KARTU CERDAS (*SMART CARD*) DAN HUBUNGANNYA DENGAN KRIPTOGRAFI

Dean Fathony Alfatwa – NIM : 13503003

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13003@students.if.itb.ac.id

Abstraksi

Untuk menjaga keamanan data, sebagian orang masih bergantung kepada sandi lewat (*password*) sebagai otentikasi pertama. Masalah yang muncul dari penggunaan *password* ini adalah bahwa *password* ini mudah ditemukan oleh *hacker* amatir menggunakan metode serangan yang sudah diketahui secara umum disebut *dictionary attack*. Serangan lain yang dapat dilakukan yaitu serangan secara *brute force*. Selain itu dapat juga digunakan aplikasi yang sudah tersedia di internet, dan tinggal menunggu waktu saja maka *password* sudah dapat diketahui. Masalah keamanan *passsword* ini biasanya dipecahkan dengan semakin merumitkan otentikasi dalam pemasukan *password*, misalkan *password* harus diganti sebulan sekali, atau panjang *pasword* minimal 15 karakter, dengan tujuan menyulitkan para penyerang untuk mengetahui *password* tersebut. Namun sayangnya hal ini juga memuat pengguna *password* merasa tidak nyaman. Solusi yang bagus untuk masalah otentikasi seperti ini adalah menggunakan kombinasi antara sertifikat pada *smart card* sebagai identifikasi dan otentikasi yang dipasangkan dengan penggunaan PIN. Hal yang bagus dari *smart card* adalah tidak diperlukan PIN yang panjang dan rumit.

Smart Card adalah kartu berbahan plastik atau sejenisnya dengan mikroprosesor yang ditanamkan pada kartu dan media penyimpanan yang besar untuk menyimpan program yang telah disediakan oleh perusahaan penerbit kartu tersebut. Struktur dari *smart card* telah disepsifikkan oleh standar internasional.

Dalam makalah ini akan dijelaskan tentang arsitektur secara umum dari *smart card* beserta penjelasan bagaimana arsitektur standard yang dipaparkan dalam ISO 7816 untuk *smart card*. Selanjutnya akan dijelaskan tentang beberapa jenis serangan yang dapat dilakukan terhadap *smart card* serta bagaimana cara pengamanan *smart card* berdasakan pendekatan dari kriptografi.

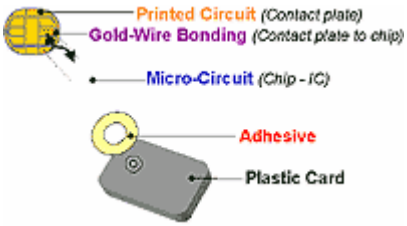
Keywords: *smart card architecture, smart card security, key management,* kriptografi, *smart card attack*

1. Pendahuluan

Munculnya dunia e-commerce seperti saat ini membuat diperlukannya sebuah pelayanan yang mudah, cepat, dan menjamin keamanan dalam bertransaksi melaui internet. Dalam hal pelayanan keamanan dalam bertransakasi melalui internet, diperlukan otentikasi user, pendistribusian kunci, integritas data, dan tanda tangan digital sebagai sarana nirpenyangkalan, *smart card* dan kriptografi merupakan alat yang sangat membantu untuk mengimplementasikan tugas pelayanan keamanan transaksi ini. *Smart card* adalah media yang membawa kunci untuk menjalankan operasi kriptografi. Sedangkan peranan dari kriptografi antara lain regristrasi pelanggan menggunakan kriptografi kunci publik, otentikasi pengguna, pendistribusian kunci, proteksi terhadap data menggunakan kriptografi kunci simetris, dan tanda tangan digital menggunakan kriptografi kunci publik dan fungsi hash

Smart card adalah kartu berbahan plastik atau sejenisnya dengan mikroprosesor yang ditanamkan pada kartu dan media penyimpanan yang besar untuk menyimpan program yang telah disediakan oleh perusahaan penerbit kartu tersebut. Struktur dari

smart card telah disepsifikkan oleh standar internasional yaitu, kartu plastik ini harus memiliki dimensi 85,60mm x 53,98mm x 0,76mm, yang mrupakan panjang, lebar, dan tebal dari kartu tersebut, selain itu kartu ini juga harus mampu menahan sejumlah tekanan udara tanpa mengalami kerusakan secara langsung. Sebuah *printed circuit* dan sebuah *integrated circuit chip* (*microcontroller*) ditanamkan pada kartu. Karena *printed circuit* dan *integrated circuit* ini tidak dapat mengatasi tekanan udara dengan baik maka *chip* yang dibuat haruslah memiliki ukuran yang sangat kecil. *Printed circuit* merupakan lempengan emas tipis yang menyediakan hubungan arus listrik dengan lingkungan luar dan juga melindungi *chip* dari tekanan yang bersifat mekanik dan melindungi dari listrik statis.



Gambar 1. *Chip* dalam *Smart Card*

Smart card menyediakan keanamanan yang lebih tinggi untuk berbagai penerapan. Kegunaan dari *smart card* didasarkan pada mobilitas dan keamanannya. Sama seperti kartu kredit kebanyakan, ketika *printed circuit* (lempengan emas) dimasukkan ke dalam *card reader*, lempengan ini akan menyediakan energi listrik untuk *microprocessor* yang terletak di dalam *smart card*, yang pada akhirnya *smart card* dapat menyimpan dan memproses informasi dengan menggunakan kunci kriptografi dan algoritma yang menyediakan tanda tangan digital untuk digunkan dengan enkripsi yang lain.

Makalah ini tersusui sesbgai berikut. Sejarah mengenai *smrat card* dijelaskan pada bab dua. Arsitektur pada *smart card* dijelaskan pada bab tiga. Protokol yang digunakan dalam *smart card* dalam berkomunikasi dijelaskan pada bab empat. Standar yang digunakan dalam *smart card* diberikan pada bab lima. Serangan dalam *smart card* yang bisa ditangani dengan kriptografi dujelaskan pada bab enam, beserta cara penanganannya. Manajemen kunci kriptografi dalam *smart card* dipaparkan pada bab tujuh, dan kesimpulan diberikan pada bab delapan.

2. Sejarah Kartu Cerdas (*Smart Card*)

Smart Card bukanlah sebuah barang hasil penemuan baru. Di tahun 1974, seorang jurnalis Perancis Roland Moreno menemukan sebuah sistem pembayaran revolusioner, dimana sebuah aplikasi elektronik ditanamkan dalam sebuah benda semacam lingkaran. Namun, baru pada Maret 1998 inovasi awal dari *smart card* ini kembali lagi (menjadi bentuk lingkaran), dengan dikenalkannya Java Ring (Gambar 3) pada saat konferensi Sun's JavaOne.

Gambar 2. Bentuk Pertama *Smart Card*

Gambar 3. Java Ring

Pada tahun 1975 kartu pertama dengan bentuk seperti kartu kredit dengan *chip* dan *printed circuit*-nya yang dibuat oleh perusahaan Perancis, CII-Honeywell-Bull.

Gambar 4. *Smart Card* Hasil dari CII-Honeywell-Bull

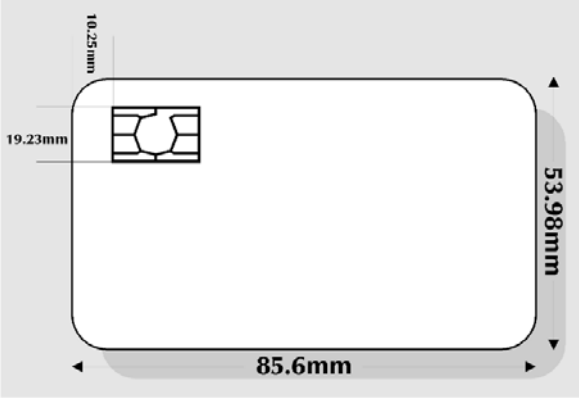
Penggunaan pertama dari *smart card* terjadi pada tahun 1983 dimana kartu ini digunakan secra luas oleh penduduk Perancis sebagai alat pembayaran telepon, Télécarte. Penggunaan keduanya terjadi pada tahun 1992 dengan integrasi antara *microchip* dengan semua kartu debit di Perancis, Carté Bleue. Penggunaan *smart card* meledak pada era 90an, pada saat dikenalkannya *smart card* berbasis *SIM* (*Subscriber Identity Module card*) yang digunakan untuk telepon selular, khususnya pada saat itu di Eropa. Dengan semakin umumnya penggunaan telepon selular di Eropa, *smart card* menjadi semakin dikenal.

3. Arsitektur Kartu Cerdas (*Smart Card*)

Meskipun istilah *smart card* sudah digunakan secara umum, namun masih ada beberapa istilah yang digunakan untuk *smart card* masih ambigu dan memiliki arti yang berbeda-beda. ISO menggunakan istilah *ICC* (*Integrated Circuit Card*) untuk mencakup semua alat yang memiliki *integrated circuit* yang termaktub dalam ISO 1, identifikasi kartu berbahan plastik. Kartu ini memiliki dimensi 85,60mm x 53,98mm x 0,76mm yang secara umum sama seperti kartu yang digunakan dalam dunia perbankan sebagai alat pembayaran atau transaksi keuangan.

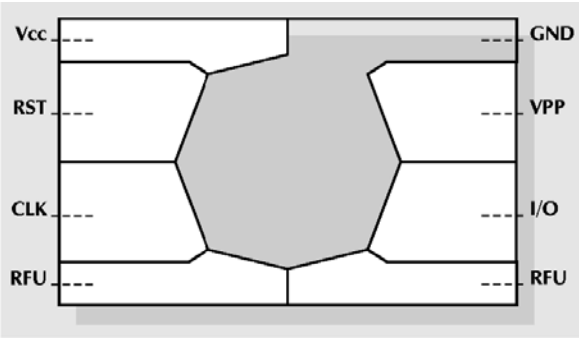
Integrated Circuit Card dibagi menjadi dua tipe, *contact* dan *contactless*. Kedua tipe ini dapat secara mudah dibedakan dari bentuknya karena adanya lempengan koneksi emas (Gambar 5) untuk tipe *contact*. Meskipun standar ISO (7816-2) mendefinisikan ada delapan kontak, namun hanya enam yang kontak yang benar-benar digunakan sebagai sarana komunikasi dengan lingkungan luar. Sedangkan untuk tipe *contactless*, kartu memiliki baterainya sendiri. Secara umum energi untuk mengoperasikan kartu *contactless* disediakan oleh putaran induksi radiasi elektromagnet.

Smart card memiliki *microprocessor* yang ditanamkan ke dalamnya, dan membutuhkan energi untuk menjalankan beberapa mekanisme komunikasi, menerima, dan mengirim data. Untuk tipe *contact smart card*, ada lempengan emas di bagian pojok kartu, sebagai lapisan kontak dengan *card reader*. Seperti yang telah disinggung pada Pendahuluan, piringan emas ini merupakan penghubung dengan *reader* untuk menyediakan energi listrik yang cukup untuk berkomunikasi secara langsung dengan *reader*. *Reader* untuk *contact smart card* biasanya merupakan alat terpisah yang ditancapkan pada port USB atau *built-in reader* seperti pada PDA atau telepon selular. Sedangkan untuk tipe *smart card* yang tidak memiliki lapisan kontak pada permukaannya atau disebut *contactless smart card*., koneksi antara *reader* dengan kartu dilakukan melalui *RF* (*Radio Frequency*). Jenis kartu ini memiliki putaran kawat yang ditanamkan di dalam kartu. Putaran kawat ini digunakan sebgai induktor untuk menyediakan energi bagi kartu dan sebagai sarana komunikasi dengan *reader*.



Gambar 5. Kartu ISO ID 1

Contact Smart Card merupakan yang paling sering dilihat karena banyaknya pengguna telepon selular yang tentunya memiliki kartu *SIM* (*Subscriber Identity Module card*). Umumnya *contact smart card* memiliki delapan kontak seperti dipaparkan pada Gambar 6.



Gambar 6. *Connector* ISO 7816-2

VCC = *Power supply*
GND = *Ground* atau *reference voltage*
CLK = *Clock* atau *timing signal*
VPP = *Programming voltage*
RST = *Reset signal*

I/O = *Serial Input/Output*

Vcc merupakan sumber tegangan yang mengendalikan *chip*, secara umum besarnya 5 volt. Dengan adanya pengembangan lanjut terhadap teknologi semikonduktor, memungkinkan besar tegangan yang dibutuhkan hanya 3 volt saja. GND merupakan *ground reference voltage* yang berlawanan dengan Vcc dimana energi potensialnya dikukur. *Reset* merupakan saluran sinyal yang digunakan untuk memulai *state* dari *integrated circuit* setelah menyala.

Clock signal digunakan untuk mengendalikan logika dari *IC* dan juga digunakan sebagai referensi pada hubungan komunikasi serial. Ada dua jenis kecepatan *clock* yang umum digunakan yaitu 3,57 MHz dan 4,92 MHz. Penghubung Vpp digunakan untuk sinyal dengan tegangan tinggi yang cocok untuk memprogram memori jenis *EPROM* (jenis memori akan dijelaskan pada bagian selanjutnya). Terakhir adalah penghubung *serial input/output* yang merupakan saluran sinyal dimana *chip* menerima perintah dan pertukaran data dari lingkungan luar.

***Power Supply* (*Vcc*)**

Power Supply untuk IC ini diberikan antara 4,75 volt sampai 5,25 volt. Dengan penggunaan arus listrik maksimum 200 mA. Kedua parameter ini memiliki masalah. Teknologi *chip* keluaran terbaru beroperasi dengan tegangan 3 volt, sehingga membutuhkan arus listrik yang lebih kecil. Kebanyakan *CAD* (*Card Acceptor Device*) atau alat pembaca kartu, beroperasi dalam tegangan 5 volt sesuai standar dari ISO.

***Clock Signal* (*CLK*)**

Meskipun *integrated circuit* memiliki *colck circuit*-nya sendiri untuk menjalankan logika internal, namun kebanyakan *IC chip* dilengkapi dengan *clock* eksternal dari perangkat antarmuka. Kecepatan komunikasi pada saluran *I/O* ditentukan oleh frekuensi dari *clock* ini. ISO memiliki standar penggunaan frekuensi dalam *clock* yaitu 3.579545 MHz dan 4.9152 MHz. Frekuensi dari *clock* dapat diubah sesuai kebutuhan tipe protokol komunikasi yang dipilih.

***Programming Voltage* (*Vpp*)**

Sinyal ini dirancang untuk menydiakan tegangan tinggi yang dibutuhkan untuk memungkinkan penulisan terhadap memori *volatile* (jenis memori akan dijelaskan pada bagian selanjutnya). Kebanyakan *IC* menggunakan memori jenis *EEPROM* dimana tegangan yang tinggi dibangkitkan dengan mengisi dari *chip*. Namun, untuk jenis memori *EPROM* dibutuhkan tegangan yang sangat tinggi (biasanya 12,5 V atau 21 V) yang disediakan dalam *IC Connector*.

***Reset Signal* (*RST*)**

Sinyal *reset* digunakan untuk memulai program yang ada di dalam *IC ROM*. Di dalam standar ISO didefinisikan tiga mode *reset*, *internal reset*, *active low reset*, dan *synchronous high active reset*. Kebanyakan *microprocessor* menggunakan mode *active low reset* dimana *IC* memindahkan kontrol ke alamat (*address*) masukan untuk program ketika *reset signal* berada pada tingkat tegangan yang tinggi. Urutan operasi untuk mengaktifkan dan menon-aktifkan *IC* didefinisikan untuk memperkecil kemungkingan kerusakan pada *IC*. Proses aktivasi untuk perangkat antarmuka adalah sebagai berikut,

1. Ambil *RST low*
2. Gunakan *Vcc*
3. Set *I/O* pada mode *receive*
4. Set *Vpp* dalam mode *idle*
5. Gunakan *clock*
6. Ambil *RST high* (*active low reset*)

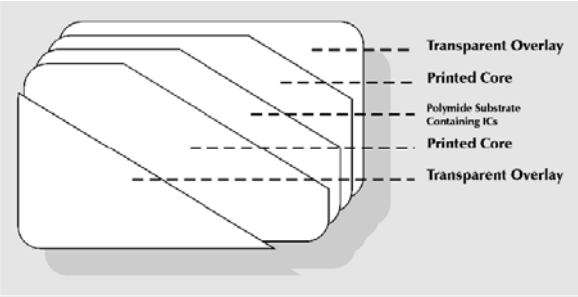
Dan untuk deaktivasi *IC* adalah sebagai berikut,

1. Ambil *RST low*
2. Ambil *clock low*
3. Deaktivasi *Vpp*
4. Set *I/O* dalam status *low*
5. Deaktivasi *Vcc*

***Serial Input/Output* (*I/O*)**

Dalam standar ISO didefinisikan satu saluran untuk pertukaran data antara *IC* dan perangkat antarmuka. Ini berarti bahwa saluran tersebut haruslah mengubah arah sesuai dengan situasi apakah *IC* sedang mengirim atau sedang menerima. Untuk itulah diperlukan protokol transmisi yang akan dijelaskan pada bagian selanjutnya dalam makalah ini.

Jenis yang kedua dari *Integrated Circut Card* (*ICC*) memiliki arsitektur yang sedikit berbeda, hal ini diakrenakan tidak adanya lemepengan emas seperti pada *contact smart card*. Untuk *contactless smart card*, bentuk dasarnya adalah kartu ini tersususn atas beberapa lapisan yang memiliki fungsi berbeda-beda.



Gambar 7. Lapisan dalam *contactless smart card*

Kegunaan dasar dari *IC card* adalah sebagai media penyimpanan *portable* dan tentunya sebagai media penerima data. Oleh sebab itu komponen pokok dari *IC* adalah modul memori. Berikut ini daftar yang menggambarkan tipe memori yang umumnya digunakan,

ROM = Read only memory (*mask ROM*)
PROM = Programmable read only memory
EPROM = Erasable programmable ROM
EEPROM = Electrically erasable PROM
RAM = Random access memory

Sebuah *chip* memungkinkan untuk memiliki lebih dari satu jenis memori. Tipe memmori memiliki ciri yang mengontrol metode yang digunakan memori tersebut. Tipe memori *ROM* merupakan yang sudah permanen dan tidak diubah lagi sejak dibuat oleh perusahaan semikonduktornya. *ROM* merupakan jenis memori yang *low-cost*, sehingga memori ini menempati sedikit ruangan dalam inti silikon. Kelemahan dari memori jenis ini tentunya dengan kepermanennannya sehingga tidak dapat diubah dan diperlukan bebrulan-bulan untuk sebuah perusahaan semikonduktor memproduksi lagi.

PROM merupakan memori yang dapat diprogram oleh pengguna. Namun, tegangan dan arus listrik yang tinggi dibutuhkan untuk beberapa pemrograman dan memori ini kurang cocok digunakan dalam *ICC* (*Integrated Circuit Card*). *EPROM* telah secara luas digunakan pada masa lalu. Memori jenis ini digunakan dalam mode *OTP* (*One Time Programmable*). Memori jenis *EEPROM* merupakan jenis memori yang dapat dihapus oleh pengguna dan dapat ditulisi kembali berulang-ulang. Semua jenis memori yang telah disebutkan merupakan memori *non volatile*, yaitu ketika daya dimatikan memori tersebut masih memiliki data di dalamnya. Satu jenis memori yang lain yaitu *RAM* yang merupakan memori *volatile* dimana ketika daya dimatikan, maka data di dalam meomori tersebut ikut terhapus.

Tentunya hanya dengan bermodalkan memori kecil *EEPROM* (128 – 512 byte) dan memori kontrol logika tidak akan didapatkan apa-apa. Untuk penerapan yang lebih lanjut dibuthkan *ROM*, *EEPROM*, *RAM*, dan *CPU* (*Central Processing Unit*). Dengan tambahan *CPU* atau *micro-controller* ini barulah sebuah kartu dinamakan "*smart*".

Pengontrol logika dibutuhkan tidak hanya dibutuhkan dalam protokol komunikasi tetapi juga dibutuhkan sebagai pelindung memori terhadap penyalahgunaan kartu. Berdasarkan isi dari *ICC* (*Integrated Circuit Card*) dapat dibedakan lagi menjadi tiga jenis,

1. Memori saja
2. Memori dengan logika keamanan (*security logic*)
3. Memori dengan *CPU*

Logika keamanan dapat digunakan dalam pengontrolan akses ke memori untuk penggunaan yang diizinkan saja. Biasanya diselesaikan oleh beberapa bentuk kode akses yang memiliki ukuran cukup besar (64 bit atau lebih). Penggunaan memori *EEPROM* harus secara ketat dikontrol untuk dapat mencegah penyalahgunaan yang dilakukan oleh orang yang tidak berkepentingan yang bermaksud mendapatkan keuntungan finansial. Hal ini berlaku juga seperti kartu telepon sebagai penerapan penggunanan *ICC* sebagai pembawa kunci kriptografi. Keuntungan dalam hal keamanan dengan penggunaan *CPU* tentu saja lebih signifikan karena *CPU* mampu untuk mengimplementasikan algoritma kriptografi, hal ini akan dibahas pada bagian selanjutnya.

4. Protokol Komunikasi dalam Kartu Cerdas (*Smart Card*)

Sifat elektronik dan karakteristik transmisi dari *ICC* (*Integrated Circuit Card*) merupakan pokok terhadap kemampuan operasi sebuah kartu. Spesifikasi ini didefiniskan dalam standar ISO 7816. Di dalam standar ini dijelaskan tentang protokol komunikasi T=1 dan tinjauan untuk *protocol type selection* (*PTS*). Subyek yang dibahas dalam standar ini adalah,

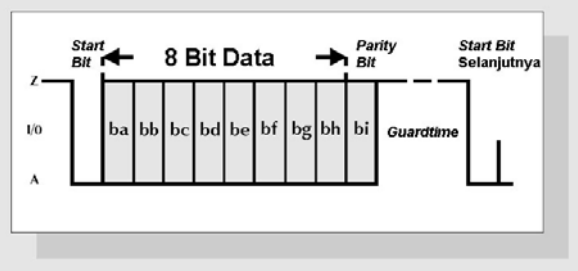
1. Karakteristik yang berhubungan dengan listrik (sudah dijelaskan pada bagian arsitektur kartu cerdas)
2. Karakter transmisi
3. *Answer to reset* (*ATR*)

4. Protokol komunikasi T=0
5. Protokol komunikasi T=1
6. *Protocol type selection* (*PTS*)

Dimulai dengan penjelasan karaketeristik transmisi yang dijalankan oleh kebanyakan *microprocessor ICC*. Karakteristik transmisi yang dijalankan oleh kebanyakan *mocropocessor IC* adalah berdasarkan mode operasi *asynchronous half duplex*. Dalam protokol komunikasi T=0 melibatkan transmisi byte, sedangkan protokol T=1 mendefinisikan operasi mode blok. Seperti yang sudah disinggung dalam penjelasan arsitektur *smart card* khususnya bagian *clock signal*, bahwa komunikasi serial dioperasikan dengan penggunaan konektor *chip* tunggal, dimana arah transmisi data harus diubah sesuai keadaan apakah *IC* atau perangkat antarmuka yang sedang mengirimkan data. Komunikasi seperti ini disebut sebagai komunikasi *half duplex*, padahal dibutuhkan dua sinyal *I/O* untuk menjalankan operasi *full duplex* dimana transmisi dapat dijalankan dua arah secara bersamaan.

Tipe asinkron dari transmisi sama dengan yang digunakan pada konektror serial RS232C di *Personal Computer* (*PC*). Namun, *PC* beroperasi dalam mode *full duplex*. Transmisi dari sebuah karakter tunggal (8 bit) membutuhkan beberapa operasi bit sbb,

1. *Start bit* (digunakan untuk sinkronisasi *frame* karakter)
2. *Parity bit* (untuk pendeteksiar kesalahan)
3. *Guardtime* (pemisahan antar karakter)



Gambar 8. *Frame* karakter asinkron

Format dari *frame* karakter dipaparkan pada Gambar 8. Penerima memeriksa pencarian transisi *I/O* dari tanda atau status tinggi rendahnya. *Parity Bit* didefinisikan untuk mendapatkan kesamaan genap yang berarti bahwa jumlah 1 di dalam dalam 8 bit data dan *parity bit* bersama harus menghasilkan nilai genap.

Guardtime didefinisikan untuk sama dengan dua periode bit, hal ini sama seperti dua *stop bit* dalam *UART* (*Universal Asynchronous Receiver Transmitter*) yang digunakan dalam *PC*.

Half duplex hanya menunjuk pada transmisi data satu arah pada satu waktu yang mana *PC* sangat mampu untuk mengatur dengan *UART*-nya. RS232C memiliki dua kabel terpisah untuk transmisi penerimaan yang membutuhkan modifikasi perangkat keras untuk menjadi antarmuka *IC* tunggal secara langsung.

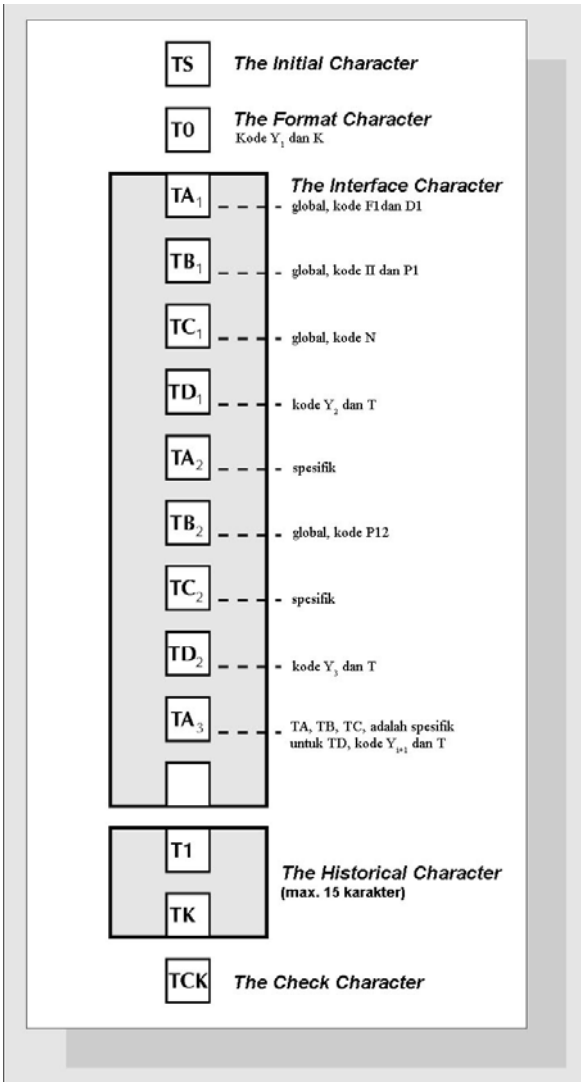
Telah dijelaskan pada bagian arsitektur *smart* card, bahwa sinyal *reset* merupakan sinyal untuk memulai status untuk menjalankan program pada *IC ROM*. Setelah sinyal *reset* dijalankan oleh perangkat antarmuka, *ICC* (*Integrated Circuit Card*) merespon dengan menjalankan *answer to reset* (*ATR*) atau pembuka untuk proses *reset*. Untuk mode *low reset* aktif, *ICC* harus merespon antara 400 sampai 40.000 putaran *clock* setelah dijalankannya sinyal *reset*. *ATR* dijalankan paling banyak dalam 33 karakter termasuk karakter inisialisasi dan memiliki lima bagian, yaitu:

1. *The initial character* (*TS*)
2. *The format character* (*TO*)
3. *The interface characters* (*TAi*,*TBii*,*TCi*,*TDii*,)
4. *The historical characters* (*T1*,*T2*. *TK*)
5. *The check character* (*TCK*)

Masing-masing bagian ini dikirimkan sesuai urutan yang dipaparkan pada Gambar 9 (pada halaman selanjutnya).

***The initial character* (*TS*)**

TS (*The Initial Character*) merupakan bentuk bit sinkronisasi yang dikirimkan sesuai urutan untuk menentukan kecepatan transmisi data juga untuk menentukan kepekaan logika. Format dari karakter *TS* ditunjukkan pada Gambar 10 (pada halaman selanjutnya). Pada gambar ini ditunjukkan dua kemungkinan dari konvensi, konvensi langsung dan konvensi terbalik. Pada konvensi terbalik dimana level logika berada pada level 1 yang merupakan *low state*, bit yang pertama kali ditransmisikan adalah bit yang paling signifikan. Dengan konvensi langsung, dimana pada level 1 merupakan *high state* maka bit yang kurang signifikan yang ditransmisikan terlebih dahulu.



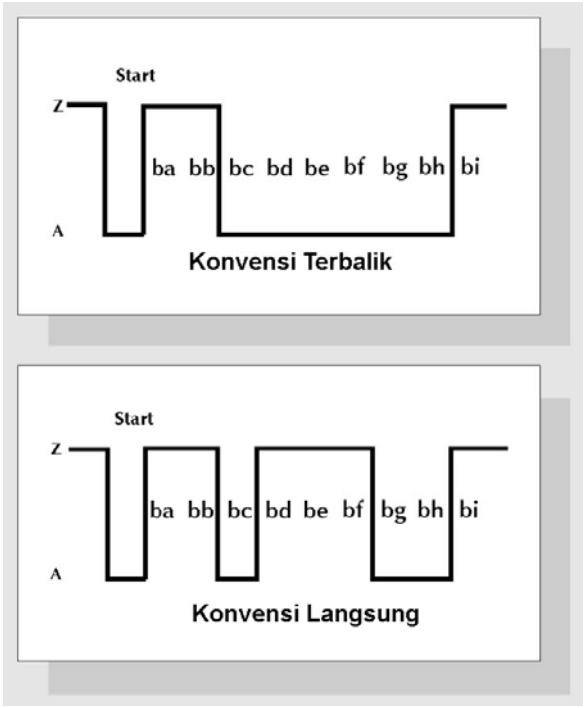
Gambar 9. Konfigurasi umum *ATR*

***The format character* (*TO*)**

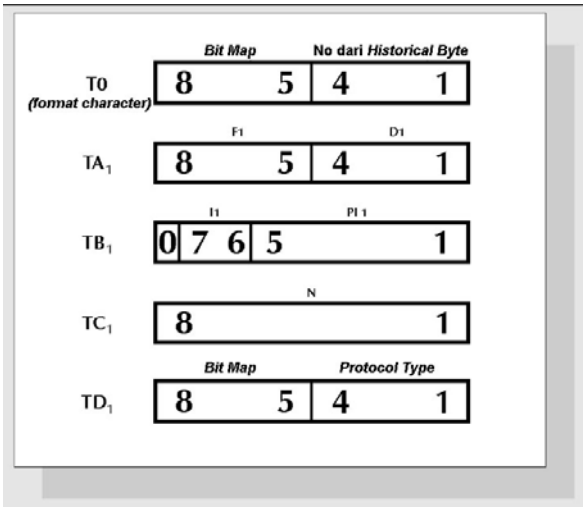
TO (*The Format Character*) menyediakan informasi yang dibutuhkan untuk menerjemahkan karakter yang tersisa dalam *ATR*.

***The interface characters* (*TAi,TBii,TCi,TDii,*)**

Ada beberapa revisi dalam ISO 7816-3 untuk menghilangkan ambigu dan memastikan metode operasi yang efektif untuk mengubah tipe protokol dan parameter protokol. Kebanyakan kompleksitas disesuaikan dengan implementasi komersial dari protokol komunikasi T=0 dan T=1. Revisi ini menyangkut *interface byte* dan *protocol type selection* (*PTS*). *Interface byte* dipaparkan pada Gambar 11



Gambar 10. *TS* (*Initial Character*)



Gambar 11. *Interface Character*

Karakter T0 dan TDi mengandung *Bit Map* yang mengindikasikan adanya byte TA_1, TB_1, TC_1, dan TD_1. Karakter TA_1, TB_1, TC_1, dan TD_1 memiliki *interface byte* global yang merupakan dasar operasi dari *smart card*.

TA_1 mendefinisikan karakter dasar dari transmisi serial F1 merupakan faktor konversi *clock rate*, dan D1 merupakan faktor penyesuaian *bit rate*. Penyandian biner dari bagian-bagian untuk mendapatkan harga sebenarnya untuk F dan D didefiniskan sbb,

$$Initial\ etu = \frac{372}{f} \text{ detik } (f \text{ biasanya} = 3.579545\text{MHz})$$

$$Work\ etu = \frac{1}{D} x \frac{F}{f} \text{ detik}$$

Elementary time unit (*etu*) merupakan satuan durasi bit yang digunakan dalam *frame* karakter. Jadi jika dijelaskan dari Gambar 8 pada halaman 6, satu frame karakter sama dengan 12 *etu* (1 *start etu*, 8 data *etu*, 1 *parity etu*, 2 *guardtime etu*).

TB_1 digunakan untuk mendefinisikan tegangan dan arus pada pemrograman *EPROM*. TC_1 menyediakan nilai untuk N yang mendefinisikan tambahan *guardtime* untuk digunakan antara karakter yang berurutan. N bisa berada dalam jarak nilai 0-254 *etu*. Ketika N sama dengan 255, ini mengindikasikan bahwa *guardtime* yang minimum (2 *etu* untuk T=0 dan 1 *etu* untuk T=1) harus digunakan.Protokol komunikasi T=0 membutuhkan *guardtime* tambahan untuk menjalankan deteksi kesalahan kesamaan (*parity error*).

TD_1 mengindikasikan tipe protokol antara 0 sampai 15 :

1. T=0 Transmisi byte *half duplex* asinkron
2. T=1 Transmisi blok *half duplex* asinkron
3. T=2/3 Hanya untuk operasi *full duplex*
4. T=4 Hanya untuk mempertinggi transmisi byte *half duplex*
5. T=5..13 Hanya untuk penggunaan lebih lanjut
6. T=14 Protokol selain ISO
7. T=15 Hanya untuk pengembangan lebih lanjut

Byte TD_1 juga mengandung *bit map* yang mengindikasikan adanya TA_2, TB_2, TC_2, dan TD_2. Revisi dari ISO mendefinisikan penggunaan *inte4rface byte* TA_2 yang memiliki aturan khusus dalam pemilihan protokol komunikasi dan parameter komunikasi, akan dijelaskan pada bagian selanjutnya.

***The historical characters* (*T1,T2. TK*)**

The historical character digunakan untuk memberikan informasi yang berhubungan dengan siklus hidup kartu.

The check character (TCK)

The check character dikirimkan sebagai karakter terakhir dalam proses *ATR*.

Saat ini dua protokol komunikasi yang umum digunakan yaitu,

1. T=0 Transmisi byte *half duplex* asinkron
2. T=1 Transmisi blok *half duplex* asinkron

Protokol T=0 merupakan protokol yang disebutkan dalam ISO 7816-3 dan protokol T=1 sebagai amandemen 1 untuk ISO 7816-3.

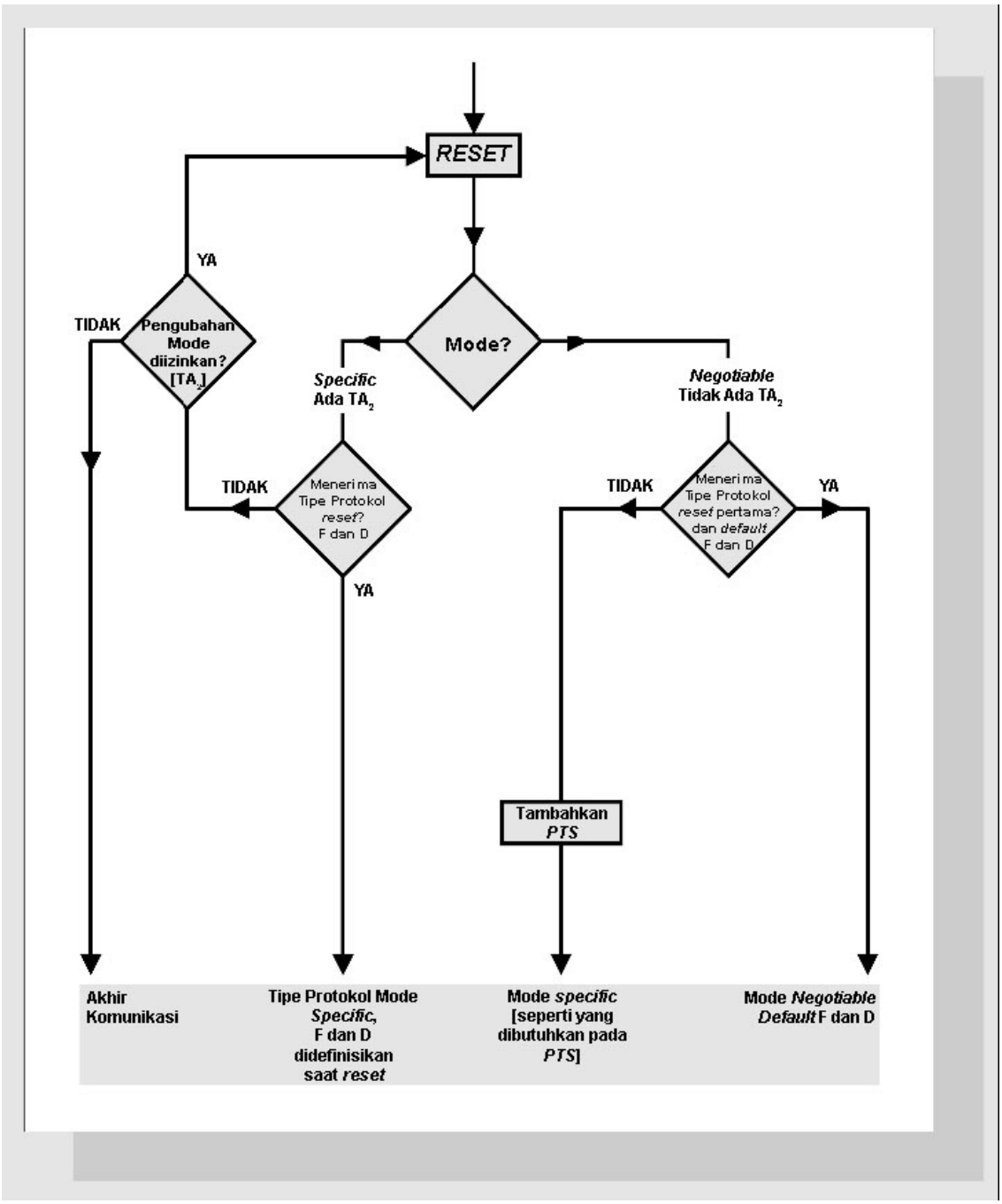
ICC (*Integrated Circuit Card*) dan perangkat antarmuka haruslah berkomunikasi menggunakan protokol. Metode yang digunakan untuk mendapatkan konfigurasi maksimal telah menjadi bahasan para ahli pada beberapa tahun ini. Hal ini dimaksudkan untuk didapatkan dari penggunaan *protocol type selection* (*PTS*). Ini merupakan perintah yang efektif untuk dikirimkan dari perangkat antarmuka ke *ICC* setelah *answer to reset* (*ATR*) terjadi. Untuk menjaga kesesuaian dengan sistem komersial yang ada, yang hanya mampu menangani protokol komunikasi T=0, beberapa perubahan dilakukan pada ISO 7816-3 yang asli. Konsep baru yang diajukan mengidentifikasikan prinsip dari dua mode operasi, yaitu:

1. Mode *Negotiable*
2. Mode *Specific*

Sebuah *ICC* yang beroperasi dalam mode *negotiable*, memungkinkan protokol komunikasi yang deapat diubah dengan penggunaan perintah *PTS*. *ICC* yang beroperasi dengan mode *specific*, tidak dapat menerima perintah *PTS* tapi dapat ditempatkan dalam mode *negotiable* dengan penanganan lebih jauh pada perintah *reset*.

Meskipun *ICC* menunjukkan pada perangkat antarmuka (dengan perantaraan TA_2) kemampuannya untuk mengubah mode ke mode *negotiable*, perangkat yang tersedia di pasaran mungkin tidak dapat menerima perubahan ini, sehingga tidak dapat menjalankan proses *reset*.

Operasi dari pengubahan mode ini ditunjukkan pada Gambar 12 (pada halaman 9). Sebagai catatan bahwa kartu *multi*-protokol yang dari definisi menyarankan mode *negotiable* dalam beroperasi, harus memberikan prioritas terhadap pprotokol komunikasi T=0. Dengan kata lain, jika tersedia protokol T=0, protokol ini haruslah menjadi protokol *default* dalam proses *ATR*.

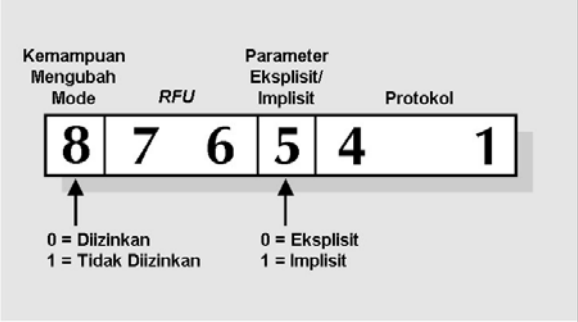


Gambar 12. Mode dari Operasi

Interface byte TA_2 yang merupakan bagian data dalam proses *ATR* memberikan informasi yang dibutuhkan untuk mengizinkan pemilihan yang tepat terhadap protokol yang digunakan. Kode dari *byte* ini saat muncul dipaparkan pada Gambar 13. Dengan adanya *byte* ini penentuan mode operasi dai kartu dapat digunakan sbb,

1. TA_2 ada pada *ATR* – Mode *Specific*
2. TA_2 tidak ada pada *ATR* – Mode *Negotiable*

Dapat diperlihatkan bahwa bit 8 di TA_2 digunakan untuk memberi tahu pada perangkat antarmuka apakah kartu dapat diubah ke mode *negotiable* atau tidak.



Gambar 13. *The Interface Byte* TA_2

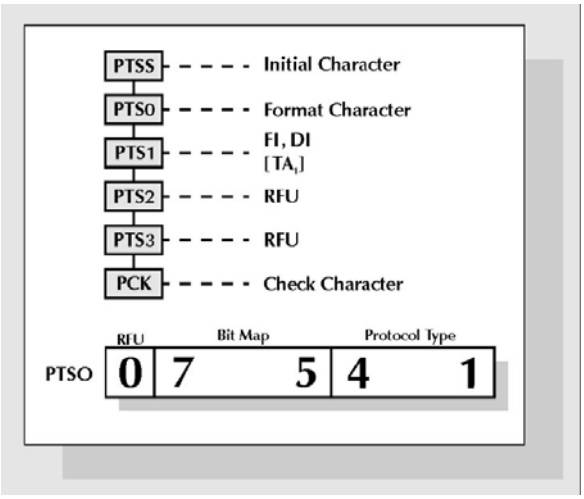
***Protocol type selection* (*PTS*)**

Protocol type selection (*PTS*) digunakan oleh perangkat antarmuka untuk protokol komunikasi dan nilai *default* dari F1 dan D1. Perintah *PTS* harus dikeluarkan segera setelah *ATR* dan ahanya diterapkan ketika *ICC* dalam mode *negotiable*.

Perangkat antarmuka mungkin memilih untuk beroperasi menggunakan protokol pertama yang teridentifikasi setelah *ATR* dan dengan menggunakan F dan D yang *default*. Untuk mendapatkan efek dari perubahan situasi ini, perangkat antarmuka harus menjalankan perintah *PTS*. Permintaan *PTS* ini mengandung sebuah karakter inisialisasi (*initial character*) *PTSS*, diikuti dengan karakter format (*format character*) *PTSO*, tiga karakter pilihan (*optional character*) *PTS1*, *PTS2*, dan *PTS3*, dan *PCK* yang merupakan karakter pengecek (*check character*). Ini dipaparkan pada Gambar 14. Respon dari *ICC* mengikuti format yang sama dengan permintaan.

karakter format (*format character*) *PTSO* disandikan seperti pada Gambar 14. *Bit map* digunakan untuk mengindikasikan adanya *PTS1*, *PTS2*, dan *PTS3* yang disandikan pada bit 5, 6, dan 7. Sedangkan tipe protokol disandikan pada bit 0, 1, 2, 3, dan 4.
Karakter *PTS1* jika ada, digunakan untuk mendefinisikan nilai untuk F1 dan D1 sebagai kode untuk TA_1 (sudah dipaparkan pada bagian sebelumnya). Parameter ini digunakan untuk mendefinisikan *work etu* (*elementary time unit*).

Karakter pengecek (*check character*) *PCK* dihitung dengan meng-*XOR*-kan semua karakter dari *PTSS* dengan *PCK* sampai didapatkan hasil sama dengan nol.



Gambar 14. Respon *PTS*

Ketika *ICC* mengimplementasikan permintaan *PTS*, kartu mengembalikan permintaan yang sama sebagai respon. Jika bit ke-5 dari karakter respon *PTS1*diset pada nol maka nilai *default* dari D dan F yang akan digunakan.

Protokol Komunikasi T=0

Perangkat antarmuka selalu memulai perintah untuk protokol komunikasi T=0. Interaksi antara perangkat antarmuka dengan *ICC* menghasilkan perintah dan respon yang berurutan. Untuk protokol komunikasi ini data hanya dapat mengalir satu arah saja untuk pasangan respon dari perintah. Dengan kata lain salah satu dari pesan perintah mengandung data untuk *ICC* atau perintah yang meminta data dari *ICC* yang akhirnya dimasukkan ke dalam respon. Arah dari aliran data ini bersifat implisit pada definisi dari perintah, oleh karenanya perangkat antarmuka dan *ICC* diharuskan memiliki pengetahuan tentang prioritas. Ketika dibutuhkan untuk mentransfer data dalam dua arah untuk perintah khusus selanjutnya

penerima respon perintah mungkin digunakan setelah perintah dasar untuk memperbaiki data respon.

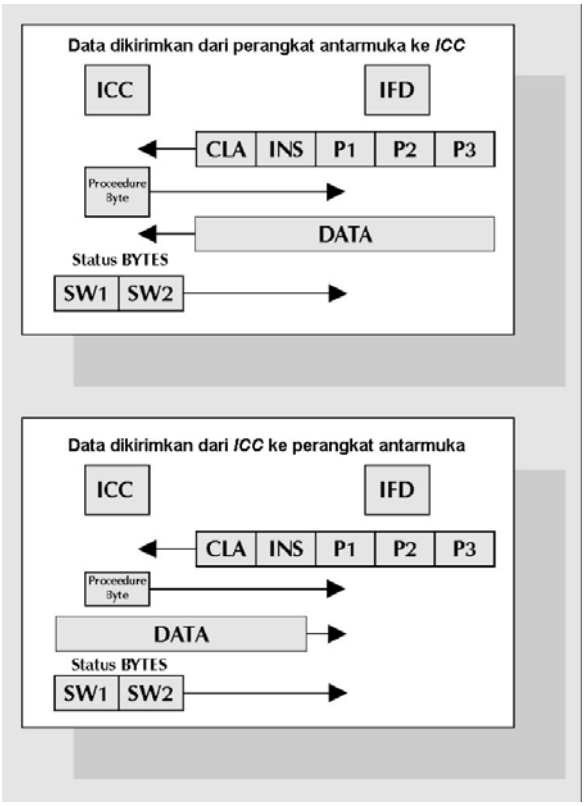
Pesan untuk perintah mengandung 5 karakter *header* yang dikirimkan oleh perangkat antarmuka ke *ICC*. Selanjutnya *ICC* membalas dengan prosedur byte setelah data telah dikirimkan ke *ICC* atau dari *ICC*, bergantung pada perintah dasarnya. Prosedur byte ini bertujuan untuk mengizinkan perangkat anatarmuka untuk mengontrol pemrograman tegangan *Vpp EPROM*. Aliran pesan untuk protokol T=0 dipaparkan pada Gambar 15. Perintah *header* mengandung 5 byte sbb,

1. CLA - *the instruction class*
2. INS - *the instruction code* (*read memory*)
3. P1 - *instruction code qualifier* (*memory address*)
4. P2 - tambahan *INS code qualifier*
5. P3 – panjang blok data

Ada dua status byte *SW1* dan *SW2*. byte ini dikirimkan dari *ICC* ke perangkat antarmuka sebagai pelengkap perintah untuk mengindikasikan status terakhir kartu. Respon yang normal adalah,

SW1, *SW2* = 90 *hex*, 00 *hex*

Ketika *SW1* = 6x atau 9x, beberapa macam kondisi kesalahan (*error*) dilaporkan oleh kartu.



Gambar 15. Protokol Komunikasi T=0

Protokol T=0 juga memiliki pendeteksi kesalahan dan mekanisme koreksinya.

Protokol Komunikasi T=1

T=1 merupakan protokol komunikasi blok *half duplex* asinkron. Protokol ini memiliki model dengan operasi dua layer, *data link layer* dan *Physical layer*. *Phisical layer* beroperasi sama seperti protokol T=0 kecuali untuk pendeteksian kesalahan dan koreksi. Protokol ini memberikan segel di sekitar blok karakter yang memungkinkan untuk:

1. *flow control*
2. *block chaining*
3. *error correction*

Keuntungan dari protokol T=1 ini adalah kemampuannya untuk mengontrol aliran data pada dua arah. Protokol T=1 ini juga menghilangkan keterbatasan dimana perangkat antarmuka selalu yamg memulai perintah untuk respon dari *ICC*. Untuk protokol blok ini, perintah bisa dimulai oleh perangkat antarmuak ataupun *ICC*.

Keuntungan yang lain dari protokol T=1 ini adalah kemampuannya untuk merangkai blok data sehingga blok data yang besar dapat dikirimkan sebagai hasil dari perintah tunggal dari transmisi dengan jumlah *frame* terurut yang tepat.

Protokol blok ini juga memiliki sistem pengaturan kesalahan yang lebih bagus. Hal ini memungkinkan penggunaan blok *error detection code* (*EDC*) dan kemampuan untuk mentrasnmisikan kembali blok yang berada dalam kondisi yang salah (*error*).

Namun ada harga yang harus dibayar untuk protokol dengan layer yang lebih tinggi ini. Terlepas dari rumitnya perangkat lunak dalam *ICC* dan perangkat antarmuka, protokol ini membutuhkan memori *RAM* pada *ICC* yang lebih besar. Memori besar ini dibutuhkan untuk menjaga blok yang dikirimkan terakhir dalam hal pengiriman kembalinya. Secara umum protokol T=1 menawarkan keuntungan dimana aplikasinya digunakan untuk mengatur blok data yang besar khususnya ketika dibutuhkan dalam tranfer data dua arah sebagai bagian dari perintah dasar. Efisisensi dari protokol ini terlihat dalam transmisi data yang lebih besar karena yang mendasari *physical layer* masih beroperasi dalam mode karakter seperti protokol T=0. Penanganan kesalahan telah ditingkatkan secara signifikan dibandingkan protokol T=0.

Blok *frame* dalam protokol T=1 terdiri dari tiga bagian, yaitu:
1. *prologue field*
2. *information field* (pilihan)
3. *epilogue field*

Tabel 1. Bagian blok *frame* dalam protokol T=1

prologue field			***information field***	***epilogue field***
Node Address	*Protocol Control Byte*	*Data Length*	(pilihan)	*Error Detetection* *LRC* atau *CRC*
NAD	*PCB*	*LEN*	*INF*	*EDC*
1 byte	1 byte	1 byte	0-254 byte	1 atau 2 byte

Prologue field memiliki tiga byte:
1. *NAD* (*node address*)
2. *PCB* (*protocol control byte*)
3. *LEN* (*data length*)

Byte *NAD* menggunakan bit 3-1 untuk mengidentifikasi alamat sumber dan bit 7-5 untuk megidentifikasi alamat tujuan. Sedangkan bit 4 dan 8 digunakan untuk kontrol *Vpp*.

Byte *PCB* mengizinkan identifikasi tiga tipe *frame* blok,
1. *information block* (*I - block*)
2. *receive ready block* (*R - block*)
3. *supervisory block* (*S - block*)

Information Block merupakan *frame* yang digunakan untuk mentransfer perintah aplikasi dan data antara *ICC* dan perangkat antarmuka. *Receive ready block* digunakan sebagai tanda ketika protokol sedang mengirimkan data sebagai urutan rangkaian blok. S*upervisory block* digunakan untuk membangun parameter kontrol dan untuk mempengaruhi proses sinkronisasi kembali dan status pembatalan sebagai hasil dari beberapa kondisi yang salah (*error*).

protocol control byte (*PCB*) mengidentifikasikan tipe yang berbeda dari blok dan membawa beberapa informasi kontrol termasuk bit tunggal urutan jumlah (*N*) dan bit rangkaian blok (*M*). Bit yang lain digunakan untuk megidentifikasi kesalahan transmisi. *PCB* dikodekan sbb,

Tabel 2. Fungsi bit dalam *PCB*

Tipe	***PCB* (bit 8-1)**								**Fungsi**
I	0	N	0	0	0	0	0	0	I blok standar
I	0	N	1	0	0	0	0	0	Set rangkaian bit
R	1	0	0	N	0	0	0	0	Tidak ada *error*
R	1	0	0	N	0	0	0	1	*EDC*
R	1	1	0	N	0	0	1	0	*Error* yang lain
S	1	1	0	0	0	0	0	0	Permintaan sinkronisasi kembali
S	1	1	1	0	0	0	0	0	Respon sinkronisasi kembali
S	1	1	0	0	0	0	0	1	Permintaan *IFS*
S	1	1	1	0	0	0	0	1	Respon *IFS*
S	1	1	0	0	0	0	1	0	Pembatalan permintaan
S	1	1	1	0	0	0	1	0	Pembatalan respon
S	1	1	0	0	0	0	1	1	Permintaan *WTX*
S	1	1	1	0	0	0	1	1	Respon *WTX*

I-block dapat menjadi sebagai blok independen atau sebagai bagian dari rangkaian blok. Nomor urutan dari pengirim bervariasi antara 0 dan 1, dan biasanya dimulai dengan 0. *R-block* digunakan untuk menyatakan keberhasilan penerimaan blok yang ditransmisikan. *S-block* digunakan untuk meminta status kontrol seperti yang telah disebutkan pada Tabel 2.

Byte *LEN* mengindikasikan jumlah dari byte dalam *information field* dari *frame*. Byte ini mengizinkan *information field* maksimum sebanyak 254 byte.

Information field digunakan untuk menyampaikan perintah aplikasi dan data. *Epilogue field* mengandung blok kode pendeteksi kesalahan yaitu *LRC* (*longitudinalredundancy check*) atau *CRC* (*cyclic redundancy check*).

Protokol T=1 menggunakan tiga karakter untuk membangun *interface character* yang digunakan dalam *ATR* untuk memulai komunikasi, yaitu:
1. *TAi* = *IFSC* (*default* = 32)
2. *Tbi*
   (bit 4 - 1) = *CWI* (*default* = 13)
   (bit 8 - 5) = *BWI* (*default* = 4)
3. *TCi*
   (bit 1 = 1) = *CRC*
   (bit 1 = 0) = *LRC* (*default*)

IFSC merupakan bagian informasi ukuran untuk kartu. Juga ada *IFSD* yang merupakan bgian

informasi ukuran untuk perangkat antarmuka. Nilai *default*-nya adalah 32 byte dan hanya dapat diubah dengan perantaraan permintaan *S - block* dari perangkat antarmuka ke *ICC*.

Protokol T=1 ini menggunakan dua parameter *waiting time* untuk membantu kontrol aliran, yaitu:
1. *Character Waiting Time* (*CWT*)
2. *Block Waiting Time* (*BWT*)

CWT merupakan wakti maksimal antara karakter yang berurutan di dalam blok, sedangkan *BWT* waktu maksimal antara awal dari karakter terakhir dalam blok yang dikirimkan oleh perangkat antarmuka dengan awal dari karakter blok selanjutnya yang dikirimkan oleh kartu.

CWT dapat digunakan untuk mendeteksi kesalahan dalam panjang blok, sedangkan *BWT* dapat digunakan untuk mendeteksi karti yang tidak merespon. Selain itu ada juga *block guard time* (*BGT*) yang didefinisikan sebagai waktu minimal antara awal dari karakter terakhir dari blok dengan awal dari karakter pertama blok baru yang akan dikirimkan melalui arah alternatif. *CWT* dan *BWT* dihitung dari nilai *CWI* dan *BWI* dengan persamaan sbb,

$CWT = (2^{BWI} + 11)$ *etu*
$BWT = (2^{BWI} \times 960 \times 372 / f)$ detik + 11 *etu*

Dimana *f* adalah frekuensi *clock*.

Operasi dalam protokol T=1 dapat dirumuskan sbb, sesuai dengan standar yang diberikan ISO

I-block; I (N,M)

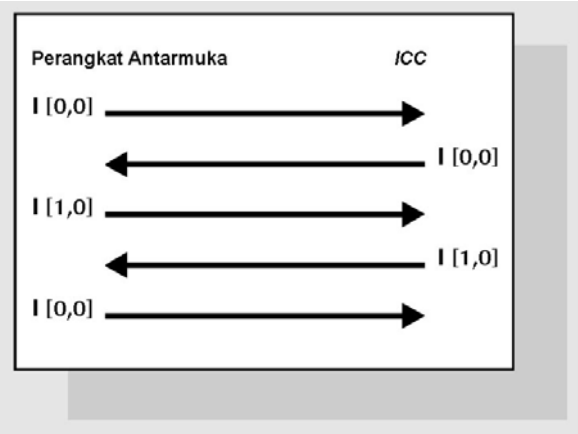
Dimana
N = nomor urutan (bervariasi antaraa 0 dan 1)
M = bit data lebih

Bit data lebih ditempatkan ketika ada tambahan *i-block* yang diikuti dengan rangkaian.

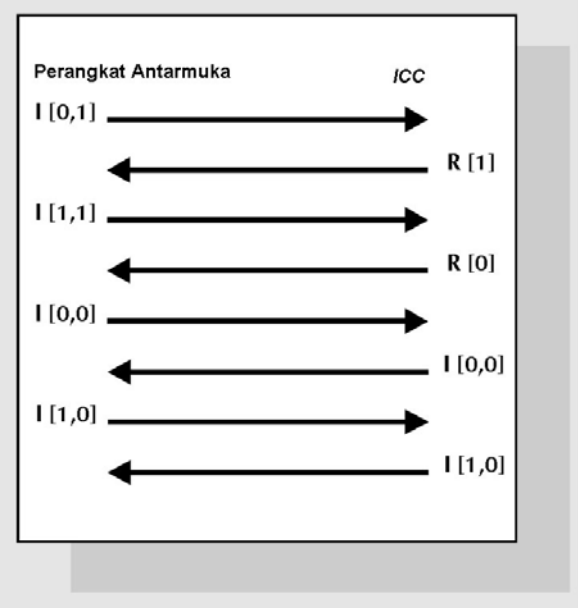
R-block; R (N)

Dimana
N = nomor urutan dari blok perkiraan selanjutnya

Dari Gambar 16 dapat dilihat bahwa perangkat antarmuka dan *ICC* masing-masing sudah bisa mentransmisikan data dimana komunikasi dimulai dengan transmisi blok dari perangkat antarmuka.

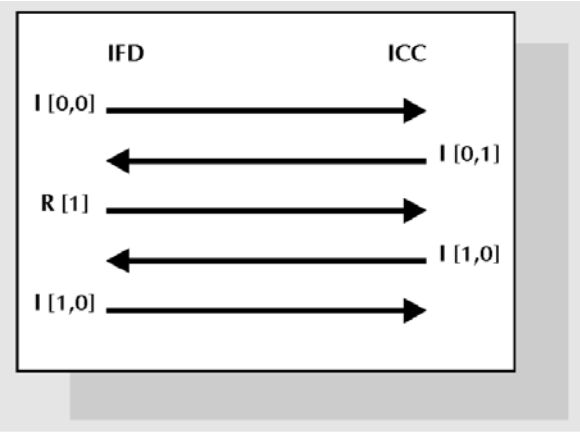


Gambar 16. Memulai komunikasi



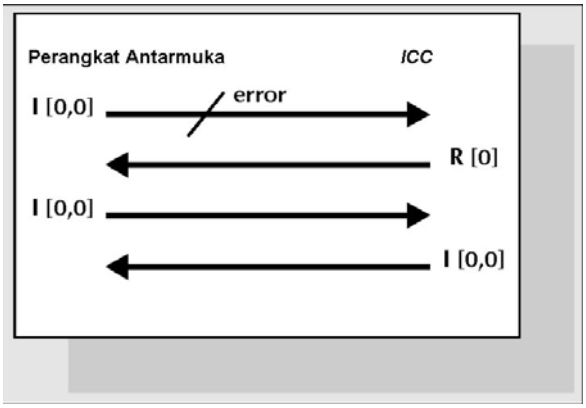
Gambar 17. Transmisi *i-block* normal

Perlu dicatat bahwa *i-block* digunakan oleh *ICC* untuk menyatakan blok terakhir dalam rangkaian yang dikirimnkan oleh perangkat antarmuka. *ICC* mungkin mengirimkan rangkaian blok dengan jalan yang sama seperti yang ditunjukkan oleh perangkat antarmuka.

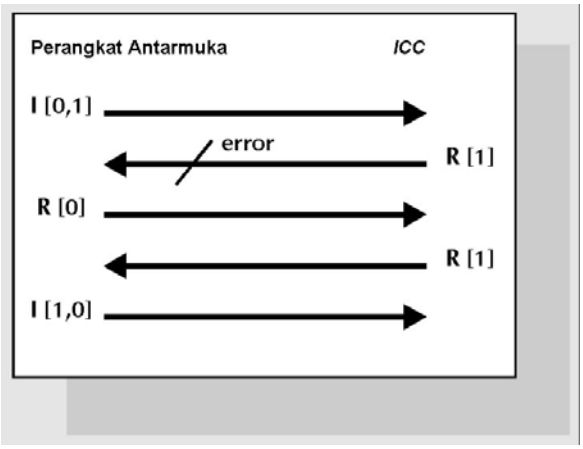


Gambar 18. Transmisi *i-block* dengan rangkaian

Penanganan kesalahan (*error*) dabagi menjadi penanganan kesalahan saat penerimaan *i-block* dan penanganan kesalahan saat respon rangkaian *i-block*.



Gambar 19. *Error* pada saat penerimaan *i-block*



Gambar 20. *Error* pada saat respon rangkaian *i-block*

Pada dua kasus kesalahan ini pengirim dinitifikasi untuk mengirimkan kembali blok yang *error* saat diterima. Tentunya banyak skenario *error* selain yang telah disebutkan, namaun pada dasarnya memiliki konsep yang sama.

5. Standar dalam Kartu Cerdas (*Smart Card*)

Seri standar ISO 7816 dan standar ETSI SCP merupakan yang paling penting dan paling relevan untuk *smart card*. Standar ISO adalah *International Standards Organization* sedangkan ETSI adalah *European Telecommunications Standards Institute*. Standar lain yang ada yaitu :

1. ISO 7810 - Identifikasi kartu (karakteristik fisik)
2. ISO/IEC 7812 - Identifikasi untuk penerbitan.
3. ISO/IEC 10536 - Identifikasi kartu (*contactless*)
4. ISO/IEC 10373 - Identifikasi kartu (metode tes)
5. ISO/IEC 14443 - Komunikasi kartu barpasanagn (*contactless*)
6. ISO TC 68 - Servis perbankan dan finanskial
7. EN 742 - Identifikasi kartu
8. EN 726 Peralatan terminal (TE) - Kebutuhan dari *ICC* dan terminal untuk berkomunikasi.

Sedangkan standar yang dasar untuk *smart card* yang dipaparkan dalam dokumen ISO 7816 adalah sbb,

1. 7816 - 1 (1987) : Karakteristik fisik.
   Smart Card harus:
   a. Berukuran 85,60mm x 53,98mm x 0,76mm
   b. Memiliki perlindungan terhadap radiasi, sinar UV, medan magnet dan listrik.
   c. Tidak rusak meskipun digunakan setiap hari
2. 7816 - 2 (1988) : Dimensi dan lokasi dari kontak. Kontak dari *smart card* harus berukuran 10,25mm x 19,23 mm (Gambar 21)

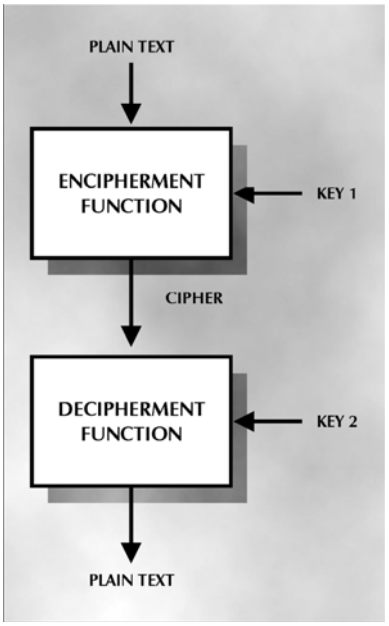
Gambar 21. ISO 7816-2

3. 7816 - 3 (1989) : Sinyal elektronik dan protokol komunikasi
4. 7816 - 4 (1995) : Respon dan perintah dalam *inter*-industri
   Diamandemen pada tahun 1998 dengan spesifikasi *APDU* (*Application Protocol Data Units*), paket data melalui komunikasi *smart card*.
5. 7816 - 5 (1994) : Sistem registrasi untuk aplikasi pengidentifikasi. Diamandemen tahun 1996
6. 7816 - 6 (1995) : Elemen data untuk pertukaran
7. 7816 - 7 (1998) : Perintah bahasa *query* untuk *smart card*.
8. 7816 - 8 : Perintah keamanan untuk *inter*-industri
9. 7816 - 9 : Perintah yang diperbaharui untuk *inter*-industri
10. 7816 - 10 (1999) : Kartu sinkron

6. Keamanan Kartu Cerdas (*Smart Card*) menggunakan Kriptografi

Keuntungan utama penggunaan *smart card* dengan kemampuan pemrosesan yang dimilikinya adalah adanya kesempatan untuk mengimplementasikan mekanisme kriptografi dalam *smart card*. *IC chip* dapat dipertimbangkan sebagai modul yang tahan terhadap serangan.

Meskipun banyak algoritma kriptografi yang telah dikembangkan, dalam prakteknya hanya dua yang sering digunakan dalam penerapan finansial yang mana masih memenuhi dalam hal keamanan. Dua algoritma tersebut adalah *DES* (*Data Encryption Standard*) dan *RSA* (*Rivest Shamir and Adleman*). Dua algoritma ini menampilkan dua kelas operasi yang berbeda, *DES* merupakan algoritma kriptografi kunci simetri, sedangkan *RSA* merupakan algoritma kriptografi kunci nirsimetri. Perbedaan dari dua algoritma ini cukup jelas dipaparkan pada Gambar 22. Pesan masukan dienkripsi menggunakan *key 1* untuk menghasilkan chiperteks. Dan pesan yang asli (plainteks) didapatkan dari hasil dekripsi terhadap cipherteks menggunakan *key 2*. Jika *key 1 = key 2* maka proses kriptografi in adalah simetri.

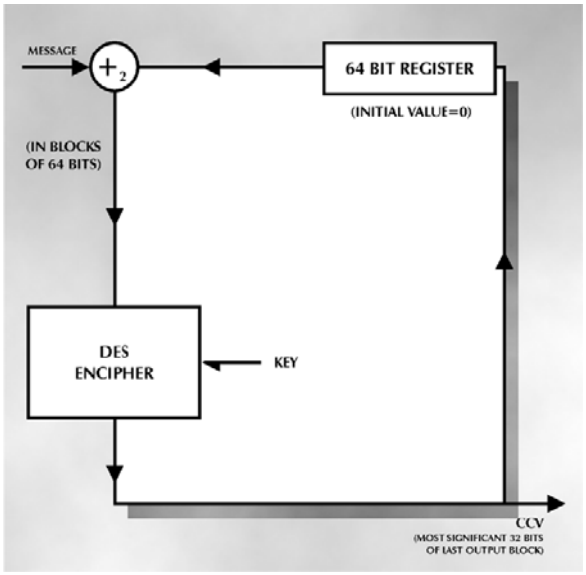


Gambar 22. Proses Kriptografi

Ada sejumlah mekanisme keamanan yang dapat digunakan dalam penerapan *smart card* namun perhatian yang lebih ditujukan untuk mekanisme yang berhubungan dengan integritas data dan otentikasi. *cryptographic check values* (*CCV*) dan tanda tangan digital, merupakan mekanisme yang digunakan secara luas.

Telah disususn penggunaan algoritma *DES* untuk menghitung *CCV*. *CCV* di sini adalah algoritma *MAC* (*Message Authentication Code*) dengan pendekatan berbasis *chiper* blok menggunakan *DES*. Jadi, *check value* ini dibangkitkan menggunakan algoritma *DES* dalam mode rangkaian *chiper* blok seperti yang digambarkan pada Gambar 23. *MAC* sendiri asalnya didefinisikan dalam standar *ANSI X9.9* dan diadopsi sebagai standar ISO 8730 untuk *financial message*. Dalam standar ini digunakan *most significant* dengan 32 bit dari *output* terakhir sebagai *check value* atau *MAC*. Jika dibutuhkan, *message* yang terakhir di ditambahkan dengan nol, sehingga masing-masing blok *input* selalu 64 bit (*DES* menggunakan ukuran

blok 64 bit). Penerima mengecek menggunakan operasi yang sama.



Gambar 23. *CCV* (*Cryptographic Check Value*)

Tujuan dasar dari *CCV* adalah untuk menyediakan fungsi integritas data yang menjamin bahwa pesan tidak dimanipulasi. Manipulasi ini termasuk modifikasi, penambahan, pengurangan data, yang pada initnya merupakan kode *hash* dari sebuah pesan. Dengan *hash* saja tentunya tidak menjamin nirpenyangkalan, sehingga dibutuhkan tanda tangan digital yang menggunakan algoritma kriptografi kunci publik.

Pelayanan keamanan untuk kerahasiaan data, otentikasi, dan itegritas data telah disediakan oleh algoritma kriptografi. Namun pelayanan keamanan yang utama ini ditentukan oleh manajemen kunci kriptografi. Hal ini dikarenakan prinsip dasar dari algoritma yang digunakan telah diketahui oleh penyerang dan dapat digunakan sebagi informasi yang berharga. Penggunaan *smart card* di sini memiliki keuntungan karena dimungkinkan untuk menyebarkan *ICC* kepada semua orang tanpa menampakkan isinya. Sifat dasar dari *ICC* yang tahan terhadap serangan ini membuat *ICC* cocok untuk membawa kunci kriptografi dan algoritma kriptografi.

Konsep bahwa *ICC* merupakan alat yang tahan terhadap serangan dapat didefinisikan sebagai berikut, alat ini harus tetap terlindung baik dalam segi fisik maupun dalam segi logis dari berbagai akses yang tidak dizinkan terhadap data rahasia. Dalam hal ini minimal kunci kriptografi, mungkin juga algoritma kriptografi, dab data keamanan lain yang umum.

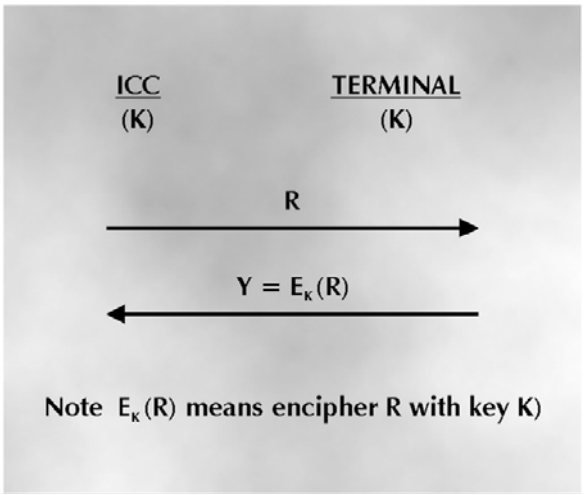
Dalam hal serangan fisik, alat yang tahan terhadap serangan harus membentuk semacam penghalang dari serangan. Dalam *ICC* data disimpan dalam memori *ROM* atau *EEPROM*. Dalam hal ini isi data yang ditanamkan dalam *ROM* atau *EEPROM* akan sulit untuk ditampilkan. Keamanan dari *chip* merupakan hal yang sangat penting, untungnya *chip* modern saat ini sudah memiliki penghalang yang efektif untuk menghadapi serangan. Beberapa jenis serangan fisik yang berda di luar kendali kriptografi yaitu,

1. *timing analysis*
2. *power analysis*
3. *reverse engineering*

Dalam hal serangan dalam bentuk logika, diperlukan rancangan kriptografi agar didapatkan keamanan yang memadai.

Manajemen kunci merupakan hal yang sangat penting dalam keamanan kriptografi. Ada beberapa perbedaan dalam hal otentikasi dengan menggunakan kriptografi kunci simetri dan kriptografi kunci publik.

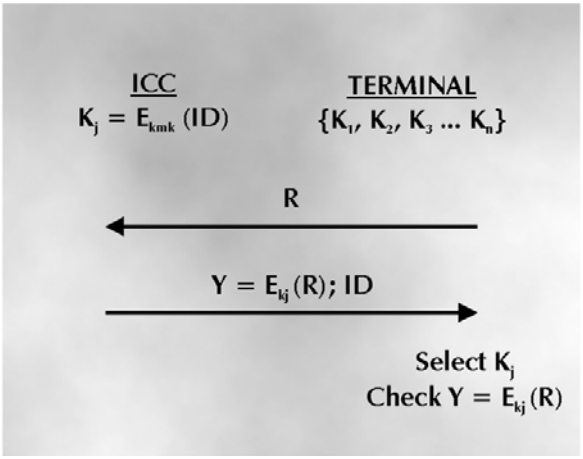
Dimulai dengan pembahasan pada kriptografi kunci simetri. Seluruh tujuan dari keamanan adalah untuk menjalankan pelayanan keamanan antara dua entitas. Sehingga diperlukan kunci umum antara dua entitas ini sebelum pelayanan keamanan dapat dijalankan. Situasi dasar dari sebuah *ICC* yang menjalankan aplikasi dengan sebuah terminal. Dalam Gambar 24 dipaparkan situasi sederhana dimana kunci umum (k) sudah dibangun sebelumnya di *ICC* dan terminal.



Gambar 24. Kunci umum sederhana

Seperti dilihat pada contoh proses otentikasi dimana terminal menegcek keaslian dari *ICC*. Di sisni terminal mengirimkan angka acak *R* ke *ICC*. *ICC* mengenkripsi angka ini dengan kunci umum *K* dan mengembalikan respon *Y* ke terminal. Dengan menggunakan algoritma dan kunci yang sama, terminal dapat mengecek apakah *ICC* mengetahui algoritma dan kunci umum *K* yang sama. Ini merupakan skenario yang dibutuhkan untuk membangun kunci rahasia global pada semua *ICC* dan terminal. Namun mekanisme dasar ini bukan hanya sulit secara operasional tapi juga juka ada penyimpangan pada kartu atau terminal, maka kunci global akan tertampilkan.

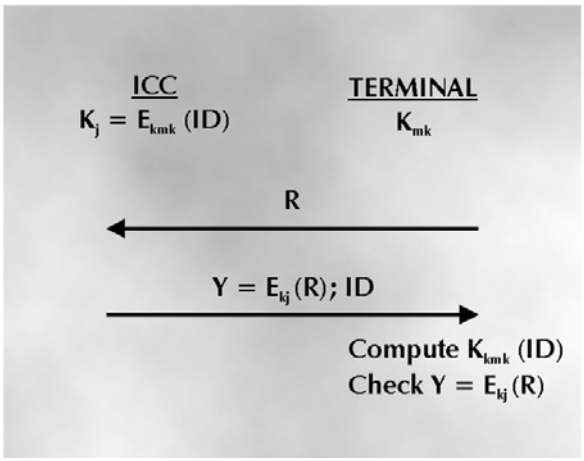
Keamanan ini dapat ditingkatkan dengan menggunakan kumpulan kunci di terminal untuk mendapatkan tingkatan keamanan yang terpisah seperti dipaparkan pada Gambar 25. Pada situasi ini, penerobosan keamanan pada *ICC* hanya akan menampilkan kunci *Kj*. Namun jika serangan terjadi pada terminal maka semua kunci akan tertampilkan.



Gambar 25. Kumpulan kunci terminal

Variasi lain dari proses manajemen kunci adalah menggunakan kunci turunan, yang dipaparkan pada Gambar 26.

Di sini, terminal mengandung kunci utama K_{mk} sedangkan *ICC* diisi dengan kunci *Kj* yang merupakan turunan dari kunci utama ini. Dalam skenario ini, keamanan global bergantung lagi pada terminal tapi masing-masing *ICC* dapat memiliki kunci turunan yang unik. Hal ini berarti efek dari serangan pada *ICC* tunggal dapat dibatasi hanya pada kartu tersebut saja. Deapat juga digunakan seperti pada contoh sebelumnya dengan menggunakan kumpulan kunci utama sehingga didapatkan kunci turunan unik yang khusus.

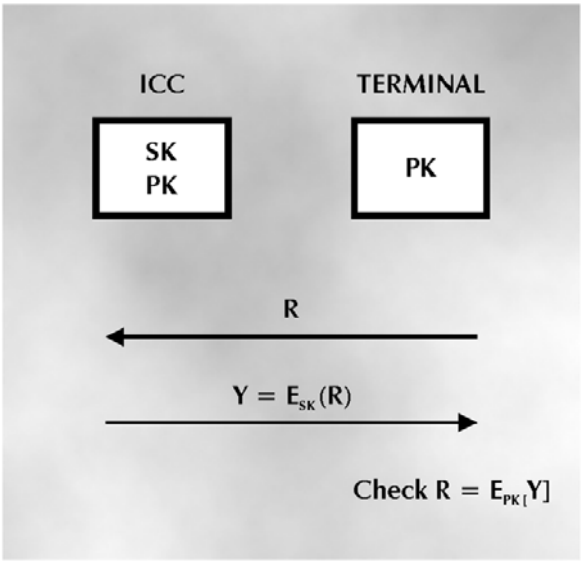


Gambar 26. Kunci turunan

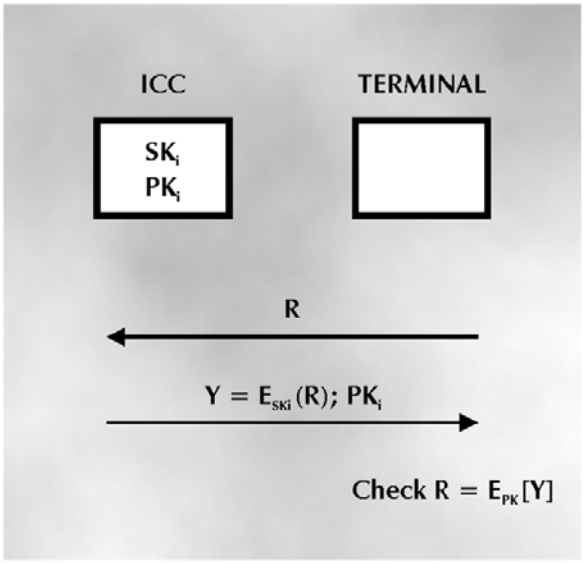
Perbedaan antara algoritma kriptografi kunci simetri dan algoritma kriptografi kunci publik sangat jelas perbedaannya saat diperiksa bermacam-macam arsitektur manajemen kunci dari dua tipe algoritma ini.

Selanjutnya akan dibahas proses manajemen kunci pada kriptografi kunci publik. Konsep untuk memasangkan kunci publik (*PK*) dengan kunci privat (*SK*) akan digunakan secara konstan dalam pembahasan ini, namun penggunanan sebenarnya dari kunci publik akan tergantung pada perancangan atau sisrem keamanan yang utama.

Dalam Gambar 27 diperlihatkan metode otentikasi menggunakan *RSA*. Di sini terminal mengirimkan angka acak *R* ke *ICC*. Dlam prakteknya bilangan *R* ini merepresentasikan bilangan pada bagian-bagian data atau mungkin *current time*. *ICC* mengenkripsi bilanagn acak *R* menggunakan kunci privat *SK* sebagai bilangan eksponen (R^{SK} mod *N*). Hasil dari perhitungan *Y* ini dkembalikan ke terminal yang kemudian menggunakan kunci publik *PK* yang berpasangan untuk mengecek keterhubunagn dengan *R* ($R = Y^{PK}$ mod *N*).



Gambar 27. Proses otentikasi kunci publik sederhana



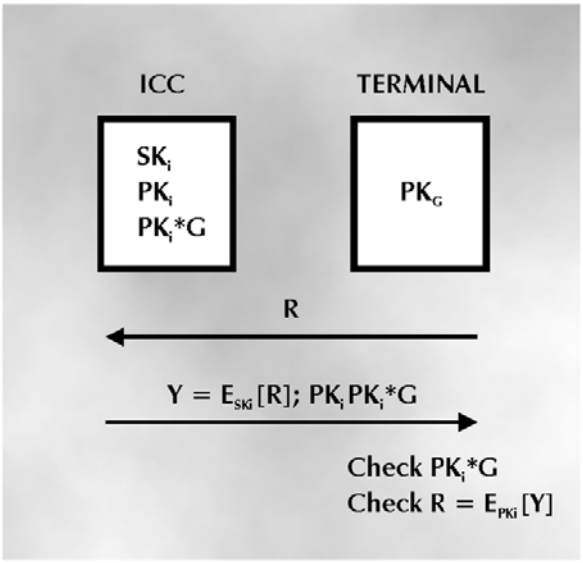
Gambar 28. Kunci unik untuk setiap kartu

Pada contoh sederhana ini diasumsikan bahwa semua *ICC* mengandung kunci privat yang umum sedangkan di terminal disimpan kunci publik global yang umum. Dengan sistem ini, kunci akan ditampilkan jika penyerang dapat mengetahui kunci dari satu kartu saja.

Sistem keamanan ini dapat diperbaiki dengan melakukan pemisahan sehingga didapatkan kunci yang unik untuk masing-masing kartu. Pada Gambar 28 penting bagi terminal untuk mengetahui kunci publik yang cocok. Pada skenario ini *ICC* mampu hanya menampilkan kunci publik saja ke trminala untuk menegecek tandatangan. Namun ada halangan pada proses otentikasi kunci publik yang ditampilkan oleh *ICC* ini. Jika tidak ada penegcakan terhadap keaslian kunci, maka bisa saja penyerang membuat sendiri kunci privat dan kunci publiknya dan akan lolos terhadap tes yang dilakukan.

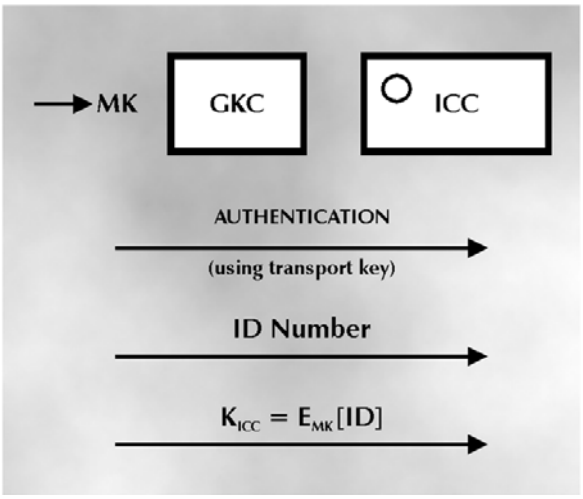
Hal ini menuju kepada dasr dari semua proses kriptografi yaitu pusat kepercayaan. Jika ada dua kelompok yang tidak diketahui akan berhubungan maka dibutuhkan suatu titik umum dimana keduanya dapat saling percaya. Pusat kepercayaan ini disebut sebagai *global key centre* (*GKC*) karena peran utamanya berhubunagn dengan manajemen kunci dari sistem yang utama.

Pada sistem kunci publik *GKC* ini memiliki kunci privat dan pasangan kunci publiknya sendiri. Penggunaan kunci privat adalah untuk menghasilkan tandatangan bagi kunci publik dari koelompok yang saling bersangkutan. Tandatangan ini sering disebut sebagai sertifikat kunci. *PK*G* digunakan sebagai representasi dari sertifikat kunci pada *PK* yang dihasilkan oleh *G*. Sekarang hanya dibutuhkan untuk mendistribusukan kunci publik PK_G dari *global key centre* (*GKC*) ke semua terminal. Pada Gambar 29, operasi ini diperlihatkan. *ICC* mengirimkan kunci publiknya PK_i dan sertifikat (dihasilkan oleh *GKC*) untuk kunci PK_i*G ini. Dalam penegcekan respon pada bilangan acak dibutuhkan dua operasi. Pertama menegcek otentikasi dari kunci publik yang diberikan oleh *ICC* menggunakan kunci publik *GKC* (mengecek $PK_i = E_{PKG}[PK_i*G]$). Setelah mengecek otentikasi dari kunci publik *ICC*, terminal lalu mengecek respon ke bilangan acak menggunakan kunci publik dari *ICC*. Dalam hal ini *ICC* perlu menyimpan kunci publik dan kunci privatnya sendiri dan tambahan sertifikat kunci dari *GKC*.



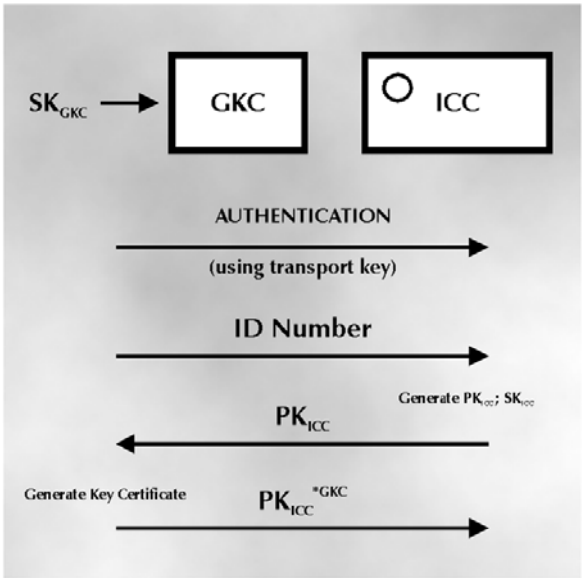
Gambar 29. Otentikasi kunci publik

Sekarang operasi manajemen kunci dasar untuk skema keamanan dapat diuji. Pada Gambar 30 *GKC* membangun kunci unik untuk masing-masing *ICC* yang diturunkan dari nomor pada *chip* dan kunci utama sistem. Nomor pada *chip* (dalam bentuk *ID*) dan kunci unik ini diisikan ke dalam memori *EEPROM chip*.



Gambar 30. Pusat kunci global simetris

Untuk sistem manajemen kunci asimetris agak berbeda dengan kunci simetris, seperti digambarkan pada Gambar 31.

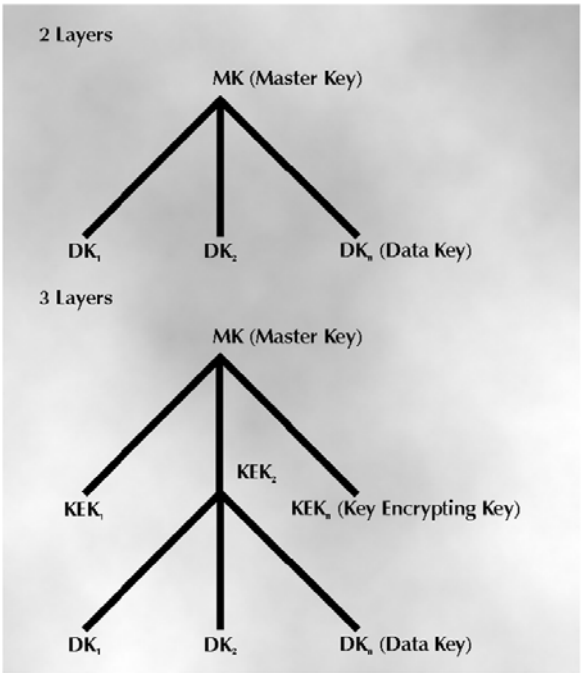


Gambar 31. Pusat kunci global asimetris

Dalam Gambar 31 ini ditampilkan bahwa *ICC* membangun kunci publik dan kunci privat pasangannya sendiri. Peranan dari *GKC* berhubungan dengan memproduksi sertifikat kunci yang akan disimpan dalam *ICC*. Pada situasi ini sudah tidak memungkinkan untuk menampilkan kunci privat pada setiap kumpulan. Kunci ini dibangkitkan, disimpan, dan disimpan dalam *ICC* yang utama.

7. Manajemen Kunci Kriptografi dalam Kartu Cerdas (*Smart Card*)

Telah dibahas penggunaan *smart card* untuk mengimplementasikan berbagai fungsi keamanan seperti integritas data dan otentikasi. Implementasi manajemen kunci kriptografi ini juga merupakan peranan dari *smart card*. Skema manajemen kunci melibatkan enam tahap, pembangkitan, penyimpanan, distribusi, penggunaan, perubahan, dan penghancuran. Setiap skema melibatkan penggunaan pusat kunci yang dipercaya dan hirarki kunci yang dibutuhkan. Skema kunci selalu disusun dalam suatu hirarki dimana kar dari struktur kunci adalah kunci *master*-nya. Beberapa skema mungkin melibatkan *layer* yang lebih, tapi di sini hanya kan dibahas dua *layer* saja seperti yang ditampilkan pada Gambar 32. Prinsipnya akan bisa dikembangkan dengan *layer* tambahan.



Gambar 32. Hirarki kunci

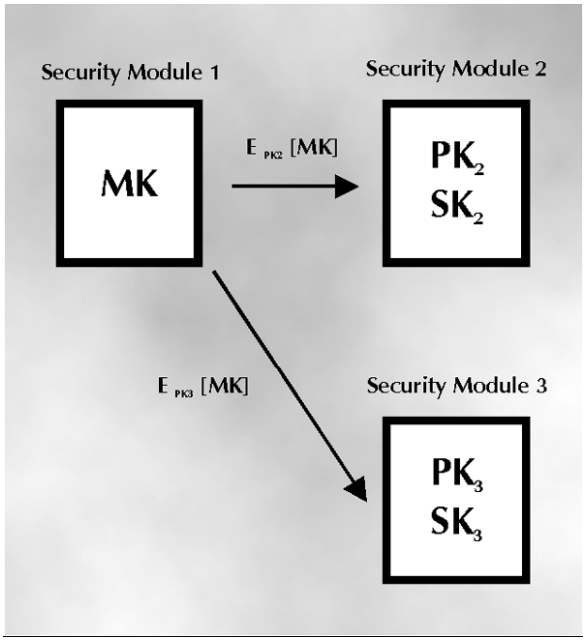
Pembangkitan dan penyimpanan Kunci

Akar dari hirarki kunci didefinisikan sebagai kunci *master* dimana merupakan perancang kunci yang lain. Ini merupakan kunci kritis di dalam sistem dan seharusnya acak. Proses manajemen ini harus didukung dengan prosedur untuk menjaga kerahasiaan kunci.

Kunci *master* ini bersifat acak atau diturunkan dari bilangan yang acak. Bilangan acak ini harus tetap dijaga kerahasiaannya setiap waktu. Ada dua jnis bilangan acak, semu dan nyata. Bilangan acak nyata umumnya menggunakan ciri ilmu fisika yang dapat menghasilkan aliran bhilangan yang berkolerasi tanpa akhir. Pembangkit bilangan semu menggunakan beberapa algoritma matematika untuk membangun aliran bilangan acak. Sayangnya banyak pembangkit sederhana memiliki kelemahan. Biasanya bilangan acak ini menuju pada sebuah urutan pendek sebelum terjadi perulangan kembali.

Secara khusus dapat digunakan algoritma kriptografi *DES* yang dari definisi dapat membangkitkan bilangan acak. Penggunaan *DES* ini yaitu dengan cara mengambil bilanagn acak pertama lalu secara berurutan mengenkripsi seluruh bagian dari bilangan-bilangan masukan (masing-masing 64 bit). Hal ini akan membangun urutan yang tidak berulang dari 2^{64} bilangan.

Namun bagaimana bilangan pertama didapatkan, seharusnya ini adalah bilangan yang acak juga atau semua urutan akan tertampilkan. Jadi diperlukan bilangan acak pertama untuk memulai urutan ini. Pertama yang dibutuhkan adalah menghasilkan sebuah modul yang mengandung kunci *master*. Hal ini dapat dibangun menggunakan algoritma kunci publik *RSA*. Situasinya digambarkan pada Gambar 33 dimana akan dibangun sebuah sumber kunci *master* dalam tiga modul keamanan. Modul keamanan pertama memiliki kemampuan untuk membangun bilanagn acak yang akan berlaku sebagai kunci *master* (*MK*). Modul keamanan kedua dan ketiga mampu untuk membangun sendiri pasangan kuni publik (*PK*) dan kunci privat (*SK*). Operasi ini membutuhkan kedua mudul untuk secara independen membangun bilangan acak pertama yang akan digunakan untuk mengkalkulasi kunci-kunci.



Gambar 33. Pembangunan kunci *master*

Dengan algoritma *RSA* sbb,

$$E_e(m) = c \equiv m^e \bmod n$$

$$D_d(c) = m \equiv c^d \bmod n$$

Dimana,

1. p dan q bilangan prima
2. $n = p.q$
3. $\Phi(n) = (p - 1)(q - 1)$
4. e (kunci enkripsi)
5. d (kunci dekripsi)
6. m (plainteks)
7. c (cipherteks)

Yang harus diperhatikan di sini adalah modul keamanan menghitung dua bilangan prima *p* dan *q* dalam mode acak. Modul ini lalu menghitung kunci privat *d* (yang tidak akan pernah keluar dari modul) modulus publik *n*.

Proses pendistribusian kunci *master* dari modul 1 ke modul 2 diproses sbb,

1. Kunci publik modul 2 (PK_2) diberikan ke modul 1.
2. Modul 1 mengenkripsi kunci *master* menggunakan PK_2.
3. Blok *chiper* yang dihasilkan diberikan ke modul 2
4. Modul 2 mendekripsi blok dengan SK_2 untuk memperoleh *MK*.

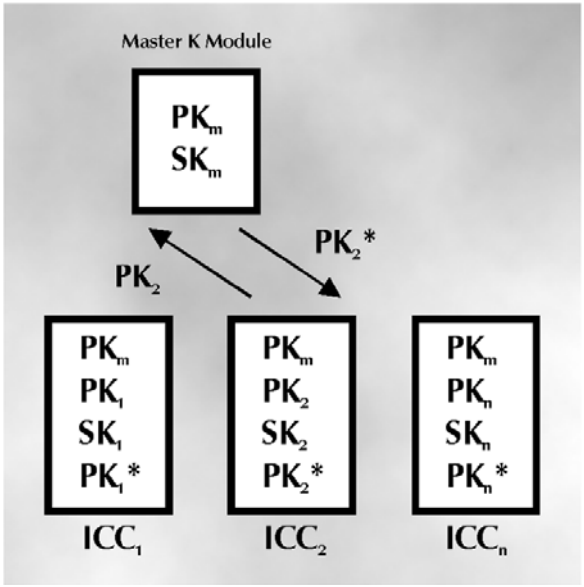
Operasi ini dapat diulang sehingga ketiga modul memiliki salinan dari kunci *master* yang dibangkitkan oleh modul 1.

Akhirnya kunci-kunci yang telah dibangkitkan ini disimpan di dalam *chip smart card* beserta sertifikat kunci tersebut.

Distribusi dan perubahan kunci

Setelah membangun kunci *master* dari modul keamanan tinggal men-*set up* kunci data untuk *smart card*. Masalah yang timbul berhubungan dengan metode pemasukan kunci turunan ke dalam *smart card* saat proses personalisasi. Prosedur yang dapat digunakan yaitu dengan menggunakan kunci untuk mengenkripsi kunci yang merupakan kunci yang sudah dimasukkan ke dalam *chip smart card* saat proses pembuatan di pabrik. Modul keamanan dapat dibuat untuk mampu menentukan kunci untuk enkripsi kunci ini dan meminta operasi yang dibutuhkan.

Masalah lain dari tahap pembangunan kunci adalah, dalam rangka penukaran *smart card* atau *ICC* dibutuhkanjuga penukaran sertifikat kunci publik (Gambar 34; * = sertifikat). Ketika kartu dipersonalisasi sertifikat ini harus disimpan di dalam *chip*. Jika *ICC* menghitung kunci publiknya sendiri makan kunci publik harus dibaca dari *chip* dan ditampilkan secara aman kepada modul kunci *master* untuk proses sertifikasi menggunakan kunci privat *master*. Proses sertifikasi ini sama dengan proses tandatangan digital. Masalah yang ditimbulkan di sini bukanlah pada masalah kerahasiaan, melainkan masalah kontrol otentikasi. Jadi prosedur yang dibangun harus menjamin tidak adanya kemungkinan memasukkan kunci palsu untuk proses sertifikasi. Solusi dari masalah ini yaitu *ICC* dapat menggunakan kunci tambahan yang akan menyediakan otentikasi yang dibutuhkan. Kunci ini dimasukkan saat awal proses pembuatan di pabrik yang hanya dapat diketahui oleh modul kunci *master*.



Gambar 34. Pembangunan kunci publik

8. Kesimpulan

Smart card yang ditemukan oleh Roland Moreno pada taun 1974 sudah smakin dikenal dan banyak digunakan dalam kehidupan sehari-hari terutama dengan adanya penggunaan kartu *SIM* (*Subscriber Identity Module card*) untuk telepon selular. Karu *SIM* ini merupakan jenis *contact smart card*, jenis yang lain dari *samrt card* adalah *contactless*. Perbedaan dua jenis kartu ini sangat jelas secara fisik dengan adanya lempengan emas yang merupakan kontak pada jenis *contact smart card*. Sedangkan secara sumber daya listrik, pada *contact smart card* tidak memiliki sumber daya sendiri. Arus listrik diberikan saat terjadi komunikasi dengan perangkat antarmuka. Sedangkan untuk jenis *contactless smart card* sumber dayanya berasal dari baterai yang ada di dalm kartu trersebut.

Untuk membangun arsitektur dari *smart card* diperlukan beberapa elemen yaitu:

1. *Central Processing Unit*
   Merupakan tempat dimana perintah-perintah yang diberikan ke *smart card* dijalankan dan pusat dimana terjadinya proses kriptografi.

2. Sistem Memori
   Secara garis besar terdapat dua jenis memori, non-*volatile dan volatile*. Jenis memori non-*volatile* yaitu memori ini masih tetap memiliki data meskipun daya dimatikan. Sedangkan jenis memori *volatile* data akan terhapus jika daya dimatikan. Bebrapa jenis memori non-*volatile* yaitu *ROM*, *PROM*, *EPROM*, dan *EEPROM*. Sedangan contoh memori *volatile* yaitu *RAM*.
3. *Input/Output*
   Di sini merupakan *port I/O* tunggal yang dikontrol oleh *processor* untuk menjamin komunikasi kartu.
4. *Interface Devices* (perangkat antarmuka)
   Merupakan sarana pembaca kartu yang digunakan untuk pertukaran data.
5. Sistem Operasi
   Mengimplementasikan perintah terhadap kartu.
6. Sistem File
   Merupakan sistem pengorganisasian file dalam bentuk hirarki.
7. Perangkat Lunak
   Biasanya merupakan bawaan dari perusahaan penerbit kartunya.
8. Bahasa Pemrograman
   Bahasa pemrograman yang dipakai adalah bahasa pemrograman tingkat rendah.

Agar terjadi komunikasi antara *smart card* dan perangkat antarmuka diperlukan protokol komunikasi. Dua protokol komunikasi yang umum adalah T=0 dan T=1. T=0 merupakan protokol komunikasi byte *half duplex* sedangkan T=1 merupakan protokol komunikasi blok *half duplex*.

Keamanan kartu cerdas dapat dibangun menggunakan kriptografi, terutama penggunaan algoritma *DES* dan *RSA* untuk otentikasi, integritas data, dan nirpenyangkalan.

9. Daftar Pustaka

[1] Ivona Bezakova, Oleg Pashko, Dinoj Surendran, "Architectural Challenges Of Smart Card Design".2002

[2] Burt Kaliski, "Some Perspectives on Smart Card Cryptography",
RSA Laboratories. 1998

[3] Dr David B Everett, "Smart Card Technology". 2002

[4] Amesh Mansukhani, "Are Smart Cards the New Way of Life?",
http://www.microsoft.com. 2006

[5] Tolga Kilici, "Smart Card How To". 2001

[6] Gary McGraw and Edward Felten, "How Smart Card and Java Mix", Willey.1999

[7] Munir, Rinaldi. Diktat Kuliah IF5054 Kriptografi. Departemen Teknik Informatika, Institut Teknologi Bandung. 2006