Jurnal Ilmiah Sinus (JIS) Vol : 21, No. 1, Januari 2023
ISSN (Print) : 1693-1173 , ISSN (Online): 2548-4028

Rancang Bangun Mobile Secure Chat dengan Mengimplementasikan Metodologi SSDLC-Agile dan Kriptografi

Hermawan Setiawan*[1)], Akhmad Rizal[2)]
[1,2)]*Jurusan Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, Bogor*
[*1]*hermawan.setiawan@poltekssn.ac.id* , [2]*akhmad.rizal@poltekssn.ac.id*

ABSTRACT

Along with the times, the exchange of information is becoming faster. People now use chat applications as a form of communication. With this application, users can easily exchange messages, whether in the form of text, images, audio, or video, without being limited by differences in distance and time. The information circulating varies, ranging from unclassified, private, to maybe confidential. Therefore, an encryption-decryption mechanism is needed so that users can exchange information safely through the mobile chat application. This study aims to implement the Rabin cryptosystem, symmetric encryption, and the use of session keys in a secure chat infrastructure for mobile applications. The test results showed that the combination of cryptographic applications and the Secure SDLC can increase security in application development, in this case for mobile secure chat.

Keywords*: Agile, Android, Chat, Crytography, Mobile, Rabin Cryptosystem, Session key, Secure SDLC*

I. PENDAHULUAN

Fasilitas penyebaran informasi pada era teknologi yang saat ini dapat dilakukan dengan cepat dan mudah melalui media aplikasi Android. Salah satu media yang paling sering digunakan untuk penyebaran informasi adalah *chatting* (Sulaksono et al., 2021). Aplikasi *mobile chat* sendiri memanfaatkan jaringan internet untuk melakukan pertukaran informasi. Akan tetapi, informasi dan jaringan bersifat selalu terbuka dan dapat diakses oleh semua penggunanya. Hal ini membuat banyak data dan sumber daya informasi menjadi mudah diserang, serta masalah terkait keamanan informasi di internet semakin berkembang. Masalah tersebut juga merupakan salah satu pemicu utama dari bahaya-bahaya tersembunyi yang ada di internet (Shen, 2020).

Ada berbagai aplikasi *mobile chat* yang dapat diakses di platform penyedia aplikasi(Kusumaningrum et al., 2022). Sejumlah besar aplikasi ini mengklaim bahwa mereka menyediakan keamanan, kerahasiaan, dan integritas informasi pengguna. Akan tetapi adanya peretasan informasi membuktikan bahwa banyak pengembang tidak menganggap keamanan sebagai pertimbangan utama dalam aplikasi mereka (H. Ali & Sagheer, 2017). Namun aplikasi yang banyak beredar tentu data yang tersimpan bukan berada pada media penyimpanan milik sendiri. Data itu rawan untuk disalah gunakan meski ujung ke ujung dienkripsi.

Penerapan metode kriptografi dalam aplikasi *chatting* diperlukan untuk mengatasi permasalahan keamanan tersebut. Berdasarkan jenis kuncinya, metode kriptografi dibedakan menjadi dua, yaitu simetris dan asimetris. Kunci simetris merupakan kunci yang dipakai dalam proses enkripsi dan dekripsi. Sedangkan kunci asimetris memiliki kunci yang berbeda dalam proses enkripsi dan dekripsi, proses enkripsi menggunakan kunci publik dan proses dekripsi menggunkan kunci privat (Hevianto Saputro et al., 2020). Masalah yang umumnya terjadi pada kriptografi kunci simetris adalah pihak yang ingin mengenkripsi ataupun mendekripsi data harus memiliki kunci yang sama, dimana jalan satu-satunya adalah dengan menyebarluaskan kunci tersebut (Mulya et al., 2021). Sementara itu, keuntungan utama

enkripsi kunci asimetris adalah memiliki enkripsi yang kuat sehingga membuat dekripsi ke teks asli menjadi sulit dan tidak dapat diprediksi oleh peretas (Mallouli et al., 2019).

Disamping itu, aspek keamanan ketika membangun aplikasi juga perlu diperhatikan. Banyak developer yang memulai proses pembuatan aplikasi tanpa mempertimbangkan cacat dan kerentanan keamanan yang mungkin terjadi. Sejumlah besar kerentanan dalam perangkat lunak terjadi karena buruknya analisis, desain, dan metode pengembangan yang digunakan. Maka dari itu, dibutuhkan cara untuk meningkatkan keamanan pada proses pengembangan aplikasi(Tung et al., 2016). Kebutuhan akan keamanan dalam proses Software Development Life Cycle (SDLC) menciptakan sebuah metodologi yang disebut sebagai *Secure Software Development Life Cycle* (SSDLC). SSDLC menambahkan parameter *security*, *safety*, dan *peformance* pada metodologi SDLC klasik (Fujdiak et al., 2019).

SSDLC memiliki banyak jenis metode. Salah satu metode SSDLC yang mengakomodasi aspek keamanan dalam setiap langkahnya adalah *Security Development Lifecycle-Agile* (SDL-Agile). *Agile* cocok untuk pengembangan sistem skala kecil karena lebih cepat beradaptasi dengan perubahan kebutuhan (Sugiantoro et al., 2020). Atas pertimbangan tersebut, penulis memilih metode agile untuk digunakan dalam penelitian ini.

Sebelumnya telah terdapat beberapa penelitian yang membahas mengenai penerapan algoritma kriptografi pada aplikasi chat, baik kriptografi simetris maupun asimetris. Namun, kebanyakan dari penelitian tersebut belum menerapkan metode SSDLC dalam pembuatan aplikasinya. Oleh karenanya, penelitian ini memadukan antara implementasi metode SSDLC dengan penerapaan kriptografi bahkan dengan pembuatan infrastruktur kunci public dalam pembangunan aplikasi mobile *chat* berbasis Android.

II. TINJAUAN PUSTAKA

2.1.Model *Agile*

Tahapan-tahapan pada model *agile* diilustrasikan pada Gambar 1: (1) *Brainstrom*, (2) *Design*, (3) *Development*, (4) *Quality Assurance/Testing (Code Review)*, (5) *Deployment (Security Assesssment)*.

Gambar 1. *Agile Life Cycle* (Hema et al., 2020)

Pada penelitian ini dikombinasikan tahapan-tahapan perancangan perangkat lunak pada model SDLC *Agile* dengan aspek-aspek keamanan, seperti *Security Requirements* dan *Misuse Case* pada *Brainstrom*, *STRIDE-DREAD* pada *Design*, implementasi kriptografi pada *Development*, *Code Review* pada *Testing*, dan yang terakhir yaitu tahap *Deployment*(Hussain et al., 2014).

2.2.*Brainstrom*

Di tahap ini menggunakan *security requirements* dan *misuse case* untuk melakukan analisis kebutuhan keamanan apa saja yang diperlukan pada rancang bangun *mobile chat* ini (Ansari et al., 2022).

 DOI : https://doi.org/10.30646/sinus.v21i1.660

2.3.*Design*

2.3.1. STRIDE

Pada tahap *design*, penulis menerapkan salah satu metode *thread modelling* yang bernama STRIDE-DREAD. STRIDE dibentuk dari huruf pertama dari masing-masing kategorinya, yaitu *Spoofing, Tampering, Repudiation, Information disclosure, Denial of service*, dan *Elevation of privilege* (Conklin & Robinson, 2017).

2.3.2. DREAD

Di sisi lain, DREAD merupakan model perankingan risiko yang dikembangkan oleh *Microsoft* untuk menghitung risiko yang dapat menghasilkan informasi peringkat risiko suatu ancaman yang terjadi, yang memiliki beberapa kategori : *Damage potential*, *Reproducibility*, *Exploitability*, *Affected user* dan *Discoverability* (Laksono & Prayudi, 2021).

Setiap ancaman akan dinilai berdasarkan parameter. Pada tabel 1, terdapat 3 jenis peringkat pada setiap sub poin DREAD, yakni 3 untuk peringkat tinggi, 2 untuk peringkat sedang, dan 1 untuk peringkat rendah. Total nilai yang diperoleh dari setiap ancaman akan menentukan peringkat ancaman. Parameter-parameter penentuan nilai didasarkan pada penelitian Azis Catur Laksono pada tahun 2020 (Laksono & Prayudi, 2021).

Tabel 1. DREAD *Ranking*

Rentang Nilai	Peringkat Risiko
5 – 7	Rendah
8 – 11	Sedang
12 – 15	Tinggi

2.4. Secure Software Development Life Cycle (SSDLC)

Secure Software Development Life Cycle (SSDLC) adalah tentang cara membangun perangkat lunak yang aman. Dengan mendesain perangkat lunak aman, yakni memastikan perangkat lunak itu aman. Lalu mengedukasi developer, arsitek, dan pengguna perangkat lunak tentang cara membangun suatu perangkat lunak yang aman (McGraw, 2006).

2.5. *Rabin Cryptosystem*

Pada tahun 1979, diperkenalkan suatu kripstosistem yang didasarkan pada ketangguhan untuk memecahkan masalah akar kuadrat modular dari bilangan bulat komposit, yaitu *Rabin Cryptosystem*. Peran eksponen publik dari enkripsi Rabin memberikan keuntungan komputasi atas Kriptosistem RSA (Asbullah et al., 2016). Dalam penelitian ini sendiri akan digunakan modifikasi dari Rabin Cryptosystem untuk membentuk suatu sertifikat digital. Sertifikat digital ini dijadikan dasar dalam pembuatan Infrastruktur Kunci Publik (IKP).

2.6. Penelitian Terkait

Pada 2017, Ammar H. Ali dan Ali M. Sagheer mengusulkan solusi membuat *end-to-end encryption* yang memberikan kerahasiaan, privasi, integritas, serta efisiensi peforma pada aplikasi chat dengan menerapkan skema kriptografi yang menggabungkan *Elliptic Curve Diffie-Hellman* (ECDH), AES, dan RC4 untuk proses enkripsi/dekripsi. Sistem yang diusulkan telah diinstal dan diuji pada beberapa perangkat ponsel yang berbasis sistem operasi android dengan berbagai kemampuan CPU dan *Random Access Memories* (RAM), untuk memastikan dapat bekerja dengan baik pada berbagai *device* (H. Ali & Sagheer, 2017).

Pada 2018, Aminudin, Ahmad Faisal Helmi, dan Sofyan Arifianto melakukan analisis mengenai penerapan kombinasi algoritma *Merkle-Hellman Knapsack* dan logaritma diskrit pada aplikasi chat. Selanjutnya, dilakukan pengujian terhadap ketahanan performa algoritma, baik algoritma knapsack standar maupun algoritma knapsack dengan algoritma

diskrit yang meliputi waktu proses pembangkitan kunci, enkripsi dan dekripsi dengan menggunakan metode *avalanche effect* dan *known plaintext attack* (Aminudin et al., 2018).

Kemudian pada tahun 2020, Ainur Rilo Taqwa dan Danang Haryo Sulaksono mengimplementasikan *Elliptic Curve Cryptography* (ECC) untuk aplikasi chatting berbasis Android. Penelitian ini bertujuan untuk menjaga keamanan dan menyembunyikan file data online pada aplikasi chatting yang bersifat privasi menggunakan algoritma ECC sehingga tidak dapat dilihat oleh user lain (Sulaksono et al., 2021).

III. METODE PENELITIAN

3.1. *Development*

Seperti SDLC, SSDLC terdapat tahapan dari Desain sampai Pengujian(*Checkmarx.Com_glossary_a-Secure-Sdlc-with-Static-Source-Code-Analysis-Tools*, n.d.). Pengembangan aplikasi ini dengan menggunakan SSDLC dijamin keamanannya. Dalam tahap awal ini diimplementasikan beberapa metode kriptografi yang terdiri dari *symmetric encryption, session key*, dan *Rabin Cryptosystem* sehingga terbentuk suatu IKP. Penggunaan metoda kriptografi untuk mengamankan tahap penggunaannya.

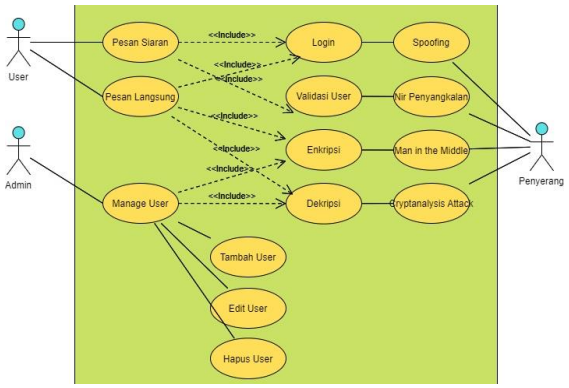
Berdasarkan penjelasan sebelumnya, disusun sebuah alur penelitian, seperti pada Gambar 5. Dari gambar tersebut dapat diketahui bahwa terdapat 2 *tools* yang akan digunakan oleh penulis dalam penelitian ini, yaitu *Microsoft Threat Modeling Tools* (MTMT) serta *Sonarcloud*(Hajrić et al., 2020). Sementara itu, tahapan lain akan dilakukan secara manual dengan acuan dari beberapa paper terkait.

Selain itu, perlu untuk mendefinisikan *requirement* apa saja yang dibutuhkan untuk dapat mengatasi hal tersebut. Beberapa *security requirements* yang dapat diterapkan pada aplikasi yakni : *Password Constraint*, *Key Management*, *Anti-CSRF*, *Session*, *End-to-End Security*, *Identity Management*, dan *Privacy*.

3.2. *Brainstrom*

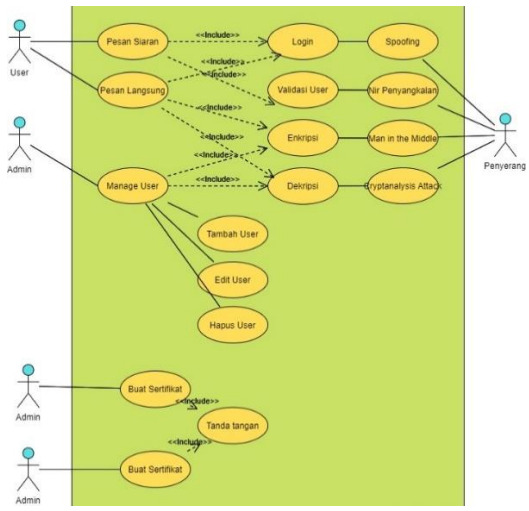
Dalam pembuatan aplikasi *secure chat*, langkah pertama yang penulis lakukan yaitu menentukan kerawanan-kerawanan apa saja yang ada pada aplikasi ini. Untuk menentukan hal tersebut, dibuatlah *misuse case diagram*(Saleh & El-Attar, 2015). Dari diagram tersebut, akan terlihat aktivitas apa saja yang dapat dilakukan oleh para aktor (*use case*) disertai dengan penyalahgunaan yang dapat terjadi nantinya(Prabowo et al., 2022). Hal ini seperti yang terlihat pada Gambar 2.

Pada gambar tersebut, dapat diketahui bahwa terdapat 2 aktor umum pada sistem, ditambah dengan 1 aktor tambahan sebagai penyerang. Dua aktor ini yaitu pengguna biasa (*user*) dan admin. *User* dapat melakukan pengiriman pesan, baik secara *broadcast* maupun privat. Pesan *broadcast* dapat dibaca oleh semua *user*, sedangkan pesan privat hanya dapat dibaca oleh *user* tertentu. Sementara itu, admin memiliki akses untuk melakukan *add*, *edit*, dan *delete* pada *user*.



Gambar 2. *Misuse Case*

 DOI : https://doi.org/10.30646/sinus.v21i1.660



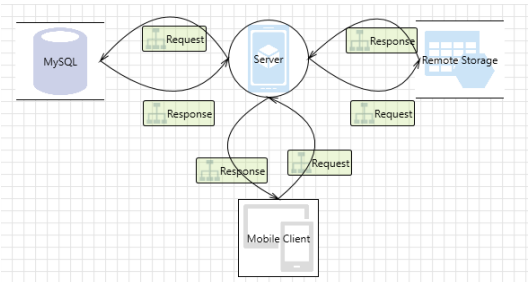
Gambar 3. *Misuse Case* dengan *Root* dan *CA*

Di sisi lain, ada aktor penyerang yang dapat melakukan tindakan yang tidak sesuai (*misuse*) pada sistem. Penyerang bisa melakukan *spoofing* pada aktivitas login untuk mengaku sebagai pengguna yang lain untuk kepentingan tertentu. Penyerang juga dapat melakukan penyangkalan atas tindakan tidak sesuai yang telah dilakukan apabila tidak terdapat mekanisme khusus untuk mengatasi hal tersebut. Selain itu, pada saat aktivitas pertukaran pesan terjadi antar pengguna, penyerang dapat melakukan *man-in-the-middle attack*, serta melakukan kriptanalisis untuk mendekrip pesan yang telah didapatkan.

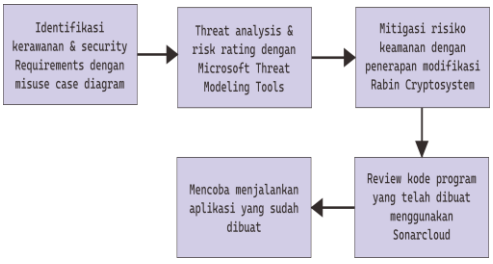
Untuk mengatasi *misuse* yang ditampilkan di Gambar 2, dapat ditambahkan 2 aktor lain, yakni *root* dan CA, seperti yang terlihat pada Gambar 3. Mereka akan bertugas untuk menerbitkan dan menandatangani sertifikat. Sertifikat ini nantinya dapat digunakan untuk melakukan tanda tangan digital. Tanda tangan digital adalah satu-satunya teknik terbaik untuk memberikan otentikasi dan nir-penyangkalan dengan menggunakan algoritma kunci publik. Oleh karenanya, penerapan tanda tangan digital akan memberikan integritas pada system(Carita & Wahyuni, 2022; Sharma & Kapoor, 2017) .

3.3. *Session Key*

Session key adalah kunci yang hanya dapat digunakan dalam suatu sesi tertentu saja. Kunci ini akan dibentuk ketika dua belah pihak telah saling setuju dan percaya untuk berkomunikasi. Dalam hal ini, *session key* tersusun atas 32 karakter, dimana 26 karakter *random* dan 6 karakter berupa *timestamp*. *Session key* akan dihapus saat salah satu atau kedua belah pihak mengakhiri sesi.



Gambar 4. Model Struktur Aplikasi



Gambar 5. Alur tahapan penelitian

3.4. *Testing dan Deployment*

Setelah itu, ketiga komponen yang berupa *symmetric encryption*, *Rabin Cryptosystem*, dan *session key* akan dipadukan dalam sebuah IKP. IKP adalah cara menyediakan langkah-langkah keamanan dengan menerapkan sarana pasangan kunci di antara pengguna (Albarqi et al., 2015) . Selanjutnya, penulis melakukan *testing* pada proyek yang telah dibuat menggunakan *code review* (Ebert et al., 2021).

Setelah aplikasi dibuat dengan menggunakan sistem yang diinginkan, aplikasi tersebut akan dicoba untuk dijalankan. Hal ini dilakukan untuk mengetahui apakah terdapat *error*, kerusakan program, program tidak berjalan sesuai dengan keinginan, ataupun terjadi *bug* pada aplikasi yang telah dibuat.

IV. HASIL DAN PEMBAHASAN

Aplikasi chat yang ditawarkan dalam penelitian ini tidak seperti aplikasi chat pada umumnya yang hasil komunikasi chat disimpan oleh pihak ketiga. Aplikasi yang ditawarkan dalam penelitian ini dalam bentuk komunikasi chat terbatas. Dibandingkan dengan aplikasi yang ada pada penelitian terkait, meski sama menggunakan kriptografi asimetrik namun yang dipilih bukan metoda Rabin. Dibandingkan dengan kriptografi asimetrik lain dari sisi kematangan, keamanan, kecepatan komputasi, konsumsi sumber daya, Metoda Rabin memiliki kesamaan dengan RSA(Wang et al., 2020). Aplikasi dibangun dengan pendekatan *SSDLC*. Integrasi sistem chatting akan membantu mengurangi ketidakamanan dalam sistem chatting. Belum ada aplikasi chat yang menggabungkan dengan SSDLC dalam pengembangannya. Bahkan belum ada juga yang menggunakan infrastruktur kunci public untuk sisi integritas dan autentikasinya (A. H. Ali & Sagheer, 2017; R. M. Ali & Alsaad, 2020; Kuliya & Abubakar, 2020; MM et al., 2016; Nayak et al., 2017; Zebua et al., 2018)

Hasil yang diperoleh untuk pengujian keamanan melalui MTMT, diperoleh 35 *threat* yang ada pada model. Selanjutnya, penulis merangkum ancaman-ancaman tersebut menjadi 10 ancaman secara garis besar. Terdapat beberapa ancaman yang mungkin terjadi pada model aplikasi *chat* yang akan dibuat. Secara detail, ada 1 kategori *spoofing*, 2 kategori *tampering*, 1 kategori *repudiation*, 2 kategori *information disclosure*, 1 kategori *denial of service*, serta 3 kategori *elevation of privileges.*

Hasil dari perhitungan peringkat setiap ancaman dapat dilihat pada Tabel 2. Dari Tabel 2, kemungkinan ancaman yang ada dapat diurutkan sesuai dengan peringkatnya, mulai dari yang tertinggi hingga terendah. Hal ini dimaksudkan untuk memprioritaskan upaya mitigasi pada ancaman dengan prioritas tertinggi terlebih dahulu. Mitigasi risiko sendiri merupakan pengambilan Langkah-langkah untuk mengurangi kerugian yang dapat ditimbulkan dari dampak atas risiko (Setyadi & Kusumawati, n.d.). Mitigasi dari setiap ancaman akan dilakukan pada tahap *development*.

4.1. *Development*

Setelah tahap *brainstorm* dan *design*, maka tahap selanjutnya yaitu untuk membuat infrastruktur yang aman untuk aplikasi *secure chat*. Pada tahap-tahap sebelumnya telah diketahui kerawanan dan risiko apa saja yang mungkin terjadi. Oleh karena itu, tahap *development* ini akan bertujuan untuk mengatasi segala permasalahan tersebut. Langkah mitigasi pertama ditujukan untuk mengatasi permasalahan terkait *repudiation*, *spoofing*, dan *elevation of privileges*. Untuk itu, diperlukan sebuah metode yang dapat mengotentikasi pengguna. Salah satu metode yang paling banyak digunakan untuk mengamankan aplikasi adalah dengan memakai *user-id* dan *password* yang dimasukkan pada *form login*. Namun demikian, penggunaan *user-id* dan *password* bukannya tanpa kelemahan. Pengguna sering

 DOI : https://doi.org/10.30646/sinus.v21i1.660

kali memilih user-id dan password yang pendek dan lemah sehingga mudah dicuri dengan teknik *brute force* (Harn & Ren, 2011).

Sertifikat digital kunci publik telah banyak digunakan dalam IKP untuk memberikan autentikasi pada kunci publik pengguna yang terdapat dalam sertifikat. Pengguna hanya akan diautentikasi jika dia dapat membuktikan bahwa dia memiliki kunci privat yang sesuai dengan kunci publik yang ditentukan dalam sertifikat digital tersebut (Subari & Iman Satoto, n.d.). Seperti yang diperlihatkan pada Gambar 7, akan terdapat satu domain pusat yang akan bertindak sebagai *Root* CA, dengan CA yang terbagi ke setiap sub domain, untuk menerbitkan sertifikat dari para pengguna di bawah domain tersebut. Sertifikat ini dibuat dengan memodifikasi algoritma *Rabin Cryptosystem* dan SHA256. Dengan penerapan sertifikat untuk proses otentikasi dan verifikasi, maka permasalahan ke-1, 4, 5, 6, dan 9 pada Tabel 2 dapat teratasi.

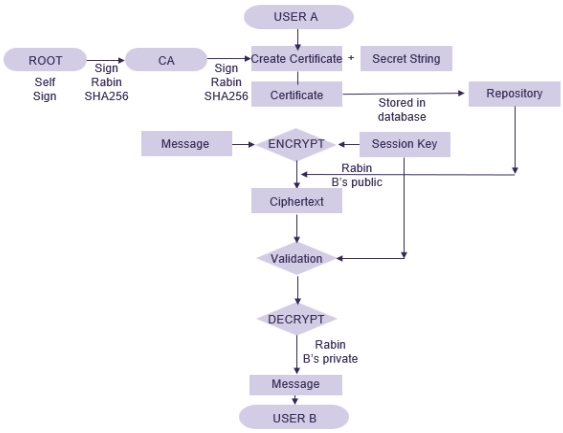
Tabel 2. *Threat Ranking*

Ancaman	**D**	**R**	**E**	**A**	**D**	**Total**	***Rank***
Musuh dapat membobol perangkat seluler dan mendapatkan hak istimewa yang lebih tinggi	2	1	1	1	1	6	Rendah
Musuh dapat memperoleh akses ke data sensitif dengan menyadap lalu lintas data dari klien *mobile*	2	2	2	2	1	9	Sedang
Musuh dapat melakukan *reverse engineering* dan merusak *binary program*	1	1	2	1	1	6	Rendah
Musuh dapat memperoleh akses secara tidak sah ke server karena konfigurasi jaringan yang lemah	3	1	1	3	1	9	Sedang
Musuh dapat menolak melakukan tindakan terhadap sumber daya terkait karena kurangnya audit yang menyebabkan masalah penolakan	2	2	2	1	1	8	Sedang
Musuh dapat melakukan tindakan atas nama pengguna lain	2	3	3	2	2	12	Tinggi
Musuh dapat merusak basis data penting yang dapat diamankan dan menolak tindakannya	2	1	1	2	2	8	Sedang
Musuh dapat memperoleh akses ke data sensitif dengan melakukan injeksi SQL	2	1	2	1	2	8	Sedang
Musuh dapat memperoleh akses tidak sah ke basis data karena kurangnya perlindungan akses jaringan	2	1	1	2	1	7	Rendah
Musuh dapat memblokir akses ke aplikasi atau server melalui serangan *denial of service*	1	2	3	2	1	9	Sedang

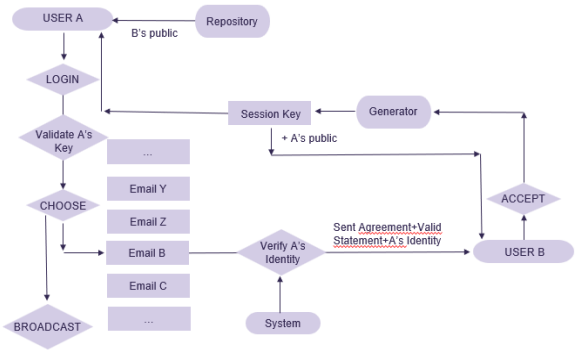
Selanjutnya, untuk memitigasi permasalahan ke-2, 7, dan 8, sertifikat yang sudah dibuat akan diimplementasikan dalam skema pertukaran pesan (*private message*) seperti yang ditampilkan dalam Gambar 6. Selain kunci publik, skema ini juga melibatkan penggunaan dari *session key* dan enkripsi simetris. *Session key* akan menetapkan privasi pada setiap node selama komunikasi data. Node disini berarti interaksi privat antara dua orang pengguna dalam jaringan. *Session key* menyediakan metode komunikasi yang aman menggunakan enkripsi simetris dan perutean otentikasi untuk melindungi pesan. Hal ini akan membuat data terlindungi dengan kunci enkripsi unik dan tepercaya yang dihasilkan menggunakan jalur yang tepercaya pula. Dengan begitu, akan menambah segi keamanan saat proses perukaran pesan berlangsung (Subari & Iman Satoto, n.d.).

Berdasarkan Gambar 6, terlihat bahwa ketika seorang pengguna hendak mengirim pesan secara privat ke satu target pengguna, pesan akan dienkrip dengan menggunakan *session key* dan *public key* dari target. Di sisi lain, target akan melakukan validasi *session key* pesan yang telah dienkrip serta melakukan dekripsi dengan *private key* miliknya. Apabila pesan tersebut benar dari pengirim dan ditujukan untuk penerima tersebut, maka akan diperoleh pesan yang

sebenarnya. Hal yang sama juga berlaku ketika pihak penerima akan membalas pesan dari pengirim.

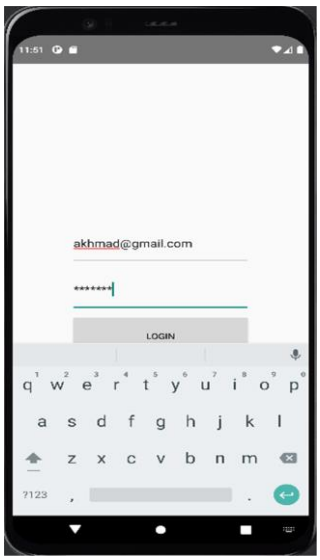


Gambar 6. Skema pengiriman pesan

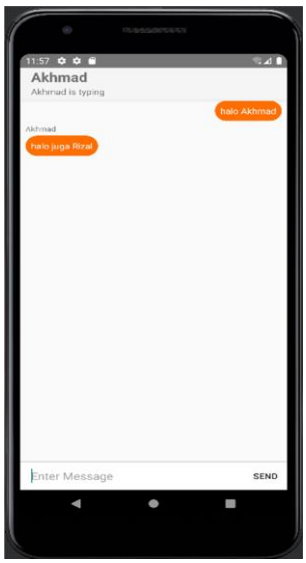


Gambar 7. Mekanisme *Agreement-Acceptance*

Selain itu, terdapat mekanisme *Agreement-Acceptance* untuk pertukaran *session key*, yang dapat dilihat pada Gambar 7. Mekanisme ini dilangsungkan sebelum komunikasi berjalan(Policy & Bodansky, 2015). Pihak yang ingin mengirim pesan, telah melakukan *login* dan divalidasi oleh sistem dengan mengecek *private key* yang disimpan oleh pengguna dengan *public key* yang disimpan pada *database*. Apabila pengguna memilih untuk mengirimkan pesan secara privat, maka sistem akan melakukan verifikasi identitas dari pengirim. Jika verifikasi berhasil, akan dikirimkan *agreement* + *valid statement* + identitas dari pengirim ke penerima. Kemudian, *generator* (sistem) akan me-*generate session key*. Pengirim akan menerima *session key* + ID dari penerima, juga *public key* penerima yang diambil dari *database*. Sedangkan, penerima akan mendapatkan *session key* + *public key* pengirim yang diambil dari *database*. Alhasil, baik pengirim ataupun penerima telah mendapatkan *session key* yang sama sehingga komunikasi bisa dilangsungkan.



Gambar 8. Halaman *login*



Gambar 9. Halaman *private chat*

 DOI : https://doi.org/10.30646/sinus.v21i1.660

4.2. *Testing*

Setelah infrastruktur dan program selesai dibuat, dilakukan tahap selanjutnya yaitu *testing*. *Sonarcloud* digunakan untuk *testing* melalui *code review* (Mathas et al., 2021). Gambar 8 menunjukkan sebelum seseorang bisa *chat* dengan melakukan login dan gambar 9 setelah berhasil login. Serangan yang mungkin terjadi adalah saat melakukan login. Dari hasil *code review* aplikasi yang dibuat memiliki aspek *Reliability, Maintainability,* dan *Security* dengan rangking A, sedangkan aspek *Security Review* memiliki rangking B. Hal tersebut karena masih terdapat beberapa *security hotspot* yang ditemukan.

Terdapat 3 jenis *security hotspot* yang ada pada program, yaitu *encryption of sensitive data*, *insecure configuration*, serta *others* (*write external storage permission* dan *backup of application data*). *Security hotspot* akan menyoroti bagian kode yang sensitif terhadap keamanan yang perlu ditinjau oleh pengembang. Ketiga *security hotspot* ini memiliki kategori *Low*. Artinya, ancaman yang mungkin ditimbulkan rendah.Di samping itu, kerawanan nomor 1 dan 2, memang diaktifkan oleh penulis untuk keperluan *debugging*. Kerawanan nomor 3 sendiri memang diperlukan oleh aplikasi untuk menulis data-data yang diperlukan pada *external storage*, diantaranya yaitu *private key*, *session key*, dan sebagainya. Kemudian, untuk kerawanan nomor 4 dapat diatasi dengan melakukan *backup* data secara berkala. Oleh karenanya, penulis memasukkan mitigasi dari ancaman tersebut kedalam kategori *acceptance*.

4.3. *Deployment*

Tahap terakhir adalah *deployment*, dimana akan dilihat apakah terdapat ketidaksesuaian (*misbehavior*) dari jalannya aplikasi. Dimulai dari halaman login, pengguna akan diminta memasukkan email dan *password* yang telah terdaftar. Apabila pengguna dapat memberikan email dan *password* yang sesuai, serta memiliki pasangan kunci yang valid, maka tampilan akan beralih ke halaman utama.

Tabel 3. *Mitigated Threat*

Ancaman	Ter mitigasi
Musuh dapat membobol perangkat seluler dan mendapatkan hak istimewa yang lebih tinggi	V
Musuh dapat memperoleh akses ke data sensitif dengan menyadap lalu lintas data dari klien *mobile*	V
Musuh dapat melakukan *reverse engineering* dan merusak *binary program*	X
Musuh dapat memperoleh akses secara tidak sah ke server karena konfigurasi jaringan yang lemah	V
Musuh dapat menolak melakukan tindakan terhadap sumber daya terkait karena kurangnya audit yang menyebabkan masalah penolakan	V
Musuh dapat melakukan tindakan atas nama pengguna lain	V
Musuh dapat merusak basis data penting yang dapat diamankan dan menolak tindakannya	V
Musuh dapat memperoleh akses ke data sensitif dengan melakukan injeksi SQL	V
Musuh dapat memperoleh akses tidak sah ke basis data karena kurangnya perlindungan akses jaringan	V
Musuh dapat memblokir akses ke aplikasi atau server melalui serangan *denial of service*	X
Musuh dapat melakukan tindakan atas nama pengguna lain karena kurangnya kontrol terhadap permintaan lintas domain	V

Sebagaimana disebutkan dalam Tabel 3, masih ada beberapa ancaman pada Tabel 3 yang belum termitigasi. Diperlukan *resource* lebih untuk mengatasi ancaman berupa *denial of service*, misalnya dengan memperbesar *bandwidth* server ataupun menggunakan *hosting* pada *provider* terpercaya. Sementara pada penelitian ini, pengujian masih menggunakan

server lokal. Di sisi lain, ancaman berupa *reverse engineering* dapat diatasi dengan melakukan obfukasi pada program. Namun, hal ini tidak memungkinkan karena terdapat beberapa parameter pada aplikasi yang harus sesuai dengan server, seperti url server, objek variabel, dan sebagainya.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penerapan SSDLC pada pembangunan perangkat lunak terbukti mampu mengatasi peluang ancaman yang mampu menimbulkan risiko keamanan. Dari Tabel 3, terlihat bahwa 9 dari 11, atau sekitar 81.81% ancaman berhasil termitigasi. Tercatat ada dua ancaman yang belum termitigasi yaitu mengenai serangan *reverse engineering* dan *denial of service*. Akan tetapi, sebagian besar ancaman yang ada mampu dideteksi sebelumnya sehingga dapat diatasi dengan baik. Hal ini karena, seluruh kemungkinan kerawanan dan risiko telah teridentifikasi, baik pada tahap *brainstorm*, *design*, serta testing. Begitu juga dengan langkah mitigasi yang dapat dilakukan pada saat berada ditahap deployment, setelah mengetahui informasi mengenai celah-celah yang ada dari tahap-tahap sebelumnya. Oleh karena itu, kombinasi penerapan kriptografi dan *SSDLC* dapat meningkatkan segi keamanan pada aplikasi.

Penelitian selanjutnya dapat dilakukan dengan membandingkan penerapan dari metode *SSDLC* yang lain. Dengan begitu dapat dibandingkan metode mana yang memberikan hasil akhir terbaik ketika diimplementasikan dalam pembuatan aplikasi *mobile secure chat*.

.

5.2 Saran

Penelitian berikutnya dengan melanjutkan ancaman yang belum bisa dimitigasi dan keamanan lebih lanjut dengan metoda kriptografi yang lebih baik.

DAFTAR PUSTAKA

Albarqi, A., Alzaid, E., Ghamdi, F. al, Asiri, S., & Kar, J. (2015). Public Key Infrastructure: A Survey. *Journal of Information Security*, *06*(01), 31–37. https://doi.org/10.4236/jis.2015.61004

Ali, A. H., & Sagheer, A. M. (2017). *Design of a secure android chatting application using end to end encryption*. *2*(1). www.jseis.org

Ali, R. M., & Alsaad, S. N. (2020). Instant messaging security and privacy secure instant messenger design. *IOP Conference Series: Materials Science and Engineering*, *881*(1). https://doi.org/10.1088/1757-899X/881/1/012117

Aminudin, A., Helmi, A. F., & Arifianto, S. (2018). Analisa Kombinasi Algoritma Merkle-Hellman Knapscak dan Logaritma Diskrit pada Aplikasi Chat. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(3), 325. https://doi.org/10.25126/jtiik.201853844

Ansari, M. T. J., Pandey, D., & Alenezi, M. (2022). STORE: Security Threat Oriented Requirements Engineering Methodology. *Journal of King Saud University - Computer and Information Sciences*, *34*(2), 191–203. https://doi.org/10.1016/j.jksuci.2018.12.005

Asbullah, M. A., Rezal, M., Ariffin, K., Asbullah, M. A., & Ariffin, M. R. K. (2016). Design of Rabin-Like Cryptosystem without Decryption Failure. In *Malaysian Journal of Mathematical Sciences* (Vol. 10).

Carita, S. S., & Wahyuni, E. S. (2022). Modifikasi Tanda Tangan Digital Pada Skema Esign Berbasis Kurva Eliptik. *Jurnal Ilmiah SINUS*, *20*(2), 33. https://doi.org/10.30646/sinus.v20i2.625

checkmarx.com_glossary_a-secure-sdlc-with-static-source-code-analysis-tools. (n.d.). Retrieved December 1, 2022, from https://checkmarx.com/glossary/a-secure-sdlc-with-static-source-code-analysis-tools/

Conklin, L., & Robinson, G. (2017). *CODE REVIEW GUIDE RELEASE*. OWASP. https://www.owasp.org

Ebert, F., Castor, F., Novielli, N., & Serebrenik, A. (2021). An exploratory study on confusion in code reviews. *Empirical Software Engineering*, *26*(1). https://doi.org/10.1007/s10664-020-09909-5

Fujdiak, R., Mlynek, P., Mrnustik, P., Barabas, M., Blazek, P., Borcik, F., & Misurec, J. (2019). Managing the Secure Software Development. *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 1–4. https://doi.org/10.1109/NTMS.2019.8763845

H. Ali, A., & Sagheer, A. M. (2017). Design of an Android Application for Secure Chatting. *International Journal of Computer Network and Information Security*, *9*(2), 29–35. https://doi.org/10.5815/ijcnis.2017.02.04

Hajrić, A., Smaka, T., Barakovic, S., & Baraković-Husić, J. (2020). Methods, Methodologies, and Tools for Threat Modeling with Case Study. *Telfor Journal*, *12*, 56–61. https://doi.org/10.5937/telfor2001056H

Harn, L., & Ren, J. (2011). Generalized digital certificate for user authentication and key establishment for secure communications. *IEEE Transactions on Wireless Communications*, *10*(7), 2372–2379. https://doi.org/10.1109/TWC.2011.042211.101913

Hema, V., Thota, S., Naresh Kumar, S., Padmaja, C., Rama Krishna, C. B., & Mahender, K. (2020). Scrum: An Effective Software Development Agile Tool. *IOP Conference Series: Materials Science and Engineering*, *981*(2). https://doi.org/10.1088/1757-899X/981/2/022060

Hevianto Saputro, T., Hidayati, N., & Ujianto, E. (2020). SURVEI TENTANG ALGORITMA KRIPTOGRAFI ASIMETRIS. *Jurnal Informatika Polinema*, *6*, 67–72. https://doi.org/10.33795/jip.v6i2.345

Hussain, S., Kamal, A., Ahmad, S., Rasool, G., & Iqbal, S. (2014). *THREAT MODELLING METHODOLOGIES: A SURVEY*. *26*, 1607–1609.

Kuliya, M., & Abubakar, H. (2020). *Secured Chatting System Using Cryptography*. www.ijcrt.org

Kusumaningrum, A., Wijayanto, H., & Raharja, B. D. (2022). Pengukuran Tingkat Kesadaran Keamanan Siber di Kalangan Mahasiswa saat Study From Home dengan Multiple Criteria Decision Analysis (MCDA). *Jurnal Ilmiah SINUS*, *20*(1), 69. https://doi.org/10.30646/sinus.v20i1.586

Laksono, A. C., & Prayudi, Y. (2021). Threat Modeling Menggunakan Pendekatan STRIDE dan DREAD untuk Mengetahui Risiko dan Mitigasi Keamanan pada Sistem Informasi Akademik. *JUSTINDO (Jurnal Sistem & Teknologi Informasi Indonesia)*, *6*(1).

Mallouli, F., Hellal, A., Sharief Saeed, N., & Abdulraheem Alzahrani, F. (2019). A Survey on Cryptography: Comparative Study between RSA vs ECC Algorithms, and RSA vs El-Gamal Algorithms. *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/ 2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, 173–176. https://doi.org/10.1109/CSCloud/EdgeCom.2019.00022

Mathas, C.-M., Vassilakis, C., Kolokotronis, N., Zarakovitis, C. C., & Kourtis, M.-A. (2021). On the Design of IoT Security: Analysis of Software Vulnerabilities for Smart Grids. *Energies*, *14*(10). https://doi.org/10.3390/en14102818

McGraw, G. (2006). Software Security: Building Security In. *2006 17th International Symposium on Software Reliability Engineering*, 6. https://doi.org/10.1109/ISSRE.2006.43

MM, R., T, A., & A, R. (2016). Development of Cryptography-Based Secure Messaging System. *Journal of Telecommunications System & Management*, *05*(03). https://doi.org/10.4172/2167-0919.1000142

Mulya, M., Rismawati, N., & Trisanto, D. (2021). Analisis Dan Perancangan Simulasi Algoritma Paillier Cryptosystem Pada Pesan Text Dengan Presentation Format Binary, Octal, Hexadecimal dan Base64. *Faktor Exacta*, *13*, 208. https://doi.org/10.30998/faktorexacta.v13i4.7429

Nayak, S., Das, S., Das, S., Sarker, S., Sarker, P., Dey, A., Sinha, A., Saha, J., Banerjee, A., Saha, N., Chowdhury, S., Chowdhury, D., Pradhan, P., Banerjee, A., Ali, S. A., Saha, A., Dey, R., & Dey, S. (2017). An application for end to end secure messaging service on Android supported device. *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 290–294. https://doi.org/10.1109/IEMCON.2017.8117222

Policy, U. S., & Bodansky, D. (2015). *LEGAL OPTIONS FOR U.S. ACCEPTANCE OF A NEW CLIMATE CHANGE AGREEMENT*. https://ssrn.com/abstract=2652008

Prabowo, I. A., YS, W. L., & Wahyudi, W. (2022). The Application of the Blowfish Algorithm and the Least Significant Bit Method for Securing Student Transcripts. *Jurnal Ilmiah SINUS*, *20*(2), 87. https://doi.org/10.30646/sinus.v20i2.622

Saleh, F., & El-Attar, M. (2015). A Scientific Evaluation of the Misuse Case Diagrams Visual Syntax. *Information and Software Technology*, *66*. https://doi.org/10.1016/j.infsof.2015.05.002

Setyadi, G., & Kusumawati, Y. (n.d.). Risk Mitigation Asset And Information Technology Component Framework Based On OCTAVE And FMEA At The Dian Nuswantoro University. *Journal of Information System*.

Sharma, S., & Kapoor, V. (2017). A Novel Approach for Improving Security by Digital Signature and Image Steganography. *International Journal of Computer Applications*, *171*(8), 7–11. https://doi.org/10.5120/ijca2017915145

Shen, Y. (2020). *Research on Internet Information Security in the Big Data Era*. *218*. https://doi.org/10.1051/e3sconf/202021804008

Subari, A., & Iman Satoto, K. (n.d.). *DESAIN WEB SECURE LOGIN DENGAN ALGORITMA ENKRIPSI SIMETRI RC-6*.

Sugiantoro, B., Anshari, M., & Sudrajat, D. (2020). Developing Framework for Web Based e-Commerce: Secure-SDLC. *Journal of Physics: Conference Series*, *1566*(1). https://doi.org/10.1088/1742-6596/1566/1/012020

Sulaksono, D. H., Prabiantissa, C. N., Yuliastuti, G. E., Taqwa, A. R., Informatika, T., Elektro, T., Informasi, T., Adhi, T., & Surabaya, T. (2021). Implementasi Kriptografi dengan Metode Elliptic Curve Cryptography (ECC) untuk Aplikasi Chatting Berbasis Android. *Seminar Nasional Sains Dan Teknologi Terapan*, 570.

Tung, Y.-H., Lo, S.-C., Shih, J.-F., & Lin, H.-F. (2016). An integrated security testing framework for Secure Software Development Life Cycle. *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 1–4. https://doi.org/10.1109/APNOMS.2016.7737238

Wang, Z., Zuo, M., Yao, S., & Aihemaiti, N. (2020). Internet of Vehicles Based on TrustZone and Optimized RSA. *IOP Conference Series: Materials Science and Engineering*, *782*(2). https://doi.org/10.1088/1757-899X/782/2/022073

Zebua, T., Kristianto Hondro, R., Ndruru, E., Stiekom, A., Utara, S., Budi, S., & Medan, D. (2018). Message Security on Chat App based on Massey Omura Algorithm. *International Journal Of Information System & Technology*, *1*(2), 16–23.