ANALISIS APLIKASI METODE KEAMANAN IOT PADA PERANGKAT SURVEILLANCE CAMERA

Disusun Oleh:

N a m a : Rauf Endro Widagdo

NIM : 16523057

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2020

HALAMAN PENGESAHAN DOSEN PEMBIMBING

ANALISIS APLIKASI METODE KEAMANAN IOT PADA PERANGKAT SURVEILLANCE CAMERA

TUGAS AKHIR

Disusun Oleh:

N a m a : Rauf Endro Widagdo
NIM : 16523057

Yogyakarta, 12 Juli 2020

Pembimbing,

(Fietyata Yudha, S.Kom., M.Kom.)

Pembimbing,

(Fayruz Rahma, S.T., M.Eng.)

HALAMAN PENGESAHAN DOSEN PENGUJI

ANALISIS APLIKASI METODE KEAMANAN IOT PADA PERANGKAT SURVEILLANCE CAMERA

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 20 Agustus 2020

Tim Penguji

Fietyata Yudha, S.Kom., M.Kom. _____

Anggota 1

Dr. Syarif Hidayat, S.Kom., M.I.T. _____

Anggota 2

Irving Vitra Paputungan, S.T., M.Sc., Ph.D. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama	:	Rauf Endro Widagdo
NIM	:	16523057

Tugas akhir dengan judul:

ANALISIS APLIKASI METODE KEAMANAN IOT PADA PERANGKAT SURVEILLANCE CAMERA

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 10 Juli 2020

(Rauf Endro Widagdo)

HALAMAN PERSEMBAHAN

Alhamdulillah dan puji syukur penulis panjatkan ke hadirat Allah SWT karena atas segala rahmat dan hidayah-Nya dapat menyelesaikan Tugas Akhir Universitas Islam Indonesia ini dengan baik. Tugas Akhir ini saya persembahkan kepada:

1. Orang tua saya yang saya sayangi dan cintai, yang tak kenal lelah mendoakan dan memberi dukungan kepada saya sehingga dapat mengenyam Pendidikan di Perguruan Tinggi.
2. Keluarga saya yang tidak pernah lupa selalu memberi semangat dan dukungan
3. Bapak Fietyata Yudha, S.Kom., M.Kom. dan Ibu Fayruz Rahma, M.Eng. sebagai dosen pembimbing Tugas Akhir yang baik hati meluangkan waktu dan tenaga untuk membimbing saya untuk menyelesaikan Tugas Akhir ini.
4. Jurusan Informatika yang menjadi tempat untuk menuntut ilmu selama di Perguruan Tinggi
5. Mas Ikhwan Alfath Nurul Fathony sebagai pemilik penelitian sebelumnya yang telah memberi saya ilmu dan dasar dari Tugas Akhir ini.
6. Seluruh pihak yang tidak dapat saya sebutkan satu persatu, terima kasih atas segala dukungan. Saya ucapkan terima kasih

HALAMAN MOTO

“It’s fine to celebrate success but it is more important to heed the lesson of failure”
– Bill Gates

KATA PENGANTAR

Assalaamu'alaikum warahmatullaahi wabarakatuh

Alhamdulillahi Robbil 'Alamin, segala puji dan syukur penulis panjatkan kepada Allah SWT karena atas segala rahmat dan hidayah-Nya dapat menyelesaikan Tugas Akhir ini dengan baik. Shalawat serta salam senantiasa dicurahkan kepada junjungan Nabi Agung Muhammad SAW beserta keluarga serta sahabatnya.

Laporan tugas akhir ini dibuat dan disusun dalam rangka mengakhiri masa Pendidikan jenjang Strata 1 Program Studi Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia. Laporan ini dapat terselesaikan secara sempurna berkat do'a dan dukungan dari berbagai kalangan. Oleh karena itu, dengan berbahagia penulis mengucapkan terima kasih kepada:

1. Orang tua saya yang saya sayangi dan cintai, yang tak kenal lelah mendoakan dan memberi dukungan kepada saya sehingga dapat mengenyam Pendidikan di Perguruan Tinggi.
2. Keluarga saya yang tidak pernah lupa selalu memberi semangat dan dukungan
3. Bapak Fietyata Yudha, S.Kom., M.Kom. dan Ibu Fayruz Rahma, M.Eng. sebagai dosen pembimbing Tugas Akhir yang baik hati meluangkan waktu dan tenaga untuk membimbing saya untuk menyelesaikan Tugas Akhir ini.
4. Jurusan Informatika yang menjadi tempat untuk menuntut ilmu selama di Perguruan Tinggi
5. Mas Ikhwan Alfath Nurul Fathony sebagai pemilik penelitian sebelumnya yang telah memberi saya ilmu dan dasar dari Tugas Akhir ini.
6. Seluruh pihak yang tidak dapat saya sebutkan satu persatu, terima kasih atas segala dukungan. Saya ucapkan terima kasih

Penulis menyadari bahwa laporan ini masih jauh dari kata sempurna dan masih banyak kesalahan serta kekurangan di dalamnya. Oleh karena itu penulis memohon maaf sebesar-besarnya. Penulis mengharapkan kritik serta saran dari pembaca untuk laporan ini agar ke depannya menjadi lebih baik.

Wassalaamu'alaikum warahmatullaahi wabarakatuh

Yogyakarta, 10 Juli 2020

(Rauf Endro Widagdo)

SARI

Internet of Things (IoT) merupakan teknologi baru yang menghubungkan mesin atau perangkat dengan jaringan internet agar dapat saling bertukar informasi sehingga dapat berjalan sesuai fungsinya. Adanya IoT dapat membantu aktivitas dalam berbagai bidang seperti bidang pertanian, energi, kesehatan, transportasi, dan lain lain. Karena IoT banyak digunakan di berbagai bidang, diperlukannya perhatian pada pengamanan perangkat IoT.

Perangkat IoT yang ada saat ini dapat dipasangkan sistem keamanan. Contohnya *Surveillance Camera* yang memiliki sistem keamanan pada sistem login pada web-nya dan keamanan pada *file*nya. Lebih tepatnya sistem keamanan dengan menggunakan fungsi *hash* algoritma pada sistem login dan enkripsi pada *file*.

Penulis akan melakukan analisis sistem keamanan login pada Surveillance Camera dengan melakukan pengujian untuk setiap fungsi algoritma *hash*. Pada pengujian pengenkripsian *file* sangat diperlukan agar dapat menjamin jika terjadi kebocoran data tidak dapat diketahui isinya. Dengan menganalisis pengamanan pada sistem keamanan perangkat IoT yang ini akan memperoleh referensi untuk menggunakan sistem keamanan tersebut

Kata kunci: Internet of Things, Sistem Keamanan, Perangkat IoT, *Surveillance Camera*

GLOSARIUM

Decrypt	Proses mengembalikan data yang terenkripsi ke data aslinya
Encrypt	Proses merubah data asli menjadi data rahasia
Hash	Data string yang diubah menjadi string acak
Internet of Things	Konsep teknologi baru yang menghubungkan perangkat dengan internet agar dapat bertukar informasi
Perangkat IoT	Perangkat yang dapat melakukan pekerjaan tertentu yang dapat terhubung ke internet agar dapat saling bertukar informasi.
Sistem Keamanan	Sistem yang dapat mengamankan sesuatu
Surveillance Camera	Kamera pengintai

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Internet of Things dapat dianggap keluarga teknologi yang bertujuan untuk membuat segala jenis objek atau perangkat yang terhubung ke internet (Tedeschi et al., 2017). Internet of Things (IoT) merupakan suatu hal yang baru untuk zaman sekarang. IoT bekerja dengan membuat mesin satu dan mesin yang lainnya saling berkomunikasi menggunakan jaringan internet, sehingga membuat mesin-mesin tersebut dapat diakses dan dikendalikan melalui jaringan internet di berbagai tempat dan waktu. Karena kemudahan tersebut banyak bidang yang telah beralih menggunakan IoT, contohnya penerapan pada bidang pertanian, energi, lingkungan hidup, otomasi rumah, medis dan kesehatan, dan transportasi.

Sisi positif dari penggunaan IoT yaitu dapat melakukan pekerjaan dari berbagai tempat dan waktu dan dapat dilakukan secara otomasi. Teknologi IoT juga tidak luput dari penyalahgunaan, seperti penyalahgunaan untuk memanfaatkan perangkat IoT untuk tindak kejahatan. Penyalahgunaan tersebut justru dapat membuat perangkat IoT beralih menjadi suatu hal yang dapat merusak. Akibat yang ditimbulkan dari penyalahgunaan tersebut tergantung pada bidang perangkat IoT-nya. Karena perangkat IoT saling berkomunikasi dan bertukar data menggunakan jaringan internet, rentan terjadi serangan pada komunikasi jaringan adalah Man-In-The-Middle (MITM) (Lutfi et al., 2018).

Teknologi yang menerapkan konsep IoT salah satunya sistem kamera pengawas atau bisa disebut sistem kamera pengawas (*Surveillance Camera*) berbasis IoT. *Surveillance Camera* berbasis IoT merupakan kamera pengawas dengan penerapan konsep teknologi IoT yang dapat melakukan pemantauan dan pengendalian melalui jaringan internet, sehingga Kamera Pengawas tersebut dapat diakses dan dikendalikan dari berbagai tempat dan waktu.

Surveillance Camera berbasis IoT juga dapat mengalami kejahatan yang dialami pada teknologi IoT. *Surveillance Camera* dapat mengalami penyalahgunaan seperti bocornya akses ke pihak yang tidak memiliki hak akses dan menggunakan kamera tersebut sebagai alat pemantau untuk tindak kejahatan. Penyerang dapat menjadi anonym dan dapat memperoleh *real-time video*, rekaman yang diarsipkan, email, FTP, kredensial lain, dan akses ke kontrol sumber daya sistem (Cusack & Tian, 2017). Karena penyalahgunaan tersebut *Surveillance Camera* yang awalnya

sebagai alat keamanan untuk mengawasi jika terjadi tindak kejahatan menjadi alat untuk melakukan tindak kejahatan. Serangan MITM juga dapat terjadi pada *Surveillance Camera* berbasis IoT pada saat mengirimkan data dari *Surveillance Camera* tersebut ke perangkat lain. Untuk protokol transmisi data, metodologi serangan yang layak disoroti sebagai serangan man-in-the-middle (MITM), di mana pihak ketiga mampu mencegat umpan kamera Wifi dan menggunakannya untuk tujuan mereka sendiri (Coole et al., 2012). Karena serangan MITM tersebut data yang dikirimkan dapat bocor atau dapat diketahui isinya oleh pihak yang seharusnya tidak mempunyai hak.

Mengingat keamanan pada teknologi IoT perlu untuk diperhatikan khususnya pada permasalahan *Surveillance Camera* di atas, penerapan *hash* pada password dan pengenkripsian pada data atau *file* perlu dilakukan analisis. Penerapan *hash* sebagai password memiliki berbagai macam fungsi *hash* yang dapat digunakan. Begitu juga dengan pengenkripsian *file* dapat menggunakan berbagai macam algoritma. Penulis ingin menganalisis sistem keamanan tersebut yang diterapkan pada sistem *Surveillance Camera* yang berbasis IoT yang sebelumnya telah dibuat dan menambahkan serta melakukan analisis terhadap *file* yang dienkripsi. Pada penelitian ini akan dilakukan analisis terhadap penerapan fungsi *hash* dan algoritma enkripsi. Penulis ingin menganalisis kelebihan dan kekurangan MD5 dan SHA-256 sebagai fungsi *hash* dan menganalisis hasil *file* yang dienkripsi menggunakan AES256-CBC. Dengan permasalahan di atas, penulis mengharapkan agar memperoleh hasil kelebihan dan kekurangan pada fungsi *hash* MD5 dan SHA-256 dan hasil *file* yang dienkripsi menggunakan AES256-CBC yang diterapkan pada sistem Surveillance Camera.

1.2 Rumusan Masalah

Rumusan masalah yang dapat diambil berdasarkan latar belakang tersebut adalah Menganalisis metode keamanan yang diterapkan pada Surveillance Camera yang memiliki permasalahan pada file rekaman video yang dihasilkan tidak dapat menjamin integritasnya dan algoritma hash yang digunakan memiliki jumlah karakter yang sedikit dalam melindungi kata sandi didalamnya, sehingga hasil dari analisis metode keamanan tersebut dapat efektif untuk Surveillance Camera.

1.3 Batasan Masalah

Agar penelitian ini tetap terarah dan dilakukan dengan semestinya, ditentukan beberapa batasan masalah yang terdiri dari:

a. Fungsi *hash* yang digunakan untuk dianalisis yaitu MD5 dan SHA-256.
b. *File* yang akan dienkripsi berformat MP4 dan AVI dan menggunakan algoritma enkripsi yaitu AES256-CBC.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah Menganalisis fungsi *hash* dan hasil *file* enkripsi dengan melakukan uji coba terhadap masing-masing fungsi *hash* dan *file* yang dienkripsi. Uji coba yang dilakukan untuk membandingkan kelebihan dan kekurangan masing-masing. Dari uji coba dapat diperoleh data dari hasil analisis tersebut.

1.5 Manfaat Penelitian

Manfaat yang diperoleh dari penelitian ini diantaranya:

1. Hasil dari analisis tersebut agar peneliti dapat mengetahui hasil dari penerapan sistem keamanan tersebut pada sistem *Surveillance Camera* berbasis IoT.
2. Dapat dijadikan sebagai referensi bagi masyarakat untuk memilih sistem keamanan yang baik.

1.6 Langkah Penyelesaian

Untuk menjawab rumusan masalah di atas, ada beberapa tahapan yang akan dilakukan. Tahapan-tahapan tersebut adalah:

1. Studi Literatur

Tahapan pertama dalam menjawab pertanyaan ini adalah dengan mengkaji literatur-literatur mengenai Surveillance Camera dan kejahatan pada Surveillance Camera.

2. Pengumpulan Data

Tahapan kedua dalam menjawab pertanyaan ini adalah dengan mengumpulkan source-code sistem Surveillance Camera dan algoritma yang akan digunakan

3. Perancangan

Tahapan ketiga dalam menjawab pertanyaan ini adalah dengan melakukan perancangan antarmuka dan sistem yang akan mempermudah dalam menganalisis pada penelitian ini.

4. Pengujian

Pada tahapan ini dilakukan pengujian agar memperoleh data dari hasil uji yang dapat dianalisis yang akan menghasilkan data hasil analisis.

BAB II
LANDASAN TEORI

2.1 Kajian Pustaka

Pada penelitian yang dilakukan didasari oleh penelitian sebelumnya yang sudah membahas mengenai Surveillance Camera dan Enkripsi *file* dengan AES. Berikut merupakan tabel tentang penelitian sebelumnya yang dapat dilihat pada Tabel 2.1

Tabel 2.1 Penelitian Sebelumnya

No.	Judul	Penulis	Pencapaian
1.	Sistem Kamera Pengawas Dengan Menggunakan Raspberry Pi Disertai Motion Detection dan Auto Backup Cloud (Google Drive)	Ikhwan Alfath Nurul Fathony	- Sistem kamera pengawas ini dapat mengakses kamera usb (webcam) dan kamera internet protocol (IP) dengan menggunakan Raspberry Pi - Sistem ini dapat membuat *file* gambar dengan format .jpg berdasarkan pergerakan yang terdeteksi oleh kamera (motion detection). - Hasil dari kamera tersebut dapat dicadangkan secara otomisasi ke dalam akun Google drive.

2.	Enkripsi dan Dekripsi Dengan Algoritma AES 256 untuk semua jenis *file*	Voni Yuniati, Gani Indriyanta, Antonius Rahmat C	- *File* dekripsi dapat kembali seperti ekstensi *file* sumber karena pada saat proses enkripsi ditambahkan header untuk menyimpan informasi *file* sumber. - Dapat membuktikan bahwa *file* yang telah dienkripsi merupakan *file* yang sama dengan sumber. - Waktu yang diperlukan pada proses enkripsi tidak sama dengan proses dekripsi karena adanya pemakaian *resource computer*. - Saat terjadi proses save pada *file* .aes dengan maupun tanpa mengganti informasi ekstensi *file* sumber maka pada saat dilakukan proses dekripsi, terdapat isi *file* yang tidak kembali seperti *file* sumber. Hal ini disebabkan *file* .aes tidak lagi berbasis *file* hasil proses enkripsi melainkan akan menjadi *file* .aes berupa *file* teks sehingga akan terjadi perubahan header *file* dalam *file* .aes.

2.2 Kamera Pengawas

Kamera pengawas adalah jenis strategi pencegahan kejahatan situasional yang berpusat pada tingkat pengawas formal ditingkatkan dalam area target. Strategi tersebut berfokus pada pencegahan kejahatan dengan mengurangi jumlah peluang criminal dan meningkatkan risiko pelanggaran melalui modifikasi lingkungan fisik (Piza et al., 2019). Kamera pengawas saat ini banyak digunakan berbagai kalangan, mulai dari pengusaha sampai perseorangan dengan kebutuhan pribadi. Biasanya kamera pengawas digunakan untuk mengawasi aktivitas usaha, mengawasi aktivitas di tempat umum, bahkan mengawasi bayi, dan lain-lain. Kamera pengawas

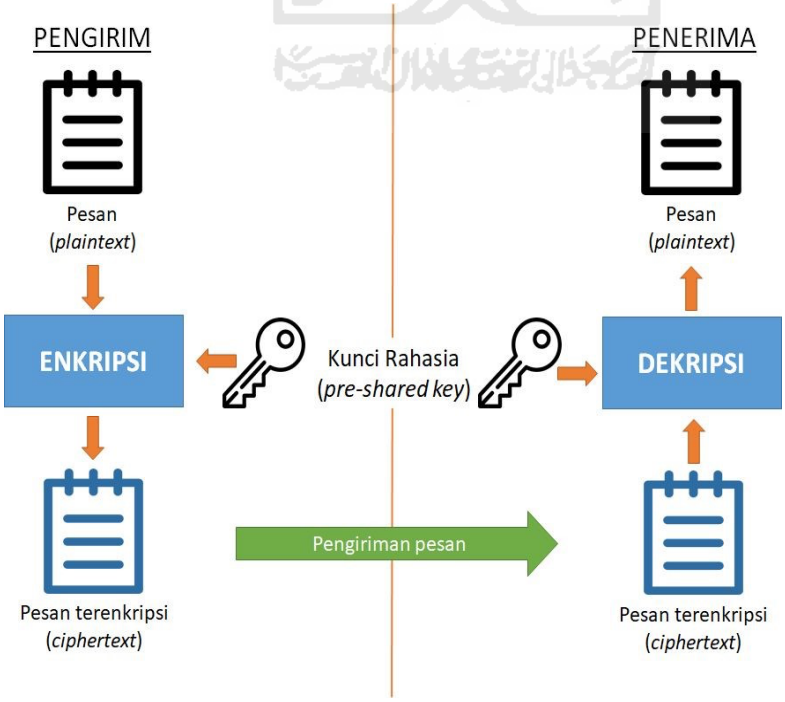
saat ini menjadi alat yang diperlukan untuk perlindungan karena dapat mengawasi tempat yang sulit dijangkau dan waktu tertentu.

2.4 Surveillance Camera Berbasis IoT

Surveillance Camera Berbasis IoT merupakan kamera pengawas dengan menerapkan konsep teknologi IoT. Surveillance camera berbasis IoT dapat menolong kota dan rumah-rumah agar lebih aman dengan fasilitas pemantauan jarak jauh pada ruang publik dalam berbagai waktu (Razalli et al., 2019). Karena itu kamera pengawas tersebut dapat diakses dan dikendalikan dari berbagai tempat dan waktu.

2.5 Enkripsi

Enkripsi adalah proses yang mengubah plaintext menjadi ciphertext yang bertujuan untuk mengamankan data atau isi pesan yang bersifat rahasia agar tidak disadap oleh kriptanalis (Yuniati et al., 2011). Hanya organisasi dan individu tertentu yang mengetahui dan memiliki proses enkripsi tersebut yang dapat membaca atau mengetahui informasi tersebut. Tujuan dari suatu informasi dienkripsi adalah untuk menjaga informasi tersebut tidak diketahui oleh suatu organisasi ataupun individu yang tidak diinginkan dan untuk menjaga integritas dari informasi tersebut. Berikut merupakan gambar yang menampilkan salah satu proses enkripsi yang ditampilkan pada Gambar 2.1.



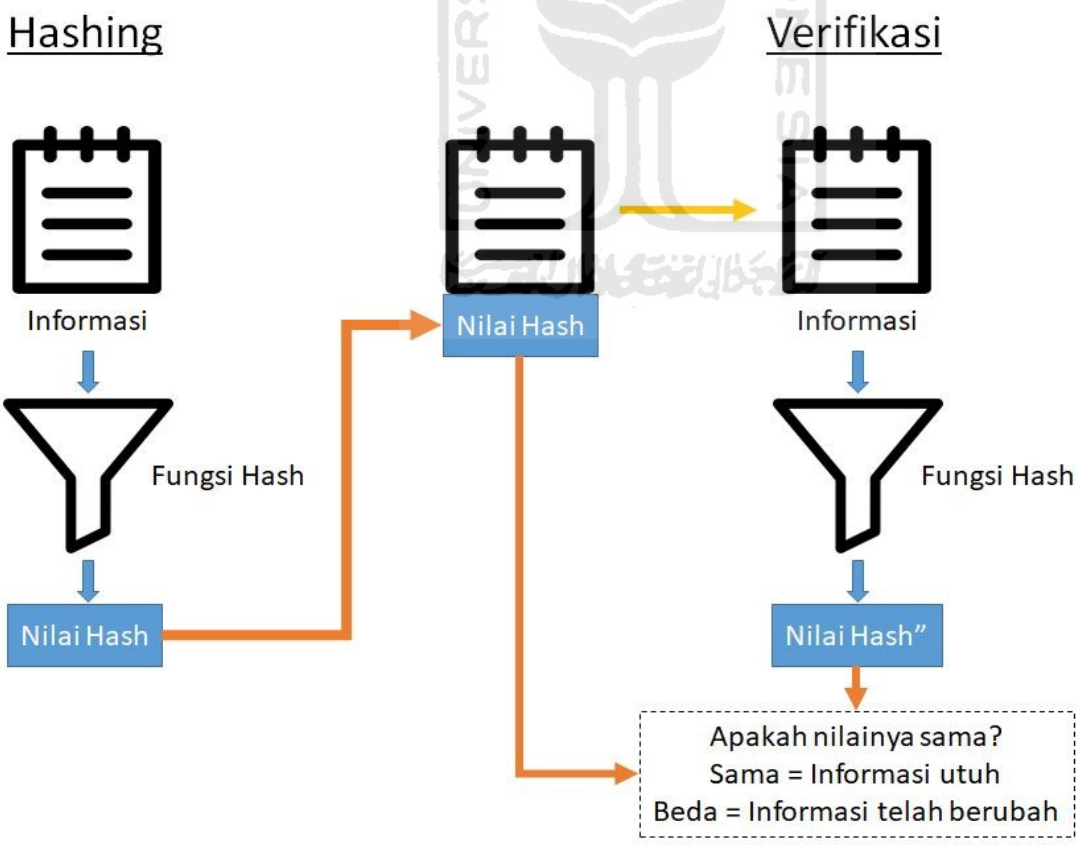
Gambar 2.1 Contoh salah satu proses enkripsi (Pratama & Rahma, 2019)

2.6 AES

AES merupakan algoritma kunci simetris, pengirim dan penerima menggunakan satu kunci untuk enkripsi dan dekripsi (Lutfi et al., 2018). AES merupakan kunci simetris yang terdiri atas 3 blok chipper yaitu AES-128, AES-192, AES-256. AES mempunyai panjang blok data 128 bit dan Panjang kunci sampai 128, 192, 256 bit.

2.7 *Hash*

Fungsi *Hash* adalah fungsi yang menerima masukan string yang panjangnya sembarang dan mengkonversinya menjadi string keluaran yang panjangnya tetap atau fixed (Angga, 2011). *Hash* biasanya digunakan sebagai metode menyimpan data string dalam bentuk nilai *hash* di database agar string tersebut tidak diketahui isinya. Nilai *hash* yang biasa disimpan dan digunakan di database adalah password untuk memverifikasi identitas pada sistem. Berikut merupakan cara kerja *hash* yang ditampilkan pada Gambar 2.2.



Gambar 2.2 Cara kerja *hash* (Pratama & Rahma, 2019)

2.7.1 *Secure Hashing Algorithm-256 (SHA-256)*

SHA-256 adalah algoritma yang diterbitkan oleh NIST pada tahun 2002 bersamaan dengan SHA-384 dan SHA-512 (Dadda et al., 2004). Pengembangan proyek SHA 2 menghasilkan 4 SHA yang dikembangkan dan dimasukkan ke dalam kategori SHA 2 yaitu SHA-224, SHA-384, SHA-512/224, SHA-512/256. SHA-512/256 ini merupakan SHA 2 dengan panjang *hash* 256 bit yang sering dikenali sebagai SHA-256.

2.7.2 MD5

MD5 (*Message-Digest algorithm* 5) adalah fungsi *hash* kriptografi yang paling banyak digunakan saat ini. MD5 dirancang pada tahun 1992 sebagai peningkatan MD4 (Wang & Yu, 2005). MD5 adalah fungsi *hash* kriptografi yang memiliki *hash* value 128 bit yang menenkripsi tulisan atau menjadi digit hex.

2.8 *Raspberry Pi*

Raspberry Pi adalah komputer single board yang dikembangkan oleh *Raspberry Pi* Foundation dan berukuran seperti kartu kredit (Bhaskar, 2015). *Raspberry Pi* dirancang sebagai pendidikan dan terinspirasi oleh keberhasilan BBC Micro untuk mengajar pemrograman komputer hingga satu generasi (Harrington, 2015). Sesuai perkembangan jaman, *Raspberry Pi* selalu berkembang dan mengeluarkan tipe-tipe baru yang memiliki spesifikasi yang diperbarui. Mulai dari tipe 1A+, 1B+, 2B, 3B, 3B+, dan yang terbaru *Raspberry Pi* 4 yang memiliki RAM hingga 4GB LPDDR4. Untuk penyimpanan, *Raspberry Pi* tidak menggunakan cakram keras atau *Solid State Drive* melainkan menggunakan SD Card atau kartu penyimpanan untuk menjalankan dan menyimpan sistem dan sebagai media penyimpanan. Berikut merupakan gambar *Raspberry Pi* yang digunakan penulis yang ditampilkan pada Gambar 2.3.

Gambar 2.3 *Raspberry Pi* 3 model B

BAB III
METODE PENELITIAN

3.1 Alur Penelitian

Alur penelitian dalam penulisan penelitian ini menjelaskan mengenai tahapan atau prosedur untuk menganalisis pengamanan IoT pada Surveillance Camera. Tahapan pertama yang dilakukan adalah studi literatur. Studi literatur dilakukan untuk mempelajari dan memahami tentang kejahatan pada Surveillance Camera. Tahapan selanjutnya melakukan pengumpulan data. Pengumpulan data tersebut meliputi sistem Surveillance Camera yang akan digunakan dalam proses menganalisis. Setelah melakukan pengumpulan data, tahapan selanjutnya adalah perancangan. Tahapan perancangan ini meliputi perancangan antarmuka dan sistem, diagram alur, use case diagram, dan perancangan basis data. Setelah tahapan perancangan dibuat, tahapan selanjutnya adalah pengujian. Pada tahapan ini dilakukan pengujian agar diperoleh hasil yang dapat dianalisis. Berikut merupakan diagram alur dari tahapan melakukan penelitian yang dapat dilihat pada Gambar 3.1.

Gambar 3.1 Tahapan melakukan penelitian

3.2 Studi Literatur

Pada bagian ini dilakukan proses mencari penelitian tentang kejahatan pada Surveillance Camera. Dari proses tersebut akan diketahui kejahatan yang mungkin terjadi pada Surveillance Camera. Dari kejahatan tersebut penulis dapat menyarankan solusi untuk kejahatan tersebut.

3.3 Pengumpulan Data

Pada tahapan ini memperbanyak pengetahuan mengenai Surveillance Camera, algoritma *hash*, dan algoritma untuk pengenkripsian *file* yang berasal dari jurnal, website, buku, artikel, paper, atau pun skripsi penelitian sebelumnya. Untuk literatur tersebut dicari menggunakan internet.

Dalam tahap ini peneliti juga mencari source-code untuk perangkat sistem kamera pengawas atau sistem Surveillance Camera yang digunakan sebagai dasar sistem yang akan dianalisis. Source-code untuk sistem tersebut menggunakan penelitian sebelumnya yang diperoleh dari studi

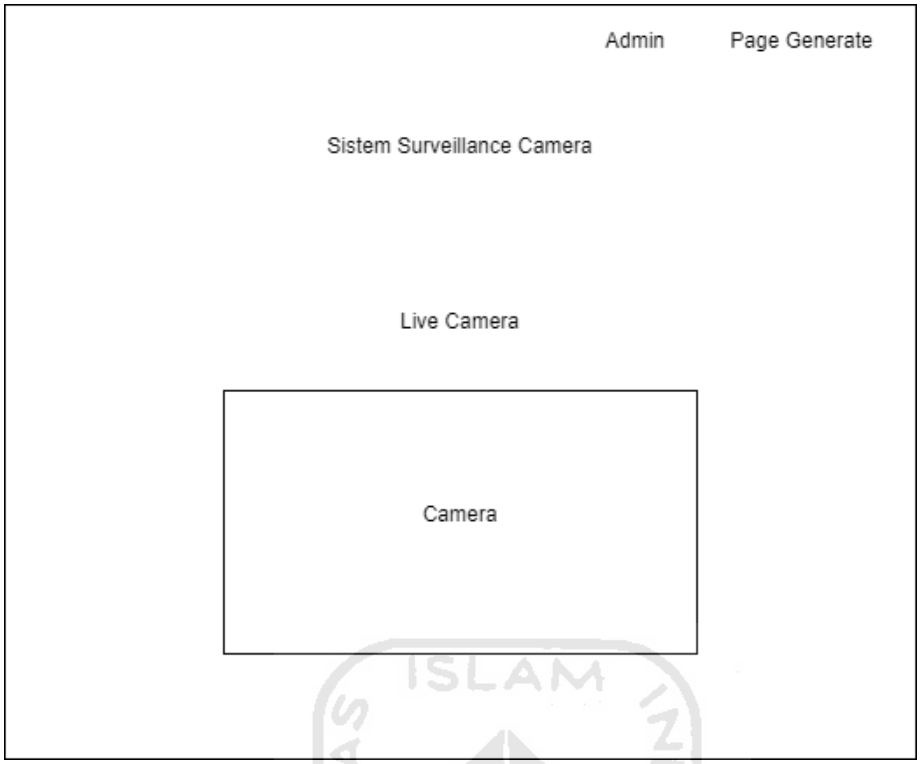
literatur. Penulis juga mendapatkan algoritma yang digunakan untuk mengenkripsi *file* menggunakan AES dan fungsi *hash* yang dapat digunakan yaitu MD5 dan SHA-256.

Penelitian berfokus pada analisis penggunaan *hash* pada sistem dan pengenkripsian pada *file*. *File* yang digunakan menggunakan 2 jenis *file* yaitu AVI dan MP4 yang diambil dari hasil rekaman pada Surveillance Camera. Kemudian kedua *file* tersebut dienkripsi menggunakan algoritma AES256-CBC.

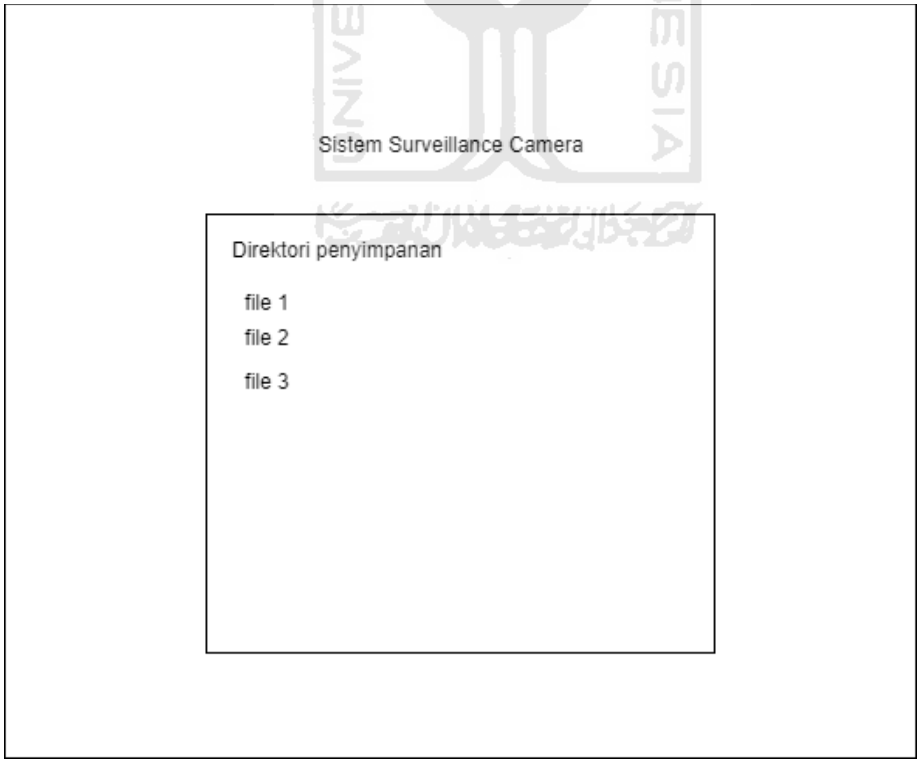
3.4 Perancangan

3.4.1 Perancangan Antarmuka

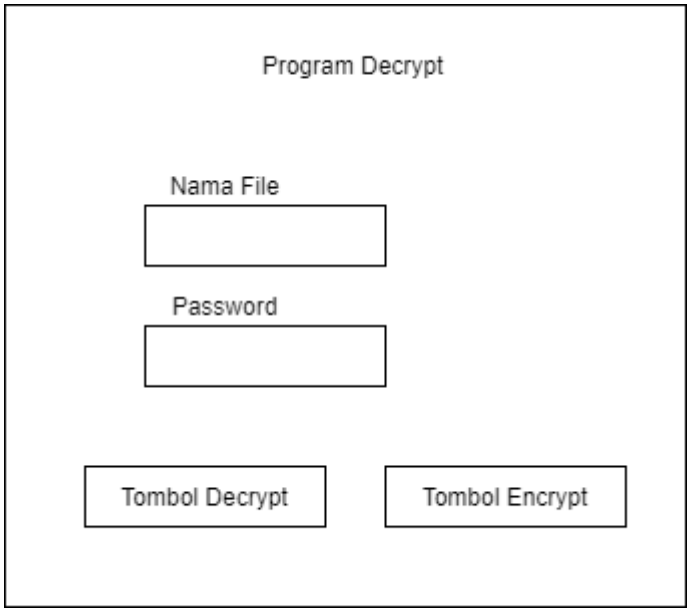
Pada tahapan ini penulis melakukan perencanaan terhadap antarmuka yang digunakan untuk membantu dalam menganalisis Surveillance Camera. Untuk perancangan antarmuka dibagi menjadi 3 yaitu antarmuka untuk menganalisis fungsi *hash*, antarmuka menyimpan *file* enkripsi, dan antarmuka program untuk melakukan *decrypt* pada *file* yang dienkripsi. Pada antarmuka analisis fungsi *hash*, sistem setelah halaman login atau beranda admin ditambahkan fungsi php page generate untuk membantu mengukur waktu login yang ditempuh pada fungsi *hash*. Satuan waktu yang digunakan untuk mengukur adalah milidetik. Alasan menggunakan milidetik karena waktu yang ditempuh tidak melebihi dari milidetik. Selanjutnya antarmuka penyimpanan *file* enkripsi. Pada halaman untuk menampilkan *file* enkripsi pada penyimpanan, halaman tersebut menggunakan fungsi php open directory. Fungsi tersebut digunakan untuk membaca isi directory dimana *file* disimpan. Pada antarmuka program *decrypt* penulis menggunakan modul tkinter yang digunakan untuk membuat tampilan *Graphical User Interface* (GUI) pada program ini. Pada program *decrypt* berisi nama *file* dan password untuk melakukan *decrypt* dan dapat mengenkripsinya kembali. Berikut merupakan gambar rancangan antarmuka halaman admin, direktori penyimpanan *file*, dan program *decrypt* yang dapat dilihat pada Gambar 3.2, 3.3, dan 3.4.



Gambar 3.2 Rancangan antarmuka halaman admin



Gambar 3.3 Rancangan antarmuka halaman direktori *file*



Gambar 3.4 Rancangan antarmuka program *decrypt*

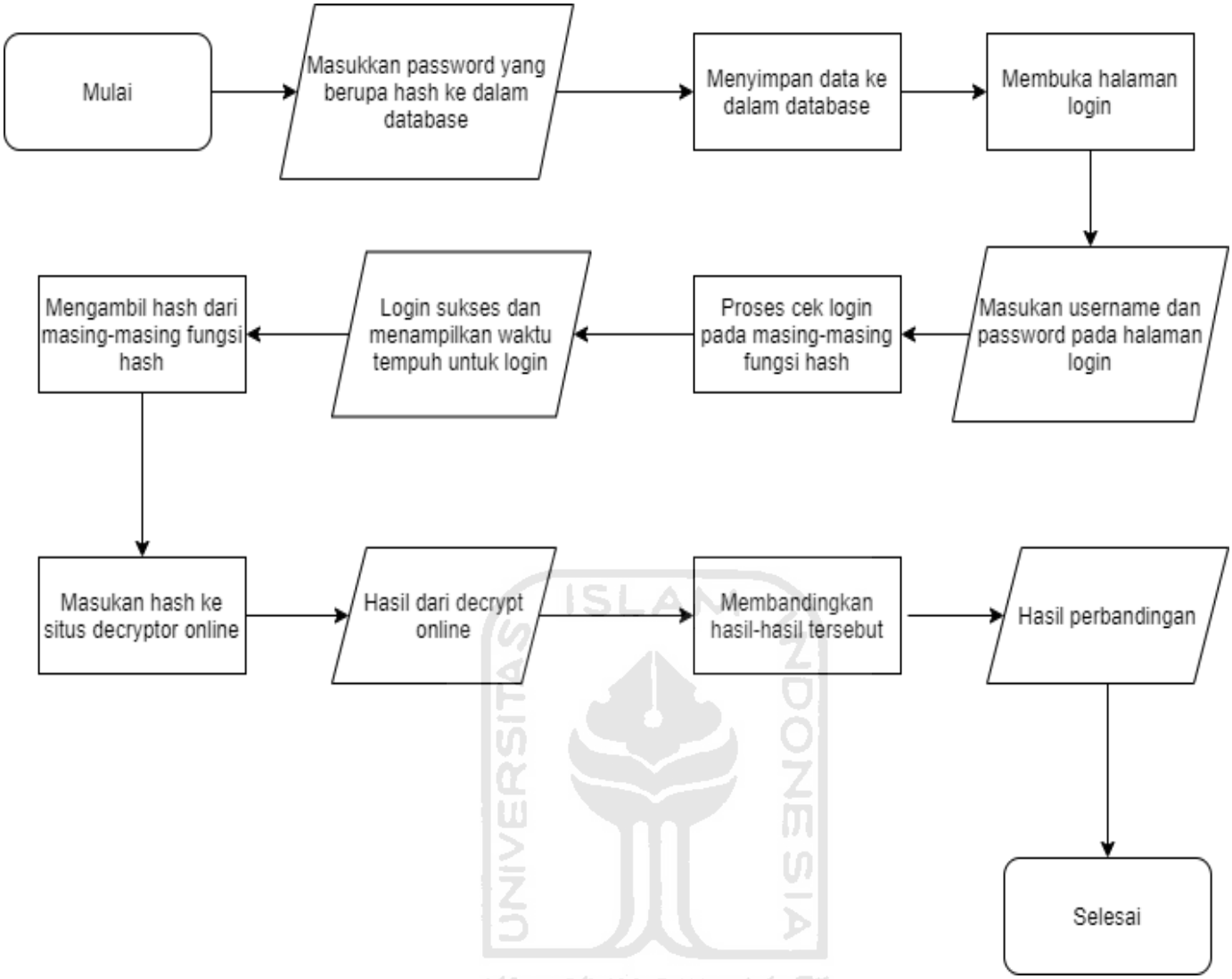
3.4.2 Perancangan Sistem

Pada tahapan ini penulis melakukan perancangan sistem untuk membantu dalam menganalisis Surveillance Camera. Pada perancangan sistem dibagi menjadi 3 yaitu sistem untuk menguji coba login pada fungsi *hash*, sistem untuk mengenkripsi, dan sistem untuk melakukan *decrypt* pada *file* yang dienkripsi. Pada perancangan sistem uji login fungsi *hash*, penulis menggunakan *file* php login check untuk menguji login pada setiap fungsi *hash*. Penulis membagi menjadi 2 untuk login check yaitu untuk login menggunakan MD5 dan SHA-256. Selain itu penulis menggunakan fungsi php page generate untuk mengukur waktu tempuh login. Penulis juga melakukan pengujian terhadap *hash* dengan menggunakan database online yang berisi kumpulan *hash* yang sebelumnya pernah diubah dari plaintext. Penulis menggunakan situs www.md5online.org/md5-*decrypt*.html dan md5*decrypt*.net/en/Sha256/ dalam melakukan pengujian. Pada sistem enkripsi penulis menggunakan python dan modul pyAesCrypt yang digunakan untuk mengenkripsi *file* video menjadi *file* yang terenkripsi (*encrypted file*). Pada penamaan *encrypted file* pada sistem enkripsi penulis menggunakan tanggal pada saat video direkam. Pada perancangan sistem untuk melakukan *decrypt*, sistem ini juga menggunakan python dan modul pyAesCrypt yang sama seperti sistem enkripsi *file*. Bedanya sistem ini kebalikan dari sistem enkripsi *file*. Sistem ini melakukan *decrypt* pada *file* yang telah dienkripsi. Sistem ini juga menggunakan *Graphical User Interface* (GUI) yang digunakan untuk antarmuka sistem agar mudah digunakan.

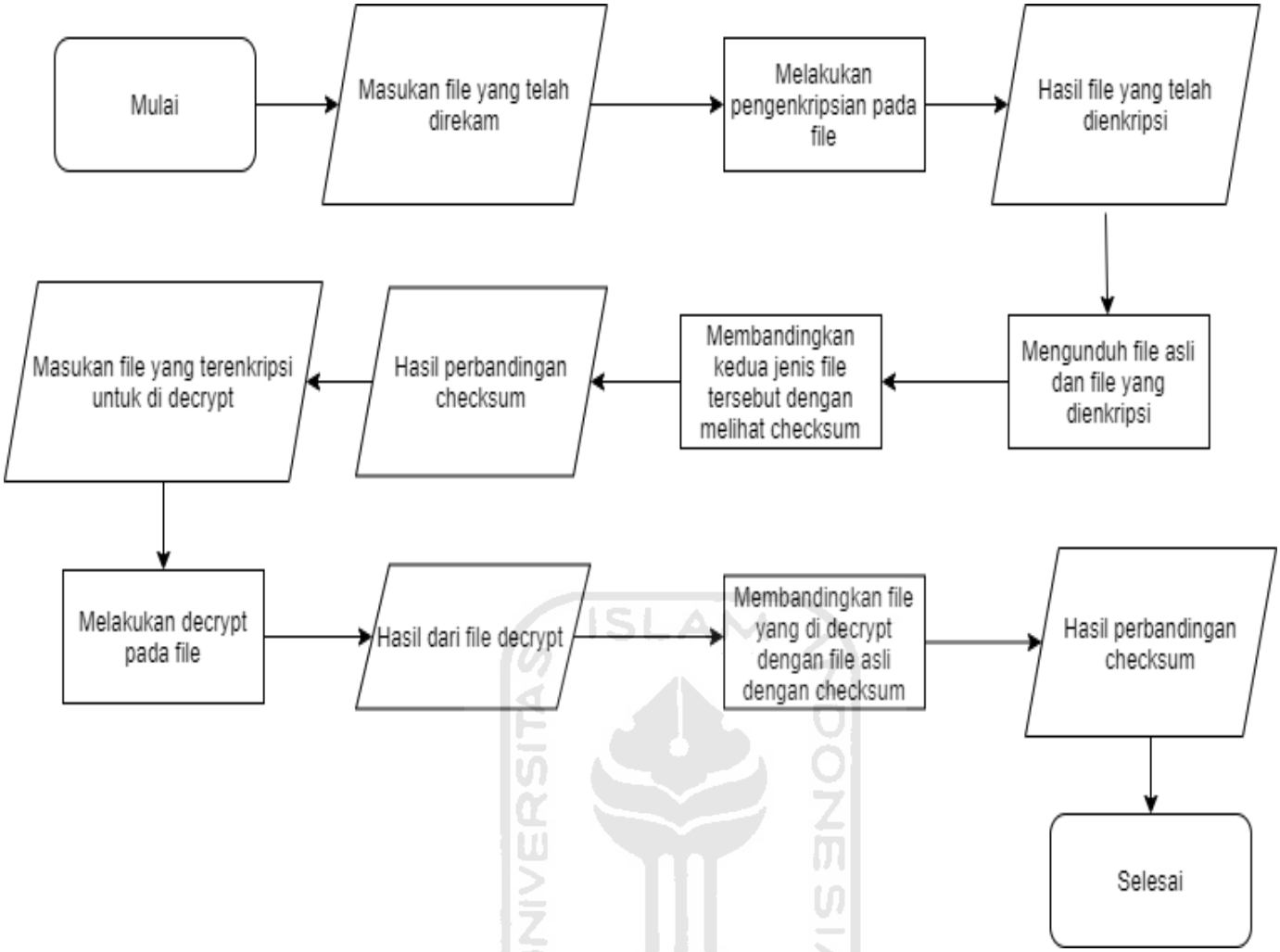
3.4.3 Diagram Alur

Pada penelitian ini, terdapat dua urutan dalam melakukan penelitian. Urutan yang pertama menjelaskan urutan untuk menganalisis fungsi *hash* pada sistem dan Urutan yang kedua menjelaskan tentang pengenkripsian pada *file*. Dalam menganalisis fungsi *hash*, hal pertama yang dilakukan adalah memasukkan data *hash* password pada database. Kemudian dilakukan pengujian login terhadap *hash* tersebut. Setelah login akan diperoleh waktu tempuh login. Setelah itu dilakukan pengujian terhadap *hash* dengan menggunakan database online. Kemudian akan diperoleh hasil dari pengujian tersebut. Dari hasil-hasil yang diperoleh kemudian membandingkan hasil dari kedua fungsi *hash* tersebut.

Dalam melakukan pengenkripsian *file*, hal pertama yang dilakukan adalah memasukkan *file* rekaman dari hasil Surveillance Camera. Kemudian *file* tersebut dienkripsi menggunakan sistem yang telah sebelumnya dibuat oleh penulis. Setelah terenkripsi *file* tersebut diunduh dan dibandingkan dengan *file* aslinya. Setelah itu *file* yang terenkripsi dibandingkan dengan *file* asli dengan melihat hasil checksum pada keduanya. Setelah itu *file* yang terenkripsi tadi di *decrypt* kembali untuk mengetahui *file* aslinya agar dapat dibandingkan dengan *file* yang tanpa dienkripsi dengan melihat checksumnya. Berikut merupakan gambar diagram alur dari kedua urutan tersebut yang ditampilkan pada Gambar 3.5 dan 3.6.



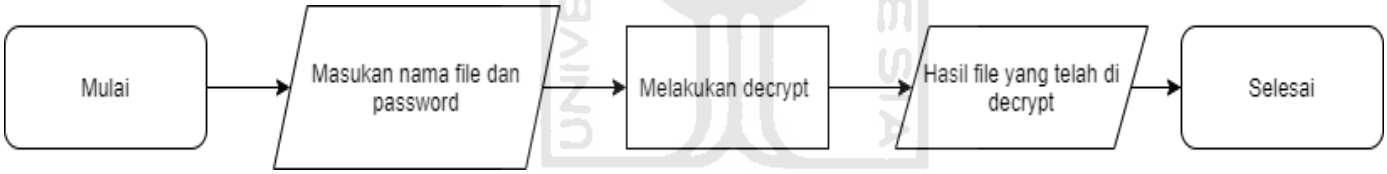
Gambar 3.5 Diagram alur pengujian fungsi *hash*



Gambar 3.6 Diagram alur pengujian pengenkripsian *file*

Pada Diagram alur pengenkripsian *file* terdapat proses sistem pengenkripsian pada *file* dan proses sistem untuk *decrypt file*. Pada proses pengenkripsian *file*, alur pertama yang dilakukan oleh sistem yaitu mengambil nama *file* pada *file* asli untuk dienkripsi. Selanjutnya *file* tersebut dienkripsi oleh sistem dan menghasilkan *file* yang telah dienkripsi. Pada proses melakukan *decrypt* pada *file* yang dienkripsi, alur pertama yang dilakukan adalah memasukkan nama dan password untuk membuka *file* enkripsi tersebut pada program *decrypt*. Setelah itu program akan melakukan *decrypt* dan menghasilkan *file* yang telah di *decrypt*. Berikut merupakan gambar diagram alur sistem enkripsi dan *decrypt file* yang ditampilkan pada Gambar 3.7 dan 3.8.

Gambar 3.7 Alur proses pengenkripsian *file*



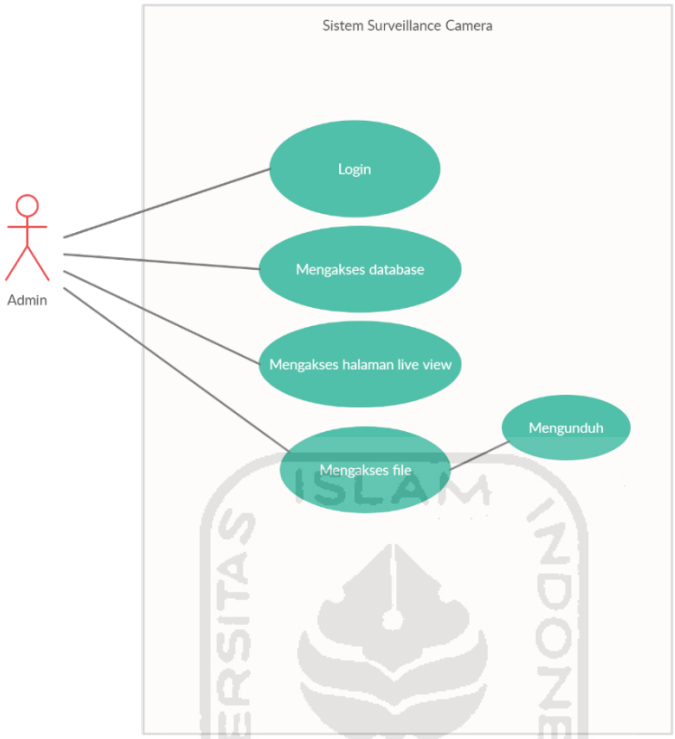
Gambar 3.8 Alur proses melakukan *decrypt*

3.4.4 Use Case Diagram

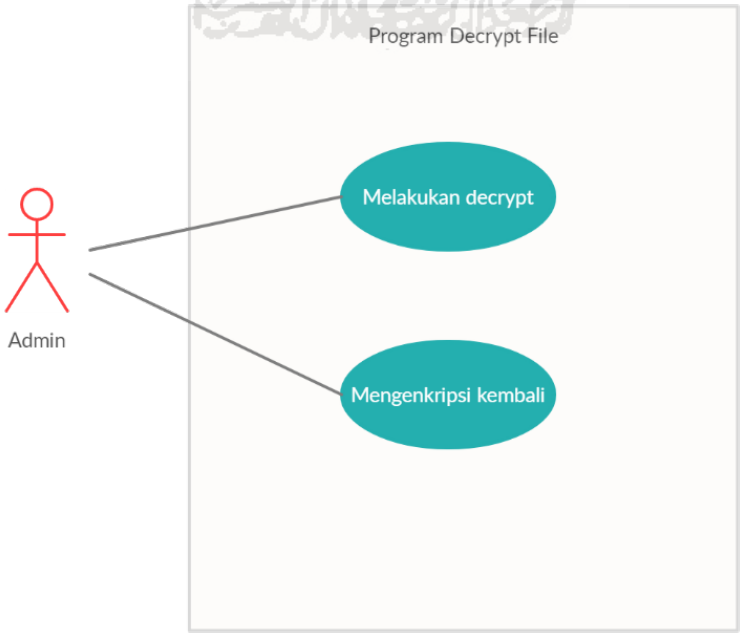
Pada penelitian ini penulis membuat use case diagram agar dapat mengetahui akses yang dapat dilakukan oleh seorang admin. Pada penelitian ini terdapat 2 use case diagram yaitu use case diagram untuk sistem Surveillance Camera dan use case diagram untuk program melakukan *decrypt file*. Aktor pada use case ini adalah admin.

Pada use case untuk sistem Surveillance Camera, admin dapat mengakses login, dapat mengakses database, Mengakses halaman live view, dan mengakses *file* untuk melakukan pengunduhan *file*. Pada use case untuk program *decrypt* admin dapat melakukan *decrypt* pada

file yang menghasilkan *file* yang telah di *decrypt* dan dapat melakukan pengenkripsian Kembali *file* jika diinginkan. Berikut merupakan use case diagram tersebut yang ditampilkan pada Gambar 3.9 dan 3.10.



Gambar 3.9 Use case diagram sistem Surveillance Camera



Gambar 3.10 Use case diagram program *decrypt*

3.4.5 Perancangan Basis Data

Pada tahapan ini penulis merancang sebuah basis data agar dapat menyimpan password dalam berbentuk *hash* dari setiap fungsi. Penulis menggunakan MySQL sebagai sistem manajemen basis data SQL. Pada basis data penulis membuat sebuah tabel yang digunakan sebagai penyimpanan akun yang diberi nama pengguna. Tabel pengguna berisi 4 kolom yaitu user_id, username, password, fullname. Untuk tipe data user_id menggunakan tipe integer, username menggunakan tipe varchar, password menggunakan tipe varchar, dan fullname menggunakan tipe varchar. Pada password menggunakan tipe varchar karena nilai *hash* terdiri dari huruf dan angka. Untuk batasannya menggunakan 255 agar MD5 yang terdiri dari 32 karakter dan SHA-256 yang terdiri dari 64 karakter akan cukup dalam kolom tersebut. Berikut merupakan tabel rancangan database yang dapat dilihat pada Tabel 3.1.

Tabel 3.1 Tabel rancangan basis data

Nama Field	Type	Size	Keterangan
user_id	int	11	Id user
username	varchar	255	Nama user untuk login
password	varchar	255	Kata sandi
fullname	varchar	255	Nama lengkap user

3.5 Pengujian

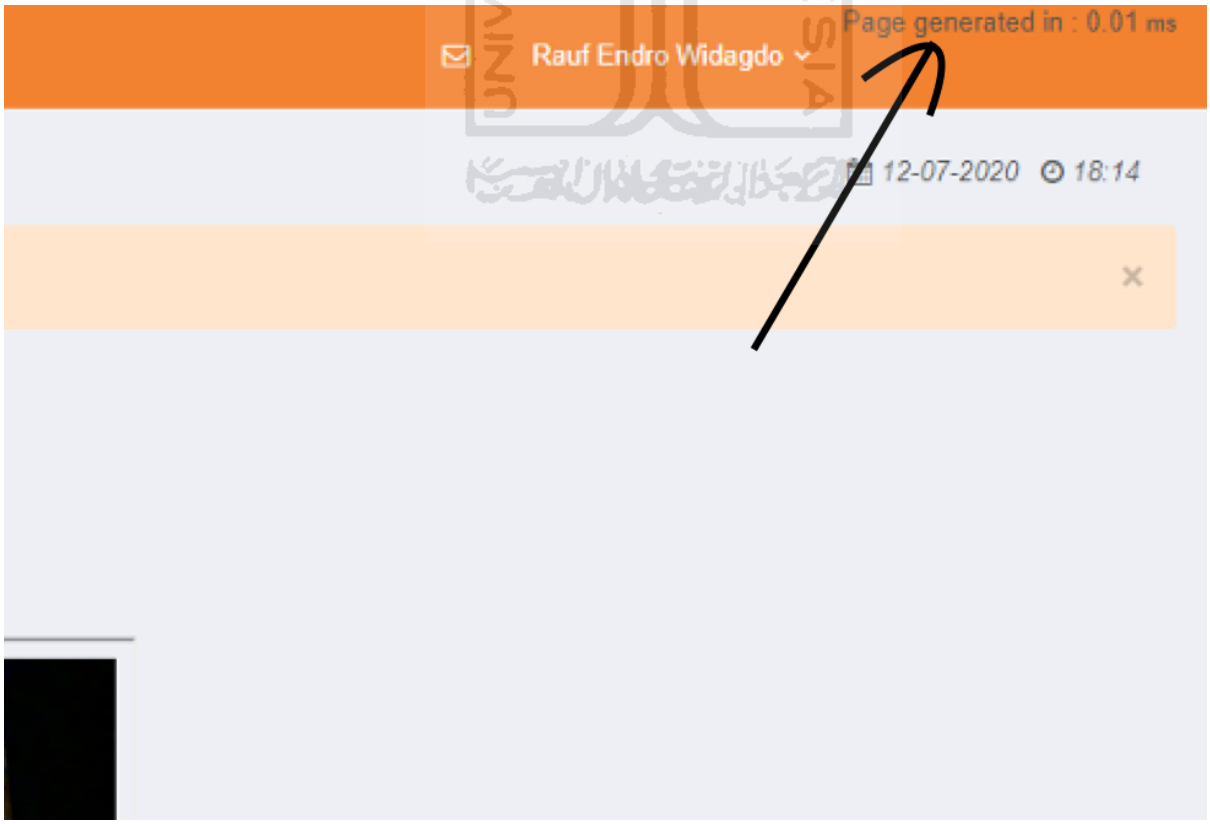
Pada penelitian ini penulis melakukan 2 pengujian. Pengujian pertama dilakukan pada fungsi *hash* dan yang kedua pada *file* yang dienkripsi. Pengujian fungsi *hash* menerapkan 3 parameter uji yaitu uji login, uji waktu tempuh login, dan uji *decrypt*or online. Kemudian hasil uji tersebut dikumpulkan agar dapat dibandingkan satu sama lain. Skenario dalam pengujian fungsi *hash* dilakukan dengan uji login pada akun yang menggunakan *hash* sebagai passwordnya. Dari uji tersebut akan diperoleh kesuksesan login dan waktu tempuh untuk melakukan login. Setelah pengujian login dilakukan pengujian dengan menguji pada situs *decrypt* online www.md5online.org/md5-*decrypt*.html dan md5*decrypt*.net/en/Sha256/ . Dari uji tersebut akan diperoleh hasil keterangan dari situs tersebut. Pada pengujian *file* enkripsi penulis menguji kesuksesan enkripsi, membandingkan *file* enkripsi dengan *file* asli, dan memeriksa kesamaan *file* yang di *decrypt* dengan *file* yang asli. Skenario dalam pengujian *file* enkripsi dilakukan dengan mengenkripsi *file* dan memeriksa checksum. Kemudian *file* yang telah dienkripsi dilakukan *decrypt* dan memeriksa checksum *file*. Dari uji tersebut akan diperoleh kesuksesan enkripsi dan pengaruhnya terhadap *file* jika dilakukan pengenkripsian.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil Implementasi

4.1.1 Implementasi Antarmuka

Tampilan yang digunakan tidak berbeda jauh dengan tampilan sistem bawaan Surveillance Camera. Perubahan dilakukan untuk mempermudah dalam melakukan proses penelitian. Implementasi yang dilakukan berdasarkan rancangan yang telah dibuat sebelumnya. Pada perancangan antarmuka fungsi *hash* penulis menggunakan fungsi php page generate untuk mengukur waktu yang ditempuh untuk melakukan login. Fungsi php tersebut diimplementasikan pada halaman di pojok kanan dengan menggunakan satuan milidetik. Fungsi php tersebut akan muncul saat melewati halaman login pada sistem. Hal tersebut dilakukan agar dapat menghitung waktu tempuh pada setiap kali login. Berikut merupakan tampilan antarmuka halaman admin dengan fungsi php page generate dan bagian kode dari halaman admin setelah login yang dapat dilihat pada Gambar 4.1 dan 4.2.



Gambar 4.1 Tampilan antarmuka halaman admin

```
      </head>
      <body class="skin-blue fixed">
        <div class="wrapper">

          <header class="main-header">
            <!-- Logo -->
            <a href="../home" class="logo"><b></b>Surveillance Camera</a>
            <!-- Header Navbar: style can be found in header.less -->
            <nav class="navbar navbar-static-top" role="navigation">
              <!-- Sidebar toggle button-->
              <a      href="#"      class="sidebar-toggle"      data-toggle="offcanvas"
role="button">
                <span class="sr-only">Toggle navigation</span>
              </a>
              <div class="navbar-custom-menu">
                <ul class="nav navbar-nav">
                  <?php
                  require_once "../config/database.php";

                  $query = mysqli_query($mysqli, "SELECT COUNT(message_id) as jumlah
FROM pesan WHERE status='n'")
                                                  or  die('Ada  kesalahan  pada  query
tampil data message: '.mysqli_error($mysqli));

                  $data = mysqli_fetch_assoc($query);

                  if ($data['jumlah']=='0') {
                    $jumlah = '';
                  } else {
                    $jumlah = $data['jumlah'];
                  }
                  $start = microtime(true);
    $finish = microtime(true);
    print  'Page  generated  in  :  '.  round(($finish  -  $start)  *  10000,  2)  .'
<small>ms</small>';
                  ?>
                  <!-- Messages: style can be found in dropdown.less-->
                  <li class="dropdown messages-menu">
                    <a href="#" class="dropdown-toggle" data-toggle="dropdown">
                      <i class="fa fa-envelope-o"></i>
                      <span   style="background:#dd4b39"   class="label"><?php   echo
$jumlah; ?></span>
                    </a>
                    <ul class="dropdown-menu">
                      <?php
                      if ($data['jumlah']=='0') {
                        echo "<li class='header'>You have no new messages</li>";
                      } else {
                        echo     "<li     class='header'>You     have     $data[jumlah]
messages</li>";
                      }
                      ?>

                      <li class="footer"><a href="?module=message">See All Message <i
class="fa fa-angle-double-right"></i></a></li>
                    </ul>
                  </li>

                  <!-- panggil file "top-menu.php" untuk menampilkan menu -->
                  <?php include "top-menu.php" ?>
```

Gambar 4.2 Potongan kode dari halaman admin

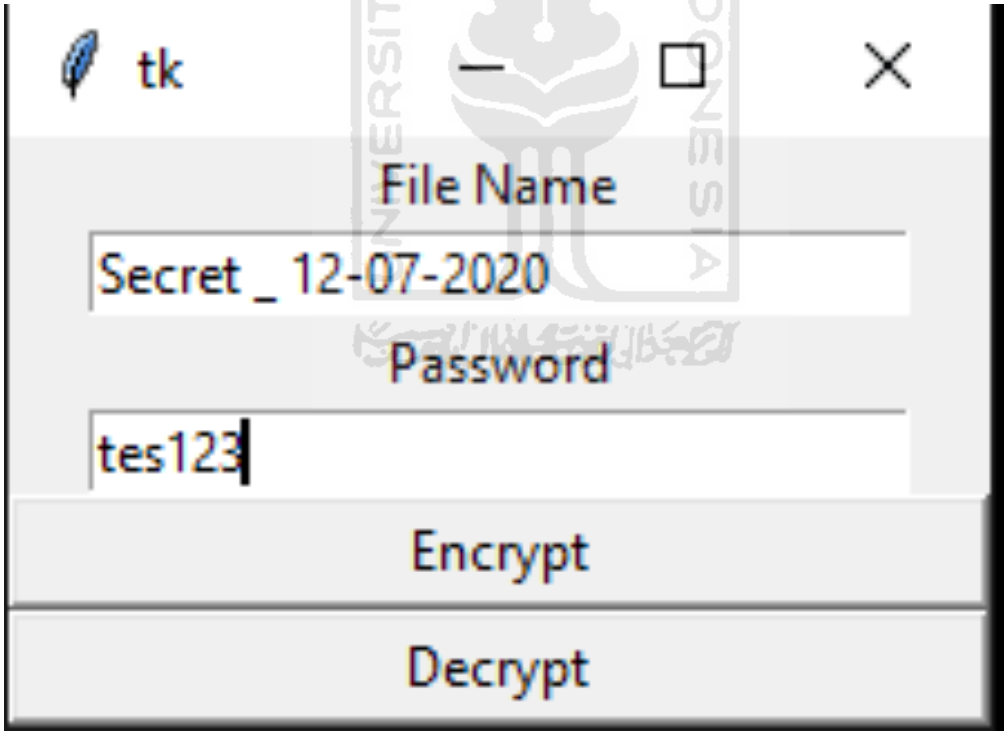
Pada tampilan antarmuka penyimpanan sebelumnya pada sistem ini menggunakan MySQL sebagai penyimpan nama *file*. Untuk menyimpan ke MySQL pada sistem ini menggunakan library python urllib2 yang akan membuka link php untuk menyimpan ke MySQL, namun library tersebut tidak terdapat pada python 3 karena penulis untuk saat ini menggunakan python 3. Untuk mengatasinya penulis menggunakan fungsi php untuk membaca direktori penyimpanan. Pada perancangan sebelumnya penulis menggunakan fungsi php open directory. Fungsi php tersebut berfungsi untuk membaca isi dari directory yang ditujukan sehingga fungsi tersebut dapat membaca isi dari directory tersebut. Berikut merupakan tampilan antarmuka direktori penyimpanan *file* dan kode untuk halaman direktori penyimpanan *file* yang dapat dilihat pada Gambar 4.3 dan 4.4.

Gambar 4.3 Halaman direktori penyimpanan *file*

```
    <!-- Content Header (Page header) -->
    <section class="content-header">
      <h1>
        <i class="fa fa-hdd-o icon-title"></i> Recording

        <a class="btn btn-primary btn-social pull-right" onclick="Refresh()">
          <i class="fa fa-refresh"></i> Refresh
        </a>
      </h1>

    </section>

    <!-- Main content -->
    <section class="content">
      <div class="row">
        <div class="col-md-12">

        <?php
    $msg = "";
    if($handle = opendir('../recording')){
    while (false !== ($file = readdir($handle)))
    {
    if (($file != ".")
    && ($file != ".."))
    {
    $msg    .=    '<li>    <a    href="../recording/'.$file.'"    download="'.$file.'"
>'.$file.'</a></li>';

    }
    }
    closedir($handle);
    }
    ?>

       <div class="box box-primary">
            <div class="box-body">
            <tbody>
                <p>
    <?php echo $msg ?>
    </p>
                </tbody>
              </table>
            </div><!-- /.box-body -->
          </div><!-- /.box -->
        </div><!--/.col -->
      </div>   <!-- /.row -->
      <script>
    function Refresh() {
        location.reload();
    }
    </script>
    </section><!-- /.content
```

Gambar 4.4 Kode halaman direktori penyimpanan file

Selanjutnya penulis membuat tampilan antarmuka untuk program *decrypt file* yang dienkripsi. Program tersebut dibuat cukup sederhana. Pada perancangan penulis menggunakan modul tkinter sebagai tampilan GUI pada program ini. Pada tampilannya hanya terdapat textfield dan tombol. Hanya terdapat 2 textfield yaitu untuk memasukkan nama *file* dan password. Untuk tombolnya terdapat 2 yaitu tombol untuk *decrypt* dan untuk mengenkripsi *file* nya kembali. Untuk melakukan *decrypt* hanya dibutuhkan nama *file* dan passwordnya lalu menekan tombol *decrypt*. *File* tersebut akan mengeluarkan *file* yang dienkripsi di dalamnya. Untuk melakukan enkripsi hanya membutuhkan nama *file* yang akan menjadi *file* terenkripsi dan password yang diinginkan untuk mengenkripsi. Selanjutnya *file* tersebut akan membuat *file* baru dengan nama yang diisi yang berisi *file* dienkripsi. Berikut merupakan tampilan dari antarmuka program *decrypt* dan kode tampilan program *decrypt* yang dapat dilihat pada Gambar 4.5 dan 4.6.



Gambar 4.5 Tampilan antarmuka program *decrypt*

```
import pyAesCrypt
from tkinter import *

root = Tk()

mylabel2=Label(root, text="File Name")
mylabel2.pack()

f = Entry(root, width=30)
f.pack()

mylabel=Label(root, text="Password")
mylabel.pack()

e = Entry(root, width=30)
e.pack()
```

Gambar 4.6 Kode tampilan program decrypt

4.1.2 Implementasi Sistem

Pada tahapan perancangan penulis menggunakan *file* php login check pada sistem Surveillance Camera untuk menguji login pada setiap fungsi *hash*. Penulis menggunakan 2 jenis login check yaitu login check untuk MD5 dan SHA-256. Pada login check MD5 pengecekan terjadi saat mengambil data dari submit pada form. Variable $password mengambil dan mengubahnya menjadi MD5 lalu variable tersebut akan dicek pada database MySQL. Berikut gambar code untuk login-check.php untuk MD5 yang dapat dilihat pada Gambar 4.7.

```
    <?php
    // panggil file untuk koneksi ke database
    require_once "../config/database.php";

    // ambil data hasil submit dari form
    $username                    =                    mysqli_real_escape_string($mysqli,
stripslashes(strip_tags(htmlspecialchars(trim($_POST['username'])))));
    $password                    =                    md5(mysqli_real_escape_string($mysqli,
stripslashes(strip_tags(htmlspecialchars(trim($_POST['password'])))))));

    // pastikan username dan password adalah berupa huruf atau angka.
    if (!ctype_alnum($username) OR !ctype_alnum($password)) {
       header("Location: index.php?alert=1");
    }
    else {
       // ambil data dari tabel user untuk pengecekan berdasarkan inputan username dan
passrword
       $query    =    mysqli_query($mysqli,    "SELECT    *    FROM    pengguna    WHERE
username='$username' AND password='$password'")
                                                                    or    die('Ada    kesalahan
pada query user: '.mysqli_error($mysqli));
       $rows  = mysqli_num_rows($query);

       // jika data ada, jalankan perintah untuk membuat session
       if ($rows > 0) {
              $data  = mysqli_fetch_assoc($query);

              session_start();
              $_SESSION['user_id']  = $data['user_id'];
              $_SESSION['username'] = $data['username'];
              $_SESSION['password'] = $data['password'];
              $_SESSION['fullname'] = $data['fullname'];

              // lalu alihkan ke halaman user
              header("Location: main.php?module=home");
       }

       // jika data tidak ada, alihkan ke halaman login dan tampilkan pesan = 1
       else {
              header("Location: index.php?alert=1");
       }
    }
    ?>
```

Gambar 4.7 login-check.php untuk MD5

Pada login check SHA-256 memiliki perbedaan dengan MD5. Pada saat mengambil data dari submit pada form, variable $password belum mengubahnya menjadi nilai *hash*. Nilai *hash* diubah pada saat ingin melakukan pengecekan pada MySQL. Berikut merupakan gambar code login-check.php untuk SHA-256 yang dapat dilihat pada Gambar 4.8.

```
    <?php
    // panggil file untuk koneksi ke database
    require_once "../config/database.php";

    // ambil data hasil submit dari form
    $username                    =                    mysqli_real_escape_string($mysqli,
stripslashes(strip_tags(htmlspecialchars(trim($_POST['username'])))));
    $password                    =                    (mysqli_real_escape_string($mysqli,
stripslashes(strip_tags(htmlspecialchars(trim($_POST['password']))))));

    // pastikan username dan password adalah berupa huruf atau angka.
    if (!ctype_alnum($username) OR !ctype_alnum($password)) {
       header("Location: index.php?alert=1");
    }
    else {
       // ambil data dari tabel user untuk pengecekan berdasarkan inputan username dan
passrword
       $query    =    mysqli_query($mysqli,    "SELECT    *    FROM    pengguna    WHERE
username='$username' AND password=sha2('$password', 256)")
                                                                    or    die('Ada    kesalahan
pada query user: '.mysqli_error($mysqli));
       $rows  = mysqli_num_rows($query);

       // jika data ada, jalankan perintah untuk membuat session
```

```
    if ($rows > 0) {
            $data  = mysqli_fetch_assoc($query);

            session_start();
            $_SESSION['user_id']  = $data['user_id'];
            $_SESSION['username'] = $data['username'];
            $_SESSION['password'] = $data['password'];
            $_SESSION['fullname'] = $data['fullname'];

            // lalu alihkan ke halaman user
            header("Location: main.php?module=home");
       }

       // jika data tidak ada, alihkan ke halaman login dan tampilkan pesan = 1
       else {
            header("Location: index.php?alert=1");
       }
    }
    ?>
```

Gambar 4.8 login-check.php untuk SHA-256

Pada tahapan perancangan sebelumnya penulis menggunakan fungsi php untuk mengukur waktu tempuh login. Untuk menghitungnya menggunakan microtime dengan mencari selisih variable $start dan variable $finish lalu dikali 10000 untuk mendapatkan satuan milidetik. Berikut gambaran kode perhitungan page generate yang dapat dilihat pada Gambar 4.9.

```
    $start = microtime(true);
    $finish = microtime(true);
    print 'Page generated in : '. round(($finish - $start) * 10000, 2) .'
<small>ms</small>';
```

Gambar 4.9 Kode fungsi php untuk menghitung waktu tempuh login

Implementasi berikutnya terdapat pada sistem pengenkripsian *file*. Sistem pengenkripsian *file* ditambahkan ke dalam sistem Surveillance Camera yang dibuat dalam *file* python baru. Pada tahapan perancangan sebelumnya penulis menggunakan modul pyAesCrypt pada python 3. Modul pyAesCrypt dapat mengenkripsi *file* dengan algoritma AES256-CBC. Berikut merupakan gambar *file* python untuk mengenkripsi *file* video mp4 dan avi yang dapat dilihat pada Gambar 4.10 dan 4.11.

```
    import pyAesCrypt
    import os
    from datetime import datetime

    os.chdir("/var/www/html/Surveilance-Camera/recording")

    waktu = datetime.now().strftime("%d-%m-%Y")
    def encrypt():
        buffer size = 64 * 1024
        password = "tes123"
        pyAesCrypt.encryptFile(str(waktu)+'.mp4',    'Secret    :    '+str(waktu),
password, buffer size)
```

Gambar 4.10 Kode untuk mengenkripsi file mp4

```
    import pyAesCrypt
    import os
    from datetime import datetime

    os.chdir("/var/www/html/Surveilance-Camera/recording")

    waktu = datetime.now().strftime("%d-%m-%Y")
    def encrypt():
        buffer size = 64 * 1024
        password = "tes123"
        pyAesCrypt.encryptFile(str(waktu)+'.avi',    'Secret    :    '+str(waktu),
password, buffer size)
```

Gambar 4.11 Kode untuk mengenkripsi file avi

Pada bagian os.chdir berfungsi untuk memindahkan direktori pekerjaan ke dalam direktori penyimpanan video. Variable waktu digunakan untuk menyimpan tanggal sekarang. Pada penelitian ini penulis menggunakan 2 buffer size yaitu 64*1024 dan 256*1024. Untuk password pengenkripsian penulis menetapkan "tes123" sebagai passwordnya. Untuk pengenkripsian *file*, sistem ini mengambil *file* berdasarkan nama dari tanggal hari ini dan diubah menjadi "Secret" yang ditambahkan tanggal hari ini.

Selanjutnya implementasi pada program *decrypt*. Program ini memiliki kesamaan dengan sistem pengenkripsian *file*. Perbedaannya ialah program ini kebalikan dari sistem pengenkripsian *file*. Program ini merupakan program dengan antarmuka GUI dan dapat dioperasikan pada sistem operasi Windows, MacOS dan Linux. Untuk membuat program ini dapat dijalankan secara GUI, penulis menggunakan modul tkinter. Untuk melakukan *decrypt* cukup memasukkan nama *file* dan password. Nama *file* dan password tersebut akan menjadi variabel yang digunakan untuk men*decrypt*. Program ini juga dapat melakukan pengenkripsian kembali seperti pada sistem pengenkripsian sebelumnya. Berikut merupakan kode untuk program *decrypt* yang dapat dilihat pada Gambar 4.12.

```
import pyAesCrypt
from tkinter import *

root = Tk()

mylabel2=Label(root, text="File Name")
mylabel2.pack()

f = Entry(root, width=30)
f.pack()

mylabel=Label(root, text="Password")
mylabel.pack()
```

```
    e = Entry(root, width=30)
    e.pack()



    def myDecrypt():
        buffer size = 256 * 1024
        password=e.get()
        pyAesCrypt.decryptFile(f.get()+".aes",  "video.mp4",  password,  buffer
size)

    def myEncrypt():
        buffer size = 256 * 1024
        password=e.get()
        pyAesCrypt.encryptFile("video.mp4",  f.get()+".aes",  password,  buffer
size)

    myButton = Button(root, text="Encrypt", width=30, command=myEncrypt)
    myButton.pack()



    myButton2 = Button(root, text="Decrypt", width=30, command=myDecrypt)
    myButton2.pack()


    root.mainloop()
```

Gambar 4.12 Kode untuk program decrypt

4.1.3 Implementasi Basis Data

Pada basis data penulis menggunakan MySQL sebagai sistem manajemen basis data SQL. Pada basis data penulis menggunakan nama "kamera-pengawas" sesuai yang terdapat pada sistem Surveillance Camera. Tabel pada basis data menggunakan nama "pengguna". Table tersebut berisi data tentang informasi akun pada sistem Surveillance Camera. Tabel tersebut juga untuk menyimpan nilai *hash*. Sesuai dengan perancangan, tabel pengguna berisi 4 kolom yaitu user_id, username, password, fullname. Pada kolom user_id menggunakan integer karena hanya id pada user. Pada kolom username menggunakan varchar agar dapat dikombinasi huruf dan angka. Pada kolom Password menggunakan varchar dengan batasan 255 agar MD5 yang memiliki 32 karakter dan SHA-256 yang memiliki 64 karakter cukup pada kolom tersebut. Pada kolom fullname menggunakan varchar dengan batasan 255 agar user bebas menamakan akunnya. Berikut merupakan kolom pada tabel pengguna yang dapat dilihat pada Gambar 4.13.

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**user_id** 🔑	int(11)			No	*None*		AUTO_INCREMENT
☐	2	**username**	varchar(255)	utf8mb4_general_ci		No	*None*		
☐	3	**password**	varchar(255)	utf8mb4_general_ci		No	*None*		
☐	4	**fullname**	varchar(255)	utf8mb4_general_ci		No	*None*		

Gambar 4.13 Kolom-kolom pada tabel pengguna

Berikut merupakan SQL INSERT INTO Statement untuk memasukkan data tersebut dengan fungsi *hash* masing-masing ke tabel pengguna.

a. Untuk MD5

INSERT INTO pengguna VALUES ('id', 'username', MD5('password'), 'Nama Lengkap')

b. SHA-256

INSERT INTO pengguna VALUES ('id', 'username', SHA2('password', 256), 'Nama Lengkap')

Pada penelitian ini menggunakan 2 tipe plaintext, yaitu “t-e-S(!=1@=2#=3)” dan “tes123”. Untuk masing-masing plaintext menggunakan MD5 dan SHA-256. Berikut merupakan hasil plaintext yang di masukkan ke dalam MySQL yang dapat dilihat pada Gambar 4.14.

+ Options

	user_id	username	password	fullname
Edit Copy Delete	1	rauf	ebd5661400190efe056dd4f6dcfb2fe6	Rauf Endro Widagdo
Edit Copy Delete	2	rauf	7c54a55ac3897f82fe37a70865b028e34e19d9797eb98292a8...	Rauf Endro Widagdo
Edit Copy Delete	3	rauf	b93939873fd4923043b9dec975811f66	Rauf Endro Widagdo
Edit Copy Delete	4	rauf	57776e8a41ff487b37a6b34186486b0e2f886e2cbf12a8e30d...	Rauf Endro Widagdo

Gambar 4.14 Isi dari tabel pengguna

4.2 Pembahasan

4.2.1 Pengujian *Hash*

Pengujian pertama dilakukan dengan menguji nilai *hash* yang sebelumnya dibuat dan disimpan pada MySQL. Nilai *hash* tersebut diuji dengan melakukan uji login. Dari Uji login tersebut diperoleh hasil uji login sukses atau gagal dan waktu yang ditempuh untuk melakukan login. Pengujian selanjutnya melakukan uji pada situs *decrypt* online. Dari uji tersebut akan menghasilkan nilai *hash* yang dapat diketahui melalui situs tersebut. Berikut merupakan tabel plaintext dan nilai *hash* dan tabel hasil dari pengujian nilai *hash* yang dapat dilihat pada Tabel 4.1 dan 4.2.

Tabel 4.1 Tabel nilai *hash* yang digunakan

Plaintext	Fungsi *Hash*	Nilai *Hash*
t-e-S(!=1@=2#=3)	MD5	ebd5661400190efe056dd4f6dcfb2fe6
t-e-S(!=1@=2#=3)	SHA-256	7c54a55ac3897f82fe37a70865b028e34e19d9797eb98292a8a2e93eb49fe19b
tes123	MD5	b93939873fd4923043b9dec975811f66
tes123	SHA-256	57776e8a41ff487b37a6b34186486b0e2f886e2cbf12a8e30d56dc67ea778193

Tabel 4.2 Tabel hasil pengujian *hash*

No.	Plaintext	Fungsi *Hash*	Nilai *Hash*	Uji Login	Kecepatan Login	Dapat di *Decrypt*
1.	t-e-S(!=1@=2#=3)	MD5	ebd5661400190efe056dd4f6dcfb2fe6	Sukses	0.02 ms	Tidak
2.	t-e-S(!=1@=2#=3)	SHA-256	7c54a55ac3897f82fe37a70865b028e34e19d9797eb98292a8a2e93eb49fe19b	Gagal	-	Tidak
3.	tes123	MD5	b93939873fd4923043b9dec975811f66	Sukses	0.01 ms	Ya
4.	tes123	SHA-256	57776e8a41ff487b37a6b34186486b0e2f886e2cbf12a8e30d56dc67ea778193	Sukses	0.01 ms	Ya

4.2.2 Hasil Analisis *Hash*

Dari hasil uji tersebut bisa dilihat data nilai *hash* pertama yang menggunakan plaintext dengan kombinasi karakter simbol dapat melakukan login dan tidak dapat di *decrypt*. Data kedua dengan plaintext yang sama tapi berbeda fungsi algoritma *hash* tidak dapat melakukan login dan tidak dapat di *decrypt*. Data ketiga dengan plaintext tanpa kombinasi karakter dapat melakukan login tapi dapat di *decrypt*. Data keempat dengan plaintext yang sama dengan data ketiga tapi berbeda fungsi algoritma *hash* dapat melakukan login dan dapat di *decrypt*.

Dilihat dari hasil uji, MD5 mampu menggunakan kombinasi karakter simbol pada plaintext nya untuk melakukan login. Karakter simbol selain memperkuat password juga dapat mengamankan nilai *hash* dari *decrypt*or online. Alasannya karena *decrypt*or online mempunyai database berupa nilai-nilai *hash* yang pernah di enkripsi sebelumnya tanpa kombinasi karakter simbol.

Kelebihan dan kekurangan kedua fungsi algoritma *hash* tersebut dapat disimpulkan MD5 memiliki keunggulan dari SHA-256 walaupun dengan panjang *hash* yang hanya berjumlah 128 bit dan 32 karakter. Penulis menyarankan jika ingin menggunakan SHA-256 sebagai password cukup menggunakan kombinasi huruf dan angka saja. Tidak semua huruf dan angka dapat di *decrypt* dengan *decrypt*or online karena selama nilai *hash* tersebut tidak berada di dalam database mereka. Berikut merupakan tabel kelebihan dan kekurangan dari masing-masing fungsi algoritma *hash* yang ditampilkan pada Tabel 4.3.

Tabel 4.3 Tabel hasil analisis *hash*

MD5	SHA-256
Memiliki panjang *hash* 128 bit dengan 32 karakter	Memiliki panjang 256 bit dengan 64 karakter
Dapat melakukan login dengan karakter huruf dan angka	Dapat melakukan login dengan karakter huruf dan angka
Dapat melakukan login dengan kombinasi karakter simbol	Tidak dapat melakukan login dengan kombinasi karakter simbol
Dapat melakukan *decrypt* dengan huruf dan angka pada *decrypt*or online	Dapat melakukan *decrypt* dengan huruf dan angka pada *decrypt*or online
Tidak dapat melakukan *decrypt* dengan kombinasi karakter simbol pada *decrypt*or online	Tidak dapat melakukan *decrypt* dengan kombinasi karakter simbol pada *decrypt*or online

4.2.3 Pengujian Enkripsi *File*

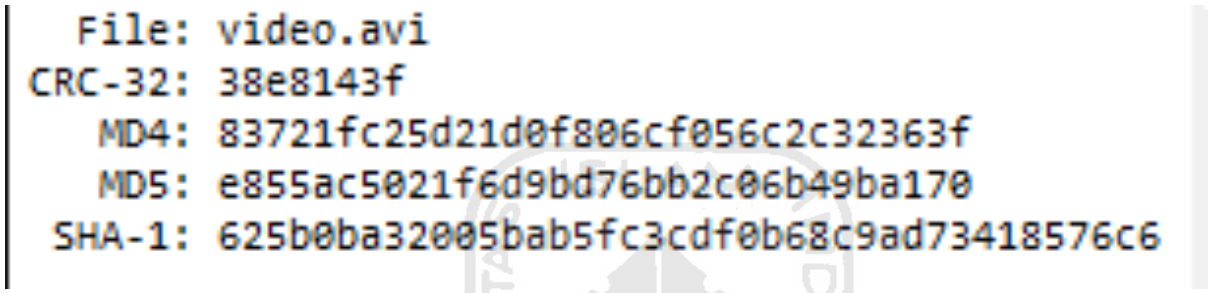
Pada pengujian enkripsi, penulis menggunakan sampel video berdurasi 1 menit untuk mengenkripsi video dengan password enkripsi yang sama. Kedua *file* tersebut menggunakan 2 format *file* mp4 dan avi. Pengujian pertama dilakukan dengan menguji mengenkripsi *file* dan memeriksa hal yang terjadi jika *file* dienkripsi. Pengenkripsian *file* dilakukan dengan 2 tahap yaitu enkripsi *file* mp4 dan enkripsi *file* avi. Pengenkripsian juga menggunakan terjadi 2 kali untuk setiap format *file*. Pengenkripsian pertama menggunakan buffer size 64*1024 dan pengenkripsian kedua menggunakan buffer size 256*1024.

Dari pengujian pertama akan dihasilkan 4 *file* yang dienkripsi dan jumlah seluruhnya ada 6 *file* termasuk 2 *file* dari *file* asli mp4 dan avi. Dari 6 *file* tersebut yang akan saling dibandingkan. Penulis lalu memeriksa checksum dari 6 *file* tersebut dan membandingkan perbedaan dari *file-file* tersebut. Penulis juga memeriksa ukuran dari *file-file* tersebut dalam satuan bytes. keduanya. Berikut merupakan tabel dari hasil uji enkripsi yang dapat dilihat pada Tabel 4.4.

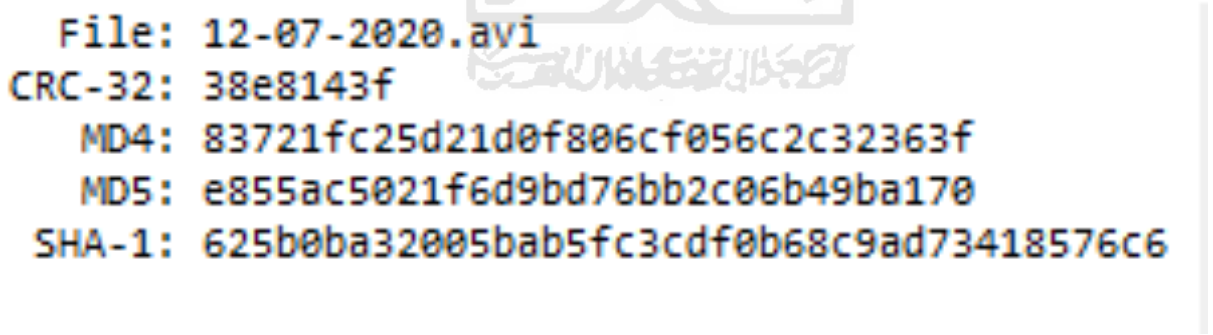
Tabel 4.4 Tabel hasil uji pengenkripsian *file*

No.	Format Video	Ukuran Video	Checksum	Ukuran saat enkripsi (Buffer size 64*1024)	Checksum MD5 *file* enkripsi (Buffer size 64*1024)	Ukuran saat enkripsi (Buffer size 256*1024)	Checksum MD5 *file* enkripsi (Buffer size 256*1024)
1.	Mp4	558,837 bytes	54c1c36a8a1bc9ad6c400b4cc253bbf5	559,143 bytes	eac7830aeb6ff1bbdee16d1f7ead5c34	559,143 bytes	0ad362d207f5008e0c287c4f259ab061
2.	Avi	9,778,834 bytes	e855ac5021f6d9bd76bb2c06b49ba170	9,779,143 bytes	a0b0f5d389836e40a98cabaf248c16d5	9,779,143 bytes	b11b948954bee82e625c69f515820a9c

Pengujian kedua penulis melakukan *decrypt* pada pada salah satu *file* yang dienkripsi. Penulis hanya melakukan salah satu *file* karena hanya ingin menguji kesuksesan *decrypt* dan kesuksesan sistem enkripsi menjaga integritas *file* aslinya. Pada uji *decrypt*, penulis membandingkan checksum pada *file* asli dan *file* hasil *decrypt* dengan kecocokan pada Berikut merupakan gambar checksum *file* hasil *decrypt* dan gambar checksum *file* asli yang dapat dilihat pada Gambar 4.15 dan 4.16.



Gambar 4.15 Checksum *file* setelah dilakukan *decrypt*



Gambar 4.16 Checksum pada *file* asli

4.2.4 Hasil Analisis Enkripsi

Dilihat dari tabel hasil uji enkripsi tersebut, format MP4 mempunyai ukuran yang lebih kecil dari format AVI. Hal itu terjadi karena pada saat video direkam, codec yang memproses mp4 mempercepat video sehingga durasinya dipercepat. Padahal FPS yang digunakan jumlahnya sama. Penulis beranggapan codec yang memproses MP4 mengalami masalah karena untuk format AVI video tidak terjadi masalah. Untuk format AVI penambahan ukuran sebanyak 306 bytes setelah di enkripsi dan untuk format MP4 penambahan ukuran terjadi 309 bytes. Untuk pengenkripsian video terjadi penambahan ukuran kurang lebih 300 bytes jika dilihat dari hasil uji kedua format tersebut. Penambahan ukuran tersebut disebabkan algoritma enkripsi tersebut menambahkan data untuk disisipkan pada *file* asli. Bisa dilihat dari checksum-nya yang telah mengalami perubahan pada datanya berbeda dengan checksum pada *file* asli. Untuk ukuran pengenkripsian tidak memiliki perbedaan dari segi ukuran.

Untuk melakukan *decrypt* juga tidak terjadi masalah dan sukses melakukan *decrypt*. Hasil dari *decrypt* juga tidak mempengaruhi *file* asli karena checksum pada *file* hasil *decrypt* tidak berubah sedikit pun. Dengan pengenkripsian ini *file* video akan terjaga integritasnya karena tujuan dari pengenkripsian *file* untuk menjaga integritas dari *file* video hasil rekam tersebut.

BAB V
KESIMPULAN

5.1 Kesimpulan

Penelitian ini menghasilkan analisis pengamanan pada IoT *Surveillance Camera* yang dapat digunakan untuk referensi dalam memilih fungsi *hash* pada IoT Surveillance Camera. Hasil dari analisis tersebut memberi referensi bahwa fungsi *hash* MD5 walaupun dengan 128 bit mampu melakukan login dengan *hash* yang berasal dari kombinasi karakter simbol, huruf, dan angka. Sebaliknya, SHA-256 yang memiliki 256 bit dengan *hash* yang berasal dari kombinasi simbol, huruf, dan angka tidak dapat melakukannya. Dari hasil analisis itu juga dapat diketahui bahwa situs *decrypt* online tersebut tidak dapat melakukan *decrypt* dengan *hash* yang berasal dari kombinasi karakter simbol, huruf, dan angka. Penelitian ini juga menghasilkan analisis sistem pengenkripsian yang mampu menjaga keamanan integritas dari *file* video yang dihasilkan. Hasil dari analisis tersebut menghasilkan *file* yang dienkripsi akan berbeda checksum dan ukurannya dengan *file* asli. *File* tersebut juga dapat mengembalikan *file* asli di dalamnya. Hal tersebut menandakan bahwa *file* asli sukses dienkripsi dan tidak dapat diketahui isinya. Hasil analisis pengamanan IoT Surveillance Camera ini menggunakan dari sistem *Surveillance Camera* yang pernah dibuat sebelumnya. Studi kasus yang digunakan adalah bimbingan atau konseling mahasiswa UII dan studi kasus terhadap penelitian sebelumnya.

5.2 Saran

Saran untuk analisis kedepannya dapat menganalisis lebih banyak algoritma *hash* dan algoritma pengenkripsian *file*. Analisis dengan banyak algoritma dapat memperbanyak referensi dalam memilih sistem keamanan yang tepat. Kedepannya juga diharapkan mampu membuat sistem Surveillance Camera dengan sistem enkripsi dan dapat menggunakan cloud sebagai penyimpanannya. Diharapkan juga dapat dikembangkan sistem pengenkripsian dengan algoritma lainnya.

DAFTAR PUSTAKA

Angga, C. (2011). Analisis Cara Kerja Beragam Fungsi Hash Yang Ada. *Informatika STEI ITB*, 1–6.

Bhaskar, P. (2015). Raspberry Pi Home Automation With Wireless Sensors Using Smart Phone. *International Journal of Computer Science and Mobile Computing*, *45*(5), 797–803.

Coole, M., Woodward, A., & Valli, C. (2012). Understanding the Vulnerabilities in Wi-Fi and the Impact on its Use in CCTV Systems. *Australian Security and Intelligence Conference Security*, *2018*(May 2014), 36–43. https://doi.org/10.4225/75/57a03670ac5ce

Cusack, B., & Tian, Z. (2017). Evaluating IP surveilance camera vulnerabilities. *Proceedings of the 15th Australian Information Security Management Conference, AISM 2017*, 25–32. https://doi.org/10.4225/75/5a84efba95b46

Dadda, L., Macchetti, M., & Owen, J. (2004). The design of a high speed ASIC unit for the hash function SHA-256 (384, 512). *Proceedings -Design, Automation and Test in Europe, DATE*, *3*, 70–75. https://doi.org/10.1109/DATE.2004.1269207

Harrington, W. (2015). *Learning Raspbian*. https://books.google.com/books?hl=en&lr=&id=O6HNBgAAQBAJ&oi=fnd&pg=PP1&dq=raspbian&ots=ZG2PnhATWF&sig=0QwaUR3tOkvYgRWReEo5sgesoHo

Lutfi, A., Aji, B., Pramukantoro, E. S., & Siregar, R. A. (2018). Analisis Mekanisme End-To-End Security Pada Komunikasi Antara Node Sensor Dengan IoT Middleware. *Analisis Mekanisme End-To-End Security Pada Komunikasi Antara Node Sensor Dengan IoT Middleware*, *2*(10), 4150–4155.

Piza, E. L., Welsh, B. C., Farrington, D. P., & Thomas, A. L. (2019). CCTV surveillance for crime prevention: A 40-year systematic review with meta-analysis. *Criminology and Public Policy*, *18*(1), 135–159. https://doi.org/10.1111/1745-9133.12419

Razalli, H., Alkawaz, M. H., & Suhemi, A. S. (2019). Smart IOT surveillance multi-camera monitoring system. *Proceeding - 2019 IEEE 7th Conference on Systems, Process and Control, ICSPC 2019*, *December*, 167–171. https://doi.org/10.1109/ICSPC47137.2019.9067984

Tedeschi, S., Mehnen, J., Tapoglou, N., & Roy, R. (2017). Secure IoT Devices for the Maintenance of Machine Tools. *Procedia CIRP*, *59*(TESConf 2016), 150–155. https://doi.org/10.1016/j.procir.2016.10.002

Wang, X., & Yu, H. (2005). How to break MD5 and other hash functions. *Lecture Notes in*

Computer Science, *3494*, 19–35. https://doi.org/10.1007/11426639_2

Yuniati, V., Indriyanta, G., & Rachmat C., A. (2011). Enkripsi Dan Dekripsi Dengan Algoritma Aes 256 Untuk Semua Jenis File. *Jurnal Informatika*, *5*(1). https://doi.org/10.21460/inf.2009.51.69

Pratama, A. R., & Rahma, F. (2019). Keamanan Siber dan Informasi Prinsip Dasar dan Ancaman Terkini. Sleman, Yogyakarta

LAMPIRAN

Lampiran tidak perlu diberi nomor halaman. Dokumen apa saja yang dimasukkan dalam lampiran cukup diberi judul dengan kata ‘LAMPIRAN’ yang dilanjutkan dengan huruf abjad besar untuk penomoran. Cukup judul ‘LAMPIRAN’ saja yang dimasukkan dalam daftar isi. Judul-judul lampiran, seperti Lampiran A, Lampiran B dan seterusnya, tidak perlu dimasukkan dalam daftar isi.