private static final Cipher a() {
99.         Cipher cipher;
100.        synchronized (c) {
101.            if (a == null) {
102.                a = Cipher.getInstance("AES/CBC/PKCS5Padding");
103.            }
104.            cipher = a;
105.        }
106.        return cipher;
107.    }
```

Gambar 14. Kode pada defpackage/djm.java

Kode tersebut artinya membuat variabel a sebagai cipherinstance dengan memanggil getInstance() metodenya dengan parameter yang diisi dengan jenis algoritma enkripsi yang digunakan yaitu

AES dengan mode operasi CBC dan padding PKCS5. Terlebih dahulu masukkan kelas java yaitu javax.crypto.Cipher untuk membuat cipher. PKCS5 identik dengan PKCS7, namun PKCS5 hanya digunakan untuk penyandian blok yang berukuran 64 bit. Keduanya bisa dipakai pada praktiknya.

Pada aplikasi Y terdapat 3 *warning severity* yaitu:

1. Aplikasi menggunakan Generator Angka Acak yang tidak aman. Generator Angka Acak adalah alat atau algoritma yang menghasilkan urutan angka yang secara statistik independen dan tidak dapat ditebak (Autobild, 2021). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 7.5 artinya memiliki peringkat kualitatif tinggi.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-330:penggunaan nilai acak yang tidak memadai dalam konteks keamanan yang bergantung pada angka yang tidak dapat diprediksi. Ketika perangkat lunak menghasilkan nilai yang dapat diprediksi dalam konteks yang membutuhkan ketidakpastian, penyerang dapat menebak nilai berikutnya yang akan dihasilkan, dan menggunakan tebakan ini untuk menyamar sebagai pengguna lain atau mengakses informasi sensitif.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai. The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-6 artinya semua nilai acak dihasilkan menggunakan generator nomor acak yang cukup aman.

   Lokasi file yaitu;
   defpackage/bhiv.java, defpackage/tks.java, defpackage/bhdz.java, defpackage/aebc.java, defpackage/bfuo.java, defpackage/apen.java, defpackage/aagk.java, defpackage/vpz.java, defpackage/dkg.java, defpackage/xrr.java, defpackage/xne.java, defpackage/qaz.java, defpackage/aglo.java, defpackage/afvn.java, defpackage/oul.java, defpackage/anmb.java, defpackage/xrp.java, defpackage/bhku.java, defpackage/bhec.java, defpackage/sot.java, defpackage/aqkh.java, defpackage/qxv.java, defpackage/diy.java, defpackage/ote.java, defpackage/ezs.java, defpackage/ahpb.java, defpackage/bhdw.java, defpackage/gnh.java, defpackage/vjv.java, defpackage/agtd.java, defpackage/bfuh.java, defpackage/abwb.java, defpackage/aeyg.java, defpackage/qbx.java, defpackage/rlx.java, defpackage/alpc.java.

Gambar 15. menunjukkan kode pada defpackage/vjv.java.

```java
/* access modifiers changed from: package-private */
public final Long a(long j) {
    int i = 0;
    Random random = new Random((long) Objects.hash(Long.valueOf(j), this.e, this.d));
    double nextDouble = random.nextDouble();
    double d2 = this.c;
    if (d2 >= 1.0d) {
        i = (int) Math.min(Math.round((d2 + d2) * nextDouble), 2147483646L);
    } else if (nextDouble < d2) {
        i = 1;
    }
```

Gambar 15. Kode pada defpackage/vjv.java

Kode tersebut artinya memasukkan kelas java yaitu java.util.Random untuk membuat angka acak pada access modifiers. Access modifiers untuk memberi hak akses kepada user, tentu tidak semua data yang berada di dalam suatu kelas atau turunannya dapat diakses karena terdapat batasan-batasan yang berlaku. Salah satunya akses private yaitu pengaksesan class hanya dapat diakses oleh class dimana tipe ini dibuat. Access modifiers termasuk data penting yang harus dilindungi dengan aman. Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman dapat melemahkan algoritma dan memiliki kemungkinan serangan offline. Serangan offline adalah serangan yang dilakukan secara offline atau tidak membutuhkan koneksi sebagai media.

2. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. MD5 yang digunakan untuk password juga rentan terhadap serangan brute-force dengan waktu yang cepat karena hanya memiliki 128 bit. Serangan brute-force adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

   Lokasi file yaitu:
   defpackage/tks.java, defpackage/dhk.java, defpackage/die.java, defpackage/rne.java, defpackage/wap.java, defpackage/vex.java, defpackage/van.java.

Gambar 16. menunjukkan kode file MD5 pada aplikasi Y.

```java
625.            try {
626.                MessageDigest instance = MessageDigest.getInstance("MD5");
627.                if (instance != null) {
628.                    return instance;
629.                }
```

Gambar 16. Kode File MD5 pada Aplikasi Y

Kode MessageDigest.getInstance(“MD5”) berarti string akan dienkripsi menggunakan MD5. Message digest adalah sebuah nilai yang dikenal juga sebagai kriptografi checksum atau secure hash. Message digest dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi user melalui password. Checksum adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. Secure hash adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi message digest tidak dapat dikembalikan menjadi pesan semula.

3. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. SHA-1 yang digunakan untuk password juga rentan terhadap serangan brute-force dengan waktu yang cepat karena hanya memiliki 160 bit. Serangan brute-force adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 5.9 artinya memiliki peringkat kualitatif sedang.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

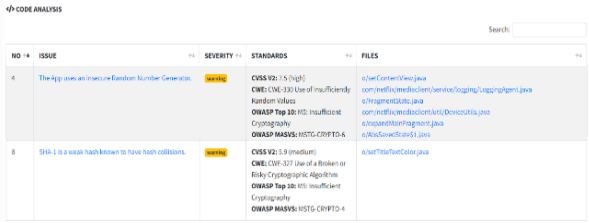
   Lokasi file yaitu;
   defpackage/aqrm.java, defpackage/amgv.java, defpackage/bgqm.java, defpackage/alhh.java, defpackage/aqpf.java, defpackage/bgcq.java, defpackage/aake.java, defpackage/wqp.java, defpackage/ahzp.java.

Gambar 17. menunjukkan kode pada defpackage/wqp.java.

```
27.         try {
28.             MessageDigest instance = MessageDigest.getInstance("SHA-1");
```

Gambar 17. Kode pada defpackage/wqp.java

Kode MessageDigest.getInstance(“SHA-1”) berarti string akan dienkripsi menggunakan SHA-1. Message digest adalah sebuah nilai yang dikenal juga sebagai kriptografi checksum atau secure hash. Message digest dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi user melalui password. Checksum adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. Secure hash adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi message digest tidak dapat dikembalikan menjadi pesan semula.



Gambar 18. Halaman Hasil Analisis File Aplikasi N pada Mobile Security Framework (MobSF) Bagian Code Analysis

Pada Gambar 18. menunjukkan halaman hasil analisis file aplikasi N pada mobile security framework (MobSF) bagian code analysis.

Pada aplikasi N terdapat 2 weak crypto yang terdiri dari 2 *warning severity* yaitu:

1. Aplikasi menggunakan Generator Angka Acak yang tidak aman. Generator Angka Acak adalah alat atau algoritma yang menghasilkan urutan angka yang secara statistik independen dan tidak dapat ditebak (Autobild, 2021). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 7.5 artinya memiliki peringkat kualitatif tinggi.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-330:penggunaan nilai acak yang tidak memadai dalam konteks keamanan yang bergantung pada angka yang tidak dapat diprediksi. Ketika perangkat lunak menghasilkan nilai yang dapat diprediksi dalam konteks yang membutuhkan ketidakpastian, penyerang dapat menebak nilai berikutnya yang akan dihasilkan, dan menggunakan tebakan ini untuk menyamar sebagai pengguna lain atau mengakses informasi sensitif.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-6 artinya semua nilai acak dihasilkan menggunakan generator nomor acak yang cukup aman.

   Lokasi file yaitu;
   o/setContentView.java, com/N/mediaclient/service/logging/LoggingAgent.java, o/FragmentState.java, com/N/mediaclient/util/DeviceUtils.java, o/expandMainFragment.java, o/AbsSavedState$1.java.

Gambar 19. menunjukkan kode pada com/netflix/mediaclient/service/logging /LoggingAgent.java.

```
16.     static {
17.         long nextLong = new Random().nextLong();
18.         if (nextLong < 0) {
19.             nextLong = 0 - nextLong;
20.         }
21.         gCritSessionId = nextLong;
22.     }
```

Gambar 19. Kode pada com/netflix/mediaclient/service/logging/LoggingAgent.java

Kode tersebut artinya memasukkan kelas java yaitu java.util.Random untuk membuat angka acak untuk *session id*. *Session id* termasuk data penting yang harus dilindungi dengan aman. Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman dapat melemahkan algoritma dan memiliki kemungkinan serangan *offline*. Serangan *offline* adalah serangan yang dilakukan secara *offline* atau tidak membutuhkan koneksi sebagai media.

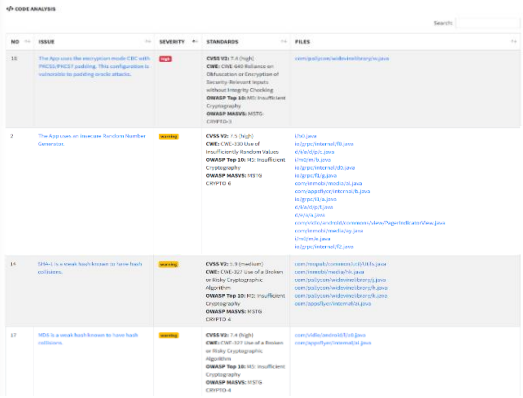
2. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. SHA-1 yang digunakan untuk *password* juga rentan terhadap serangan *brute-force* dengan waktu yang cepat karena hanya memiliki 160 bit. Serangan *brute-force* adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 5.9 artinya memiliki peringkat kualitatif sedang.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

Lokasi file yaitu o/setTitleTextColor.java. Gambar 20. menunjukkan kode pada o/setTitleTextColor.java.

```
94.         try {
95.             Result.Companion companion = Result.Companion;
96.             MessageDigest instance = MessageDigest.getInstance("SHA-1");
97.             StringBuilder sb = new StringBuilder("sha1 ");
```

Gambar 20. Kode pada o/setTitleTextColor.java

Kode MessageDigest.getInstance(“SHA-1”) berarti string akan dienkripsi menggunakan SHA-1. *Message digest* adalah sebuah nilai yang dikenal juga sebagai kriptografi *checksum* atau *secure hash*. *Message digest* dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi *user* melalui *password*. *Checksum* adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. *Secure hash* adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi *message digest* tidak dapat dikembalikan menjadi pesan semula.



Gambar 21. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada Gambar 21. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *code analysis.*

Pada aplikasi V terdapat 4 *weak crypto* yang terdiri dari 1 *high severity* yaitu aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding*. *Advanced* Encryption *Standard* (AES) merupakan algoritma yang menggunakan kunci yang sama (*private key*) untuk enkripsi dan dekripsi. Mode *Cipher Block Chaining* (CBC) adalah suatu mode dimana blok yang satu dengan blok lain saling terkait (*chained*), enkripsi dan dekripsi suatu blok data selalu melibatkan ciphertext (hasil enkripsi) blok sebelumnya. *Public Key Cryptographic Standard* (PKCS) adalah penambahan data pada awal, tengah, atau akhir sebelum enkripsi, nilai tiap bit yang ditambahkan adalah banyak bit yang ditambahkan. Serangan *oracle padding* adalah serangan yang menggunakan validasi padding dari pesan kriptografi untuk mendekripsi ciphertext (Ichi.pro, 2021). Dengan standar:

1. *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
2. *Common Weakness Enumeration* (CWE) terdeteksi CWE-649:ketergantungan pada kebingungan atau enkripsi input keamanan yang relevan tanpa pemeriksaan integritas.
3. *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
4. OWASP *The Mobile Application Security Verification Standard* (MASVS) terdeteksi MSTG-CRYPTO-3 artinya aplikasi menggunakan primitif kriptografi yang sesuai untuk kasus penggunaan tertentu, serta dikonfigurasi dengan parameter yang mematuhi praktik terbaik industri.

Lokasi file yaitu com/pallycon/widevinelibrary /w.java. Gambar 22. menunjukkan kode pada com/pallycon/widevinelibrary/w.java.

```
30.             try {
31.                 this.a = Cipher.getInstance("AES/CBC/PKCS7Padding");
```

Gambar 22. Kode pada com/pallycon/widevinelibrary/w.java

Kode tersebut artinya membuat cipherinstance dengan memanggil getInstance() metodenya dengan parameter yang diisi dengan jenis algoritma enkripsi yang digunakan yaitu AES dengan mode operasi CBC dan padding PKCS5. Terlebih dahulu masukkan kelas java yaitu javax.crypto.Cipher untuk membuat cipher. PKCS5 identik dengan PKCS7, namun PKCS5 hanya digunakan untuk penyandian blok yang berukuran 64 bit. Keduanya bisa dipakai pada praktiknya.

Pada aplikasi V terdapat 3 *warning severity* yaitu:

1. Aplikasi menggunakan Generator Angka Acak yang tidak aman. Generator Angka Acak adalah alat atau algoritma yang menghasilkan urutan angka yang secara statistik independen dan tidak dapat ditebak (Autobild, 2021). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 7.5 artinya memiliki peringkat kualitatif tinggi.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-330:penggunaan nilai acak yang tidak memadai dalam konteks keamanan yang bergantung pada angka yang tidak dapat diprediksi. Ketika perangkat lunak menghasilkan nilai yang dapat diprediksi dalam konteks yang membutuhkan ketidakpastian, penyerang dapat menebak nilai berikutnya yang akan dihasilkan, dan menggunakan tebakan ini untuk menyamar sebagai pengguna lain atau mengakses informasi sensitif.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai. *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-6 artinya semua nilai acak dihasilkan menggunakan generator nomor acak yang cukup aman.

Lokasi file yaitu;
i/b0.java, io/grpc/internal/f0.java, d/i/a/d/p/c.java, i/m0/m/b.java, io/grpc/internal/d0 .java, io/grpc/f1/g.java, com/inmobi/media/al.java, com/appsflyer/internal/b.java, io/grpc/i1/a.java, d/i/a/ d/p/f.java, d/e/a/a.java, com/V/android/commons/ view/PagerIndicatorView.java, com/inmobi/media/ ay.java, i/m0/m/e.java, io/grpc/internal/f2.java.

Gambar 23. menunjukkan kode pada com/appsflyer/internal/b.

```
try {
    Random random2 = new Random();
    short s8 = (short) 951;
    try {
        byte[] bArr11 = AppsFlyerConversionListener;
        zArr5 = zArr2;
        try {
            try {
                random2.setSeed(((Long) Class.forName($$c(s8, (byte) bArr11[72],
                Object obj14 = null;
                Object obj15 = null;
                Object obj16 = null;
                Object obj17 = null;
                while (obj14 == null) {
                    if (obj15 == null) {
                        int i37 = AppsFlyerInAppPurchaseValidatorListener;
                        int i38 = (i37 & 123) + (i37 | 123);
                        obj12 = obj14;
                        getSdkVersion = i38 % 128;
                        i10 = i38 % 2 == 0 ? 101 : 6;
```

Gambar 23. Kode pada com/appsflyer/internal/b

Kode pada gambar 23. artinya memasukkan kelas java yaitu java.util.Random untuk membuat angka acak, implementasi antarmuka untuk menangani keberhasilan dan kegagalan validasi pembelian adalah arti kode dari AppsFlyerInAppPurchaseValidatorListener. Validasi pembelian termasuk data penting yang harus dilindungi dengan aman. Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman dapat melemahkan algoritma dan memiliki kemungkinan serangan *offline*. Serangan *offline* adalah serangan yang dilakukan secara *offline* atau tidak membutuhkan koneksi sebagai media.

2. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. SHA-1 yang digunakan untuk *password* juga rentan terhadap serangan *brute-force* dengan waktu yang cepat karena hanya memiliki 160 bit. Serangan *brute-force* adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 5.9 artinya memiliki peringkat kualitatif sedang.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

Lokasi file yaitu;
com/mopub/common/util/Utils.java, com/inmobi/media/hk.java, com/pallycon/widevine library/j.java, com/pallycon/widevinelibrary/h.java, com/pallycon/widevinelibrary/k.java, com/appsflyer/ internal/ai.java.

Gambar 24. menunjukkan kode pada com/appsflyer/internal/ai.java.

```
38.         try {
39.             MessageDigest instance = MessageDigest.getInstance("SHA-1");
```

Gambar 24. Kode pada com/appsflyer/internal/ai.java

Kode MessageDigest.getInstance(“SHA-1”) berarti string akan dienkripsi menggunakan SHA-1. *Message digest* adalah sebuah nilai yang dikenal juga sebagai kriptografi *checksum* atau *secure hash*. *Message digest* dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi *user* melalui *password*. *Checksum* adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. *Secure hash* adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang

sudah diubah menjadi *message digest* tidak dapat dikembalikan menjadi pesan semula.

3. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. MD5 yang digunakan untuk *password* juga rentan terhadap serangan *brute-force* dengan waktu yang cepat karena hanya memiliki 128 bit. Serangan *brute-force* adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

Lokasi file yaitu *com/V/android/l/a0.java dan com/appsflyer/internal/ai.java*. Gambar 25. menunjukkan kode file MD5 pada aplikasi V.

```
82.     public final String f() {
83.         List J = p.J(this.a, this.f23673b);
84.         MessageDigest instance = MessageDigest.getInstance("MD5");
```

Gambar 25. Kode File MD5 pada Aplikasi V

Kode MessageDigest.getInstance("MD5") berarti string akan dienkripsi menggunakan MD5. *Message digest* adalah sebuah nilai yang dikenal juga sebagai kriptografi *checksum* atau *secure hash*. *Message digest* dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi *user* melalui *password*. *Checksum* adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. *Secure hash* adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi *message digest* tidak dapat dikembalikan menjadi pesan semula.

4.A.iii. Root Detection

Root detection adalah deteksi akses root terhadap perangkat android yang digunakan (Alviansyah dan Ramadhani, 2021). Pada *handphone* yang sudah di-*root* membuat *handphone* rentan terhadap serangan *spyware* atau virus dan memungkinkan pengguna untuk memodifikasi aplikasi sehingga menyebabkan aplikasi rusak atau rentan terhadap kecurangan.

Pada aplikasi Y tidak memiliki kemampuan *root detection* karena pada bagian *code analysis* tidak terdapat file atau kode tentang *root detection*. Pada *handphone* yang sudah di-*root*, memungkinkan pengguna untuk memodifikasi aplikasi sehingga menyebabkan aplikasi rusak atau rentan terhadap kecurangan. Saat peneliti menggunakan *handphone* yang sudah di-*root*, walaupun tidak menggunakan akun premium pada aplikasi Y tetapi dapat menggunakan fitur premium.

Gambar 26. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada gambar 26. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian *code analysis*.

Pada gambar 27. menunjukkan kode pada file com/bugsnag/android/RootDetector.java.

Gambar 27. Kode pada com/bugsnag/android/RootDetector.java

Pada aplikasi N memiliki kemampuan *root detection*. Kode pada gambar 27. memiliki fungsi untuk menemukan file *Super User* (SU). *Super user* adalah pengguna (*user*) yang memiliki hak penuh untuk mengelola sistem. Saat peneliti menggunakan *handphone* yang sudah di-*root*, aplikasi N tidak dapat ditemukan di *playsore* yang berarti aplikasi N tidak dapat diinstal pada *handphone* yang sudah di-*root*. Hal ini bagus untuk keamanan aplikasi N karena membuat pengguna tidak dapat memodifikasi aplikasi melalui *handphone* yang sudah di-*root*. Sehingga kemungkinan aplikasi rusak atau rentan terhadap kecurangan dapat dicegah.

Pada gambar 28. yang menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *code analysis*.

Gambar 28. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada gambar 29. menunjukkan kode pada file *d/h/a/b.java*.

```
if (!z2) {
    String str7 = Build.TAGS;
    if (!(str7 != null && str7.contains("test-keys"))) {
        try {
            process = Runtime.getRuntime().exec(new String[]{"which", "su"});
```

Gambar 29. Kode pada d/h/a/b.java

Pada aplikasi V memiliki kemampuan *root detection*. Kode pada gambar 29. memiliki fungsi untuk menemukan *test-keys* yang terdapat string SU. *Test-keys* berarti file APK ditandatangani dengan kunci khusus yang dibuat oleh pengembang pihak ketiga. *Super user* adalah pengguna (*user*) yang memiliki hak penuh untuk mengelola sistem.

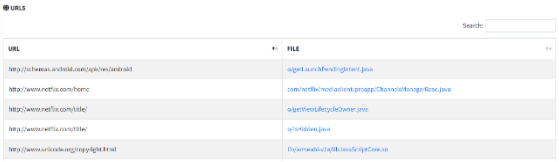
4.A.iv. SSL Bypass

Pada gambar 30. menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian URLs.

Gambar 30. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian URLs

Pada aplikasi Y terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). HTTPS adalah protokol jaringan dengan SSL. *Secure Socket Layers* (SSL) adalah lapisan keamanan tambahan untuk melindungi komunikasi data antara *client* dan *server*. Pada gambar 4.25 terdapat url http://youtube.com/streaming/metadata/segment/102015 dan http://youtube.com/streaming/metadata/segment/102015 yang menggunakan protokol jaringan HTTP. Seharusnya metadata menggunakan HTTPS yang terdapat SSL, karena untuk *video streaming* metadata video digunakan untuk memberikan informasi tentang aliran atau file video dan audio tertentu, juga dikenal sebagai esensi. Metadata dapat disematkan langsung ke dalam video atau dimasukkan sebagai file terpisah dalam wadah seperti MP4 atau MKV. Metadata memerlukan perlindungan dari SSL agar pihak ketiga tidak dapat mengaksesnya.

Pada gambar 31. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian URLs.

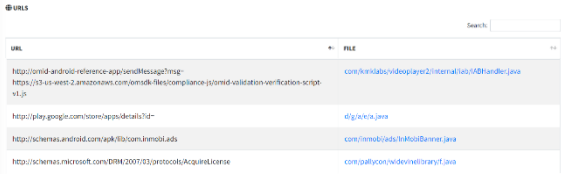


Gambar 31. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian URLs

Pada gambar 32. menunjukkan kode pada o/getViewLifecycleOwner.java.

Gambar 32. Kode pada o/getViewLifecycleOwner.java

Pada aplikasi N terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). HTTPS adalah protokol jaringan dengan SSL. *Secure Socket Layers* (SSL) adalah lapisan keamanan tambahan untuk melindungi komunikasi data antara *client* dan *server*. Pada gambar 31. terdapat url http://www.netflix.com/title/ yang menggunakan protokol jaringan HTTP. Seharusnya url tersebut menggunakan HTTPS yang terdapat SSL, karena pada o/getViewLifecycleOwner.java terdapat kode seperti pada gambar 32. yang berarti membuat tampilan pada *signup* dan *profilechange*. onCreateView() artinya membuat dan mengembalikan hierarki tampilan yang terkait dengan fragmen. *Signup* dan *profilechange* berisi data pribadi pengguna sehingga memerlukan perlindungan dari SSL agar pihak ketiga tidak dapat mengaksesnya.



Gambar 33. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian URLs

Pada gambar 33. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian URLs.

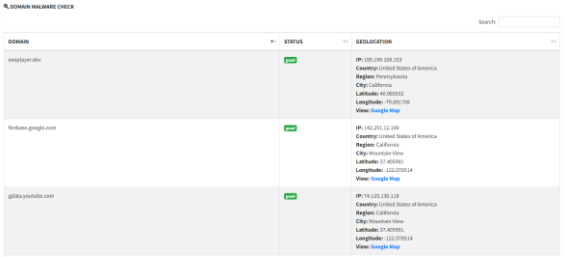
Pada gambar 34. menunjukkan kode pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java.

Gambar 34. Kode pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java

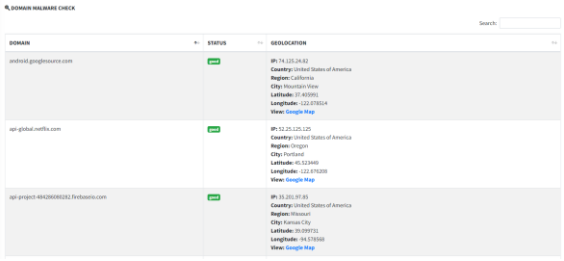
Pada aplikasi V terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). HTTPS adalah protokol jaringan dengan SSL. *Secure Socket Layers* (SSL) adalah lapisan keamanan tambahan untuk melindungi komunikasi data antara *client* dan *server*. Pada gambar 33. terdapat url http://omid-android-reference-app/sendMessage?msg= dan https://s3-us-west-2.amazonaws.com/omsdk-files/compliance-js/omid-validation-verification-script-v1.js yang menggunakan protokol jaringan HTTP. Seharusnya url tersebut menggunakan HTTPS yang terdapat SSL, karena pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java terdapat kode seperti pada gambar 34. yang berarti memverifikasi vendor (Garner, 2017) sehingga memerlukan perlindungan dari SSL agar pihak ketiga tidak dapat mengaksesnya.

4.A.v. Domain Malware Check

Pada aplikasi Y tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware* seperti pada Gambar 35. yang menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian *domain malware check*.



Gambar 35. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian *Domain Malware Check*

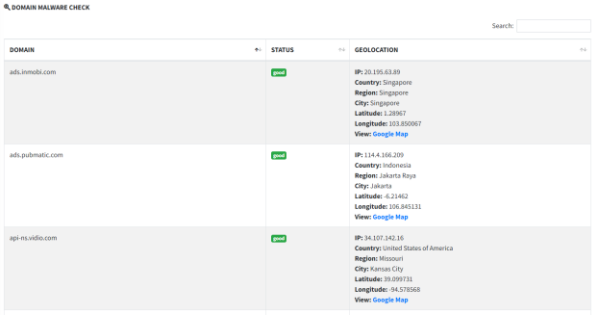


Gambar 36. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian *Domain Malware Check*

Pada aplikasi N tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware* seperti pada Gambar 36. yang menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian *domain malware check*.

Pada aplikasi V tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware* seperti pada Gambar 37. yang menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *domain malware check*.

Berdasarkan analsis statis yang dilakukan didapatkan hasil seperti yang ditampilkan dalam Tabel 4. yang menunjukkan hasil analisis statik.



Gambar 37 Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Domain Malware Check*

Tabel 4. Hasil Analisis Statik

Apk	*Dangerous Permissions*	*Weak Crypto*	*Root Detection*	SSL *Bypass*	*Domain Malware Check*	*Security Score*
Y	*Yes*	*Yes*	*No*	*Yes*	*Good*	10
N	*Yes*	*Yes*	*Yes*	*Yes*	*Good*	35
V	*Yes*	*Yes*	*Yes*	*Yes*	*Good*	10

Pada Tabel 4. dapat disimpulkan bahwa:
Aplikasi Y memiliki *dangerous permissions* yang meliputi:
1. *android.permission.CAMERA*
2. *android.permission.GET_ACCOUNTS*
3. *android.permission.MANAGE_ACCOUNTS*
4. *android.permission.RECORD_AUDIO*
5. *android.permission.WRITE_EXTERNAL_STORAGE*
6. *android.permission.USE_CREDENTIALS*

Weak crypto yang meliputi:
1. Aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding* yang termasuk *high severity*.
2. Aplikasi menggunakan Generator Angka Acak yang tidak aman yang termasuk *warning severity*.
3. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.
4. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.

Pada SSL *Bypass* menggunakan protokol jaringan HTTP pada url http://youtube.com/streaming/metadata/segment/102015 dan http://youtube.com/streaming/metadata/segment/102015 yang berisi metadata yang seharusnya memerlukan perlindungan SSL.

Aplikasi Y tidak memiliki kemampuan *root detection* karena pada bagian *code analysis* tidak terdapat file atau kode tentang *root detection*.

Pada *domain malware check* tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware*.

Mobile Security Framework (MobSF) menampilkan s*ecurity score* sebesar 10 yang artinya memiliki risiko keamanan yang kritis.

Apikasi N memiliki *dangerous permissions* yang meliputi:
1. *android.permission.GET_ACCOUNTS*
2. *android.permission.WRITE_EXTERNAL_STORAGE*

Weak crypto yang meliputi:
1. Aplikasi menggunakan Generator Angka Acak yang tidak aman yang termasuk *warning severity*.
2. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.

Aplikasi N memiliki kemampuan *root detection*, karena ditemukan kode untuk *root detection* pada com/bugsnag/android/RootDetector.java.

Pada SSL *Bypass* menggunakan protokol jaringan HTTP pada url http://www.netflix.com/title yang pada o/getViewLifecycleOwner.java berisi kode yang berarti membuat tampilan pada *signup* dan *profilechange* yang seharusnya memerlukan perlindungan SSL.

Pada *domain malware check* tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware*.

Mobile Security Framework (MobSF) menampilkan s*ecurity score* sebesar 35 yang artinya memiliki risiko keamanan yang tinggi.

Aplikasi V memiliki *dangerous permissions*,yang meliputi:

1. *android.permission.RECORD_ AUDIO*
2. *android.permission.CAMERA*
3. *android.permission.READ_EXTERNAL_STORAGE*
4. *android.permission.WRITE_EXTERNAL_STORAGE*

Weak crypto yang meliputi:

1. Aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding* yang termasuk *high severity*.
2. Aplikasi menggunakan Generator Angka Acak yang tidak aman yang termasuk *warning severity*.
3. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.
4. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.

Aplikasi V memiliki kemampuan *root detection*, karena ditemukan kode untuk *root detection* pada d/h/a/b.java.

Pada SSL *Bypass* menggunakan protokol jaringan HTTP pada url http://omid-android-reference-app/sendMessage?msg= dan https://s3-us-west-2.amazonaws.com/omsdk-files/compliance-js/omid-validation-verification-script-v1.js yang pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java berisi kode yang berarti memverifikasi vendor yang seharusnya memerlukan perlindungan SSL.

Pada *domain malware check* tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware*.

Mobile Security Framework (MobSF) menampilkan s*ecurity score* sebesar 10 yang artinya memiliki risiko keamanan yang kritis.

4.B. Rekomendasi

Berdasarkan hasil penelitian, peneliti memberikan saran yang diajukan sesuai dengan masalah pada latar belakang penelitian. Asumsi dari hasil penelitian ini dapat dijadikan sebagai bahan pertimbangan oleh pengembang dan pengguna aplikasi video *streaming* yaitu Y, N, dan V untuk dilanjutkan dalam penelitian terkait analisis statik pada *mobile security framework* (MobSF).

4.B.i. Dangerous Permissions

Sejak android versi 11 (API level 30), setiap kali aplikasi meminta izin untuk mikrofon atau kamera, dialog izin pengguna akan menyertakan opsi yaitu hanya kali ini (Developer.android.com, 2020a). *Application Programming Interface* (API) Level adalah nilai integer yang secara unik mengidentifikasi revisi API framework yang ditawarkan oleh versi platform Android. Peneliti merekomendasikan bagi pengguna aplikasi untuk memilih opsi hanya kali ini atau agar lebih terjamin keamanannya, pengguna bisa menutup kamera dengan stiker atau benda lainnya agar terhindar dari *spy* atau kamera tidak bisa mengambil foto atau video tanpa pengguna ketahui.

Peneliti merekomendasikan bagi pengguna aplikasi untuk izin akses penyimpanan dan daftar akun, apabila pengguna merasa izin ini tidak begitu diperlukan, sebaiknya pengguna menonaktifkan izin tersebut agar kemungkinan data pada penyimpanan dan data pribadi lainnya disalahgunakan tidak terjadi.

4.B.ii. Weak Crypto

Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman untuk data penting dapat melemahkan algoritma dan memiliki kemungkinan serangan *offline*. Peneliti merekomendasikan bagi pengembang aplikasi untuk menggunakan generator angka acak yang aman (*SecureRandom*) berfungsi untuk menginisialisasi kunci kriptografis yang diciptakan oleh *KeyGenerator* (Developer.android.com, 2020b).

Untuk memperkuat enkripsi data bagian *password*, peneliti merekomendasikan bagi pengembang aplikasi agar algoritma MD5 bisa dikombinasikan dengan algoritma kriptografi yang lain seperti vigenere cipher. Vigenere cipher adalah sebuah enkripsi dengan melakukan beberapa pergeseran yang direpresentasikan menggunakan satu kata kunci. Kombinasi algoritma MD5 dan vigenere cipher ini bisa menjadi lebih kuat jika dibandingkan hanya menggunakan MD5 saja karena kombinasi algoritma ini akan melakukan proses enkripsi dua kali (Sibyan, 2017). Untuk SHA-1 bisa digantikan dengan SHA-3 yang memiliki ketahanan terhadap serangan *brute-force* lebih baik karena waktu yang ditempuh lebih lama untuk mendapatkan plaintext 8, 9 dan 10 karakter dari hash SHA-3 (Kurniawan, Kusyanti dan Nurwarsito, 2017).

4.B.iii. Root Detection

Pada *handphone* yang sudah di-*root* membuat *handphone* rentan terhadap serangan *spyware* atau virus (Nguyen-Vu et al., 2017) dan memungkinkan pengguna untuk memodifikasi aplikasi sehingga menyebabkan aplikasi rusak atau rentan terhadap kecurangan (Sulistio, 2021). Karena hal tersebut, peneliti merekomendasikan bagi pengembang aplikasi Y untuk menambahkan kemampuan *root detection*. Peneliti merekomendasikan untuk pengguna aplikasi *video streaming* agar tidak melakukan *root* terhadap *handphone* sehingga terhindar dari serangan *spyware* atau virus.

4.B.iv. SSL Bypass

Pada aplikasi Y, N dan V terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). Peneliti merekomendasikan agar pengembang aplikasi Y, N dan V tidak menggunakan protokol jaringan *Hypertext Transfer Protocol* (HTTP) untuk data yang bersifat pribadi seperti video yang dikhususkan untuk pelanggan berbayar atau premium, data saat *login*, dan data profil pengguna

sebaiknya menggunakan *Hypertext Transfer Protocol Secure* (HTTPS) karena terdapat *Secure Socket Layer* (SSL) yang terenkripsi dengan baik sehingga pihak ketiga sulit untuk mengaksesnya (Prayama, Yuhefizar dan Amelia Yolanda, 2021).

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa cara melakukan analisis statik menggunakan *mobile security framework* (MobSF) adalah dengan cara meng-upload file aplikasi *video streaming* dengan format APK ke *mobile security framework*. Saat proses telah selesai, maka akan muncul hasil analisis file tersebut. Tingkat keamanan pada aplikasi *video streaming* berbasis android pada aplikasi Y mendapatkan skor keamanan sebesar 10 yang artinya memiliki risiko keamanan yang kritis, aplikasi N mendapatkan skor keamanan sebesar 35 yang artinya memiliki risiko keamanan yang tinggi, aplikasi V mendapatkan skor keamanan sebesar 10 yang artinya memiliki risiko keamanan yang kritis. 3. Terdapat celah keamanan pada aplikasi video streaming berbasis android yaitu pada aplikasi Y, N dan V memiliki *dangerous permissions*, *weak crypto*, dan SSL *bypass*. Hanya Aplikasi Y yang tidak memiliki *root detection*.

Berdasarkan hasil penelitian dan kesimpulan, berikut saran dari penulis untuk penelitian selanjutnya yaitu menganalisis aplikasi *video streaming* lebih banyak lagi, menambahkan tahapan analisis statik pada aplikasi *video streaming*, menggunakan metode analisis lainnya seperti analisis dinamis.

DAFTAR PUSTAKA

Zen, B. P., Gultom, R. A., & Reksoprodjo, A. H. (2020). Analisis Security Assessment Menggunakan Metode Penetration Testing dalam Menjaga Kapabilitas Keamanan Teknologi Informasi Pertahanan Negara. Teknologi Penginderaan, 2(1).

Abraham, A., 2021. *Mobile Security Framework MobSF*. [daring] https://github.com. Tersedia pada: <https://github.com/MobSF/Mobile-Security-Framework-MobSF> [Diakses 30 Mar 2021].

Alviansyah, F.A. dan Ramadhani, E., 2021. Implementasi Dynamic Application Security Testing pada Aplikasi Berbasis Android. *Automata*, [daring] 2(1), hal.1–6. Tersedia pada: <https://journal.uii.ac.id/AUTOMATA/article/view/17387>.

Anwar, N., Akbar, S.A., Elektro, T. dan Dahlan, U.A., 2020. Ekstraksi Logis Forensik Mobile Pada Aplikasi E-Commerce Android. 2(1), hal.1–10.

Arsam, A., 2017. Jurnal Ilmiah Komputer dan Informatika (KOMPUTA) Pembangunan Aplikasi Video Streaming Berbasis Android di STV Bandung. *Jurnal Ilmiah Komputer dan Informatika (KOMPUTA)*.

Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020. *LAPORAN SURVEI INTERNET APJII 2019 – 2020 (Q2)*.

Asrianti, S., 2020. *April 2020, Pengguna Aktif Aplikasi Vidio Capai 62 Juta*. [daring] Tersedia pada: <https://www.republika.co.id/berita/qaefoi328/april-2020-pengguna-aktif-aplikasi-vidio-capai-62-juta>.

Autobild, 2021. *Acak Angka Online (Random Number Generator)*. [daring] Autobild. Tersedia pada: <https://www.autobild.co.id/p/acak-angka-online-random-number.html>.

Developer.android.com, 2020a. *Pembaruan izin di Android 11*. [daring] developer.android.com. Tersedia pada: <https://developer.android.com/about/versions/11/privacy/permissions?hl=id>.

Developer.android.com, 2020b. *Tips keamanan*. [daring] developer.android.com. Tersedia pada: <https://developer.android.com/training/articles/security-tips?hl=id>.

Dify Virginia Rizaldy, 2021. VIDEO ON DEMAND : CARA MUDAH MENONTON FILM (Studies on Consumer Behavior). *STIE PGRI Dewantara Jombang*.

Dvorak, M., 2021. *Video Streaming Site Data Breach Exposes 2M Customers*. [daring] Tersedia pada: <www.askcybersecurity.com/adult-video-streaming-data-breach-2m-customers>.

Dynatrace, 2021. *Improve DevSecOps Processes - Download Free Research*. [daring] Tersedia pada: <https://www.dynatrace.com/monitoring/solutions/devsecops-ciso/?utm_source=google&utm_medium=cpc&utm_term=devsecops&utm_campaign=id-application-security&utm_content=none&gclid=CjwKCAiAsNKQBhAPEiwAB-I5zXzMTYBTjZyM4VaxuqE9_qh1NC1ND26rAjBftfXyIo3lj7DbJk9WRxoC>.

Garner, N., 2017. *Class VerificationScriptResource*. [daring] docs.iabtechlab.com. Tersedia pada: <https://docs.iabtechlab.com/omsdk-1.3/android/com/iab/omid/library/adsession/VerificationScriptResource.html>.

Hanifurohman, C. dan Hutagalung, D.D., 2020. Analisis Statis Menggunakan Mobile Security Framework Untuk Pengujian Keamanan Aplikasi Mobile E-Commerce Berbasis Android. *Sebatik*, 24(1), hal.22–28.

Ichi.pro, 2021. *Kriptanalisis AES-CBC (Bagian-1)*. [daring] ichi.pro. Tersedia pada: <https://ichi.pro/id/kriptanalisis-aes-cbc-

bagian-1-34012510098655>.

Idan, 2016. *Droidmon*. [daring] Tersedia pada: <https://github.com/idanr1986/droidmon>.

ISOCENTER INDONESIA, 2022. *ISO 27001 Information Security*. [daring] Tersedia pada: <https://isoindonesiacenter.com/iso-27001-information-security/>.

Kartono, A., Sularsa, A. dan Ismail, S.J.I., 2019. Membangun Sistem Pengujian Keamanan Aplikasi Android Menggunakan Mobsf. 5(1), hal.146–151.

Kurniawan, F., Kusyanti, A. dan Nurwarsito, H., 2017. Analisis dan Implementasi Algoritma SHA-1 dan SHA-3 pada Sistem Autentikasi Garuda Training Cost. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, [daring] 1(9), hal.803–812. Tersedia pada: <http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/247>.

NDK, F., 2020. *Agile vs DevOps vs CI/CD: Apa Saja Perbedaanya?* [daring] PT. Logique Digital Indonesia. Tersedia pada: <https://www.logique.co.id/blog/2020/12/16/agile-vs-devops-vs-cicd/>.

Nguyen-Vu, L., Chau, N.T., Kang, S. dan Jung, S., 2017. Android Rooting: An Arms Race between Evasion and Detection. *Security and Communication Networks*, 2017(4).

NKD, F., 2020. *Data Breaches 2020: Kasus Pelanggaran Data Terbesar yang Terjadi di Tahun 2020*. [daring] Tersedia pada: <https://www.logique.co.id/blog/2020/12/29/data-breaches-2020/>.

Prayama, D., Yuhefizar dan Amelia Yolanda, 2021. Protokol HTTPS, Apakah Benar-benar Aman? *Journal of Applied Computer Science and Technology*, 2(1), hal.7–11.

Pusparisa, Y., 2020. *Pelanggan Netflix Naik 15,8 Juta di Tengah Pandemi Covid-19*. [daring] Tersedia pada: <https://databoks.katadata.co.id/datapublish/2020/04/24/pelanggan-netflix-naik-158-juta-di-tengah-pandemi-covid-19>.

Putra, A.A., Nurhayati, O.D. dan Windasari, I.P., 2016. Perencanaan dan Implementasi Information Security Management System Menggunakan Framework ISO/IEC 20071. *Jurnal Teknologi dan Sistem Komputer*, 4(1), hal.60.

Rovo, 2019. *Xposed Installer*. [daring] Tersedia pada: <https://repo.xposed.info/module/de.robv.android.xposed.installer>.

Shahriar, H., Arabin Talukder, M. dan Saiful Islam, M., 2019. An Exploratory Analysis of Mobile Security Tools. [daring] (February 2020). Tersedia pada: <https://digitalcommons.kennesaw.edu/ccerp https://digitalcommons.kennesaw.edu/ccerp/2019/research/4>.

Sibyan, H., 2017. Implementasi Enkripsi Basis Data Dengan Algoritma Demgan Algoritma MD5 (Message Digest Algorithm 5) Dan Vigenere Cipher. *Ppkm I*, 5, hal.114–121.

Simon Kemp, 2021. *DIGITAL 2021: THE LATEST INSIGHTS INTO THE 'STATE OF DIGITAL.'* [daring] Tersedia pada: <https://wearesocial.com/blog/2021/01/digital-2021-the-latest-insights-into-the-state-of-digital>.

Skylot, 2020. *Jadx*. [daring] Tersedia pada: <https://github.com/skylot/jadx>.

Sulistio, P., 2021. *Dampak Negatif Root HP Android*. [daring] kominfo.bengkulukota.go.id. Tersedia pada: <https://kominfo.bengkulukota.go.id/dampak-negatif-root-hp-android/>.

Suroso, S., Ciksadan, C. dan Sholihatun, S., 2020. Analisis Quality of Service Video Streaming Youtube Dan Rma Wlan Di Politeknik Negeri Sriwijaya. *TESLA: Jurnal Teknik Elektro*, 22(2), hal.93.

Tanjung, I., 2016. *Live-streaming viewers exposed to malware, data theft: Study*. [daring] Tersedia pada: <https://www.thejakartapost.com/life/2016/07/12/live-streaming-viewers-exposed-to-malware-data-theft-study.html>.

Tansen, E. dan Nurdiarto, D.W., 2020. Analisis Dan Deteksi Malware Dengan Metode Hybrid Analysis Menggunakan Framework Mobsf. 4(2), hal.191–201.

Tesalonica, 2020. *Jumlah pengguna unik YouTube di Indonesia capai 93 juta*. [daring] Tersedia pada: <https://www.tek.id/tek/jumlah-pengguna-unik-youtube-di-indonesia-capai-93-juta-b1ZT79iPE>.

Tugas, M. dan El, K., 2016. Fuzzing untuk Menemukan Kerentanan Aplikasi Web. hal.1–17.

Winder, D., 2020. *235 Million Instagram, TikTok And YouTube User Profiles Exposed In Massive Data Leak*. [daring] Tersedia pada: <https://www.forbes.com/sites/daveywinder/2020/08/19/massive-data-leak235-million-instagram-tiktok-and-youtube-user-profiles-exposed/>.

ANALISIS STATIK KEAMANAN APLIKASI VIDEO STREAMING BERBASIS ANDROID MENGGUNAKAN MOBILE SECURITY FRAMEWORK (MOBSF)

By Bita Parga Zen

WORD COUNT 10558 TIME SUBMITTED 08-AUG-2023 01:26PM

PAPER ID 101836705

CyberSecurity dan Forensik Digital e-ISSN: 2615-8442
Vol. 4, No. 2, November 2021, hlm. 63-80

ANALISIS STATIK KEAMANAN APLIKASI *VIDEO STREAMING* BERBASIS ANDROID MENGGUNAKAN *MOBILE SECURITY FRAMEWORK* (MOBSF)

Fitri Nurindahsari[1], Bita Parga Zen[2]
[1,2]Program Studi Informatika
Institut Teknologi Telkom Purwokerto
Email: [1]fitrinurindahsari26@gmail.com, [2]bita@ittelkom-pwt.ac.id

Penulis korespondensi : bita@ittelkom-pwt.ac.id

Abstrak

Video streaming yaitu layanan transmisi video dan audio melalui internet. Layanan ini dibroadcast kepada banyak pengguna yang mengakses suatu situs *video streaming*. Pengguna aplikasi *video streaming* di Indonesia mengalami peningkatan dari tahun 2018 sampai 2021. Kondisi pada tahun 2021 sampai 2022 yang masih dalam masa pandemi semakin meningkatkan keinginan masyarakat untuk menggunakan aplikasi *video streaming*, terkhusus aplikasi *video streaming* berbasis android. Pengguna aplikasi *video streaming* tidak bisa mengabaikan keamanan dari aplikasi tersebut. Dengan banyaknya akses ke aplikasi *video streaming* berbasis android dapat menjadi target utama oleh *Cracker* dalam melakukan tindak kejahatan. Apalagi terdapat data pribadi pengguna aplikasi *video streaming* seperti nama pengguna, kata sandi dan nomor handphone, jika hal ini sampai dapat diakses oleh oknum yang tidak bijak maka dapat disalahgunakan. Tujuan penelitian ini untuk menemukan celah keamanan yang terdapat pada aplikasi *video streaming* berbasis android. Peneliti menggunakan *mobile security framework* (MobSF) untuk menganalisis statik keamanan dengan parameter *dangerous permissions*, *weak crypto*, *root detection*, SSL *bypass* dan *domain malware check* pada aplikasi *video streaming* berbasis android. Hasil yang diperoleh dari penelitian ini yaitu aplikasi Y, N dan V memiliki *dangerous permissions*, *weak crypto*, dan SSL *bypass*, dan *domain malware check* dengan status *good*. Hanya aplikasi Y yang tidak memiliki *root detection*.

Kata kunci: *analisis statik*, *android*, *keamanan*, MobSF, *video streaming*.

SECURITY STATIC ANALYSIS OF ANDROID-BASED VIDEO STREAMING APPLICATION USING MOBILE SECURITY FRAMEWORK (MOBSF)

Abstract

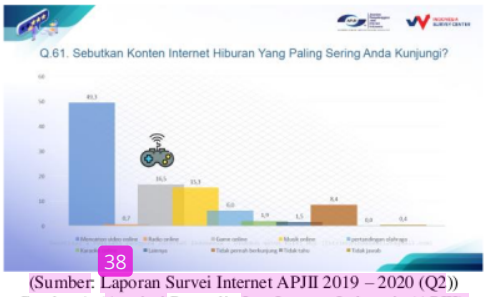
Video streaming is an internet-based video and audio transmission service. This service is transmitted to a large number of people that visit an online video website. Video streaming application users in Indonesia have increased from 2018 to 2021. The current situation, which is still in a pandemic phase, is increasing the public's desire to use video streaming programs, particularly those based on Android. Users of streaming video applications cannot overlook the app's security. Because so many people have access to Android-based video streaming apps, crackers may use them to conduct crimes. Furthermore, there is personal data of users of video streaming programs, such as usernames, passwords, and phone numbers, which could be accessed by unscrupulous individuals. The goal of this research is to identify security flaws in an Android-based video streaming software. On android-based video streaming apps, researchers employed the mobile security framework (MobSF) to examine static security with parameters such as dangerous permissions, weak crypto, root detection, SSL bypass and domain malware check. Applications Y, N, and V have dangerous permissions, weak crypto, SSL bypass and domain malware check with a good status, according to the findings of this study. Only application Y is not able to root detection.

***Keywords**: android, MobSF, security, static analysis, video streaming.*

1. PENDAHULUAN

Kemajuan teknologi informasi dan sistem pertahanan siber saat ini berkembang begitu pesat (BP Zen, 2020) Seiring kemajuan teknologi pada 2021 dan 2022 yang semakin berkembang, pemakaian internet pun semakin meningkat. Dalam kehidupan masyarakat, internet mencakup berbagai sektor termasuk sektor hiburan. Kondisi pada tahun 2021 sampai 2022 yang masih dalam masa pandemi semakin meningkatkan keinginan masyarakat untuk menggunakan aplikasi *video streaming*, terkhusus aplikasi *video streaming* berbasis android.

Asosiasi Penyedia Jasa Internet Indonesia (APJII) menyebutkan hingga kuartal II tahun 2020 di sektor hiburan, video daring menjadi akses hiburan terbesar dengan 49,3%, disusul game daring 16,5%, dan musik daring 15,3% (Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020). Gambar 1. menunjukkan hasil survei konten internet hiburan yang paling sering dikunjungi.



(Sumber: Laporan Survei Internet APJII 2019 – 2020 (Q2))
Gambar 1. (Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020)

We Are Social mengungkapkan laporan "*Digital 2021: The Latest Insights Inti The State of Digital*" yang diterbitkan pada 11 Februari 2021, alasan masyarakat menggunakan internet untuk menonton video online, acara tv dan film sebesar 51,7% (Simon Kemp, 2021). Dapat disimpulkan jika pengaksesan *video streaming* di Indonesia mengalami peningkatan selama 4 tahun terakhir. Tabel 1. menunjukkan data akses aplikasi *video streaming*.

Tabel 1. Data Akses Aplikasi *Video streaming* (Dify Virginia Rizaldy, 2021)(Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020)(Simon Kemp, 2021)

No.	Tahun	Presentase Akses Aplikasi *Video streaming*
1.	2018	19%
2.	2019	49,3%
3.	2020	49,3%
4.	2021	51,7 %

Data dari dari ComScore menunjukkan, ada lebih dari 93 juta penonton unik di Indonesia (berusia di atas 18 tahun) yang menonton video di Y setiap bulannya selama 2021. Jumlah itu tercatat meningkat hingga 10 juta dibanding tahun 2020 (Tesalonica, 2020). Table 2. menunjukkan data pelanggan berbayar N di Indonesia.

Tabel 2. Data Pelanggan Berbayar N di Indonesia (Pusparisa, 2020)

No.	Tahun	Banyak Pelanggan Berbayar
1.	TW 2-2019	151,6 Juta
2.	TW 3-2019	158,3 Juta
3.	TW 4-2019	167,1 Juta
4.	TW 1-2020	182,9 Juta

Pada gambar 2. menunjukkan data pelanggan N. V mencatat peningkatan signifikan jumlah pengguna aplikasi aktif dan pengunduhan aplikasi selama masa pembatasan sosial berskala besar (PSBB). Pengguna aktif layanan *streaming video on demand* itu mencapai 62 juta pada April 2020. Peningkatan mulai terlihat setelah kehadiran program "V Bebas Nonton" (Asrianti, 2020).

(Sumber: https://databoks.katadata.co.id)
Gambar 2. Data Pelanggan N (Pusparisa, 2020)

Tabel 3. menunjukkan data pengguna aplikasi *video streaming* pada tahun 2020.

Tabel 3. Data Pengguna Aplikasi *Video streaming* pada tahun 2020, (Tesalonica, 2020)(Pusparisa, 2020)(Asrianti, 2020)

No.	Nama Aplikasi *Video streaming*	Pengguna Aplikasi *Video streaming*
1.	Y	93 Juta
2.	N	182 Juta
3.	V	62 Juta

Situs web *streaming* membuat pemirsa tertarik untuk menonton acara *online*, baik acara olahraga, pesta, saluran TV, atau lainnya. Terlepas dari risiko menonton *streaming* langsung gratis, banyak pemirsa mungkin tidak menyadari bahaya sebenarnya yang mengintai. Di antara skema termasuk menempatkan iklan *overlay* dengan tombol "tutup" palsu ke *video streaming* yang dapat mengelabui pemirsa agar mengunduh *malware* berbahaya (Tanjung, 2016).

Dibalik kemudahan penggunaan dan keuntungan yang diberikan oleh aplikasi *video streaming*, pengguna juga tidak bisa mengabaikan keamanan dari aplikasi *video streaming* tersebut. Dengan semakin banyaknya akses ke aplikasi *video streaming* berbasis android dapat menjadi target utama oleh *Cracker* dalam melakukan tidak kejahatan. Dari banyaknya aplikasi android yang beredar tidak semuanya sudah menerapkan pengujian keamanan dengan baik sesuai ISO/27001 (Putra, Nurhayati dan Windasari, 2016). ISO/27001 merupakan suatu standar Internasional dalam menerapkan sistem manajemen kemanan informasi atau lebih dikenal dengan *Information Security Management Systems* (ISMS) (ISOCENTER INDONESIA, 2022). Apalagi terdapat data pribadi pengguna aplikasi *video streaming* seperti email dan nomor handphone, jika hal ini sampai dapat diakses

oleh oknum yang tidak bijak maka dapat disalahgunakan.

Situs *video streaming* CAM4 pada Bulan Maret menjadi *website* yang terkena serangan data *breach*. Hal ini diungkapkan oleh Anurag Sen dari Safety Detectives. Kebocoran data ini muncul karena adanya kesalahan konfigurasi *server ElasticSearch* yang berakibat pada *database* pengguna dan anggota *platform* menjadi tidak aman. Akibatnya, sebesar 10,88 miliar data pribadi dapat terakses (NKD, 2020).

Data pelanggan sebanyak 10.000 dicuri dari situs *video streaming* MyFreeCams.com dalam pelanggaran data pada tahun 2021. Penjahat dunia maya telah menjual data pelanggan yang dicuri seharga 1.500 dolar AS secara *online* meliputi nama pengguna, kata sandi tidak terenkripsi, alamat email, saldo MyFreeTokens pelanggan untuk setiap akun. Data dieksfiltrasi pada Desember 2020 melalui serangan injeksi SQL (Dvorak, 2021).

Pada tahun 2020 ada sekumpulan laporan tentang data akun yang muncul di forum kejahatan *dark web*. Dari audit *dark web* menunjukkan ada 15 miliar *login* yang dicuri dari 100.000 pelanggaran, hingga peretas yang memberikan 386 juta catatan curian secara gratis. Basis data yang tidak aman dengan cepat menjadi masalah perlindungan data yang sangat besar, sehingga meninggalkan data profil pribadi hampir 235 juta pengguna Instagram, TikTok, dan Y untuk diperebutkan (Winder, 2020).

Pada penelitian sebelumnya, peneliti pertama meneliti tentang analisis dan deteksi *malware* dengan metode *hybrid analysis* menggunakan *framework* MobSF yang memiliki masalah pada mencari hal-hal yang janggal pada *malware* yang menginfeksi perangkat android dan mendapatkan beberapa karakteristik dari *malware Bouncing* dan *Riltok*. Peneliti kedua meneliti tentang analisis statis menggunakan *mobile security framework* untuk pengujian keamanan aplikasi *mobile e-commerce* berbasis android yang memiliki masalah pada mencari celah-celah keamanan pada aplikasi *mobile e-commerce* dan mendapatkan lima besar *mobile e-commerce* di Indonesia memiliki beberapa celah keamanan masih ada dengan tingkat kemanan yang relatif sama. Peneliti ketiga meneliti tentang *a comprehensive analysis of the android permissions system* yang memiliki masalah pada mencari masalah keamanan dan cara aplikasi menangani informasi dan sumber daya berisiko tinggi dari sistem izin android dan mendapatkan beberapa ancaman keamanan yang dapat membuka OS untuk berbagai serangan.

Pada penelitian kali ini, penulis menggunakan *mobile security framework* untuk melakukan analisis statik pada aplikasi *video streaming* dengan subyek yang digunakan yaitu situs *video streaming* di Indonesia inisial Y, N dan V, dengan harapan penelitian ini dapat mendukung analisis sistem keamanan pada situs *video streaming* yang dapat menjadi rujukan untuk menilai tingkat keamanannya.

Langkah yang akan dilakukan peneliti yaitu parameter analisis *dangerous permissions*, *weak crypto*, *root detection*, SSL *bypass* dan *domain malware check*, sehingga dapat ditemukan celah keamanan pada aplikasi *video streaming* jika memang ada. Kemudian peneliti menganalisis statik dan merekomendasikan untuk celah keamanan yang ditemukan.

Berdasarkan latar belakang yang sudah dijabarkan, peneliti memilih topik tersebut untuk diteliti dengan judul Analisis Statik Keamanan Aplikasi *Video Streaming* Berbasis Android Menggunakan *Mobile Security Framework* (MobSF) dengan tujuan adanya penelitian ini dapat berguna untuk memberikan kesadaran terhadap masyarakat maupun pengguna aplikasi, memberikan masukan kepada pihak pengembang aplikasi untuk terus meningkatkan aspek keamanan dan dari pengguna aplikasi menyadari akan risiko keamanan terhadap setiap aplikasi *video streaming*.

2. TINJAUAN PUSTAKA

2.A. Aplikasi Video Streaming

Streaming adalah suatu teknologi untuk menjalankan file audio atau video secara langsung menggunakan jaringan internet maupun lokal (Suroso, Ciksadan dan Sholihatun, 2020).

File audio atau video yang terletak pada sebuah *server* dapat secara langsung dijalankan pada komputer *client* sesaat setelah ada permintaan dari pengguna sehingga proses *download* file tersebut yang membutuhkan waktu cukup lama dapat dihindari. Saat file tersebut diputar maka akan terbentuk sebuah *buffer* di komputer *client* dan data audio atau video tersebut akan mulai di-*download* ke dalam *buffer* yang telah terbentuk pada mesin *client*. Setelah *buffer* terisi dalam waktu beberapa detik, maka secara otomatis file video ataupun audio akan di jalankan oleh sistem. Sistem akan membaca informasi dari *buffer* sambil tetap melakukan proses *download* file sehingga proses *streaming* tetap berlangsung ke mesin *client* (Arsam, 2017).

2.B. Mobile Security Framework (MobSF)

Mobile Security Framework (MobSF) adalah *framework* yang digunakan untuk pengujian penetrasi terhadap aplikasi seluler (android/iOS/windows) otomatis yang mampu melakukan analisis statis, dinamis, dan *malware*. MobSF digunakan untuk analisis keamanan yang efektif dan cepat dari aplikasi seluler dan mendukung kedua binari (*Android Package Kit* (APK), *iPhone Application* (IPA) & APPX yang merupakan format file *windows store*) dan kode sumber zip. MobSF dapat melakukan pengujian aplikasi dinamis saat *runtime* untuk aplikasi Android dan memiliki kemampuan *fuzzing* API Web yang didukung oleh boofuzz, pemindai keamanan khusus Web API. *Fuzzing* merupakan suatu metode mencari kesalahan piranti lunak dengan menyediakan input yang tidak diduga lalu mengamati hasilnya (Tugas dan El, 2016). MobSF dirancang untuk membuat integrasi *Continuous Integration*/

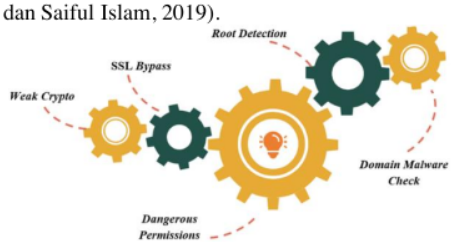
Continuous Delivery (CI/CD) atau *Development, Security, and Operations* (DevSecOps) secara bagus (Abraham, 2021). CI/CD merupakan metode untuk mengirimkan aplikasi ke pelanggan secara rutin dengan memperkenalkan otomatisasi ke dalam tahapan pengembangan aplikasi (NDK, 2020). DevSecOps adalah kerangka kerja kolaborasi yang memperluas dampak *Development and Operations* (DevOps) dengan menambahkan praktik keamanan ke proses pengembangan dan pengiriman perangkat lunak (Dynatrace, 2021).

Mobile Security Framework adalah kerangka kerja gabungan yang melakukan analisis statis dan dinamis dari sebuah APK. MobSF dikembangkan berdasarkan Python. Untuk menghasilkan laporan MobSF menggunakan html. Ini menjalankan server lokal melalui baris perintah di komputer *host* (Shahriar, Arabin Talukder dan Saiful Islam, 2019). Semua alat analisis statis dan dinamis yang ada melakukan analisis menggunakan *DroidMon-Dalvik Monitoring Framework* (Idan, 2016) dan *Xposed Module Repository*. Xposed adalah kerangka kerja untuk modul yang dapat mengubah perilaku sistem dan aplikasi tanpa menyentuh APK apa pun. Itu bagus karena itu berarti modul dapat bekerja untuk versi yang berbeda dan bahkan ROM tanpa perubahan apa pun (selama kode aslinya tidak terlalu banyak diubah) (Tansen dan Nurdiarto, 2020).

2.C. Analisis Statik

Metode statis analis dilakukan tanpa benar-benar menjalankan programnya dan lebih seperti menyelidiki apa yang terjadi pada *source code* dengan tujuan utama yaitu untuk mengetahui kode berbahaya seperti apa yang tertanam dalam aplikasi tersebut (Skylot, 2020).

Analisis statis mengacu pada dekompilasi APK aplikasi ke file Java dan XML yang sesuai. Untuk melakukan analisis statis, penganalisis statis harus memiliki fitur untuk memeriksa XML dengan benar dan presisi. File Java dapat diekstraksi dengan dekompiler DEXtoJar (Rovo, 2019). Alat analisis Statis mendekompilasi kode APK ke format yang dapat dibaca manusia. Sehingga penganalisa dapat membaca kode dan mengidentifikasi kerentanan yang mungkin dimiliki aplikasi (Shahriar, Arabin Talukder dan Saiful Islam, 2019).



Gambar 3. Parameter Analisis Statik

Dari hasil analisis statis maka didapat hasil daftar *malwords* yang sering muncul sebagai daftar string berbahaya dengan tingkat resiko masing-masing, setelah mendapatkan daftar malwords, peneliti akan menganalisisnya (Anwar et al., 2020). Gambar 3. menunjukkan parameter pada analisis statik.

2.D. Weak Crypto

Analisis *weak crypto* dilakukan dengan acuan ada atau tidaknya implementasi algoritma kriptografi yang lemah atau penggunaan algoritma kriptografi yang sudah usang atau sudah dianggap tidak layak.

2.E. SSL Bypass

Analisis SSL *bypass* dilakukan dengan melakukan cek ada atau tidaknya *service* yang melibatkan protokol http yang tidak mewajibkan penggunaan SSL sebagai persyaratan keamanan transaksi dalam protokol http menggunakan SSL seperti mengizinkan http di manifest, atau terdapat string berkonten http:// yaitu *weak implementation*.

2.F. Dangerous Permissions

Aplikasi Android dibangun untuk melakukan serangkaian tindakan, beberapa di antaranya memerlukan izin dari pengguna. Analisis terhadap *permission* dilakukan dengan melihat pada seberapa banyak *dangerous permissions* yang digunakan oleh aplikasi.

2.G. Root Detection

Analisis *root detection* dilakukan dengan melakukan cek ada atau tidaknya aplikasi mempunyai fungsi untuk melakukan deteksi akses root terhadap perangkat android yang digunakan. Dimana akses memungkinkan untuk akses langsung ke dalam sistem termasuk data-data yang dimiliki aplikasi.

2.H. Domain Malware Check

Analisis *domain malware check* dilakukan dengan melakukan cek ada atau tidaknya domain-domain yang terdapat dalam aplikasi terindikasi dalam kategori domain yang mengandung malware atau tidak (Hanifurohman dan Hutagalung, 2020).

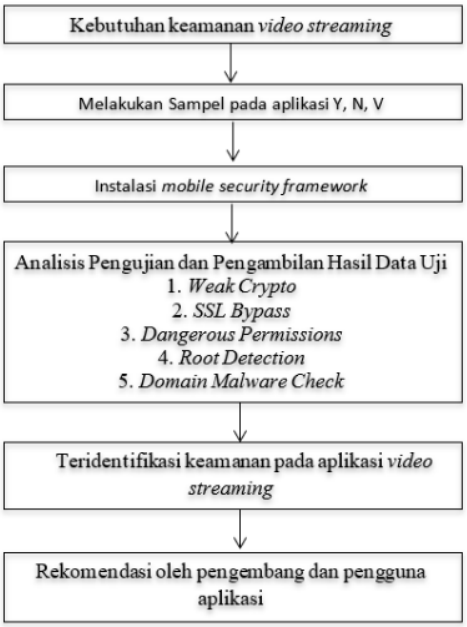
3. METODE PENELITIAN

Metode penelitian yang digunakan dalam melaksanakan penelitian ini adalah dengan menggunakan metode kuantitatif dengan uji analisis *video streaming* menggunakan *Mobile Security Framework* (MobSF) dan obyek penelitian yaitu keamanan aplikasi *video streaming* di Indonesia yaitu Y, N dan V, dengan tujuan penelitian ini mendukung sistem keamanan pada situs *video streaming* tersebut. Dari 3 situs *video streaming*, kriteria yang akan dijadikan sebagai parameter penelitian adalah *dangerous permissions, weak crypto, root detection, SSL bypass dan domain malware check*.

3.A. Diagram Alir Penelitian

Diagram alir penelitian pada ini dengan mengidentifikasi suatu permasalahan atau fenomena yang dianggap perlu untuk diteliti, yaitu penelitian tentang analisis statik keamanan aplikasi *video*

streaming berbasis android menggunakan *mobile security framework*. Permasalahan terjadi pada banyaknya kasus pencurian data. Pada masa pandemik tahun 2020 sampai 2021 banyak masyarakat Indonesia yang menggunakan aplikasi *video streaming* berbasis android. Terdapat 3 aplikasi *video streaming* berbasis android yang memiliki pengguna banyak yaitu Y, N dan V.



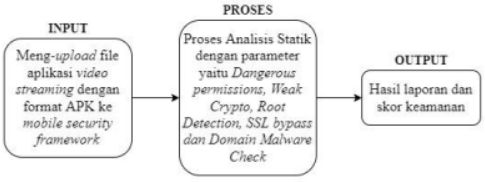
Gambar 4. Diagram Alir Penelitian

Peneliti menggunakan *mobile security framework* untuk mengetahui celah keamanan yang terdapat pada aplikasi *video streaming* berbasis android, kemudian menganalisis statik dan merekomendasikan untuk celah keamanan yang ditemukan sehingga masyarakat dapat mengetahui tingkat keamanan dari aplikasi *video streaming* berbasis android yang telah diteliti.

3.B. Melakukan Pengujian dan Pengambilan Hasil Data Uji

Tahap ini untuk melakukan pengujian, kemudian melihat hasil pengujian menggunakan *mobile security framework* terhadap aplikasi *video streaming* berbasis android dengan format APK.

1. Cara Pengujian
   Gambar 5. menunjukkan blok diagram analisis statik menggunakan *Mobile Security Framework* (MobSF).



Gambar 5. Blok Diagram Analisis Statik Menggunakan *Mobile Security Framework* (MobSF)

Input: meng-*upload* file aplikasi *video streaming* dengan format APK ke *mobile security framework*, tunggu hingga proses *upload* selesai.
Proses: *mobile security framework* akan melakukan proses analisis statik dengan parameter yaitu *dangerous permissions, weak crypto, root detection, ssl bypass dan domain malware check*.
Output: saat proses telah selesai, maka akan muncul hasil pengujian file tersebut yang berupa laporan dan skor keamanan (Kartono, Sularsa dan Ismail, 2019).

2. Parameter yang Diuji
   Dangerous permissions, *weak crypto*, *root* detection, SSL *bypass* dan *domain malware check*.
3. Data yang Diambil
   Tingkat dan ceklah keamanan dari 3 aplikasi *video streaming* berbasis android yaitu Y, N dan V

3.C. Hasil Analisis dan Pembahasan

Tahap ini untuk membahas hasil pengujian keamanan dari 3 aplikasi *video streaming* yang dilakukan menggunakan *mobile security framework*. Peneliti akan menganalisis celah keamanan yang ditemukan pada aplikasi *video streaming* dari hasil pengujian sebelumnya. Tahap terakhir adalah berisi kesimpulan dari penelitian yang dilakukan sesuai dengan ruang lingkup pengujian. Bagian ini juga berisi saran untuk penelitian kedepannya.

4. HASIL DAN PEMBAHASAN

4.A. Hasil Analisis

File aplikasi *video streaming* dengan format APK diunduh terlebih dahulu sebagai tahap awal. Analisis dilakukan dengan cara meng-*upload* file aplikasi *video streaming* dengan format APK ke *mobile security framework*. Saat proses telah selesai, maka akan muncul hasil analisis file tersebut.

Pada gambar 6. menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF).

Gambar 6. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF)

Pada aplikasi Y bagian *app score* terdapat:

1. *The Common* Vulnerability *Scoring System* (CVSS) yang memberikan representasi numerik 0 sampai 10 dari tingkat keparahan kerentanan keamanan informasi, aplikasi Y mendapat skor 6.6 yang artinya memiliki peringkat kualitatif sedang,
2. Security score mendapatkan skor 10/100 yang artinya memiliki risiko yang kritis.
3. Trackers detection mendapatkan skor 2/407 yang artinya memiliki 2 pelacak.

Pada file information terdeteksi: Nama file yaitu com.google.android.Y-15.22.33.apk dengan ukuran 15.53 MB. Pada *app information* terdeteksi:

1. Nama app yaitu Y dengan package name yaitu com.google .android.Y.
2. Aktifitas utama pada com.google.android .apps.Y.app.application.Shell$SettingsActivity.
3. Software Development Kit (SDK) adalah sebuah koleksi dari alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal Target SDK adalah versi yang ditargetkan untuk dijalankan oleh aplikasi, tercantum 29, min SDK adalah versi minimal yang dapat dijalankan oleh aplikasi, tercantum 21 dan tidak ada max SDK.
4. Nama versi android 15.22.33 dan kode versi android 1512951232.

Pada *playstore information* terdeteksi:

1. Judul yaitu Y dengan skor 4.278066 yaitu rating pada playstore.
2. Diinstal sebanyak 10,000,000,000+ kali.
3. Dengan harga aplikasi 0.
4. Dukungan versi android bervariasi menurut perangkat.
5. Kategori aplikasi yaitu pemutar video & editor.
6. URL play store yaitu com.google.android.Y.
7. Pengembang aplikasi yaitu Google LLC.
8. ID pengembang yaitu 5700313618786177705
9. Alamat pengembang yaitu 1600 Amphitheatre Parkway, Mountain View 94043.
10. Website pengembang yaitu https://support.google.com/Y/topic/2422554?rd=1
11. Email pengembang yaitu ytandroid-support@google.com.
12. Tanggal rilis aplikasi yaitu 20 Oktober 2010.
13. Privacy policy yaitu link privasi.

Pada gambar 7. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF).

Gambar 7. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF)

Pada aplikasi N bagian *app score* terdapat:

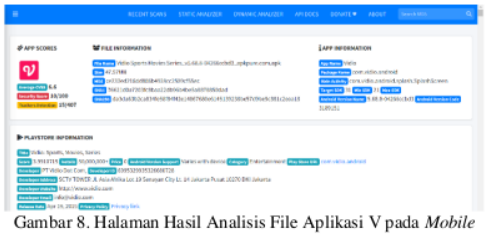
1. The Common Vulnerability Scoring System (CVSS) yang memberikan representasi numerik 0 sampai 10 dari tingkat keparahan kerentanan keamanan informasi, aplikasi N mendapatkan skor 6.2 yang artinya memiliki peringkat kualitatif sedang.
2. Security score mendapatkan skor 35/100 yang artinya memiliki risiko keamanan yang tinggi.
3. Trackers detection mendapatkan skor 2/407 yang artinya memiliki 2 pelacak.

Pada file information terdeteksi, nama file yaitu: com.N.ninja_8.3.1_build_45464546_minAPI22(armeabi-v7a)(nodpi)_apkmirror.com.apk. Dengan ukuran file sebesar 79.37 MB. Pada *app information* terdeteksi:

1. Nama app yaitu N dengan package name yaitu com.N.ninja.
2. Aktifitas utama pada .MainActivity.
3. Software Development Kit (SDK) adalah sebuah koleksi dari alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal. Target SDK adalah versi yang ditargetkan untuk dijalankan oleh aplikasi, tercantum 31, min SDK adalah versi minimal yang dapat dijalankan oleh aplikasi, tercantum 22 dan tidak ada max SDK.
4. Nama versi android 8.3.1 build 4546 dan kode versi android 4546.

Pada *playstore information* terdeteksi:

1. Judul yaitu N dengan skor 2.28 yaitu rating pada playstore.
2. Diinstal sebanyak 100,000,000+ kali.
3. Dengan harga aplikasi 0.
4. Dukungan versi android bervariasi menurut perangkat.
5. Kategori aplikasi yaitu hiburan.
6. URL play store yaitu com.N.ninja.
7. Pengembang aplikasi yaitu N, Inc.
8. ID pengembang yaitu N,+I20
9. Alamat pengembang yaitu 100 Winchester Circle Los Gatos, CA 95032-1815 USA.
10. Website pengembang yaitu http://www.N.com.
11. Email pengembang yaitu playstore@N.com.
12. Tanggal rilis aplikasi yaitu 27 Juni 2014.
13. Privacy policy yaitu link privasi.
14. Pada gambar 8. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF).



Gambar 8. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF)

Pada aplikasi V bagian *app score* terdapat:

1. The Common Vulnerability Scoring System (CVSS) yang memberikan representasi numerik 0

sampai 10 dari tingkat keparahan kerentanan keamanan informasi, aplikasi V mendapatkan skor 6.6 yang artinya memiliki peringkat kualitatif sedang.
2. Security score mendapatkan skor 10/100 yang artinya memiliki risiko keamanan yang kritis.
3. Trackers detection mendapatkan skor 15/407 yang artinya memiliki 15 pelacak.

Pada *file information* terdeteksi nama file yaitu V Sports Movies Series_v5.68.8-04266ccbd3_apkpure.com .apk dengan ukuran file sebesar 47.57 MB. Pada *app information* terdeteksi:
1. Nama app yaitu V dengan package name yaitu com.V.android.
2. Aktifitas utama pada com.V.android.splash .SplashScreen.
3. Software Development Kit (SDK) adalah sebuah koleksi dari alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal. Target SDK adalah versi yang ditargetkan untuk dijalankan oleh aplikasi, tercantum 31, min SDK adalah versi minimal yang dapat dijalankan oleh aplikasi, tercantum 21 dan tidak ada max SDK.
4. Nama versi android 5.68.8-04266ccbd3 dan kode versi android 3189151.

Pada *playstore information* terdeteksi:
1. Judul yaitu V: Sports, Movies, Series dengan skor 3.9910715 yaitu rating pada playstore.
2. Diinstal sebanyak 50,000,000+ kali.
3. Dengan harga aplikasi 0.
4. Dukungan versi android bervariasi menurut perangkat.
5. Kategori aplikasi yaitu hiburan.
6. URL play store yaitu com.V.android.
7. Pengembang aplikasi yaitu PT V Dot Com.
8. ID pengembang yaitu 699[illegible]935326680728.
9. Alamat pengembang yaitu SCTV TOWER Jl. Asia Afrika Lot 19 Senayan City Lt. 14 Jakarta Pusat 10270 DKI Jakarta.
10. Website pengembang yaitu http://www.V.com.
11. Email pengembang yaitu info@V.com.
12. Tanggal rilis aplikasi yaitu 19 April 2015.
13. Privacy policy yaitu link privasi.

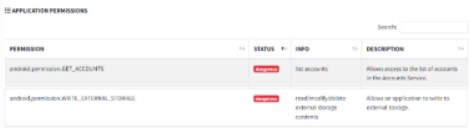
4.A.i. Dangerous Permissions

Pada gambar 9. yang menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian *application permissions*.

Gambar 9. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian *Application Permissions*

Pada aplikasi Y terdapat 6 *dangerous permissions* yaitu:
1. android.permission.CAMERA yang mengizinkan akses perangkat kamera sehingga aplikasi dapat mengambil gambar atau video melalui perangkat kamera kapan saja.
2. android.permission.GET_ACCOUNTS yang mengizinkan akses ke daftar akun di layanan akun sehingga aplikasi dapat memperoleh informasi tentang akun.
3. android.permission.MANAGE_ACCOUNTS yang mengizinkan aplikasi melakukan operasi seperti menambah dan menghapus akun serta menghapus sandinya sehingga aplikasi bisa mengambil alih akun.
4. android.permission.RECORD_AUDIO yang mengizinkan aplikasi mengakses jalur rekaman audio atau mikrofon sehingga aplikasi dapat merekam audio kapan saja.
5. android.permission.WRITE_EXTERNAL_STORAGE yang mengizinkan aplikasi untuk membaca/memodifikasi/menghapus konten penyimpanan eksternal sehingga aplikasi dapat menyalahgunakan konten yang ada pada penyimpanan eksternal.
6. android.permission.USE_CREDENTIALS yang mengizinkan aplikasi meminta token autentikasi atau menggunakan kredensial otentikasi akun.

Pada gambar 10. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian *application permissions*.



Gambar 10. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian *Application Permissions*

Pada aplikasi N terdapat 2 *dangerous permissions* yaitu:
1. android.permission.GET_ACCOUNTS yang mengizinkan akses ke daftar akun di layanan akun sehingga aplikasi dapat memperoleh informasi tentang akun.
2. android.permission.WRITE_EXTERNAL_STORAGE yang mengizinkan aplikasi untuk membaca/memodifikasi/menghapus konten penyimpanan eksternal sehingga aplikasi dapat menyalahgunakan konten yang ada pada penyimpanan eksternal.

Pada gambar 11. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *application permissions*.

Gambar 11. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Application Permissions*

Pada aplikasi V terdapat 4 *dangerous permissions* yaitu:

1. android.permission.RECORD_ AUDIO yang mengizinkan aplikasi mengakses jalur rekaman audio atau mikrofon sehingga aplikasi dapat merekam audio kapan saja.
2. android.permission.CAMERA yang mengizinkan akses perangkat kamera sehingga aplikasi dapat mengambil gambar atau video melalui perangkat kamera kapan saja.
3. android.permission.READ_EXTERNAL_STORAGE yang mengizinkan aplikasi membaca dari penyimpanan eksternal sehingga aplikasi dapat memperoleh informasi tentang konten yang ada pada penyimpanan eksternal.
4. android.permission.WRITE_EXTERNAL_STORAGE yang mengizinkan aplikasi untuk membaca/memodifikasi/menghapus konten penyimpanan eksternal sehingga aplikasi dapat menyalahgunakan konten yang ada pada penyimpanan eksternal.

4.A.ii. Weak Crypto

Pada gambar 12. menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian *code analysis*.

Gambar 12. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada aplikasi Y terdapat 4 *weak crypto* yang terdiri dari 1 *high severity* yaitu aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding*. Serangan *oracle padding* adalah serangan yang menggunakan validasi *padding* dari pesan kriptografi untuk mendekripsi ciphertext (Ichi.pro, 2020). Dengan standar:

1. Common Vulnerability Scoring System (CVSS) adalah suatu nilai pengukuran yang dipakai untuk menilai suatu kerentanan terhadap sistem, pada CVSS V2 terdeteksi 7.4 artinya memiliki peringkat kritatif tinggi.
2. Common Weakness Enumeration (CWE) adalah daftar yang menampilkan keberadaan bug pada software atau hardware yang berbahaya, terdeteksi CWE-649:ketergantungan pada kebingungan atau enkripsi input keamanan yang relevan tanpa pemeriksaan integritas.
3. Open Web Application Security Project (OWASP) Top 10 adalah sebuah panduan bagi para developers dan security team tentang kelemahan-kelemahan pada web apps yang mudah diserang dan harus segera disiasati, terdeteksi M5:kriptografi tidak memadai.
4. OWASP The Mobile Application Security Verification Standard (MASVS) adalah standar untuk keamanan aplikasi seluler, terdeteksi MSTG-CRYPTO-3 artinya aplikasi menggunakan primitif kriptografi yang sesuai untuk kasus penggunaan tertentu, serta dikonfigurasi dengan parameter yang mematuhi praktik terbaik industri.

Lokasi file yaitu defpackage/wpr.java dan defpackage/djm.java. Gambar 13. menunjukkan kode pada defpackage/wpr.java.

```
93.     public final Cipher c() {
94.         try {
95.             Cipher instance = Cipher.getInstance("AES/CBC/PKCS7Padding");
96.             instance.init(1, (SecretKey) this.d.getKey("YouTubeFingerprintKey", null));
97.             return instance;
```

Gambar 13. Kode pada defpackage/wpr.java

Kode tersebut artinya membuat cipherinstance dengan memanggil getInstance() metodenya dengan parameter yang diisi dengan jenis algoritma enkripsi yang digunakan yaitu AES dengan mode operasi CBC dan padding PKCS7. Terlebih dahulu masukkan kelas java yaitu javax.crypto.Cipher untuk membuat cipher. *Advanced Encryption Standard* (AES) merupakan algoritma yang menggunakan kunci yang sama (*private key*) untuk enkripsi dan dekripsi. Mode *Cipher Block Chaining* (CBC) adalah suatu mode dimana blok yang satu dengan blok lain saling terkait (*chained*), enkripsi dan dekripsi suatu blok data selalu melibatkan ciphertext (hasil enkripsi) blok sebelumnya. *Public Key Cryptographic Standard* (PKCS) adalah penambahan data pada awal, tengah, atau akhir sebelum enkripsi, nilai tiap bit yang ditambahkan adalah banyak bit yang ditambahkan. Gambar 14. menunjukkan kode pada defpackage/djm.java.

```
98.     private static final Cipher a() {
99.         Cipher cipher;
100.        synchronized (c) {
101.            if (a == null) {
102.                a = Cipher.getInstance("AES/CBC/PKCS5Padding");
103.            }
104.            cipher = a;
105.        }
106.        return cipher;
107.    }
```

Gambar 14. Kode pada defpackage/djm.java

Kode tersebut artinya membuat variabel a sebagai cipherinstance dengan memanggil getInstance() metodenya dengan parameter yang diisi dengan jenis algoritma enkripsi yang digunakan yaitu

AES dengan mode operasi CBC dan padding PKCS5. Terlebih dahulu masukkan kelas java yaitu javax.crypto.Cipher untuk membuat cipher. PKCS5 identik dengan PKCS7, namun PKCS5 hanya digunakan untuk penyandian blok yang berukuran 64 bit. Keduanya bisa dipakai pada praktiknya.

Pada aplikasi Y terdapat 3 *warning severity* yaitu:

1. Aplikasi menggunakan Generator Angka Acak yang tidak aman. Generator Angka Acak adalah alat atau algoritma yang menghasilkan urutan angka yang secara statistik independen dan tidak dapat ditebak (Autobild, 2021). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 7.5 artinya memiliki peringkat kualitatif tinggi.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-330:penggunaan nilai acak yang tidak memadai dalam konteks keamanan yang bergantung pada angka yang tidak dapat diprediksi. Ketika perangkat lunak menghasilkan nilai yang dapat diprediksi dalam konteks yang membutuhkan ketidakpastian, penyerang dapat menebak nilai berikutnya yang akan dihasilkan, dan menggunakan tebakan ini untuk menyamar sebagai pengguna lain atau mengakses informasi sensitif.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai. The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-6 artinya semua nilai acak dihasilkan menggunakan generator nomor acak yang cukup aman.

   Lokasi file yaitu;
   defpackage/bhiv.java, defpackage/tks.java, defpackage/bhdz.java, defpackage/aebc.java, defpackage/bfuo.java, defpackage/apen.java, defpackage/aagk.java, defpackage/vpz.java, defpackage/dkg.java, defpackage/xrr.java, defpackage/xne.java, defpackage/qaz.java, defpackage/aglo.java, defpackage/afvn.java, defpackage/oul.java, defpackage/anmb.java, defpackage/xrp.java, defpackage/bhku.java, defpackage/bhec.java, defpackage/sot.java, defpackage/aqkh.java, defpackage/qxv.java, defpackage/diy.java, defpackage/ote.java, defpackage/ezs.java, defpackage/ahpb.java, defpackage/bhdw.java, defpackage/gnh.java, defpackage/vjv.java, defpackage/agtd.java, defpackage/bfuh.java, defpackage/abwb.java, defpackage/aeyg.java, defpackage/qbx.java, defpackage/rlx.java, defpackage/alpc.java.

Gambar 15. menunjukkan kode pada defpackage/vjv.java.

```
/* access modifiers changed from: package-private */
public final Long a(long j) {
    int i = 0;
    Random random = new Random((long) Objects.hash(Long.valueOf(j), this.c, this.d));
    double nextDouble = random.nextDouble();
    double d2 = this.c;
    if (d2 >= 1.0d) {
        i = (int) Math.min(Math.round((d2 + d2) * nextDouble), 2147483646L);
    } else if (nextDouble < d2) {
        i = 1;
    }
```

Gambar 15. Kode pada defpackage/vjv.java

Kode tersebut artinya memasukkan kelas java yaitu java.util.Random untuk membuat angka acak pada access modifiers. Access modifiers untuk memberi hak akses kepada user, tentu tidak semua data yang berada di dalam suatu kelas atau turunannya dapat diakses karena terdapat batasan-batasan yang berlaku. Salah satunya akses private yaitu pengaksesan class hanya dapat diakses oleh class dimana tipe ini dibuat. Access modifiers termasuk data penting yang harus dilindungi dengan aman. Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman dapat melemahkan algoritma dan memiliki kemungkinan serangan offline. Serangan offline adalah serangan yang dilakukan secara offline atau tidak membutuhkan koneksi sebagai media.

2. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. MD5 yang digunakan untuk password juga rentan terhadap serangan brute-force dengan waktu yang cepat karena hanya memiliki 128 bit. Serangan brute-force adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

   Lokasi file yaitu:
   defpackage/tks.java, defpackage/dhk.java, defpackage/die.java, defpackage/rne.java, defpackage/wap.java, defpackage/vex.java, defpackage/van.java.

Gambar 16. menunjukkan kode file MD5 pada aplikasi Y.

```
625.            try {
626.                MessageDigest instance = MessageDigest.getInstance("MD5");
627.                if (instance != null) {
628.                    return instance;
629.                }
```

Gambar 16. Kode File MD5 pada Aplikasi Y

Kode MessageDigest.getInstance("MD5") berarti string akan dienkripsi menggunakan MD5. Message digest adalah sebuah nilai yang dikenal juga sebagai kriptografi checksum atau secure hash. Message digest dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi user melalui password. Checksum adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. Secure hash adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi message digest tidak dapat dikembalikan menjadi pesan semula.

3. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. SHA-1 yang digunakan untuk password juga rentan terhadap serangan brute-force dengan waktu yang cepat karena hanya memiliki 160 bit. Serangan brute-force adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 5.9 artinya memiliki peringkat kualitatif sedang.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

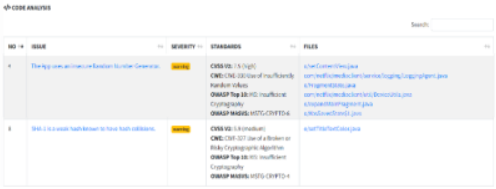
   Lokasi file yaitu;
   defpackage/aqrm.java, defpackage/amgv.java, defpackage/bgqm.java, defpackage/alhh.java, defpackage/aqpf.java, defpackage/bgcq.java, defpackage/aake.java, defpackage/wqp.java, defpackage/ahzp.java.

Gambar 17. menunjukkan kode pada defpackage/wqp.java.

```
27.         try {
28.             MessageDigest instance = MessageDigest.getInstance("SHA-1");
```

Gambar 17. Kode pada defpackage/wqp.java

Kode MessageDigest.getInstance("SHA-1") berarti string akan dienkripsi menggunakan SHA-1. Message digest adalah sebuah nilai yang dikenal juga sebagai kriptografi checksum atau secure hash. Message digest dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi user melalui password. Checksum adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. Secure hash adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi message digest tidak dapat dikembalikan menjadi pesan semula.



Gambar 18. Halaman Hasil Analisis File Aplikasi N pada Mobile Security Framework (MobSF) Bagian Code Analysis

Pada Gambar 18. menunjukkan halaman hasil analisis file aplikasi N pada mobile security framework (MobSF) bagian code analysis.

Pada aplikasi N terdapat 2 weak crypto yang terdiri dari 2 *warning severity* yaitu:

1. Aplikasi menggunakan Generator Angka Acak yang tidak aman. Generator Angka Acak adalah alat atau algoritma yang menghasilkan urutan angka yang secara statistik independen dan tidak dapat ditebak (Autobild, 2021). Dengan standar:
   - Common Vulnerability Scoring System (CVSS) V2 terdeteksi 7.5 artinya memiliki peringkat kualitatif tinggi.
   - Common Weakness Enumeration (CWE) terdeteksi CWE-330:penggunaan nilai acak yang tidak memadai dalam konteks keamanan yang bergantung pada angka yang tidak dapat diprediksi. Ketika perangkat lunak menghasilkan nilai yang dapat diprediksi dalam konteks yang membutuhkan ketidakpastian, penyerang dapat menebak nilai berikutnya yang akan dihasilkan, dan menggunakan tebakan ini untuk menyamar sebagai pengguna lain atau mengakses informasi sensitif.
   - Open Web Application Security Project (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - The Mobile Application Security Verification Standard terdeteksi MSTG-CRYPTO-6 artinya semua nilai acak dihasilkan menggunakan generator nomor acak yang cukup aman.

   Lokasi file yaitu;
   o/setContentView.java, com/N/mediaclient/service/logging/LoggingAgent.java, o/FragmentState.java, com/N/mediaclient/util/DeviceUtils.java, o/expandMainFragment.java, o/AbsSavedState$1.java.

Gambar 19. menunjukkan kode pada com/netflix/mediaclient/service/logging /LoggingAgent.java.

```
16.     static {
17.         long nextLong = new Random().nextLong();
18.         if (nextLong < 0) {
19.             nextLong = 0 - nextLong;
20.         }
21.         gCritSessionId = nextLong;
22.     }
```

Gambar 19. Kode pada com/netflix/mediaclient/service/logging/LoggingAgent.java

Kode tersebut artinya memasukkan kelas java yaitu java.util.Random untuk membuat angka acak untuk *session id*. *Session id* termasuk data penting yang harus dilindungi dengan aman. Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman dapat melemahkan algoritma dan memiliki kemungkinan serangan *offline*. Serangan *offline* adalah serangan yang dilakukan secara *offline* atau tidak membutuhkan koneksi sebagai media.

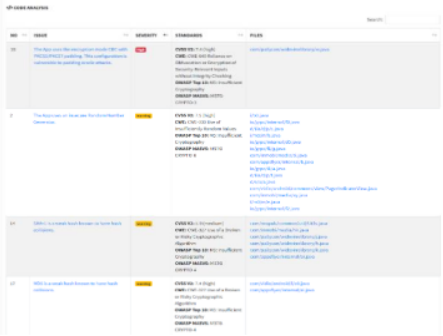
2. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. SHA-1 yang digunakan untuk *password* juga rentan terhadap serangan *brute-force* dengan waktu yang cepat karena hanya memiliki 160 bit. Serangan *brute-force* adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 5.9 artinya memiliki peringkat kualitatif sedang.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

Lokasi file yaitu o/setTitleTextColor.java. Gambar 20. menunjukkan kode pada o/setTitleTextColor.java.

```
94.         try {
95.             Result.Companion companion = Result.Companion;
96.             MessageDigest instance = MessageDigest.getInstance("SHA-1");
97.             StringBuilder sb = new StringBuilder("sha1 ");
```

Gambar 20. Kode pada o/setTitleTextColor.java

Kode MessageDigest.getInstance("SHA-1") berarti string akan dienkripsi menggunakan SHA-1. *Message digest* adalah sebuah nilai yang dikenal juga sebagai kriptografi *checksum* atau *secure hash*. *Message digest* dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi *user* melalui *password*. *Checksum* adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. *Secure hash* adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi *message digest* tidak dapat dikembalikan menjadi pesan semula.



Gambar 21. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada Gambar 21. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *code analysis*.

Pada aplikasi V terdapat 4 *weak crypto* yang terdiri dari 1 *high severity* yaitu aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding*. *Advanced* Encryption *Standard* (AES) merupakan algoritma yang menggunakan kunci yang sama (*private key*) untuk enkripsi dan dekripsi. Mode *Cipher Block Chaining* (CBC) adalah suatu mode dimana blok yang satu dengan blok lain saling terkait (*chained*), enkripsi dan dekripsi suatu blok data selalu melibatkan ciphertext (hasil enkripsi) blok sebelumnya. *Public Key Cryptographic Standard* (PKCS) adalah penambahan data pada awal, tengah, atau akhir sebelum enkripsi, nilai tiap bit yang ditambahkan adalah banyak bit yang ditambahkan. Serangan *oracle padding* adalah serangan yang menggunakan validasi padding dari pesan kriptografi untuk mendekripsi ciphertext (Ichi.pro, 2021). Dengan standar:

1. *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
2. *Common Weakness Enumeration* (CWE) terdeteksi CWE-649:ketergantungan pada kebingungan atau enkripsi input keamanan yang relevan tanpa pemeriksaan integritas.
3. *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
4. OWASP *The Mobile Application Security Verification Standard* (MASVS) terdeteksi MSTG-CRYPTO-3 artinya aplikasi menggunakan primitif kriptografi yang sesuai untuk kasus penggunaan tertentu, serta dikonfigurasi dengan parameter yang mematuhi praktik terbaik industri.

Lokasi file yaitu com/pallycon/widevinelibrary/w.java. Gambar 22. menunjukkan kode pada com/pallycon/widevinelibrary/w.java.

```
30.             try {
31.                 this.a = Cipher.getInstance("AES/CBC/PKCS7Padding");
```

Gambar 22. Kode pada com/pallycon/widevinelibrary/w.java

Kode tersebut artinya membuat cipherinstance dengan memanggil getInstance() metodenya dengan parameter yang diisi dengan jenis algoritma enkripsi yang digunakan yaitu AES dengan mode operasi CBC dan padding PKCS5. Terlebih dahulu masukkan kelas java yaitu javax.crypto.Cipher untuk membuat cipher. PKCS5 identik dengan PKCS7, namun PKCS5 hanya digunakan untuk penyandian blok yang berukuran 64 bit. Keduanya bisa dipakai pada praktiknya.

Pada aplikasi V terdapat 3 *warning severity* yaitu:

1. Aplikasi menggunakan Generator Angka Acak yang tidak aman. Generator Angka Acak adalah alat atau algoritma yang menghasilkan urutan angka yang secara statistik independen dan tidak dapat ditebak (Autobild, 2021). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 7.5 artinya memiliki peringkat kualitatif tinggi.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-330:penggunaan nilai acak yang tidak memadai dalam konteks keamanan yang bergantung pada angka yang tidak dapat diprediksi. Ketika perangkat lunak menghasilkan nilai yang dapat diprediksi dalam konteks yang membutuhkan ketidakpastian, penyerang dapat menebak nilai berikutnya yang akan dihasilkan, dan menggunakan tebakan ini untuk menyamar sebagai pengguna lain atau mengakses informasi sensitif.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai. *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-6 artinya semua nilai acak dihasilkan menggunakan generator nomor acak yang cukup aman.

Lokasi file yaitu;
i/b0.java, io/grpc/internal/f0.java, d/i/a/d/p/c.java, i/m0/m/b.java, io/grpc/internal/d0 .java, io/grpc/f1/g.java, com/inmobi/media/al.java, com/appsflyer/internal/b.java, io/grpc/i1/a.java, d/i/a/ d/p/f.java, d/e/a/a.java, com/V/android/commons/ view/PagerIndicatorView.java, com/inmobi/media/ ay.java, i/m0/m/e.java, io/grpc/internal/f2.java.

Gambar 23. menunjukkan kode pada com/appsflyer/internal/b.

```
try {
    Random random2 = new Random();
    short s8 = (short) 951;
    try {
        byte[] bArr11 = AppsFlyerConversionListener;
        zArr5 = zArr2;
        try {
            try {
                random2.setSeed(((Long) Class.forName($$c(s8, (byte) bArr11[72],
                Object obj14 = null;
                Object obj15 = null;
                Object obj16 = null;
                Object obj17 = null;
                while (obj14 == null) {
                    if (obj15 == null) {
                        int i37 = AppsFlyerInAppPurchaseValidatorListener;
                        int i38 = (i37 & 123) + (i37 | 123);
                        obj12 = obj14;
                        getSdkVersion = i38 % 128;
                        i10 = i38 % 2 == 0 ? 101 : 6;
```

Gambar 23. Kode pada com/appsflyer/internal/b

Kode pada gambar 23. artinya memasukkan kelas java yaitu java.util.Random untuk membuat angka acak, implementasi antarmuka untuk menangani keberhasilan dan kegagalan validasi pembelian adalah arti kode dari AppsFlyerInAppPurchaseValidatorListener. Validasi pembelian termasuk data penting yang harus dilindungi dengan aman. Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman dapat melemahkan algoritma dan memiliki kemungkinan serangan *offline*. Serangan *offline* adalah serangan yang dilakukan secara *offline* atau tidak membutuhkan koneksi sebagai media.

2. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. SHA-1 yang digunakan untuk *password* juga rentan terhadap serangan *brute-force* dengan waktu yang cepat karena hanya memiliki 160 bit. Serangan *brute-force* adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 5.9 artinya memiliki peringkat kualitatif sedang.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

Lokasi file yaitu;
com/mopub/common/util/Utils.java, com/inmobi/media/hk.java, com/pallycon/widevine library/j.java, com/pallycon/widevinelibrary/h.java, com/pallycon/widevinelibrary/k.java, com/appsflyer/ internal/ai.java.

Gambar 24. menunjukkan kode pada com/appsflyer/internal/ai.java.

```
38.     try {
39.         MessageDigest instance = MessageDigest.getInstance("SHA-1");
```

Gambar 24. Kode pada com/appsflyer/internal/ai.java

Kode MessageDigest.getInstance("SHA-1") berarti string akan dienkripsi menggunakan SHA-1. *Message digest* adalah sebuah nilai yang dikenal juga sebagai kriptografi *checksum* atau *secure hash*. *Message digest* dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi *user* melalui *password*. *Checksum* adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. *Secure hash* adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang

sudah diubah menjadi *message digest* tidak dapat dikembalikan menjadi pesan semula.

3. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash. Tabrakan hash adalah ada 2 atau lebih teks yang menghasilkan nilai hash yang sama. MD5 yang digunakan untuk *password* juga rentan terhadap serangan *brute-force* dengan waktu yang cepat karena hanya memiliki 128 bit. Serangan *brute-force* adalah serangan dengan mencoba segala kombinasi huruf, angka dan simbol agar didapatkan plaintext dari suatu hash (Kurniawan, Kusyanti dan Nurwarsito, 2017). Dengan standar:
   - *Common Vulnerability Scoring System* (CVSS) V2 terdeteksi 7.4 artinya memiliki peringkat kualitatif tinggi.
   - *Common Weakness Enumeration* (CWE) terdeteksi CWE-327:penggunaan algoritma kriptografi rusak atau berisiko.
   - *Open Web Application Security Project* (OWASP) Top 10 terdeteksi M5:kriptografi tidak memadai.
   - *The Mobile Application Security Verification Standard* terdeteksi MSTG-CRYPTO-4 artinya aplikasi tidak menggunakan protokol atau algoritme kriptografi yang secara luas dianggap tidak digunakan lagi untuk tujuan keamanan.

Lokasi file yaitu *com/V/android/l/a0.java dan com/appsflyer/internal/ai.java*. Gambar 25. menunjukkan kode file MD5 pada aplikasi V.

```
82.     public final String f() {
83.         List J = p.J(this.a, this.f23673b);
84.         MessageDigest instance = MessageDigest.getInstance("MD5");
```

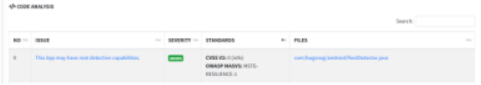
Gambar 25. Kode File MD5 pada Aplikasi V

Kode MessageDigest.getInstance("MD5") berarti string akan dienkripsi menggunakan MD5. *Message digest* adalah sebuah nilai yang dikenal juga sebagai kriptografi *checksum* atau *secure hash*. *Message digest* dimaksudkan untuk meningkatkan keamanan dalam transformasi data, kelas yang sering digunakan dalam aplikasi yang membutuhkan autentikasi *user* melalui *password*. *Checksum* adalah urutan angka dan huruf yang digunakan untuk memeriksa kesalahan data. *Secure hash* adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi *message digest* tidak dapat dikembalikan menjadi pesan semula.

4.A.iii. Root Detection

Root detection adalah deteksi akses root terhadap perangkat android yang digunakan (Alviansyah dan Ramadhani, 2021). Pada *handphone* yang sudah di-*root* membuat *handphone* rentan terhadap serangan *spyware* atau virus dan memungkinkan pengguna untuk memodifikasi aplikasi sehingga menyebabkan aplikasi rusak atau rentan terhadap kecurangan.

Pada aplikasi Y tidak memiliki kemampuan *root detection* karena pada bagian *code analysis* tidak terdapat file atau kode tentang *root detection*. Pada *handphone* yang sudah di-*root*, memungkinkan pengguna untuk memodifikasi aplikasi sehingga menyebabkan aplikasi rusak atau rentan terhadap kecurangan. Saat peneliti menggunakan *handphone* yang sudah di-*root*, walaupun tidak menggunakan akun premium pada aplikasi Y tetapi dapat menggunakan fitur premium.



Gambar 26. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada gambar 26. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian *code analysis*.

Pada gambar 27. menunjukkan kode pada file com/bugsnag/android/RootDetector.java.

Gambar 27. Kode pada com/bugsnag/android/RootDetector.java

Pada aplikasi N memiliki kemampuan *root detection*. Kode pada gambar 27. memiliki fungsi untuk menemukan file *Super User* (SU). *Super user* adalah pengguna (*user*) yang memiliki hak penuh untuk mengelola sistem. Saat peneliti menggunakan *handphone* yang sudah di-*root*, aplikasi N tidak dapat ditemukan di *playsore* yang berarti aplikasi N tidak dapat diinstal pada *handphone* yang sudah di-*root*. Hal ini bagus untuk keamanan aplikasi N karena membuat pengguna tidak dapat memodifikasi aplikasi melalui *handphone* yang sudah di-*root*. Sehingga kemungkinan aplikasi rusak atau rentan terhadap kecurangan dapat dicegah.

Pada gambar 28. yang menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *code analysis*.

Gambar 28. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Code Analysis*

Pada gambar 29. menunjukkan kode pada file *d/h/a/b.java*.

```
if (!z2) {
    String str7 = Build.TAGS;
    if (!(str7 != null && str7.contains("test-keys"))) {
        try {
            process = Runtime.getRuntime().exec(new String[]{"which", "su"});
```

Gambar 29. Kode pada d/h/a/b.java

Pada aplikasi V memiliki kemampuan *root detection*. Kode pada gambar 29. memiliki fungsi untuk menemukan *test-keys* yang terdapat string SU. *Test-keys* berarti file APK ditandatangani dengan kunci khusus yang dibuat oleh pengembang pihak ketiga. *Super user* adalah pengguna (*user*) yang memiliki hak penuh untuk mengelola sistem.

4.A.iv. SSL Bypass

Pada gambar 30. menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian URLs.

Gambar 30. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian URLs

Pada aplikasi Y terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). HTTPS adalah protokol jaringan dengan SSL. *Secure Socket Layers* (SSL) adalah lapisan keamanan tambahan untuk melindungi komunikasi data antara *client* dan *server*. Pada gambar 4.25 terdapat url http://youtube.com/streaming/metadata/segment/102 dan http://youtube.com/streaming/metadata/segment/102015 yang menggunakan protokol jaringan HTTP. Seharusnya metadata menggunakan HTTPS yang terdapat SSL, karena untuk *video streaming* metadata video digunakan untuk memberikan informasi tentang aliran atau file video dan audio tertentu, juga dikenal sebagai esensi. Metadata dapat disematkan langsung ke dalam video atau dimasukkan sebagai file terpisah dalam wadah seperti MP4 atau MKV. Metadata memerlukan perlindungan dari SSL agar pihak ketiga tidak dapat mengaksesnya.

Pada gambar 31. menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian URLs.

Gambar 31. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian URLs

Pada gambar 32. menunjukkan kode pada o/getViewLifecycleOwner.java.

Gambar 32. Kode pada o/getViewLifecycleOwner.java

Pada aplikasi N terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). HTTPS adalah protokol jaringan dengan SSL. *Secure Socket Layers* (SSL) adalah lapisan keamanan tambahan untuk melindungi komunikasi data antara *client* dan *server*. Pada gambar 31. terdapat url http://www.netflix.com/title/ yang menggunakan protokol jaringan HTTP. Seharusnya url tersebut menggunakan HTTPS yang terdapat SSL, karena pada o/getViewLifecycleOwner.java terdapat kode seperti pada gambar 32. yang berarti membuat tampilan pada *signup* dan *profilechange*. onCreateView() artinya membuat dan mengembalikan hierarki tampilan yang terkait dengan fragmen. *Signup* dan *profilechange* berisi data pribadi pengguna sehingga memerlukan perlindungan dari SSL agar pihak ketiga tidak dapat mengaksesnya.

Gambar 33. Halaman Hasil Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian URLs

Pada gambar 33. menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian URLs.

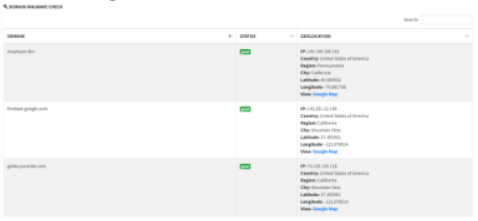
Pada gambar 34. menunjukkan kode pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java.

Gambar 34. Kode pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java

Pada aplikasi V terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). HTTPS adalah protokol jaringan dengan SSL. *Secure Socket Layers* (SSL) adalah lapisan keamanan tambahan untuk melindungi komunikasi data antara *client* dan *server*. Pada gambar 33. terdapat url http://omid-android-reference-app/sendMessage?msg= dan https://s3-us-west-2.amazonaws.com/omsdk-files/compliance-js/omid-validation-verification-script-v1.js yang menggunakan protokol jaringan HTTP. Seharusnya url tersebut menggunakan HTTPS yang terdapat SSL, karena pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java terdapat kode seperti pada gambar 34. yang berarti memverifikasi vendor (Garner, 2017) sehingga memerlukan perlindungan dari SSL agar pihak ketiga tidak dapat mengaksesnya.

4.A.v. Domain Malware Check

Pada aplikasi Y tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware* seperti pada Gambar 35. yang menunjukkan halaman hasil analisis file aplikasi Y pada *mobile security framework* (MobSF) bagian *domain malware check*.



Gambar 35. Halaman Hasil Analisis File Aplikasi Y pada *Mobile Security Framework* (MobSF) Bagian *Domain Malware Check*

Gambar 36. Halaman Hasil Analisis File Aplikasi N pada *Mobile Security Framework* (MobSF) Bagian *Domain Malware Check*

Pada aplikasi N tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware* seperti pada Gambar 36. yang menunjukkan halaman hasil analisis file aplikasi N pada *mobile security framework* (MobSF) bagian *domain malware check*.

Pada aplikasi V tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware* seperti pada Gambar 37. yang menunjukkan halaman hasil analisis file aplikasi V pada *mobile security framework* (MobSF) bagian *domain malware check*.

Berdasarkan analsis statis yang dilakukan didapatkan hasil seperti yang ditampilkan dalam Tabel 4. yang menunjukkan hasil analisis statik.

Gambar 37 Analisis File Aplikasi V pada *Mobile Security Framework* (MobSF) Bagian *Domain Malware Check*

Tabel 4. Hasil Analisis Statik

Apk	*Dangerous Permissions*	*Weak Crypto*	*Root Detection*	SSL *Bypass*	*Domain Malware Check*	*Security Score*
Y	*Yes*	*Yes*	*No*	*Yes*	*Good*	10
N	*Yes*	*Yes*	*Yes*	*Yes*	*Good*	35
V	*Yes*	*Yes*	*Yes*	*Yes*	*Good*	10

Pada Tabel 4. dapat disimpulkan bahwa:
Aplikasi Y memiliki *dangerous permissions* yang meliputi:
1. *android.permission.CAMERA*
2. *android.permission.GET_ACCOUNTS*
3. *android.permission.MANAGE_ACCOUNTS*
4. *android.permission.RECORD_AUDIO*
5. *android.permission.WRITE_EXTERNAL_STORAGE*
6. *android.permission.USE_CREDENTIALS*

Weak crypto yang meliputi:
1. Aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding* yang termasuk *high severity*.
2. Aplikasi menggunakan Generator Angka Acak yang tidak aman yang termasuk *warning severity*.
3. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.
4. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.

Pada SSL *Bypass* menggunakan protokol jaringan HTTP pada url http://youtube.com/streaming/metadata/segment/102015 dan http://youtube.com/streaming/metadata/segment/102015 yang berisi metadata yang seharusnya memerlukan perlindungan SSL.

Aplikasi Y tidak memiliki kemampuan *root detection* karena pada bagian *code analysis* tidak terdapat file atau kode tentang *root detection*.

Pada *domain malware check* tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware*.

Mobile Security Framework (MobSF) menampilkan *security score* sebesar 10 yang artinya memiliki risiko keamanan yang kritis.

Apikasi N memiliki *dangerous permissions* yang meliputi:
1. *android.permission.GET_ACCOUNTS*
2. *android.permission.WRITE_EXTERNAL_STORAGE*

Weak crypto yang meliputi:
1. Aplikasi menggunakan Generator Angka Acak yang tidak aman yang termasuk *warning severity*.
2. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.

Aplikasi N memiliki kemampuan *root detection*, karena ditemukan kode untuk *root detection* pada com/bugsnag/android/RootDetector.java.

Pada SSL *Bypass* menggunakan protokol jaringan HTTP pada url http://www.netflix.com/title yang pada o/getViewLifecycleOwner.java berisi kode yang berarti membuat tampilan pada *signup* dan *profilechange* yang seharusnya memerlukan perlindungan SSL.

Pada *domain malware check* tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware*.

Mobile Security Framework (MobSF) menampilkan *security score* sebesar 35 yang artinya memiliki risiko keamanan yang tinggi.

Aplikasi V memiliki *dangerous permissions*,yang meliputi:

1. *android.permission.RECORD_ AUDIO*
2. *android.permission.CAMERA*
3. *android.permission.READ_EXTERNAL_STORAGE*
4. *android.permission.WRITE_EXTERNAL_STORAGE*

Weak crypto yang meliputi:

1. Aplikasi menggunakan mode enkripsi CBC dengan *padding* PKCS5/PKCS7 yang rentan terhadap serangan *oracle padding* yang termasuk *high severity*.
2. Aplikasi menggunakan Generator Angka Acak yang tidak aman yang termasuk *warning severity*.
3. SHA-1 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.
4. MD5 adalah hash lemah yang diketahui memiliki tabrakan hash yang termasuk *warning severity*.

Aplikasi V memiliki kemampuan *root detection*, karena ditemukan kode untuk *root detection* pada d/h/a/b.java.

Pada SSL *Bypass* menggunakan protokol jaringan HTTP pada url http://o[redacted]-android-reference-app/sendMessage?msg= dan https://s3-us-west-2.amazonaws.com/omsdk-files/compliance-js/omid-validation-verification-script-v1.js yang pada com/kmklabs/videoplayer2/internal/iab/IABHandler.java berisi kode yang berarti memverifikasi vendor yang seharusnya memerlukan perlindungan SSL.

Pada *domain malware check* tidak terdapat *domain malware* karena semua domain berstatus *good* yang berarti domain tidak terindikasi *malware*.

Mobile Security Framework (MobSF) menampilkan *security score* sebesar 10 yang artinya memiliki risiko keamanan yang kritis.

4.B. Rekomendasi

Berdasarkan hasil penelitian, peneliti memberikan saran yang diajukan sesuai dengan masalah pada latar belakang penelitian. Asumsi dari hasil penelitian ini dapat dijadikan sebagai bahan pertimbangan oleh pengembang dan pengguna aplikasi video *streaming* yaitu Y, N, dan V untuk dilanjutkan dalam penelitian terkait analisis statik pada *mobile security framework* (MobSF).

4.B.i. Dangerous Permissions

Sejak android versi 11 (API level 30), setiap kali aplikasi meminta izin untuk mikrofon atau kamera, dialog izin pengguna akan menyertakan opsi yaitu hanya kali ini (Developer.android.com, 2020a). *Application Programming Interface* (API) Level adalah nilai integer yang secara unik mengidentifikasi revisi API framework yang ditawarkan oleh versi platform Android. Peneliti merekomendasikan bagi pengguna aplikasi untuk memilih opsi hanya kali ini atau agar lebih terjamin keamanannya, pengguna bisa menutup kamera dengan stiker atau benda lainnya agar terhindar dari *spy* atau kamera tidak bisa mengambil foto atau video tanpa pengguna ketahui.

Peneliti merekomendasikan bagi pengguna aplikasi untuk izin akses penyimpanan dan daftar akun, apabila pengguna merasa izin ini tidak begitu diperlukan, sebaiknya pengguna menonaktifkan izin tersebut agar kemungkinan data pada penyimpanan dan data pribadi lainnya disalahgunakan tidak terjadi.

4.B.ii. Weak Crypto

Menggunaan kunci yang tidak dibuat dengan generator angka acak yang aman untuk data penting dapat melemahkan algoritma dan memiliki kemungkinan serangan *offline*. Peneliti merekomendasikan bagi pengembang aplikasi untuk menggunakan generator angka acak yang aman (*SecureRandom*) berfungsi untuk menginisialisasi kunci kriptografis yang diciptakan oleh *KeyGenerator* (Developer.android.com, 2020b).

Untuk memperkuat enkripsi data bagian *password*, peneliti merekomendasikan bagi pengembang aplikasi agar algoritma MD5 bisa dikombinasikan dengan algoritma kriptografi yang lain seperti vigenere cipher. Vigenere cipher adalah sebuah enkripsi dengan melakukan beberapa pergeseran yang direpresentasikan menggunakan satu kata kunci. Kombinasi algoritma MD5 dan vigenere cipher ini bisa menjadi lebih kuat jika dibandingkan hanya menggunakan MD5 saja karena kombinasi algoritma ini akan melakukan proses enkripsi dua kali (Sibyan, 2017). Untuk SHA-1 bisa digantikan dengan SHA-3 yang memiliki ketahanan terhadap serangan *brute-force* lebih baik karena waktu yang ditempuh lebih lama untuk mendapatkan plaintext 8, 9 dan 10 karakter dari hash SHA-3 (Kurniawan, Kusyanti dan Nurwarsito, 2017).

4.B.iii. Root Detection

Pada *handphone* yang sudah di-*root* membuat *handphone* rentan terhadap serangan *spyware* atau virus (Nguyen-Vu et al., 2017) dan memungkinkan pengguna untuk memodifikasi aplikasi sehingga menyebabkan aplikasi rusak atau rentan terhadap kecurangan (Sulistio, 2021). Karena hal tersebut, peneliti merekomendasikan bagi pengembang aplikasi Y untuk menambahkan kemampuan *root detection*. Peneliti merekomendasikan untuk pengguna aplikasi *video streaming* agar tidak melakukan *root* terhadap *handphone* sehingga terhindar dari serangan *spyware* atau virus.

4.B.iv. SSL Bypass

Pada aplikasi Y, N dan V terdapat url dengan protokol jaringan *Hypertext Transfer Protocol* (HTTP) dan *Hypertext Transfer Protocol Secure* (HTTPS). Peneliti merekomendasikan agar pengembang aplikasi Y, N dan V tidak menggunakan protokol jaringan *Hypertext Transfer Protocol* (HTTP) untuk data yang bersifat pribadi seperti video yang dikhususkan untuk pelanggan berbayar atau premium, data saat *login*, dan data profil pengguna

sebaiknya menggunakan *Hypertext Transfer Protocol Secure* (HTTPS) karena terdapat *Secure Socket Layer* (SSL) yang terenkripsi dengan baik sehingga pihak ketiga sulit untuk mengaksesnya (Prayama, Yuhefizar dan Amelia Yolanda, 2021).

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa cara melakukan analisis statik menggunakan *mobile security framework* (MobSF) adalah dengan cara meng-upload file aplikasi *video streaming* dengan format APK ke *mobile security framework*. Saat proses telah selesai, maka akan muncul hasil analisis file tersebut. Tingkat keamanan pada aplikasi *video streaming* berbasis android pada aplikasi Y mendapatkan skor keamanan sebesar 10 yang artinya memiliki risiko keamanan yang kritis, aplikasi N mendapatkan skor keamanan sebesar 35 yang artinya memiliki risiko keamanan yang tinggi, aplikasi V mendapatkan skor keamanan sebesar 10 yang artinya memiliki risiko keamanan yang kritis. 3. Terdapat celah keamanan pada aplikasi video streaming berbasis android yaitu pada aplikasi Y, N dan V memiliki *dangerous permissions*, *weak crypto*, dan SSL *bypass*. Hanya Aplikasi Y yang tidak memiliki *root detection*.

Berdasarkan hasil penelitian dan kesimpulan, berikut saran dari penulis untuk penelitian selanjutnya yaitu menganalisis aplikasi *video streaming* lebih banyak lagi, menambahkan tahapan analisis statik pada aplikasi *video streaming*, menggunakan metode analisis lainnya seperti analisis dinamis.

DAFTAR PUSTAKA

Zen, B. P., Gultom, R. A., & Reksoprodjo, A. H. (2020). Analisis Security Assessment Menggunakan Metode Penetration Testing dalam Menjaga Kapabilitas Keamanan Teknologi Informasi Pertahanan Negara. Teknologi Penginderaan, 2(1).

Abraham, A., 2021. *Mobile Security Framework MobSF*. [daring] https://github.com. Tersedia pada: <https://github.com/MobSF/Mobile-Security-Framework-MobSF> [Diakses 30 Mar 2021].

Alviansyah, F.A. dan Ramadhani, E., 2021. Implementasi Dynamic Application Security Testing pada Aplikasi Berbasis Android. *Automata*, [daring] 2(1), hal.1–6. Tersedia pada: <https://journal.uii.ac.id/AUTOMATA/article/view/17387>.

Anwar, N., Akbar, S.A., Elektro, T. dan Dahlan, U.A., 2020. Ekstraksi Logis Forensik Mobile Pada Aplikasi E-Commerce Android. 2(1), hal.1–10.

Arsam, A., 2017. Jurnal Ilmiah Komputer dan Informatika (KOMPUTA) Pembangunan Aplikasi Video Streaming Berbasis Android di STV Bandung. *Jurnal Ilmiah Komputer dan Informatika (KOMPUTA)*.

Asosiasi Penyedia Jasa Internet Indonesia (APJII), 2020. *LAPORAN SURVEI INTERNET APJII 2019 – 2020 (Q2)*.

Asrianti, S., 2020. *April 2020, Pengguna Aktif Aplikasi Vidio Capai 62 Juta*. [daring] Tersedia pada: <https://www.republika.co.id/berita/qaefoi328/april-2020-pengguna-aktif-aplikasi-vidio-capai-62-juta>.

Autobild, 2021. *Acak Angka Online (Random Number Generator)*. [daring] Autobild. Tersedia pada: <https://www.autobild.co.id/p/acak-angka-online-random-number.html>.

Developer.android.com, 2020a. *Pembaruan izin di Android 11*. [daring] developer.android.com. Tersedia pada: <https://developer.android.com/about/versions/11/privacy/permissions?hl=id>.

Developer.android.com, 2020b. *Tips keamanan*. [daring] developer.android.com. Tersedia pada: <https://developer.android.com/training/articles/security-tips?hl=id>.

Dify Virginia Rizaldy, 2021. VIDEO ON DEMAND : CARA MUDAH MENONTON FILM (Studies on Consumer Behavior). *STIE PGRI Dewantara Jombang*.

Dvorak, M., 2021. *Video Streaming Site Data Breach Exposes 2M Customers*. [daring] Tersedia pada: <www.askcybersecurity.com/adult-video-streaming-data-breach-2m-customers>.

Dynatrace, 2021. *Improve DevSecOps Processes - Download Free Research*. [daring] Tersedia pada: <https://www.dynatrace.com/monitoring/solutions/devsecops-ciso/?utm_source=google&utm_medium=cpc&utm_term=devsecops&utm_campaign=id-application-security&utm_content=none&gclid=CjwKCAiAsNKQBhAPEiwAB-I5zXzMTYBTjZyM4VaxuqE9_qh1NC1ND26rAjBftfXyIo3lj7DbJk9WRxoC>.

Garner, N., 2017. *Class VerificationScriptResource*. [daring] docs.iabtechlab.com. Tersedia pada: <https://docs.iabtechlab.com/omsdk-1.3/android/com/iab/omid/library/adsession/VerificationScriptResource.html>.

Hanifurohman, C. dan Hutagalung, D.D., 2020. Analisis Statis Menggunakan Mobile Security Framework Untuk Pengujian Keamanan Aplikasi Mobile E-Commerce Berbasis Android. *Sebatik*, 24(1), hal.22–28.

Ichi.pro, 2021. *Kriptanalisis AES-CBC (Bagian-1)*. [daring] ichi.pro. Tersedia pada: <https://ichi.pro/id/kriptanalisis-aes-cbc-

bagian-1-34012510098655>.

Idan, 2016. *Droidmon*. [daring] Tersedia pada: <https://github.com/idanr1986/droidmon>.

ISOCENTER INDONESIA, 2022. *ISO 27001 Information Security*. [daring] Tersedia pada: <https://isoindonesiacenter.com/iso-27001-information-security/>.

Kartono, A., Sularsa, A. dan Ismail, S.J.I., 2019. Membangun Sistem Pengujian Keamanan Aplikasi Android Menggunakan Mobsf. 5(1), hal.146–151.

Kurniawan, F., Kusyanti, A. dan Nurwarsito, H., 2017. Analisis dan Implementasi Algoritma SHA-1 dan SHA-3 pada Sistem Autentikasi Garuda Training Cost. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, [daring] 1(9), hal.803–812. Tersedia pada: <http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/247>.

NDK, F., 2020. *Agile vs DevOps vs CI/CD : Apa Saja Perbedaanya?* [daring] PT. Logique Digital Indonesia. Tersedia pada: <https://www.logique.co.id/blog/2020/12/16/agile-vs-devops-vs-cicd/>.

Nguyen-Vu, L., Chau, N.T., Kang, S. dan Jung, S., 2017. Android Rooting: An Arms Race between Evasion and Detection. *Security and Communication Networks*, 2017(4).

NKD, F., 2020. *Data Breaches 2020: Kasus Pelanggaran Data Terbesar yang Terjadi di Tahun 2020*. [daring] Tersedia pada: <https://www.logique.co.id/blog/2020/12/29/data-breaches-2020/>.

Prayama, D., Yuhefizar dan Amelia Yolanda, 2021. Protokol HTTPS, Apakah Benar-benar Aman? *Journal of Applied Computer Science and Technology*, 2(1), hal.7–11.

Pusparisa, Y., 2020. *Pelanggan Netflix Naik 15,8 Juta di Tengah Pandemi Covid-19*. [daring] Tersedia pada: <https://databoks.katadata.co.id/datapublish/2020/04/24/pelanggan-netflix-naik-158-juta-di-tengah-pandemi-covid-19>.

Putra, A.A., Nurhayati, O.D. dan Windasari, I.P., 2016. Perencanaan dan Implementasi Information Security Management System Menggunakan Framework ISO/IEC 20071. *Jurnal Teknologi dan Sistem Komputer*, 4(1), hal.60.

Rovo, 2019. *Xposed Installer*. [daring] Tersedia pada: <https://repo.xposed.info/module/de.robv.android.xposed.installer>.

Shahriar, H., Arabin Talukder, M. dan Saiful Islam, M., 2019. An Exploratory Analysis of Mobile Security Tools. [daring] (February 2020). Tersedia pada: <https://digitalcommons.kennesaw.edu/ccerp https://digitalcommons.kennesaw.edu/ccerp/2019/research/4>.

Sibyan, H., 2017. Implementasi Enkripsi Basis Data Dengan Algoritma Demgan Algoritma MD5 (Message Digest Algorithm 5) Dan Vigenere Cipher. *Ppkm I*, 5, hal.114–121.

Simon Kemp, 2021. *DIGITAL 2021: THE LATEST INSIGHTS INTO THE 'STATE OF DIGITAL.'* [daring] Tersedia pada: <https://wearesocial.com/blog/2021/01/digital-2021-the-latest-insights-into-the-state-of-digital>.

Skylot, 2020. *Jadx*. [daring] Tersedia pada: <https://github.com/skylot/jadx>.

Sulistio, P., 2021. *Dampak Negatif Root HP Android*. [daring] kominfo.bengkulukota.go.id. Tersedia pada: <https://kominfo.bengkulukota.go.id/dampak-negatif-root-hp-android/>.

Suroso, S., Ciksadan, C. dan Sholihatun, S., 2020. Analisis Quality of Service Video Streaming Youtube Dan Rma Wlan Di Politeknik Negeri Sriwijaya. *TESLA: Jurnal Teknik Elektro*, 22(2), hal.93.

Tanjung, I., 2016. *Live-streaming viewers exposed to malware, data theft: Study*. [daring] Tersedia pada: <https://www.thejakartapost.com/life/2016/07/12/live-streaming-viewers-exposed-to-malware-data-theft-study.html>.

Tansen, E. dan Nurdiarto, D.W., 2020. Analisis Dan Deteksi Malware Dengan Metode Hybrid Analysis Menggunakan Framework Mobsf. 4(2), hal.191–201.

Tesalonica, 2020. *Jumlah pengguna unik YouTube di Indonesia capai 93 juta*. [daring] Tersedia pada: <https://www.tek.id/tek/jumlah-pengguna-unik-youtube-di-indonesia-capai-93-juta-b1ZT79iPE>.

Tugas, M. dan El, K., 2016. Fuzzing untuk Menemukan Kerentanan Aplikasi Web. hal.1–17.

Winder, D., 2020. *235 Million Instagram, TikTok And YouTube User Profiles Exposed In Massive Data Leak*. [daring] Tersedia pada: <https://www.forbes.com/sites/daveywinder/2020/08/19/massive-data-leak235-million-instagram-tiktok-and-youtube-user-profiles-exposed/>.

ANALISIS STATIK KEAMANAN APLIKASI VIDEO STREAMING BERBASIS ANDROID MENGGUNAKAN MOBILE SECURITY FRAMEWORK (MOBSF)

ORIGINALITY REPORT

13%
SIMILARITY INDEX

PRIMARY SOURCES

1	www.logique.co.id Internet	68 words — 1%
2	sads.instiki.ac.id Internet	64 words — 1%
3	informatika.stei.itb.ac.id Internet	52 words — 1%
4	Submitted to Telkom University Your Indexed Documents	48 words — < 1%
5	gudangssl.id Internet	45 words — < 1%
6	randi212.blogspot.com Internet	44 words — < 1%
7	docplayer.info Internet	40 words — < 1%
8	developer.android.com Internet	38 words — < 1%
9	eprints.amikom.ac.id Internet	38 words — < 1%

10	www.maxrooted.com Internet	38 words — < 1%
11	ela.kpi.ua Internet	36 words — < 1%
12	media.neliti.com Internet	35 words — < 1%
13	repository.unpas.ac.id Internet	35 words — < 1%
14	misc.teads.tv Internet	34 words — < 1%
15	www.coursehero.com Internet	34 words — < 1%
16	Submitted to Telkom University Your Indexed Documents	33 words — < 1%
17	id.scribd.com Internet	32 words — < 1%
18	if.polibatam.ac.id Internet	25 words — < 1%
19	journal2.uad.ac.id Internet	25 words — < 1%
20	play.google.com Internet	25 words — < 1%
21	republika.co.id Internet	25 words — < 1%

22	www.joesandbox.com Internet	24 words — < 1%
23	edocs.ilkom.unsri.ac.id Internet	21 words — < 1%
24	e-jurnal.pnl.ac.id Internet	20 words — < 1%
25	www.jsp.fisip-unmul.ac.id Internet	20 words — < 1%
26	hdl.handle.net Internet	19 words — < 1%
27	jurnal.pancabudi.ac.id Internet	18 words — < 1%
28	www.360modunduh.com Internet	18 words — < 1%
29	beta.pithus.org Internet	17 words — < 1%
30	Shikha Badhani, Sunil K. Muttoo. "Evading android anti-malware by hiding malicious application inside images", International Journal of System Assurance Engineering and Management, 2017 Crossref	16 words — < 1%
31	budi.rahardjo.id Internet	16 words — < 1%
32	hosteko.com Internet	16 words — < 1%
	jim.unindra.ac.id	

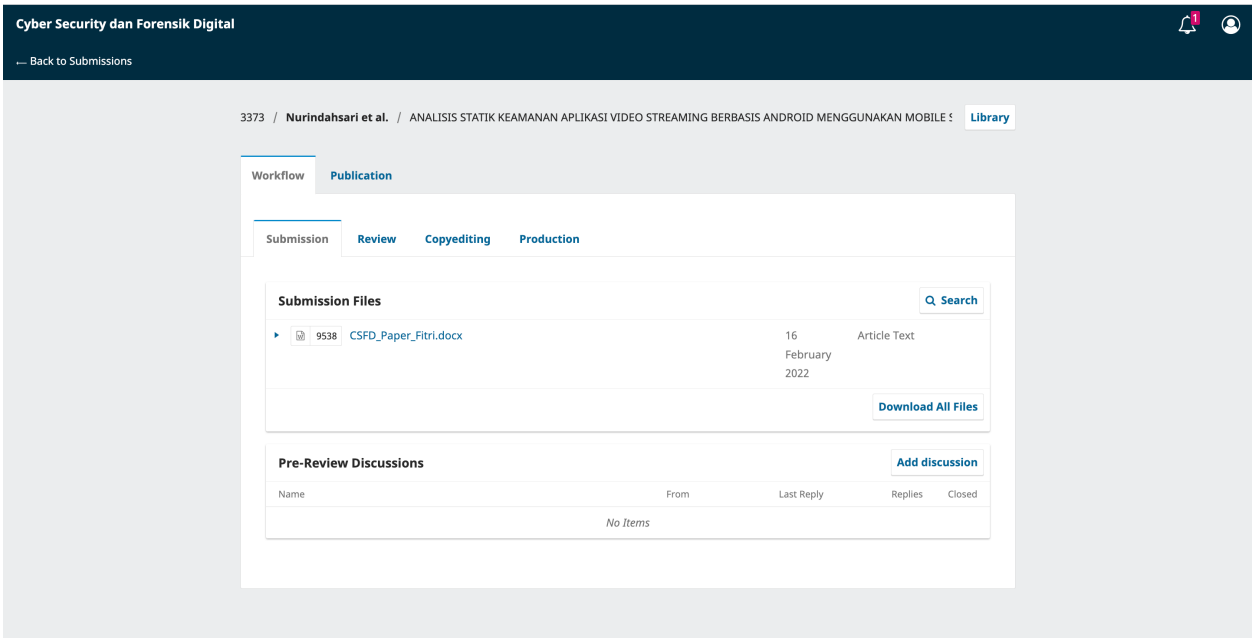
33	Internet	16 words — < 1%
34	openjournal.unpam.ac.id Internet	16 words — < 1%
35	ojs.unud.ac.id Internet	15 words — < 1%
36	ejournal.uin-suka.ac.id Internet	13 words — < 1%
37	journal.stmikjayakarta.ac.id Internet	13 words — < 1%
38	repository.uph.edu Internet	13 words — < 1%
39	webapps.bps.go.id Internet	13 words — < 1%
40	Kalisha Latifa Kartawijaya, Kyla Aisha Humaira, Marcellino Wijaya, Muhammad Gavin Wicaksono. "MEDIA SOSIAL SEBAGAI SARANA UNTUK MEMBANGUN KARAKTER MASYARAKAT YANG BERLANDASKAN NILAI PANCASILA", Jurnal Kewarganegaraan, 2021 Crossref	12 words — < 1%
41	ejournal.stmikgici.ac.id Internet	12 words — < 1%
42	lib.ui.ac.id Internet	12 words — < 1%
43	repository.uinsaizu.ac.id Internet	12 words — < 1%

44	journal.ittelkom-pwt.ac.id Internet	11 words — < 1%
45	repositori.uma.ac.id Internet	11 words — < 1%
46	socs.binus.ac.id Internet	11 words — < 1%
47	android.googlesource.com Internet	10 words — < 1%
48	ejurnal.umri.ac.id Internet	10 words — < 1%
49	repository.uin-suska.ac.id Internet	10 words — < 1%
50	repository.unpar.ac.id Internet	10 words — < 1%
51	Khaled Sarieddine, Mohammad Ali Sayed, Sadegh Torabi, Ribal Atallah, Chadi Assi. "Investigating the Security of EV Charging Mobile Applications As an Attack Surface", ACM Transactions on Cyber-Physical Systems, 2023 Crossref	9 words — < 1%
52	www.kutuphane.sakarya.edu.tr Internet	9 words — < 1%
53	www.scribd.com Internet	9 words — < 1%
54	000copas-sanasini000.blogspot.com Internet	8 words — < 1%

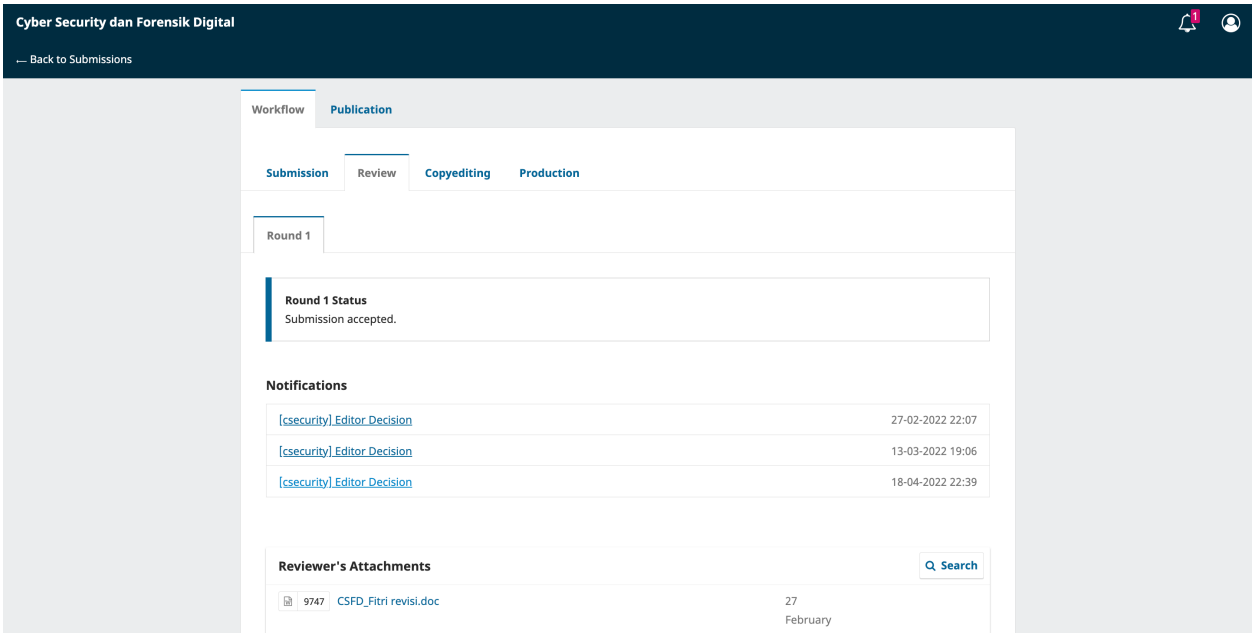
55	Eslam Amer, Shaker El-Sappagh. "Robust deep learning early alarm prediction model based on the behavioural smell for android malware", Computers & Security, 2022 Crossref	8 words — < 1%
56	digilib.binadarma.ac.id Internet	8 words — < 1%
57	fliphtml5.com Internet	8 words — < 1%
58	www.jurnal.wicida.ac.id Internet	8 words — < 1%
59	Submitted to Telkom University Your Indexed Documents	6 words — < 1%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	OFF
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	OFF

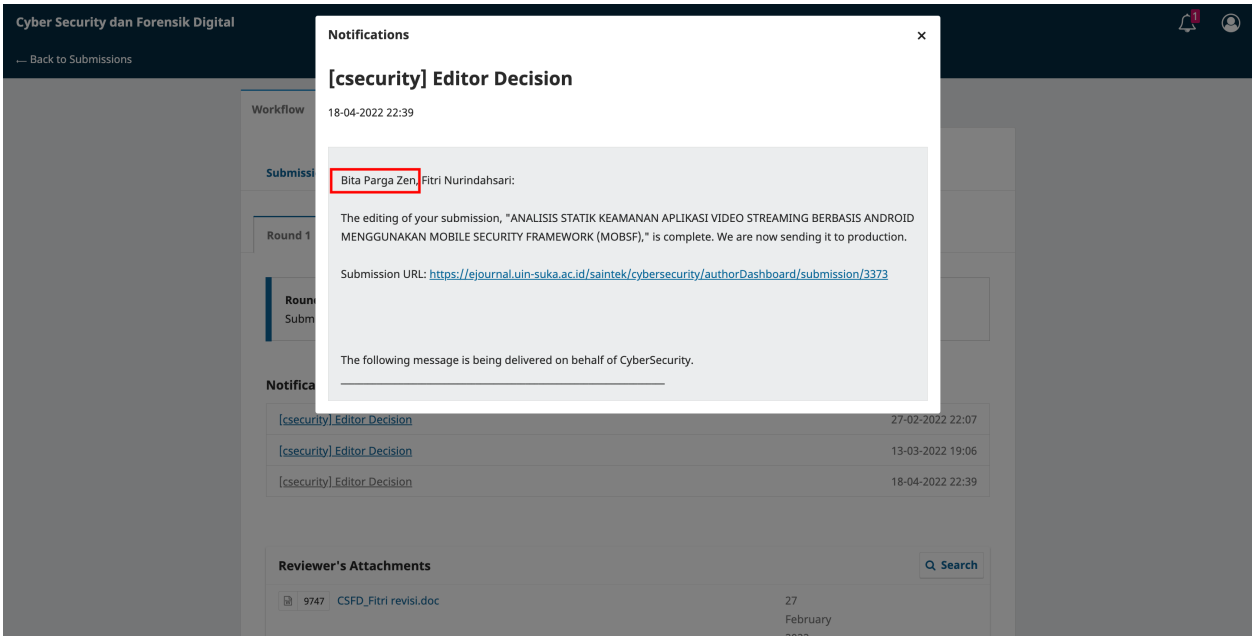
Proses Submission



Proses Review



Proses Accepet



Bukti penulis korespondensi pada naskah