BAB 2
TINJAUAN PUSTAKA

2.1 Profil Tempat Penelitian

Badan Penanaman Modal Dan Pelayanan Perizinan Terpadu dibentuk berdasarkan Peraturan Daerah Kabupaten Bandung Barat Nomor 4 Tahun 2011 tentang perubahan atas peraturan daerah Kabupaten Bandung Barat Nomor 10 Tahun 2008 Tentang Pembentukan dan susunan organisasi lembaga teknis daerah, Badan Penanaman Modal dan Pelayanan Perizinan Terpadu, berkedudukan sebagai unsur pendukung tugas Bupati dan bertanggung jawab kepada Bupati melalui Sekretaris Daerah berupa pertanggungjawaban administratif meliputi penyusunan kebijakan, perencanaan, pelaksanaan, *monitoring*, evaluasi, dan pelaporan pelaksanaan tugas lembaga teknis daerah.

Dalam RPJMD Kabupaten Bandung Barat, berkaitan dengan tugas pokok dan fungsi Badan Penanaman Modal dan Pelayanan Perizin**a**n Terpadu terkait dengan misi RPJM Daerah (2008-2013) sebagai berikut:

1. Meningkatkan penyelenggaraan pemerintahan yang amanah, profesional, efektif, efisien dan ekonomis yang berbasis pada sistem penganggaran yang pro-publik.
2. Meningkatkan kualitas sumber daya manusia yang berakhlak, cerdas, sehat dan berdaya saing.
3. Memberdayakan perekonomian daerah berbasis ekonomi kerakyatan yang berorientasi pada pengembangan sektor agribisnis dan agro wisata dalam upaya pengentasan kemiskinan.
4. Mewujudkan pembangunan berkelanjutan berwawasan lingkungan

2.1.1 Tupoksi BPMPPT KBB

Berdasarkan Peraturan Daerah Kabupaten Bandung Barat Nomor 4 Tahun 2011 Tentang Perubahan Atas Peraturan Daerah Kabupaten Bandung Barat Nomor 10 Tahun 2008 Tentang Pembentukan dan Susunan Organisasi Lembaga Teknis Daerah, Badan Penanaman Modal dan Pelayanan Perizinan Terpadu memiliki tugas

pokok melaksanakan penyusunan dan pelaksanaan kebijakan daerah di bidang penanaman modal serta melaksanakan koordinasi dan menyelenggarakan pelayanan administrasi di bidang penanaman modal dan perizinan secara terpadu dengan prinsip koordinasi, integrasi, sinkronisasi, simplifikasi, keamanan dan kepastian.

Fungsi Badan Penanaman Modal dan Pelayanan Perizinan Terpadu Dalam menyelenggarakan tugas pokok tersebut Badan Penanaman Modal dan Pelayanan Perizinan Terpadu Kabupaten Bandung Barat, mempunyai fungsi sebagai berikut:

1. Perumusan kebijakan teknis di bidang penanaman modal;
2. Pengkoordinasian dan penyelenggaraan sebagian urusan pemerintahan di bidang penanaman modal;
3. Pembinaan dan pelaksanaan pengelolaan penanaman modal yang meliputi pengembangan penanaman modal, promosi dan kerja sama serta data dan sistem informasi penanaman modal;
4. Pelaksanaan penyusunan program kerja;
5. Penyelenggaraan pelayanan administrasi perizinan, sistem informasi dan pengaduan;
6. Pelaksanaan koordinasi proses pelayanan perizinan;
7. Pemantauan dan evaluasi proses pemberian pelayanan perizinan;
8. Pelaksanaan administrasi ketatausahaan; dan
9. Pelaksanaan tugas lain yang didelegasikan oleh Bupati sesuai dengan tugas dan fungsinya.

2.2 Landasan Teori

Landasan teori di gunakan sebagai dasar penyusunan tugas akhir pengembangan SIMPPTSP dengan penambahan fitur QR code dan TLS.

2.2.1 Perizinan [2]

Perizinan dapat diartikan sebagai salah satu bentuk pelaksanaan fungsi pengaturan dan bersifat pengendalian yang dimiliki oleh pemerintah terhadap kegiatan-kegiatan yang dilakukan oleh masyarakat. Bentuk perizinan antara lain: pendaftaran, rekomendasi, sertifikasi, penentuan kuota dan izin untuk melakukan sesuatu usaha yang biasanya harus memiliki atau diperoleh suatu organisasi

perusahaan atau seseorang sebelum yang bersangkutan dapat melaksanakan suatu kegiatan atau tindakan. Dengan memberi izin, pengusaha memperkenankan orang yang memohonnya untuk melakukan tindakan tindakan tertentu yang sebenarnya dilarang demi memperhatikan kepentingan umum yang mengharuskan adanya pengawasan. (Andrian Sutedi).

Perizinan dapat didefenisikan dalam arti luas dan dalam arti sempit. Dalam arti luas yakni merupakan salah satu instrumen yang paling banyak digunakan dalam hukum administrasi. Pemerintah menggunakan izin sebagai sarana yuridis untuk mengemudikan tingkah laku para warga. Izin ialah suatu persetujuan dari penguasa berdasarkan undang-undang atau peraturan pemerintah untuk dalam keadaan tertentu menyimpang dari ketentuan-ketentuan larangan peraturan perundang-undangan. Dengan memberi izin, penguasa memperkenankan orang yang memohonnya untuk melakukan tindakan-tindakan tertentu yang sebenarnya dilarang. Ini menyangkut perkenaan bagi suatu tindakan yang demi kepentingan umum mengharuskan pengawasan khusus atasnya. Sedangkan dalam arti sempit yakni pengikatan-pengikatan pada suatu peraturan izin pada umumnya didasarkan pada keinginan pembuat undang-undang untuk mencapai suatu tatanan tertentu atau untuk menghalangi keadaan-keadaan yang buruk. Tujuannya ialah mengatur tindakan-tindakan yang oleh pembuat undang-undang tidak seluruhnya dianggap tercela, namun di mana ia menginginkan dapat melakukan pengawasan sekedarnya. Hal pokok pada izin dalam arti sempit adalah bahwa suatu tindakan dilarang, terkecuali diperkenankan dengan tujuan agar dalam ketentuan-ketentuan yang disangkutkan dengan perkenaan dapat dengan teliti diberikan batas-batas tertentu bagi tiap kasus. Jadi persoalannya bukanlah untuk hanya memberi perkenaan dalam keadaan-keadaan yang sangat khusus, tetapi agar tindakan-tindakan yang diperkenankan dilakukan dengan cara tertentu/ dicantumkan dalam ketentuan-ketentuan (N.M. Spelt dan J.B.J.M Ten Berge).

2.2.2 Sistem Infomasi Manajemen [3]

Sistem Informasi Manajemen adalah sebuah sistem informasi yang selain melakukan pengolahan transaksi yang sangat berguna untuk kepentingan

organisasi, juga banyak memberikan dukungan informasi dan pengolahan untuk fungsi manajemen dalam pengambilan keputusan.

Secara historis, gagasan sebuah sistem informasi tidak dikenal sebelum munculnya komputer. Akan tetapi, komputer telah banyak memberikan dampak dengan terwujudnya gagasan tersebut menjadi realitas. Organisasi dalam segala motif selalu membutuhkan sistem-sistem untuk mengklasifikasi, mengolah, menyimpan, melihat kembali / menemukan kembali, dan mendistribusikan informasi. Komputer telah memberikan sumbangan yang tinggi berupa teknologi canggih dan terandalkan pada sistem informasi. Dampaknya, sebuah sistem informasi berdasarkan komputer akan mengalami perbedaan dengan sistem-sistem yang diolah secara manual.

Sistem Informasi manajemen secara umum dapat di katakan sebagai sebuah sistem manusia dan mesin yang terintergrasi dalam menyediakan informasi guna mendukung fungsi operasi manajemen dan penentuan alternatif tindakan dalam sebuah organisasi sistem tersebut. Dalam operasinya, sistem informasi manajemen menggunakan perangkat keras (*hardware*), perangkat lunak (*software*), prosedur, model manajemen dan keputusan serta sebuah terminal data. Sistem informasi manajemen sebagai suatu kumpulan manusia dan sumber modal di dalam suatu organisasi bertanggung jawab untuk pengumpulan dan pengolahan data sewaktu menghasilkan informasi yang berguna untuk setiap hierarki manajemen dalam perencanaan dan pengendalian kegiatan-kegiatan organisasi.

2.2.3 Sistem Informasi Manajemen Pelayanan Perizinan Terpadu Satu Pintu (SIMPPTSP)

Sistem Informasi Pelayanan Perizinan Terpadu Satu Pintu (SIMPPTSP) adalah sebuah sistem yang digunakan oleh BPMPPT KBB untuk mengatur dan memonitor alur pelayanan perizinan dari mulai proses pengajuan perizinan sampai dengan pencetakan dan penerbitan perizinan, SIMPPTSP memiliki fasilitas pencatatan data pemohon perizinan, pencatatan status pengajuan perizinan, pencetakan resi pemohon perizinan, pencetakan surat tugas tim teknis dan pencetakan warkah perizinan.

Tujuan pembangunan SIMPPTSP adalah untuk menanggulangi berbagai kendala yang terjadi dalam proses pelayanan perizinan, contohnya seperti sulitnya mengetahui status pengajuan sebuah perizinan, dengan adanya SIMPPTSP status pengajuan perizinan dapat dengan mudah diketahui, selain itu SIMPPTSP juga mempermudah proses pengawasan pelayanan perizinan karena dengan adanya SIMPPTSP, Kepala Badan BPMPPT KBB dapat dengan mudah mendapatkan informasi pelayanan perizinan yang sedang dilakukan.

2.2.4 QR Code [4]

Sejak di perkenalkan pada tahun 1994 QR code di terima secara luas oleh berbagai jenis industri seperti manufakturing, *warehousing* dan logistik, ritel, pelayanan kesehatan, *life sciences*, transportasi dan otomasi kantor. Seiring dengan pertumbuhan pengguna *smartphone* yang meledak, QR code juga di gunakan pada *mobile marketing* dan periklanan sebagai cara cepat dan efektif untuk terhubung dengan pelanggan dan menyediakan konten kepada pengguna, termasuk tautan *web*, kupon *mobile*, tiket pesawat terbang dan lain lain.



QR Code

Gambar 2.1 QR code

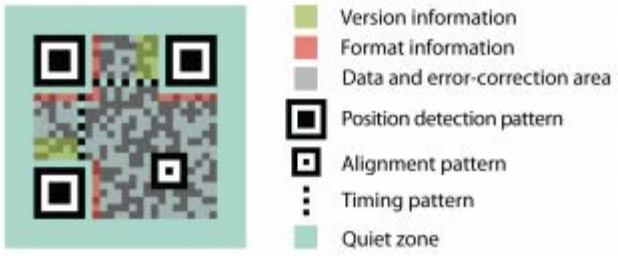
QR (*Quick Response*) code adalah *matrix* code dua dimensi (2-D) yang merupakan *machine-readable codes* yang sering kali disebut *barcodes*. Di bandingkan dengan *barcode* 1-D, code 2-D dapat menampung lebih banyak data pada ruang yang lebih kecil, dan jika di bandingkan dengan *barcode* 2-D lainnya QR code masih memiliki kapasitas yang lebih besar. Selain itu QR code juga menyertakan metode koreksi *error* yang membuat QR code lebih handal dan dapat di baca dengan lebih cepat. QR code menyimpan data melalui pengaturan *element*

gelap dan terang yang disebut *modules* yang di tata pada kolom dan baris (Vertikal dan Horizontal).



Gambar 2.2 Perbandingan QR code dan Bar code

Setiap modul gelap dan terang pada QR code mewakili 0 dan 1 sehingga memungkinkan QR code di baca oleh mesin.

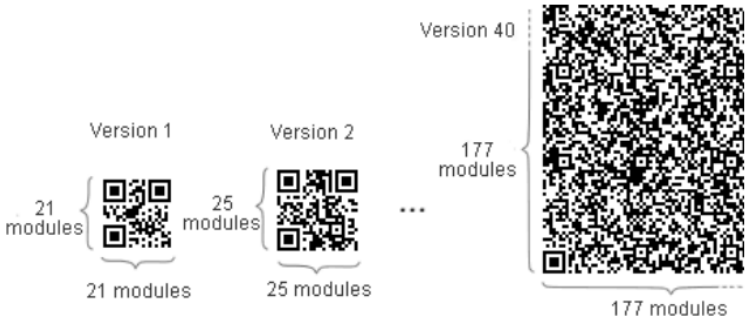


Gambar 2.3 Struktur QR code

Modul QR code memiliki beberapa fungsi, sebagian modul berisi data sedangkan bagian lainnya merupakan modul yang berfungsi sebagai *function pattern* yang meningkatkan performa pembacaan, *error correction* dan kompensasi distorsi. *Timing pattern* menyimpan informasi ukuran dari simbol QR code yang nantinya di gunakan oleh pembaca QR code. Selain itu QR code juga membutuhkan "*Quiet Zone*" yaitu *buffer* seluas 4 modul yang tidak berisi data untuk memastikan teks ataupun tanda di sekitar QR code tidak di salah kenali sebagai data QR code.

2.2.5 Kapasitas QR Code [5]

QR code dapat di buat dalam berbagai ukuran yang di sebut dengan *version*. Ada 40 *version* QR code, *version* terkecil dari QR code adalah *version* 1 dengan ukuran 21x21 *pixel* (modul) sedangkan *version* terbesar adalah *version* 40 dengan ukuran 177 x 177 *pixel* (modul). Kenaikan ukuran tiap *version* QR code adalah sebesar 4 *pixel* (modul).



Gambar 2.4 Perbandingan ukuran version QR code

Kapasitas penyimpanan dari QR code di tentukan oleh *version* QR code, mode *encoding* dan *level error correction* yang di gunakan. Ada 4 tingkat *error correction* yang dapat di gunakan, seperti di jelaskan pada tabel berikut :

Tabel 2.1 Level *error correction* QR code

Level Error Correction	**Error Correction Capability**
L	*Recovers 7% of data*
M	*Recovers 15% of data*
Q	*Recovers 25% of data*
H	*Recovers 30% of data*

Ukuran QR code terbesar saat ini adalah QR code *version* 40 dengan level *error correction* L berikut tabel kapasitas QR code 40-L pada berbagai *mode encodin*g.

Tabel 2.2 Kapasitas QR code 40-dalam berbagai mode

Encoding Mode	**Jumlah Karakter QR Code 40-L**	**Set Karakter**
Numeric	7089	0-9
Alphanumeric	4296	0-9, A-Z (Upper Case) $, %, *, +, -, ., /,:,space
Byte	2953	ISO-8859-1
Kanji	1817	Shift JIS

Sedangkan tabel kapasitas QR code untuk setiap versi, *level error correction* dan *mode encoding* dapat dilihat pada Tabel 2.3.

Tabel 2.3 Kapasitas QR code

Version	**EC**	**Numeric**	**Alpha**	**Byte**	**Kanji**		**Version**	**EC**	**Numeric**	**Alpha**	**Byte**	**Kanji**
1	L	41	25	17	10		21	L	2232	1352	929	572
	M	34	20	14	8			M	1708	1035	711	438
	Q	27	16	11	7			Q	1224	742	509	314
	H	17	10	7	4			H	969	587	403	248
2	L	77	47	32	20		22	L	2409	1460	1003	618
	M	63	38	26	16			M	1872	1134	779	480
	Q	48	29	20	12			Q	1358	823	565	348

	H	34	20	14	8			H	1056	640	439	270
3	L	127	77	53	32		23	L	2620	1588	1091	672
	M	101	61	42	26			M	2059	1248	857	528
	Q	77	47	32	20			Q	1468	890	611	376
	H	58	35	24	15			H	1108	672	461	284
4	L	187	114	78	48		24	L	2812	1704	1171	721
	M	149	90	62	38			M	2188	1326	911	561
	Q	111	67	46	28			Q	1588	963	661	407
	H	82	50	34	21			H	1228	744	511	315
5	L	255	154	106	65		25	L	3057	1853	1273	784
	M	202	122	84	52			M	2395	1451	997	614
	Q	144	87	60	37			Q	1718	1041	715	440
	H	106	64	44	27			H	1286	779	535	330
6	L	322	195	134	82		26	L	3283	1990	1367	842
	M	255	154	106	65			M	2544	1542	1059	652
	Q	178	108	74	45			Q	1804	1094	751	462
	H	139	84	58	36			H	1425	864	593	365
7	L	370	224	154	95		27	L	3517	2132	1465	902
	M	293	178	122	75			M	2701	1637	1125	692
	Q	207	125	86	53			Q	1933	1172	805	496
	H	154	93	64	39			H	1501	910	625	385
8	L	461	279	192	118		28	L	3669	2223	1528	940
	M	365	221	152	93			M	2857	1732	1190	732
	Q	259	157	108	66			Q	2085	1263	868	534
	H	202	122	84	52			H	1581	958	658	405
9	L	552	335	230	141		29	L	3909	2369	1628	1002
	M	432	262	180	111			M	3035	1839	1264	778
	Q	312	189	130	80			Q	2181	1322	908	559
	H	235	143	98	60			H	1677	1016	698	430
10	L	652	395	271	167		30	L	4158	2520	1732	1066
	M	513	311	213	131			M	3289	1994	1370	843
	Q	364	221	151	93			Q	2358	1429	982	604
	H	288	174	119	74			H	1782	1080	742	457
11	L	772	468	321	198		31	L	4417	2677	1840	1132
	M	604	366	251	155			M	3486	2113	1452	894
	Q	427	259	177	109			Q	2473	1499	1030	634
	H	331	200	137	85			H	1897	1150	790	486
12	L	883	535	367	226		32	L	4686	2840	1952	1201
	M	691	419	287	177			M	3693	2238	1538	947
	Q	489	296	203	125			Q	2670	1618	1112	684
	H	374	227	155	96			H	2022	1226	842	518
13	L	1022	619	425	262		33	L	4965	3009	2068	1273
	M	796	483	331	204			M	3909	2369	1628	1002
	Q	580	352	241	149			Q	2805	1700	1168	719
	H	427	259	177	109			H	2157	1307	898	553
14	L	1101	667	458	282		34	L	5253	3183	2188	1347
	M	871	528	362	223			M	4134	2506	1722	1060
	Q	621	376	258	159			Q	2949	1787	1228	756
	H	468	283	194	120			H	2301	1394	958	590
15	L	1250	758	520	320		35	L	5529	3351	2303	1417
	M	991	600	412	254			M	4343	2632	1809	1113
	Q	703	426	292	180			Q	3081	1867	1283	790
	H	530	321	220	136			H	2361	1431	983	605
16	L	1408	854	586	361		36	L	5836	3537	2431	1496
	M	1082	656	450	277			M	4588	2780	1911	1176
	Q	775	470	322	198			Q	3244	1966	1351	832
	H	602	365	250	154			H	2524	1530	1051	647
17	L	1548	938	644	397		37	L	6153	3729	2563	1577

	M	1212	734	504	310			M	4775	2894	1989	1224
	Q	876	531	364	224			Q	3417	2071	1423	876
	H	674	408	280	173			H	2625	1591	1093	673
18	L	1725	1046	718	442		38	L	6479	3927	2699	1661
	M	1346	816	560	345			M	5039	3054	2099	1292
	Q	948	574	394	243			Q	3599	2181	1499	923
	H	746	452	310	191			H	2735	1658	1139	701
19	L	1903	1153	792	488		39	L	6743	4087	2809	1729
	M	1500	909	624	384			M	5313	3220	2213	1362
	Q	1063	644	442	272			Q	3791	2298	1579	972
	H	813	493	338	208			H	2927	1774	1219	750
20	L	2061	1249	858	528		40	L	7089	4296	2953	1817
	M	1600	970	666	410			M	5596	3391	2331	1435
	Q	1159	702	482	297			Q	3993	2420	1663	1024
	H	919	557	382	235			H	3057	1852	1273	784

2.2.5.1 *Encoding* QR Code [5]

Ada beberapa tahapan pengkodean QR code, langkah langkah tersebut adalah sebagai berikut:

1. Menganalisa data

   QR code memiliki 4 mode *encoding* teks yaitu numerik, *alphanumeric*, *byte* dan kanji. Setiap mode mengubah teks sebagai *string bit* (1 dan 0), tetapi setiap mode menggunakan metode yang berbeda untuk mengubah teks menjadi *bit*. Setiap metode dioptimalkan untuk menghasilkan *bit* terpendek untuk setiap jenis data. Oleh karena itu, langkah pertama yang harus di lakukan adalah analisis data untuk menentukan apakah teks yang akan dikodekan adalah numerik, *alphanumeric*, *byte*, atau Kanji, detail set karakter yang untuk setiap mode dapat dilihat pada tabel Tabel 2.2.

2. Menyandikan data

   Setelah dipilih mode pengkodean yang tepat, dilakukan proses pemilihan level *error correction* yang detailnya dapat dilihat pada tabel Tabel 2.1, setelah di lakukan pemilihan level *error correction* dipilih juga *version* QR code terkecil yang memungkinkan untuk menampung data, dilakukan juga penambahan *Mode indicator* dan *character count indicator.*

   Mode indicator digunakan untuk mengenali mode pengkodean yang di gunakan oleh QR code. Setiap mode memiliki *mode indicator* yang di tulis dalam bentuk biner seperti di jelaskan pada tabel Tabel 2.4.

Tabel 2.4 *Mode indicator*

Mode Name	**Mode Indicator**
Numeric Mode	0001
Alphanumeric Mode	0010
Byte Mode	0100
Kanji Mode	1000
ECI Mode	0111

Sedangkan *character count indicator* berfungsi sebagai penanda jumlah dari karakter yang di kodekan, setiap *version* QR code memiliki panjang *character count indicator* yang berbeda beda. Seperti di jelaskan pada tabel Tabel 2.5.

Tabel 2.5 *Character count indicator*

Version	**Panjang Character Count Indicator**			
	Numeric	**Alphanumeric**	**Byte**	**Kanji**
1-9	10 bits	9 bits	8 bits	8bits
10-26	12 bits	11 bits	16 bits	10 bits
27-40	14 bits	13 bits	16 bits	12 bits

Character count indicator ditambahkan dalam bentuk biner contohnya pada teks “HELLO WORLD” , panjang dari teks adalah 11 karakter jika teks di kodekan untuk QR code *version* 1 dengan menggunakan mode *alphanumeric* maka *character count indicator nya adalah 000001011* (11 dalam biner dengan *padding* disebelah kiri sehingga panjangnya menjadi 9 bit), *mode indicator dan character indicator* kemudian di gabungkan menjadi 00100000001011. Setelah di lakukan perhitungan mode dan *character count indicator* di lakukan pengkodean teks, setiap mode memiliki metode pengkodean teks yang berbeda. Contohnya pada teks “HELLO WORLD” yang akan di kodekan dengan menggunakan mode *alphanumeric*. pengkodean di lakukan dengan menggunakan rumus “*(45 * Nilai alphanumeric Karakter Kedua) + Nilai alphanumeric Karakter Pertama”* yang kemudian nilai hasilnya di konversi ke dalam bentuk biner.

Tabel 2.6 Tabel alphanumeric

Kode	Karakter	Kode	Karakter	Kode	Karakter
0	0	19	J	38	%
1	1	20	K	39	*
2	2	21	L	40	+
3	3	22	M	41	-
4	4	23	N	42	.
5	5	24	O	43	/
6	6	25	P	44	:
7	7	26	Q		
8	8	27	R		
9	9	28	S		
10	A	29	T		
11	B	30	U		
12	C	31	V		
13	D	32	W		
14	E	33	X		
15	F	34	Y		
16	G	35	Z		
17	H	36	SPACE		
18	I	37	$		

Mengacu pada Tabel 2.6 dapat di hitung hasil pengkodean dari teks “HELLO WORLD”, nilai dari H -> 17, nilai dari E -> 14, (45*17) + 14 = 779 -> 01100001011, proses pengkodean di lanjutkan per pasangan karakter, jika teks yang di proses berjumlah ganjil maka cukup dilakukan konversi nilai *alphanumeric* karakter terakhir dan diubah menjadi biner sepanjang 6bit. Berikut hasil dari pengkodean dengan menggunakan mode *alphanumeric* teks “HELLO WORLD” yang telah di gabungkan dengan mode dan *character count indicator* dijelaskan pada tabel Tabel 2.7.

Tabel 2.7 Representasi hasil pengkodean

Mode Indicator	Character Count Indicator	Encoded Data
0010	000001011	01100001011 01111000110 10001011100 10110111000 10011010100 001101

Data yang telah di kodekan di pecah menjadi data *codeword* sepanjang 8-*bit* dengan ditambahkan *terminator* (0) jika di perlukan, berikut data bit dari *string* " HELLO WORLD"

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000 11101100 00010001

setiap *version* dan level *error correction* memiliki aturan jumlah *codeword* yang berbeda-beda. Contohnya QR code *version* 1 dengan *error correction* level Q di *haruskan* memiliki sebanyak 13 *codeword*, dengan total bits 13 * 8 = 104 bit. Jumlah dari panjang data bit untuk teks "HELLO WORLD" adalah 74 bit, maka perlu di tambahkah *terminator* 0, jumlah maksimum *terminator* 0 yang boleh di tambahkan adalah 4 bit. Jika panjang bit data setelah penambahan *terminator* 0 bukan kelipatan delapan maka perlu di tambahkan lagi 0 secukupnya agar data bit menjadi kelipatan 8 (78 bit + 2bit 0 = 80 bit). Dikarenakan jumlah data bit belum mencapai nilai kapasitas maksimum (104 bit) maka perlu ditambahkan nilai berikut 11101100 00010001 hingga data bit mencapai nilai 104.

3. *Error Correction Coding*

Dilakukan perhitungan *error correction codeword* dengan menggunakan algoritma koreksi kesalahan reed-solomon. Tahapan pertama yang dilakukan adalah menentukan jumlah *error correction codeword* yang di perlukan (ditentukan oleh spesifikasi QR code) dan mengubah data bit menjadi bilangan desimal. Berikut hasil konversi data bits menjadi bilangan desimal :

32, 91, 11, 120, 209, 114, 220, 77, 67, 64, 236, 17, 236, 17, 236, 17

Nilai desimal hasil konversi di gunakan sebagai koefisien dari *message polynomial* yang ditentukan oleh spesifikasi QR code, berikut *message polynomial* dari data bit :

$32x^{15} + 91x^{14} + 11x^{13} + 120x^{12} + 209x^{11} + 114x^{10} + 220x^{9} + 77x^{8} + 67x^{7} + 64x^{6} + 236x^{5} + 17x^{4} + 236x^{3} + 17x^{2} + 236x^{1} + 17$

Kemudian dilakukan pembangkitan *generator polynomial*, untuk setiap kombinasi versi dan level *error correction* memiliki generator *polynomial* yang berbeda, generator *polynomial* untuk QR code 1-Q adalah sebagai berikut

$\alpha\, x^{10} + \alpha^{251}x^9 + \alpha^{67}x^8 + \alpha^{46}x^7 + \alpha^{61}x^6 + \alpha^{118}x^5 + \alpha^{70}x^4 + \alpha^{64}x^3 + \alpha^{94}x^2 + \alpha^{32}x + \alpha^{45}$.

α adalah elemen primitif pada *Galois Field* (2^8), yang dapat dikonversi menjadi bilangan integer menggunakan Tabel 2.8.

Tabel 2.8 *Log antilog table*

Exp α	**Integer**	**Integer**	**Exp α**	**Exp α**	**Integer**	**Integer**	**Exp α**
0	1			128	133	128	7
1	2	1	0	129	23	129	112
2	4	2	1	130	46	130	192
3	8	3	25	131	92	131	247
4	16	4	2	132	184	132	140
5	32	5	50	133	109	133	128
6	64	6	26	134	218	134	99
7	128	7	198	135	169	135	13
8	29	8	3	136	79	136	103
9	58	9	223	137	158	137	74
10	116	10	51	138	33	138	222
11	232	11	238	139	66	139	237
12	205	12	27	140	132	140	49
13	135	13	104	141	21	141	197
14	19	14	199	142	42	142	254
15	38	15	75	143	84	143	24
16	76	16	4	144	168	144	227
17	152	17	100	145	77	145	165
18	45	18	224	146	154	146	153
19	90	19	14	147	41	147	119
20	180	20	52	148	82	148	38
21	117	21	141	149	164	149	184
22	234	22	239	150	85	150	180
23	201	23	129	151	170	151	124
24	143	24	28	152	73	152	17
25	3	25	193	153	146	153	68
26	6	26	105	154	57	154	146
27	12	27	248	155	114	155	217
28	24	28	200	156	228	156	35
29	48	29	8	157	213	157	32
30	96	30	76	158	183	158	137
31	192	31	113	159	115	159	46
32	157	32	5	160	230	160	55

33	39	33	138	161	209	161	63
34	78	34	101	162	191	162	209
35	156	35	47	163	99	163	91
36	37	36	225	164	198	164	149
37	74	37	36	165	145	165	188
38	148	38	15	166	63	166	207
39	53	39	33	167	126	167	205
40	106	40	53	168	252	168	144
41	212	41	147	169	229	169	135
42	181	42	142	170	215	170	151
43	119	43	218	171	179	171	178
44	238	44	240	172	123	172	220
45	193	45	18	173	246	173	252
46	159	46	130	174	241	174	190
47	35	47	69	175	255	175	97
48	70	48	29	176	227	176	242
49	140	49	181	177	219	177	86
50	5	50	194	178	171	178	211
51	10	51	125	179	75	179	171
52	20	52	106	180	150	180	20
53	40	53	39	181	49	181	42
54	80	54	249	182	98	182	93
55	160	55	185	183	196	183	158
56	93	56	201	184	149	184	132
57	186	57	154	185	55	185	60
58	105	58	9	186	110	186	57
59	210	59	120	187	220	187	83
60	185	60	77	188	165	188	71
61	111	61	228	189	87	189	109
62	222	62	114	190	174	190	65
63	161	63	166	191	65	191	162
64	95	64	6	192	130	192	31
65	190	65	191	193	25	193	45
66	97	66	139	194	50	194	67
67	194	67	98	195	100	195	216
68	153	68	102	196	200	196	183
69	47	69	221	197	141	197	123
70	94	70	48	198	7	198	164
71	188	71	253	199	14	199	118
72	101	72	226	200	28	200	196
73	202	73	152	201	56	201	23
74	137	74	37	202	112	202	73
75	15	75	179	203	224	203	236
76	30	76	16	204	221	204	127
77	60	77	145	205	167	205	12
78	120	78	34	206	83	206	111
79	240	79	136	207	166	207	246
80	253	80	54	208	81	208	108
81	231	81	208	209	162	209	161
82	211	82	148	210	89	210	59
83	187	83	206	211	178	211	82
84	107	84	143	212	121	212	41
85	214	85	150	213	242	213	157

86	177	86	219	214	249	214	85
87	127	87	189	215	239	215	170
88	254	88	241	216	195	216	251
89	225	89	210	217	155	217	96
90	223	90	19	218	43	218	134
91	163	91	92	219	86	219	177
92	91	92	131	220	172	220	187
93	182	93	56	221	69	221	204
94	113	94	70	222	138	222	62
95	226	95	64	223	9	223	90
96	217	96	30	224	18	224	203
97	175	97	66	225	36	225	89
98	67	98	182	226	72	226	95
99	134	99	163	227	144	227	176
100	17	100	195	228	61	228	156
101	34	101	72	229	122	229	169
102	68	102	126	230	244	230	160
103	136	103	110	231	245	231	81
104	13	104	107	232	247	232	11
105	26	105	58	233	243	233	245
106	52	106	40	234	251	234	22
107	104	107	84	235	235	235	235
108	208	108	250	236	203	236	122
109	189	109	133	237	139	237	117
110	103	110	186	238	11	238	44
111	206	111	61	239	22	239	215
112	129	112	202	240	44	240	79
113	31	113	94	241	88	241	174
114	62	114	155	242	176	242	213
115	124	115	159	243	125	243	233
116	248	116	10	244	250	244	230
117	237	117	21	245	233	245	231
118	199	118	121	246	207	246	173
119	147	119	43	247	131	247	232
120	59	120	78	248	27	248	116
121	118	121	212	249	54	249	214
122	236	122	229	250	108	250	244
123	197	123	172	251	216	251	234
124	151	124	115	252	173	252	168
125	51	125	243	253	71	253	80
126	102	126	167	254	142	254	88
127	204	127	87	255	1	255	175

Tahapan penghitungan *error correction codeword* di mulai dengan melakukan perkalian eksponen dari *message polynomial* dengan n (n di sini adalah jumlah *error correction codeword* yang dibutuhkan yaitu contohnya 10 untuk QR code versi 1-Q) , berikut hasil perkalian *message polynomial* dengan n :

$$32x^{25} + 91x^{24} + 11x^{23} + 120x^{22} + 209x^{21} + 114x^{20} + 220x^{19} + 77x^{18} + 67x^{17} + 64x^{16} + 236x^{15} + 17x^{14} + 236x^{13} + 17x^{12} + 236x^{11} + 17x^{10}$$

Kemudian dilakukan perkalian generator *polynomial*, dikarenakan Eksponen terdepan dari generator *polynomial* harus memiliki nilai eksponen yang sama dengan eksponen terdepan dari *message polynomial*, berikut hasil perkalian eksponen generator *polynomial* dengan 15 :

$$\alpha^{0}x^{25} + \alpha^{251}x^{24} + \alpha^{67}x^{23} + \alpha^{46}x^{22} + \alpha^{61}x^{21} + \alpha^{118}x^{20} + \alpha^{70}x^{19} + \alpha^{64}x^{18} + \alpha^{94}x^{17} + \alpha^{32}x^{16} + \alpha^{45}x^{15}$$

Kemudian di lakukan pembagian *polynomial* antara *message polynomial* dibagi dengan generator *polynomial* dengan aturan sebagai berikut:

a. Temukan nilai satuan untuk mengalikan generator *polynomial*. Eksponen terdepan generator *polynomial* hasil perkalian harus memiliki nilai yang sama dengan eksponen dari *message polynomial*.
b. XOR hasil dengan *message polynomial*
c. Ulangi pembagian sebanyak n kali di mana n adalah jumlah dari *codeword*.

Koefisien dari hasil sisa bagi proses ini adalah nilai *error codeword* dari data bit.

4. Menyusun *Final Message*

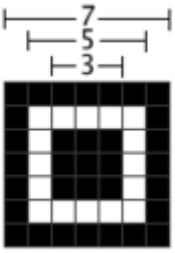
   Pada tahapan ini dilakukan proses *interleaving* pada data *codeword* dan *error correction codeword*, tahapan ini diperlukan untuk QR code di atas versi 5-Q dikarenakan memiliki lebih dari 1 *codeblock*, sedangkan untuk versi QR code yang lebih kecil tahapan ini tidak perlu di lakukan dikarenakan hanya ada 1 *codeblock* sehingga data dapat langsung di tempatkan ke dalam matriks.

5. Menempatkan Modul pada Matriks

   Ada beberapa tahapan penempatan modul pada *matrix*, yaitu:

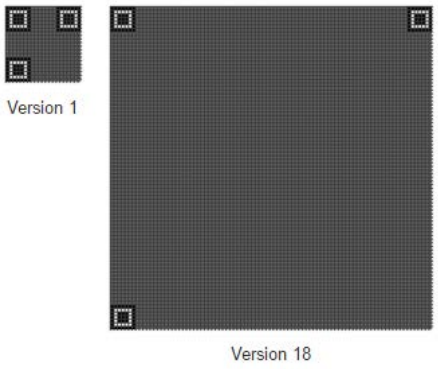
a. Menambahkan *finder pattern*

Finder pattern digunakan oleh pembaca QR code untuk menentukan orientasi posisi dari QR code , *finder pattern* dapat dilihat pada Gambar 2.5.



Gambar 2.5 Finder pattern QR code

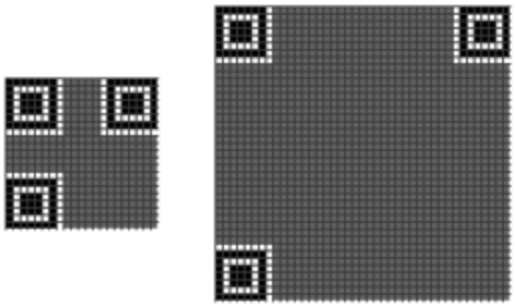
Sedangkan contoh dari *finder pattern* yang telah ditempatkan pada QR code dapat dilihat pada Gambar 2.6.



Gambar 2.6 Penempatan finder pattern QR code

b. Menambahkan *separator*

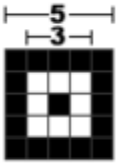
Separator adalah garis module berwarna putih yang ditempatkan disebalah *finder pattern* dengan tujuan memisahkan antar *finder pattern* dari bagian QR code lainnya. Contoh dari separator dapat dilihat pada Gambar 2.7.



Gambar 2.7 Separator QR code

c. Menambahkan *alignment pattern*

QR code dengan version 2 keatas diharuskan memiliki *alignment pattern* , contoh dari *alignment pattern* dapat dilihat pada Gambar 2.8.



Gambar 2.8 Alignment pattern QR code

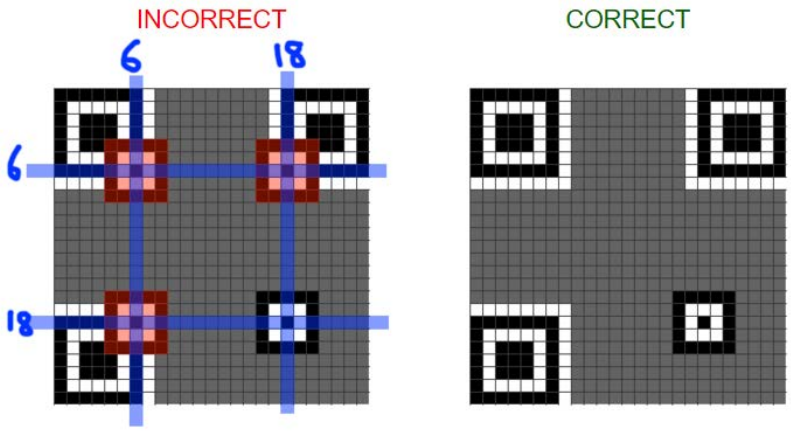
Kordinat penempatan center module *alignment pattern* untuk setiap version QR code dijelaskan Tabel 2.9.

Tabel 2.9 Kordinat penempatan allignment pattern

Version	**Center Module Row and Column**						
QR Version 2	6	18					
QR Version 3	6	22					
QR Version 4	6	26					
QR Version 5	6	30					
QR Version 6	6	34					
QR Version 7	6	22	38				
QR Version 8	6	24	42				
QR Version 9	6	26	46				
QR Version 10	6	28	50				
QR Version 11	6	30	54				
QR Version 12	6	32	58				
QR Version 13	6	34	62				
QR Version 14	6	26	46	66			
QR Version 15	6	26	48	70			
QR Version 16	6	26	50	74			
QR Version 17	6	30	54	78			
QR Version 18	6	30	56	82			
QR Version 19	6	30	58	86			

QR Version 20	6	34	62	90			
QR Version 21	6	28	50	72	94		
QR Version 22	6	26	50	74	98		
QR Version 23	6	30	54	78	102		
QR Version 24	6	28	54	80	106		
QR Version 25	6	32	58	84	110		
QR Version 26	6	30	58	86	114		
QR Version 27	6	34	62	90	118		
QR Version 28	6	26	50	74	98	122	
QR Version 29	6	30	54	78	102	126	
QR Version 30	6	26	52	78	104	130	
QR Version 31	6	30	56	82	108	134	
QR Version 32	6	34	60	86	112	138	
QR Version 33	6	30	58	86	114	142	
QR Version 34	6	34	62	90	118	146	
QR Version 35	6	30	54	78	102	126	150
QR Version 36	6	24	50	76	102	128	154
QR Version 37	6	28	54	80	106	132	158
QR Version 38	6	32	58	84	110	136	162
QR Version 39	6	26	54	82	110	138	166
QR Version 40	6	30	58	86	114	142	170

Dengan aturan penempatan setiap *alignment pattern* tidak boleh menimpah *finder pattern* dan *separator*, jika menimpah maka *alignment pattern* pada kordinat tersebut tidak ditambahkan. Contoh penempatan *alignment pattern* dapat dilihat pada Gambar 2.9.

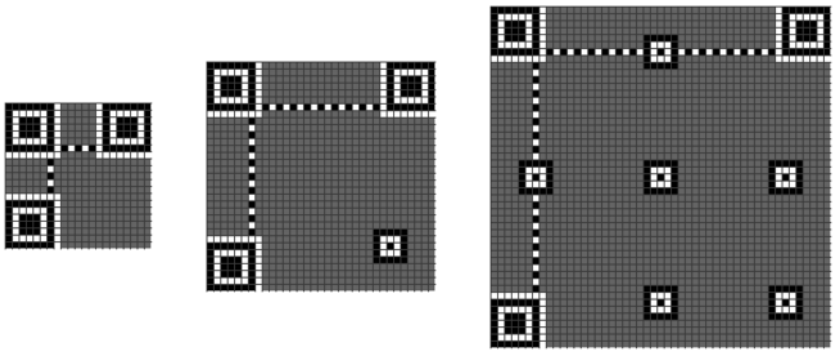


Gambar 2.9 Penempatan allignment pattern QR code

d. Menambahkan *timming pattern*

Timming pattern adalah dua garis horizintal dan vertical yang berisi module gelap dan terang secara bergantian. Horizontal *timming pattern* ditempatkan pada baris ke 6 sedangkan vertical *timming pattern* ditempatkan pada kolom ke 6, *timming pattern* selalu

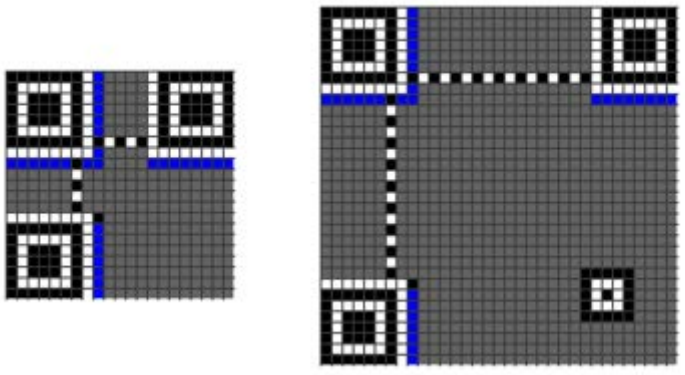
dimulai dan diakhiri oleh modul gelap. Contoh penempatan *timming pattern* dapat dilihat pada Gambar 2.10.



Gambar 2.10 Penempatan timming pattern QR code

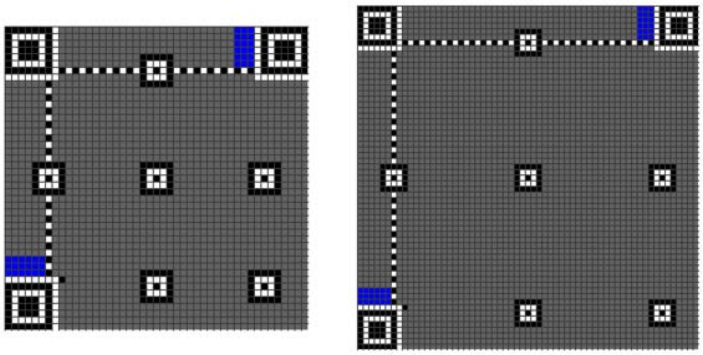
e. Menambahkan *dark module* dan *reserved area*

Sebelum dilakukan penempatan bit data harus dilakukan penempatan *dark module* dan *reserved area* yang nantinya digunakan untuk menyimpan *format dan information bits, dark module* ditempatkan pada kordinat yang dihitung dengan formula ([(4 * V) + 9], 8) dimana V adalah version dari QR code. Sedangkan untuk penempatan *reserved area* dapat dilihat pada Gambar 2.11.



Gambar 2.11 Penempatan reserved area QR code

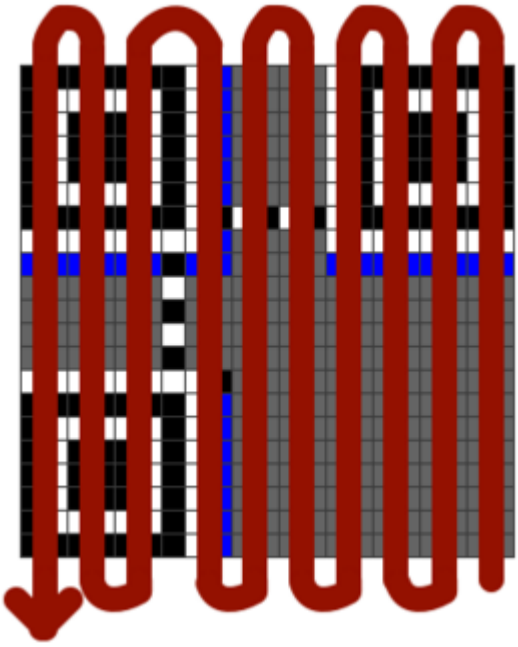
Untuk QR code version 7 keatas diperlukan *version information area* seluas 6x3 module yang penempatanya dapat dilihat pada Gambar 2.12.



Gambar 2.12 Version information area QR code

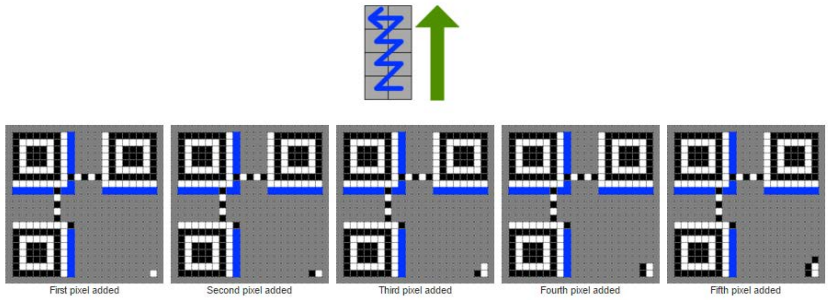
f. Menempatkan bit data dan *error correction* data pada QR code sesuai dengan aturan penempatan bit data.

Pola umum penempatan bit data dan *error correction* data dapat dilihat pada Gambar 2.13.



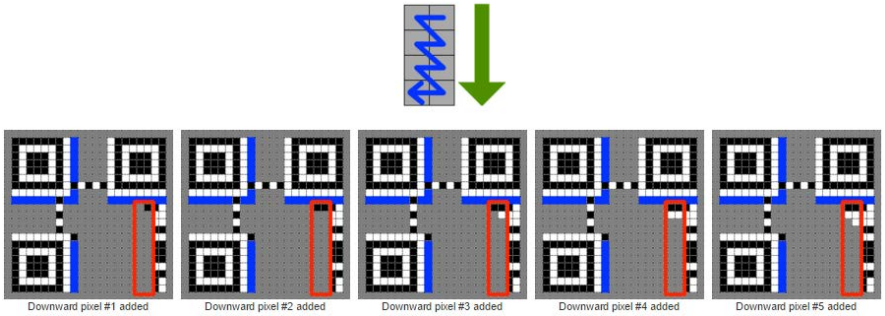
Gambar 2.13 Pola umum penempatan bit data QR code

Jika penempatan data mengarah keatas maka penempatan data dilakukan sesuai dengan pola yang digambarkan pada Gambar 2.14.



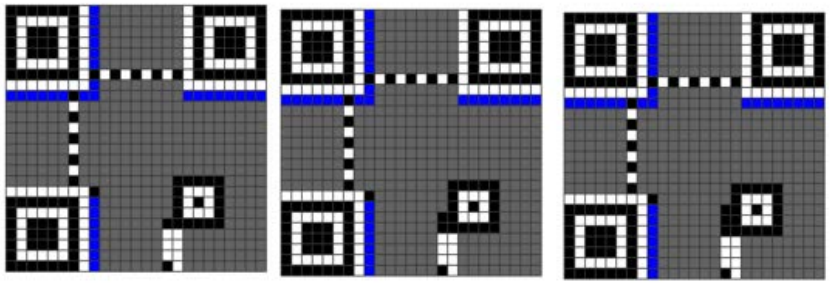
Gambar 2.14 Penempatan bit data dengan arah atas

Sedangkan jika penempatan data mengarah kebawah maka dilakukan sesuai dengan pola yang digambarkan pada Gambar 2.15.



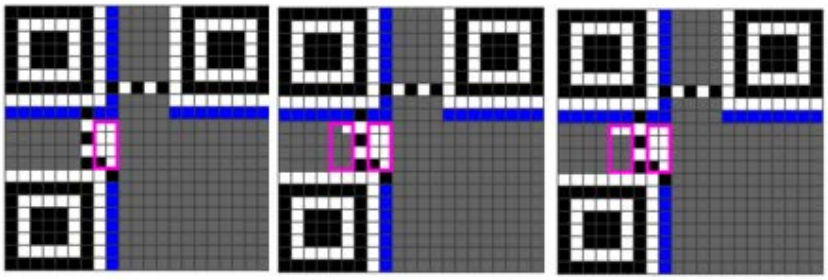
Gambar 2.15 Penempatan bit data dengan arah bawah

Jika penempatan melewati *function pattern* seperi *aligment pattern, timing pattern dan reserved area* maka, dipilih modul terdekat yang tidak digunakan seperti digambarkan pada Gambar 2.16.



Gambar 2.16 Penempatan bit data jika melewati function pattern

Aturan khusus digunakan jika penempatan data melewati *vertical timming pattern*, selalu gunakan modul disebelah kiri dari *vertical timming pattern* seperti digambarkan pada Gambar 2.17.



Gambar 2.17 Penempatan bit data jika melewati vertical timming pattern

6. *Masking*

Masking adalah proses mengubah modul QR code dengan cara menukar warna dengan tujuan untuk mempermudah pembaca QR code untuk mengenali data, spesifikasi QR code mendefinisikan 8 jenis *mask pattern* yang dapat di gunakan. Berikut aturan dari setiap *mask pattern* dijelaskan pada Tabel 2.10.

Tabel 2.10 Formula *mask pattern*

Mask Number	**Jika Hasil Evaluasi Formula True Maka Ganti Nilai Data Pada Kordinat**
0	(row + column) mod 2 == 0
1	(row) mod 2 == 0
2	(column) mod 3 == 0
3	(row + column) mod 3 == 0
4	(floor(row / 2) + floor(column / 3)) mod 2 == 0
5	((row * column) mod 2) + ((row * column) mod 3) == 0
6	(((row * column) mod 2) + ((row * column) mod 3)) mod 2 == 0
7	(((row + column) mod 2) + ((row * column) mod 3)) mod 2 == 0

Masking digunakan untuk mengurangi nilai penalti yang di tentukan oleh 4 aturan yang di gunakan untuk mengevaluasi kemudahan pembacaan sebuah QR code. Aturan-aturan tersebut adalah sebagai berikut:

1. Periksa setiap kolom dan baris ,tambahkan 3 nilai penalti jika di temukan 5 modul dengan warna yang sama secara berurut dalam satu kolom atau baris, jika ditemukan modul dengan warna yang sama setelah 5 modul pertama maka tambahkan 1 nilai penalti untuk

setiap modul dengan warna yang sama (misalkan ditemukan 6 modul maka nilai penalti adalah 4) .

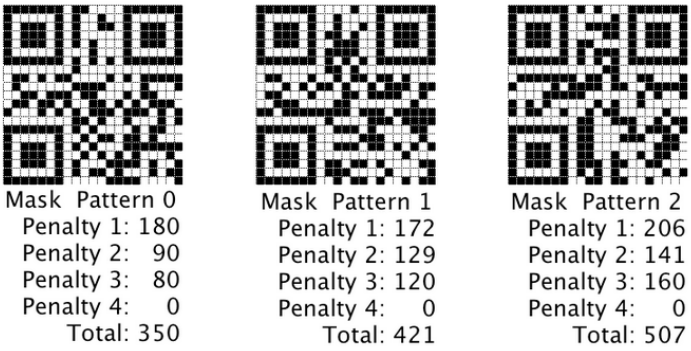
2. Tambahkan 3 nilai penalti untuk setiap area 2x2 yang memiliki warna yang sama pastikan menghitung setiap modul yang saling menimpah (3x2 dihitung sebagai dua 2x2 block).
3. Tambahkan 40 nilai penalti jika di temukan pola yang menyerupai dengan *finder pattern* , pola tersebut dapat dilihat pada Gambar 2.18.



Gambar 2.18 Pola menyerupai finder pattern

4. Tambahkan nilai penalti jika sebagian dari QR code berwarna putih atau hitam, dengan penalti semakin besar untuk semakin besar tingkatan perbedaannya, dengan langkah perhitungan sebagai berikut:
    i. Hitung jumlah total modul dalam matriks.
    ii. Hitung berapa banyak modul gelap yang ada dalam matriks.
    iii. Hitung persentase modul gelap yang ada dalam (modul gelap / total modul) * 100
    iv. Tentukan berapa kelipatan lima persen sebelum dan sesudah dari nilai. Misalnya, untuk 43 persen, lima persen sebelum nya adalah 40 dan 5 persen sesudah nya adalah 45.
    v. Kurangi 50 dari masing-masing nilai kelipatan ini dan ambil nilai absolut dari hasilnya. Misalnya, | 40 - 50 | = | -10 | = 10 dan | 45 - 50 | = | -5 | = 5.
    vi. Bagi masing masing nilai dengan 5. Misalnya, 10/5 = 2 dan 5/5 = 1.
    vii. Ambil nilai terkecil dari kedua angka dan kalikan dengan 10. Dalam contoh ini, angka yang lebih rendah adalah 1, sehingga hasilnya adalah 10.

Contoh nilai penalti dari beberapa masking dapat dilihat pada Gambar 2.19.



Gambar 2.19 Contoh nilai penalti beberapa *masking*

7. Format dan *version Information*

Dilakukan penambahan Format *information* yang berisi level *error* koreksi dan pola *masking* yang digunakan, berikut format *information string* untuk setiap level koreksi *error* dan pola *masking*, dijelaskan pada Tabel 2.11.

Tabel 2.11 *Format information string*

ECC Level	**Mask Pattern**	**Format Information Bits**
L	0	111011111000100
L	1	111001011110011
L	2	111110110101010
L	3	111100010011101
L	4	110011000101111
L	5	110001100011000
L	6	110110001000001
L	7	110100101110110
M	0	101010000010010
M	1	101000100100101
M	2	101111001111100
M	3	101101101001011
M	4	100010111111001
M	5	100000011001110
M	6	100111110010111
M	7	100101010100000
Q	0	011010101011111
Q	1	011000001101000
Q	2	011111100110001
Q	3	011101000000110
Q	4	010010010110100
Q	5	010000110000011
Q	6	010111011011010
Q	7	010101111101101
H	0	001011010001001
H	1	001001110111110
H	2	001110011100111

H	3	001100111010000
H	4	000011101100010
H	5	000001001010101
H	6	000110100001100
H	7	000100000111011

Jika *version* QR code yang digunakan lebih besar dari *version* 8 maka ditambahkan juga *version information* yang mengindikasikan *version* QR code yang di gunakan. Berikut *version information string* untuk setiap *version* QR code, dijelaskan pada Tabel 2.12.

Tabel 2.12 *Version information string*

Version	**Version Information String**
7	000111110010010100
8	001000010110111100
9	001001101010011001
10	001010010011010011
11	001011101111110110
12	001100011101100010
13	001101100001000111
14	001110011000001101
15	001111100100101000
16	010000101101111000
17	010001010001011101
18	010010101000010111
19	010011010100110010
20	010100100110100110
21	010101011010000011
22	010110100011001001
23	010111011111101100
24	011000111011000100
25	011001000111100001
26	011010111110101011
27	011011000010001110
28	011100110000011010
29	011101001100111111
30	011110110101110101
31	011111001001010000
32	100000100111010101
33	100001011011110000
34	100010100010111010
35	100011011110011111
36	100100101100001011
37	100101010000101110
38	100110101001100100
39	100111010101000001
40	101000110001101001

2.2.5.2 *Decoding* QR Code [6]

Tahapan *decoding* QR code dari pembacaan sampai menjadi *output* data merupakan kebalikan dari prosedur *encoding*, tahapan – tahapan *decoding* adalah sebagai berikut:

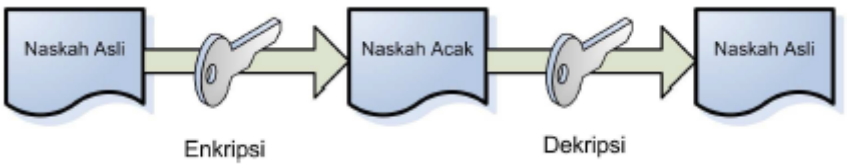
1. Cari dan temukan citra simbol, kenali modul gelap dan terang sebagai *array* “0” dan "1”.
2. Baca informasi format.
3. Baca *version information* dan tentukan *version* dari QR code
4. Lepaskan *masking* dengan cara menaplikasi operasi XOR pada daerah yang di *masking* dengan *pattern masking* yang di dapatkan dari ekstraksi informasi format.
5. Baca simbol karakter sesuai dengan aturan penempatan dan kembalikan data dan *error correction codeword* dari pesan.
6. Deteksi *erro*r menggunakan *error correction codeword*, jika di temukan *error* , perbaiki.
7. Pisahkan data *codeword* menjadi *segment* sesuai dengan *mode indicator* dan *character count indicator*.
8. *Decode* karakter sesuai dengan mode yang digunakan dan keluarkan hasil.

2.2.6 Kriptografi [7]

Kriptografi adalah ilmu mengenai teknik enkripsi di mana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi. Dekripsi menggunakan kunci dekripsi mendapatkan kembali data asli. Proses enkripsi dilakukan menggunakan suatu algoritma dengan beberapa parameter. Biasanya algoritma tidak dirahasiakan, bahkan enkripsi yang mengandalkan kerahasiaan algoritma dianggap sesuatu yang tidak baik. Rahasia terletak di beberapa parameter yang digunakan, jadi kunci ditentukan oleh parameter. Parameter yang menentukan kunci dekripsi itulah yang harus dirahasiakan (parameter menjadi ekuivalen dengan kunci).

Dalam kriptografi klasik, teknik enkripsi yang digunakan adalah enkripsi simetris di mana kunci dekripsi sama dengan kunci enkripsi. Untuk *public key cryptography*, diperlukan teknik enkripsi asimetris di mana kunci dekripsi tidak

sama dengan kunci enkripsi. Enkripsi, dekripsi dan pembuatan kunci untuk teknik enkripsi asimetris memerlukan komputasi yang lebih intensif dibandingkan enkripsi simetris, karena enkripsi asimetris menggunakan bilangan-bilangan yang sangat besar. Namun, walaupun enkripsi asimetris lebih "mahal" dibandingkan enkripsi simetris, *public key cryptography* sangat berguna untuk *key management* dan *digital signature*.



Gambar 2.20 Gambaran proses enkripsi

Gambar di atas menunjukkan efek dari proses enkripsi dan proses dekripsi. Secara garis besar, proses enkripsi adalah proses pengacakan "naskah asli" (*plaintext*) menjadi "naskah acak" (*ciphertext*) yang "sulit untuk dibaca" oleh seseorang yang tidak mempunyai kunci dekripsi. Yang dimaksud dengan "sulit untuk dibaca" di sini adalah probabilitas mendapat kembali naskah asli oleh seseorang yang tidak mempunyai kunci dekripsi dalam waktu yang tidak terlalu lama adalah sangat kecil. Jadi suatu proses enkripsi yang baik menghasilkan naskah acak yang memerlukan waktu yang lama (contohnya satu juta tahun) untuk didekripsi oleh seseorang yang tidak mempunyai kunci dekripsi. Satu cara untuk mendapatkan kembali naskah asli tentunya dengan menerka kunci dekripsi, jadi proses menerka kunci dekripsi harus menjadi sesuatu yang sulit. Tentunya naskah acak harus dapat didekripsi oleh seseorang yang mempunyai kunci dekripsi untuk mendapatkan kembali naskah asli.

2.2.7 Kriptografi *Public Key* [8]

Dalam kriptografi klasik (simetris), jika seseorang mengetahui cara mengenkripsi naskah asli menjadi naskah acak, maka orang tersebut juga mengetahui cara mendekripsi naskah acak yang dihasilkan. Demikian juga jika seseorang mengetahui cara mendekripsi naskah acak, maka orang tersebut juga mengetahui cara mengenkripsi naskah asli untuk menghasilkan naskah acak.

Sekitar pertengahan tahun 1970an, muncul konsep baru dalam kriptografi yaitu kriptografi public key (asimetris). Seseorang yang mengetahui cara mengenkripsi naskah asli belum tentu mengetahui juga cara mendekripsi naskah acak yang dihasilkan. Demikian juga seseorang yang mengetahui cara mendekripsi naskah acak belum tentu mengetahui juga cara mengenkripsi naskah asli untuk menghasilkan naskah acak tersebut. Enkripsi dan dekripsi dalam kriptografi *public key* menggunakan sepasang kunci yaitu kunci publik (*public key*) dan kunci privat (*private key*). Naskah yang telah dienkripsi menggunakan kunci privat hanya dapat didekripsi menggunakan kunci publik dan naskah yang dapat didekripsi menggunakan kunci publik dapat dipastikan telah di enkripsi menggunakan kunci privat. Sebaliknya, naskah yang telah dienkripsi menggunakan kunci publik hanya dapat didekripsi menggunakan kunci privat. Mekanisme ini memungkinkan berbagai aplikasi, dua yang terpenting di antaranya adalah distribusi kunci sesi dan tanda tangan digital (*digital signature*).

Jika A ingin mengirim kunci sesi atau rahasia lainnya ke B dan hanya ingin B yang dapat membacanya, maka A mengenkripsi rahasia tersebut menggunakan kunci publik milik B. Dengan asumsi hanya B yang memiliki kunci privat B, maka hanya B yang dapat mendekripsi rahasia yang telah dienkripsi tersebut. Cara inilah yang kerap digunakan untuk mendistribusikan kunci sesi menggunakan kriptografi public key.

Jika A ingin menanda-tangan suatu naskah secara digital, maka A mengen kripsi naskah tersebut menggunakan kunci privat miliknya dan hasil enkripsi merupakan “tanda tangan,” Jika seseorang (sebut saja B) ingin memeriksa apakah naskah tersebut telah ditandatangan oleh A, maka B mendekripsi “tanda tangan” tersebut dengan kunci publik milik A dan membandingkan hasil dekripsi dengan naskah yang ditandatangan. Jika sama maka B dapat meyakinkan dirinya sendiri bahwa A telah menanda-tangan naskah tersebut karena hanya A yang memiliki kunci privat yang digunakan untuk mengen- kripsi naskah untuk menghasilkan “tanda-tangan.” Dalam prakteknya, yang dienkripsi bukan naskah penuh melainkan digest dari naskah tersebut.

Jadi dalam kriptografi public key, kunci publik dapat disebar-luaskan kepada umum dan sebaiknya disebar luaskan. Sebaliknya, kunci privat harus dirahasiakan oleh pemiliknya.

2.2.8 Tanda Tangan Digital [9]

Sejak ditemukan pada tahun 1970-an, tanda tangan digital di harapkan dapat berperan sebagaimana tanda tangan pada dokumen kertas. Perkembangan teknologi yang pesat menjadikan tanda tangan digital sebagai komponen penting dalam bisnis di *cyberspace* saat ini. Bahkan di bebeberapa negara maju tanda tangan digital telah memiliki kekuatan hukum.

Di saat tanda tangan digital mulai lazim digunakan dan bahkan berkekuatan hukum, sebagian ahli di bidang informatika justru mulai mempertanyakan keabsahan tanda tangan digital. Bukan karena lemahnya kekuatan matematis dari algoritma yang di gunakan, tetapi lebih keapa banyaknya celah kelemahan pada implementasi tanda tangan digital tersebut.

Tanda tangan digital merupakan tanda tangan yang di lakukan dengan memakai alat elektronik yang berfungsi sama dengan tanda tangan manual. Tanda tangan digital merupakan kumpulan bit yang bisa melakukan fungsi elektronik yang memakai fungsi Hash satu arah.

Pada dasarnya tanda tangan digital dari setiap dokumen berbeda dengan dokumen lain, karena di ambil dari dokumen itu sendiri. Bilamana terjadi perubahan pada dokumen maka hal itu tentu akan menciptakan tanda tangan digital yang berbeda. Tanda tangan digital juga mempunyai fungsi yang sama dengan tanda tangan manual yang mengabsahkan sesuatu dokumen yang bisa di jadikan persetujuan, tanda tterima dan lain sebagainya.

Sifat dari tanda tangan digital di antaranya:

1. *Authentication:* Jaminan dari suatu pesan yang belum di modifikasi di dalam pergiriman (keaslian pesan atau intergritas pesan). Juga kunci yang membuktikan keaslian untuk kunci publik, pemakai atau identifikasi sumber, yang boleh memverifikasi hak untuk mengirimkan pesan. Penerima pesan dapat memastika keaslian pengirimnya. Kriptanalis tidak bisa

berpura-pura mengirim pesan tersebut. Selain itu juga merupakan suatu proses dalam rangka validasi user pada saat masuk kedalam sistem.

2. Cuma berlaku untuk sekali pengiriman dokumen yang mana tanda tangan digital tersebut tidak bisa di pindahkan ke dokumen lain dan jika dokumen tersebut diubah maka dokumen yang dikirim tidak berlaku lagi.
3. Keabsahan tanda tangan digital itu dapat diperiksa oleh pihak yang menerima pesan, walaupun belum pernah saling bertemu sekalipun.

Penandatanganan digital terhadap suatu dokumen adalah sidik jadi dari dokumen tersebut, yang beserta timestamp-nya dienkripsi menggunakan kunci rahasia pihak yang menandatangani. Tanda tanda tangan digital memanfaatkan fungsi Hash satu arah untuk menjadi bahwa tanda tangan itu hanya berlaku untuk dokumen yang bersangkutan saja.

2.2.9 DSA [10]

DSA (*Digital Signature Algorithm*) adalah salah satu algoritma yang digunakan dalam DSS (*Digital Signature Standard*), standard untuk *digital signature* yang dibuat oleh FIPS. DSS juga memperbolehkan penggunaan RSA. Karena DSS mewajibkan penggunaan SHA-1. maka DSA atau RSA digunakan untuk mengenkripsi *digest* sebesar 160 bit.

Parameter yang digunakan oleh DSA adalah sebagai berikut:

1. Suatu bilangan prima p yang dipilih menggunakan *random number generator* minimum 512 bit, sebaiknya 1024 bit.
2. Suatu bilangan prima q yang dipilih menggunakan *random number generator* sebesar 160 bit dimana q membagi $p - 1$. Untuk implementasi, mungkin lebih mudah untuk memilih q terlebih dahulu kemudian memilih p dimana $p \equiv 1 \pmod{q}$.
3. Suatu bilangan $g \in \mathbf{GF}(p)*$ yang mempunyai *order* q yang dipilih sebagai berikut: Pilih bilangan g_0 menggunakan *random number generator* dimana $1 < g_0 < p - 1$ lalu komputasi $g = g_0^{(p-1)/q} \bmod p$. Jika $g > 1$ maka itulah g yang dipilih. Jika tidak maka pilih g_0 yang lain sampai kita dapatkan $g > 1$.
4. Suatu bilangan x yang dipilih menggunakan *random number generator* dimana $0 < x < q$.

5. Bilangan $y = g^x \bmod p$.
6. Suatu bilangan k yang dipilih menggunakan *random number generator* dimana $0 < k < q$. Setiap kali menanda-tangan, k yang baru harus dipilih.

Parameter p,q,g,x dan y merupakan parameter semi-permanen, sedangkan k yang baru harus dipilih setiap kali menanda-tangan. Parameter p,q,g dan y merupakan parameter publik, jadi harus disebar-luaskan, sedangkan parameter x dan k harus dirahasiakan.

Berikut adalah cara DSA membuat *digital signature* untuk naskah M menggunakan parameter diatas. Hasil pembuatan *digital signature* adalah sepasang bilangan bulat (r,s) dimana $0 \leq r,s < q$. Rumus untuk r dan s adalah sebagai berikut:

$$r = (g^k \bmod p) \bmod q,\ s = (k^{-1}(\text{SHA-1}(M) + xr) \bmod q,$$

dimana SHA-1(M) adalah hasil komputasi *digest* menggunakan SHA. Jika ternyata $r = 0$ atau $s = 0$ (kemungkinannya sangat kecil), maka pembuatan *digital signature* dapat diulang menggunakan nilai k yang lain.

Seseorang yang ingin memeriksa *digital signature* akan mendapatkan M,r dan s. Pertama r dan s diperiksa nilainya apakah $0 < r < q$ dan $0 < s < q$. Jika tidak maka *digital signature* ditolak. Pemeriksa kemudian mengkomputasi:

$$\begin{aligned} w &= s^{-1} \bmod q, \\ u_1 &= \text{SHA-1}(M)w \bmod q, \\ u_2 &= rw \bmod q, \text{ dan} \\ v &= (g^{u1}y^{u2} \bmod p) \bmod q. \end{aligned}$$

Jika $v = r$ maka pemeriksa dapat cukup yakin bahwa M telah ditanda-tangan menggunakan kunci pasangan dari y. Mari kita buktikan bahwa jika semua komputasi dilakukan sesuai dengan aturan DSA maka $v = r$. Untuk itu kita buat M' = SHA-1(M), jadi

$$\begin{aligned} s &= (k^{-1}(M' + xr) \bmod q, \\ u_1 &= M'w \bmod q. \end{aligned}$$

Semua *inverse* dalam pembuktian adalah dalam **GF**(q). Menggunakan rumus komputasi kita dapatkan

$$\begin{aligned} v &= (g^{u1}y^{u2} \bmod p) \bmod q \\ &= (g^{M'w}y^{rw} \bmod p) \bmod q \\ &= (g^{M'w}g^{xrw} \bmod p) \bmod q \\ &= (g(M'+xr)w \bmod p) \bmod q. \end{aligned}$$

Karena $s = k^{-1}(M' + xr) \bmod q$,

maka

$$w = k(M' + xr)^{-1} \bmod q,$$

jadi

$$(M' + xr)w \bmod q = k \bmod q.$$

Kembali ke v:

$$v = (g^k \bmod p) \bmod q = r.$$

Selesailah pembuktian kita.

Suatu hal yang menarik dengan DSA adalah fungsi dari parameter k. Setiap kali menanda-tangan, nilai k yang baru dipilih secara acak. Walaupun naskah yang ditanda-tangan sama, jika ditanda-tangan dua kali oleh pemilik a, tanda tangannya berbeda. Jadi k berfungsi seperti *initialization vector*.

2.2.10 Kriptografi *Elliptic Curve* [11]

Sistem kriptografi *public key* yang berbasis pada sukarnya mengkomputasi logaritma diskrit seperti Diffie-Hellman, DSA dan ElGamal bekerja menggunakan suatu *multiplicative group* **GF**(q)$*$. Suatu *elliptic curve over a finite field* juga memberikan *Abelian group* yang dapat digunakan untuk mekanisme kriptografi yang serupa dengan sistem berbasis logaritma diskrit. Lebih menarik lagi, *elliptic curve over a finite field* memberikan lebih banyak fleksibilitas dibandingkan *finite field* yang terbatas pada **GF**(p) dan **GF**(p^n).

Logaritma diskrit untuk *elliptic curve* jauh lebih sukar dibandingkan logaritma diskrit untuk **GF**(q) (kecuali untuk *supersingular elliptic curve* yang sangat jarang). Untuk *elliptic curve*, secara umum logaritma diskrit hanya dapat dikomputasi menggunakan algoritma Shanks atau algoritma Pollard, dan kedua algoritma mempunyai kompleksitas *full exponential*. Untuk *finite field*, kompleksitas

logaritma diskrit mirip dengan kompleksitas penguraian yaitu *sub-exponential*. Jadi kompleksitas untuk *elliptic curve* tumbuh lebih cepat dibandingkan kompleksitas untuk *finite field*. Berdasarkan perkiraan oleh Alfred Menezes, penggunaan kriptografi *finite field* seperti DSA atau RSA dengan kunci sebesar 1024 bit sama kuatnya dengan penggunaan kripografi *elliptic curve* dengan kunci sebesar 160 bit. Jadi cukup menyolok perbedaan besar kunci untuk kekuatan yang sama. Untuk kekuatan yang lebih besar, perbedaan semakin menyolok karena kekuatan *elliptic curve* tumbuh lebih cepat (*exponential* dibandingkan *sub-exponential* untuk *finite field*), seperti terlihat pada Tabel 2.13 ECDSA adalah DSA versi *elliptic curve*

Tabel 2.13 Besar kunci untuk kekuatan yang sama

DSA/ElGamal	**RSA**	**ECDSA/ECES**
1024	1024	160
2048	2048	224
3072	3072	256
7680	7680	384
15360	15360	512

Sedangkan ECES adalah ElGamal versi *elliptic curve*. Tabel II.8 menunjukkan relatif waktu yang dibutuhkan untuk berbagai operasi. Untuk *key generation*, RSA jauh lebih lambat dibandingkan DSA/ElGamal

Tabel 2.14 Waktu untuk berbagai operasi

	DSA/ElGamal	**RSA**	**ECDSA/ECES**
	1024 bit	1024 bit	160 bit
encryption	480	17	120
decryption	240	384	60
signing	240	384	60
verification	480	17	120

dan ECDSA/ECES. Jadi cukup jelas mengapa sistem kriptografi *public key* dengan *elliptic curve* sangat menarik, terutama untuk aplikasi di perangkat kecil dengan kemampuan terbatas seperti *smartcard* atau perangkat *Bluetooth*. Kriptografi *elliptic curve* juga semakin menarik untuk penggunaan masa depan karena pertumbuhan kunci yang dibutuhkan tidak sebesar kriptografi *finite field*. Sebetulnya versi *elliptic curve* untuk RSA juga ada, namun berbeda dengan logaritma diskrit dimana versi *elliptic curve* lebih sukar untuk dipecahkan

dibandingkan versi *finite field*, penguraian bilangan bulat tetap merupakan penguraian bilangan bulat, jadi tidak ada keuntungan dengan menggunakan *elliptic curve* untuk RSA.

2.2.11 SHA [12]

Algoritma *secure hashing* SHA dirancang oleh National Security Agency (NSA) dan dijadikan standard FIPS. Ada 4 varian SHA dalam standard FIPS-180-2 dengan parameter yang berbeda. 4 varian SHA dapat dilihat pada tabel II.9. Keamanan algoritma didasarkan pada fakta bahwa *birthday attack* pada *digest* sebesar *n* bit menghasilkan *collision* dengan faktor kerja sekitar $2^{n/2}$.

Tabel 2.15 Varian SHA

Algoritma	**Naskah (bit)**	**Block (Bit)**	**Word (Bit)**	***Digest* (bit)**	**Keamanan (bit)**
SHA-1	$< 2^{64}$	512	32	160	80
SHA-256	$< 2^{64}$	512	32	256	128
SHA-384	$< 2^{128}$	1024	64	384	192
SHA-512	$< 2^{128}$	1024	64	512	256

SHA-1 menggunakan fungsi f_t, dimana $0 \leq t \leq 79$, dengan masukan 3 *word* masing-masing sebesar 32 bit dan menghasilkan keluaran 1 *word*:

$$f(x) = \begin{cases} Ch(x,y,z) = (x \wedge a) \oplus (\neg x \wedge z), 0 \leq t \leq 19 \\ Parity(x,y,z) = x \oplus y \oplus z, 20 \leq t \leq 39 \\ Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z), 40 \leq t \leq 59 \\ Parity(x,y,z) = x \oplus y \oplus z, 60 \leq t \leq 79 \end{cases}$$

SHA-1 menggunakan konstan k_t sebagai berikut (dengan menggunakan *hexadecimal*):

$$f(x) = \begin{cases} 5a827999 \; 0 \leq t \leq 19 \\ 6ed9eba1 \; 20 \leq t \leq 39 \\ 8f1bbcdc \; 40 \leq t \leq 59 \\ ca62c1d6 \; 60 \leq t \leq 79 \end{cases}$$

Seperti halnya dengan MD5, SHA-1 terdiri dari dua tahap yaitu *preprocessing* dan *hashing*. *Preprocessing* dimulai dengan *padding* yang prosesnya persis sama dengan MD5, yaitu setelah akhir naskah, 1 bit dengan nilai 1 ditambahkan, disusul oleh bit dengan nilai 0 sebanyak 0 sampai dengan 511 tergantung panjang naskah, dan diahiri dengan 64 bit yang merepresentasikan besar naskah asli. Setelah *padding*, naskah terdiri dari *n word M*[0 . . . n − 1] dimana n adalah kelipatan 16.

Langkah berikutnya dalam preprocessing adalah menyiapkan SHA-1 *buffer* sebesar 5 word:

$$\left(H_0^{(0)}, H_1^{(0)}, H_2^{(0)}, H_3^{(0)}, H_4^{(0)}\right).$$

Buffer diberi nilai awal sebagai berikut (nilai dalam hexadecimal):

$$H_0^{(0)} \leftarrow 67452301$$
$$H_1^{(0)} \leftarrow efcdab89$$
$$H_2^{(0)} \leftarrow 98badcfe$$
$$H_3^{(0)} \leftarrow 10325476$$
$$H_4^{(0)} \leftarrow c3d2e1f0$$

Setelah *buffer* diberi nilai awal, tahap kedua yaitu *hashing* dilakukan terhadap setiap blok ($M^{(1)}$, $M^{(2)}$, ..., $M^{(n)}$) sebagai berikut (i = 1, 2, ..., n):

1. Siapkan *message schedule* $\{W_t\}$:

$$W_t \leftarrow \begin{cases} M_t^{(i)}, 0 \le t \le 15 \\ (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) <<< 1, 16 \le t \le 79 \end{cases}$$

2. Berikan nilai awal untuk variabel *a, b, c, d,* dan *e*

$$a \leftarrow H_0^{(i-1)}$$
$$b \leftarrow H_1^{(i-1)}$$
$$c \leftarrow H_2^{(i-1)}$$
$$d \leftarrow H_3^{(i-1)}$$
$$e \leftarrow H_4^{(i-1)}$$

3. Untuk t = 0, 1, 2, ..., 79:

$$T \leftarrow (a <<< 5) + f_t(b,c,d) + e + K_t + W_t (mod\ 2^{23})$$
$$e \leftarrow d$$
$$d \leftarrow c$$
$$c \leftarrow b <<< 30$$
$$b \leftarrow a$$
$$a \leftarrow T$$

4. Lakukan kalkulasi *hash value* tahap *i*:

$$H_0^i \leftarrow a + H_0^i$$
$$H_1^i \leftarrow b + H_1^i$$
$$H_2^i \leftarrow c + H_2^i$$
$$H_3^i \leftarrow d + H_3^i$$
$$H_4^i \leftarrow e + H_4^i$$

Setelah *hashing* dilakukan pada semua blok ($M^{(1)}$, $M^{(2)}$,....., $M^{(n)}$), kita dapatkan *digest* sebagai berikut

$$\left(H_0^{(n)}, H_1^{(n)}, H_2^{(n)}, H_3^{(n)}, H_4^{(n)}\right).$$

Beberapa ahli kriptografi telah mencoba mencari kelemahan SHA-1. Xiaoyun Wang dan koleganya berhasil memperkecil ruang pencarian untuk *collision* SHA-1 dari 2^{80} operasi SHA-1 (yang merupakan "kekuatan" teoritis SHA-1 jika SHA-1 tidak memiliki kelemahan, berdasarkan birthday attack) menjadi 2^{69} operasi. Walaupun demikian, penggunaan SHA-1 masih dianggap cukup aman, dan jika ingin lebih aman lagi, maka SHA-256, SHA-384 atau SHA-512 dapat digunakan.

SHA-256, SHA-384 dan SHA-512, bersama dengan SHA-224 secara kolektif masuk dalam standard SHA-2. Saat bab ini ditulis, NIST sedang mengadakan sayembara pembuatan standard SHA-3 yang diharapkan akan lebih tangguh dari SHA-2.

2.2.12 SSL/TLS [13]

SSL adalah singkatan dari *Secure Socket Layer*, suatu *defacto standard* yang dibuat oleh Netscape, perusahaan pembuat *web browser* terpopuler tahun 1995. Ada dua versi SSL yang digunakan secara umum, yaitu SSL2 dan SSL3. Tahun 1999, standard SSL diambil alih oleh Internet Engineering Task Force (IETF) dan namanya diubah menjadi TLS, yang merupakan singkatan dari *Transport Layer Security* (SSL3 menjadi TLS versi 1). Perubahan nama ini mungkin agar nama menjadi lebih netral karena *socket* adalah istilah Unix. Versi terbaru dari TLS adalah versi 1.2. Karena nama SSL sudah sangat melekat, meskipun nama sudah berganti, dalam skripsi ini menyebutnya sebagai SSL/TLS.

SSL/TLS dimaksudkan untuk mengamankan sesi komunikasi antar proses. Dalam suatu *operating system* yang *multi-tasking* seperti Unix, Microsoft Windows, Linux atau MacOS, berbagai proses berjalan secara bersamaan. (Untuk melihat berbagai proses yang aktif, jika menggunakan Microsoft Windows, kita dapat menggunakan Windows Task Manager. Jika menggunakan Unix, Linux atau MacOS 10, kita dapat menggunakan *command line program* ***ps*** untuk melihat berbagai proses yang aktif.) Pengamanan komunikasi tidak terbatas pada pengamanan komunikasi antar proses dalam satu komputer, akan tetapi juga pengamanan komunikasi antar proses di dua komputer yang berbeda. Bahkan pengamanan komunikasi antar proses di dua komputer yang berbeda jauh lebih penting karena terdapat lebih banyak ancaman. Sebagai contoh, penggunaan terbesar dari SSL/TLS adalah untuk "sesi aman" antara proses *web browser* dengan proses *web server* yang biasanya berada pada dua komputer yang berbeda.

2.2.13 Standar SSL/TLS [13]

Dalam sesi menggunakan SSL/TLS, proses yang disebut *client* berkomunikasi dengan proses yang disebut *server*. Secara garis besar, sesi antara *client* dan *server* diamankan dengan, pertama melakukan *handshake*, lalu mengenkripsi komunikasi antara *client* dan *server* selama sesi berlangsung. Tujuan dari *handshake* adalah:

1. *Server authentication (optional)*.
2. Menentukan parameter enkripsi.
3. *Client authentication (optional)*.

Bagian *handshake* mungkin merupakan yang terpenting dalam sesi SSL/TLS. Yang jelas *handshake* merupakan bagian paling rumit dari segi protokol. Secara garis besar, protokol *handshake* adalah sebagai berikut:

1. *Client* mengirim ke *server* nomor versi SSL/TLS yang digunakan *client*, parameter enkripsi, data yang dibuat secara acak, dan informasi lain yang dibutuhkan oleh *server* untuk berkomunikasi dengan *client*. Jika dibutuhkan, *client* juga meminta *server certificate*.
2. *Server* mengirim ke *client* nomor versi SSL/TLS yang digunakan *server*, parameter enkripsi, data yang dibuat secara acak, dan informasi lain yang dibutuhkan oleh *client* untuk berkomunikasi dengan *server*. Jika diminta

dalam langkah 1, *server* juga mengirim *server certificate*. Jika dibutuhkan, *server* juga meminta *client* untuk mengirim *client certificate*.

3. Jika *client* meminta *server certificate* dalam langkah 1, *client* melakukan *server authentication* (akan dijelaskan secara lebih rinci) menggunakan *server certificate* dan informasi lain yang didapat. Jika *authentication* sukses, *server certificate* tidak diminta, atau pengguna mengizinkan, *client* meneruskan ke langkah 4. Jika tidak, sesi dihentikan.
4. Menggunakan data yang telah didapat, *client* membuat suatu premaster secret untuk sesi. Tergantung jenis enkripsi yang digunakan, ini dapat dilakukan dengan partisipasi *server*. Premaster secret dienkripsi menggunakan kunci publik *server* (diambil dari *server certificate*), lalu dikirim ke *server*.
5. Jika *server* meminta *client certificate* pada langkah 2, *client* menandatangan secara digital data yang unik untuk sesi yang diketahui oleh *client* dan *server*. Data berikut *digital signature* dan *client certificate* dikirim oleh *client* ke *server*.
6. Jika *server* meminta *client certificate* pada langkah 2, *server* melakukan *client authentication*. Jika *authentication* diminta dan gagal, maka sesi dihentikan.
7. *Client* dan *server* membuat master secret menggunakan premaster secret. Master secret digunakan oleh *client* dan *server* untuk membuat kunci sesi yang merupakan kunci enkripsi simetris. Enkripsi simetris adalah teknik kriptografi yang dalam proses enkripsi maupun dekripsi menggunakan kunci yang sama.
8. *Client* memberi tahu *server* bahwa kunci sesi akan digunakan untuk mengenkripsi komunikasi lebih lanjut. *Client* kemudian mengirim pesan yang dienkripsi ke *server* yang mengatakan bahwa ia selesai dengan *handshake*.
9. *Server* memberi tahu *client* bahwa kunci sesi akan digunakan untuk mengenkripsi komunikasi lebih lanjut. *Server* kemudian mengirim pesan

yang dienkripsi ke *client* yang mengatakan bahwa ia selesai dengan *handshake*.

10. *Handshake* selesai.

    Server authentication dilakukan dengan memeriksa *server certificate*. Dalam *server certificate* terdapat informasi antara lain:

    a. kunci publik *server*,
    b. masa berlaku *certificate*,
    c. domain name untuk *server*, dan
    d. domain name untuk pembuat *certificate* (biasanya *certificate* dibuat oleh suatu *certificate authority*).

    Server certificate ditanda-tangan secara digital oleh pembuatnya. Pemeriksaan *certificate* dilakukan dengan menjawab berbagai pertanyaan sebagai berikut:

    a. Apakah tanggal pemeriksaan berada dalam masa berlaku *certificate*?
    b. Apakah pembuat *certificate* (teridentifikasi dengan domain name pembuat) dapat kita percayai? Pembuat bisa merupakan suatu *certificate authority* atau sumber lain.
    c. Apakah *certificate* ditanda-tangan secara digital dengan benar oleh pembuatnya?
    d. Apakah domain name untuk *server* yang tertera dalam *certificate* sesuai dengan domain name untuk *server* yang sebenarnya?

Jika semua pertanyaan terjawab secara memuaskan, maka *authentication* sukses. Jika ada pertanyaan yang jawabannya tidak memuaskan, maka *authentication* gagal. *Client authentication* juga dilakukan serupa. Setelah *handshake* selesai, komunikasi antara *client* dan *server* dienkripsi menggunakan kunci sesi. Standard SSL/TLS tentunya juga menentukan format yang harus digunakan dalam komunikasi antara *client* dan *server*. Standard juga menentukan *cipher suite* yang dapat digunakan. *Cipher* suite adalah kombinasi metode enkripsi yang terdiri dari:

1. Metode enkripsi untuk *key* exchange atau *key agreement*.

2. Metode enkripsi untuk *certificate*.
3. Metode enkripsi menggunakan kunci sesi.
4. Metode enkripsi untuk message *authentication* code (MAC).

Untuk *key* exchange atau *key agreement*, metode enkripsi yang dapat digunakan antara lain RSA, dan Diffie-Hellman (DH). Untuk *certificate*, metode enkripsi yang dapat digunakan antara lain RSA, dan DSA/DSS. Untuk enkripsi menggunakan kunci sesi, metode enkripsi yang dapat digunakan antara lain RC4, 3DES, AES 128 bit, dan AES 256 bit.

Jika menggunakan *block cipher* seperti 3DES atau AES, maka metode enkripsi harus menggunakan mode *cipher* block chaining (CBC). Untuk message *authentication* code, metode enkripsi yang dapat digunakan antara lain:

1. HMAC-MD5,
2. HMAC-SHA-1,
3. HMAC-SHA256.

Standard TLS versi 1.2 mengharuskan implementasi sedikitnya bisa mendukung *cipher suite* yang terdiri dari RSA untuk *key* exchange dan *certificate*, AES 128 bit mode CBC untuk enkripsi menggunakan kunci sesi, dan HMACSHA-1 untuk *message authentication code*.

2.2.14 Pengunaan SSL/TLS [13]

Seperti dikatakan sebelumnya, penggunaan terbesar SSL/TLS adalah untuk *secure web browsing*. Semua *web browser* yang populer mendukung *secure web browsing* menggunakan SSL/TLS. Biasanya pengguna *web browser* tidak perlu mengetahui SSL/TLS. Saat *web browser* akan menampilkan web page dengan prefix https (jadi bukan http), maka *web browser* secara otomatis akan memulai sesi SSL/TLS. Dalam melakukan *handshake* SSL/TLS, *web browser* akan melakukan *server authentication* dengan memeriksa *certificate* untuk web *server*. *Web browser* biasanya sudah memiliki daftar *certificate authority* yang dapat dipercaya, dan ada *web browser* yang memperbolehkan pengguna untuk menambah pembuat *certificate* yang dipercaya ke dalam daftar. Jika ada masalah dalam *authentication* maka *web browser* biasanya memberi tahu pengguna dan menanyakan pengguna apakah sesi diteruskan atau tidak. Biasanya ada juga opsi untuk menambah

certificate yang bermasalah ke daftar pengecualian dimana *certificate* yang ada dalam daftar tidak perlu diperiksa. Jika *handshake* SSL/TLS berhasil maka *web page* dapat ditampilkan setelah terlebih dahulu di unduh dengan proteksi sesi SSL/TLS.

Selain *web browser*, perangkat lunak untuk web *server* seperti Apache juga sudah mendukung SSL/TLS. Jadi kita dapat membuat web *server* yang dapat melayani pengguna internet dengan *secure web page*. *Apache* memberi opsi untuk menggunakan *server certificate* yang dibuat sendiri atau dibuat oleh suatu *certificate authority*. Tentunya ada kemungkinan *web browser* pengguna akan berpendapat bahwa *certificate* bermasalah, akan tetapi ini dapat diatasi seperti dibahas di paragraf sebelum ini.

Karena *web browser* dan web *server* mendukung SSL/TLS, maka semua sistem informasi *client-server* yang berbasis web dapat menggunakan fasilitas SSL/TLS dengan mudah. Penggunaan sistem informasi akan sangat fleksibel karena pengguna dapat berada dimana saja asalkan ada koneksi TCP/IP dengan *server*, contohnya:

1. di komputer yang sama dengan *server*,
2. di komputer lain yang terhubung dengan *server* melalui *local area network*, atau
3. di lokasi lain, bahkan di negara yang berbeda waktu 12 jam dengan *server*, asalkan ada koneksi internet ke *server*.

Jika sistem informasi dibuat menggunakan sesuatu yang *platform independent* seperti Apache-MySQL-PHP, maka sistem informasi akan lebih fleksibel lagi karena *server* dapat ditempatkan di komputer dengan operating system apa saja, baik Microsoft Windows, Linux, MacOS 10, FreeBSD, OpenBSD, atau varian Unix lainnya, dan dapat dengan mudah dipindahkan. Tentunya penggunaan SSL/TLS tidak harus melalui web. Pada prinsipnya apa saja yang memerlukan pengamanan komunikasi antar proses dapat menggunakan SSL/TLS.

Ada beberapa alat yang dapat membantu pembuatan sistem yang menggunakan SSL/TLS. Contohnya adalah *cryptographic* library seperti RSA BSafe atau Cryptlib. Contoh lain adalah OpenSSL yang juga memberikan fasilitas

cryptographic library selain memberikan fasilitas SSL/TLS. Cryptlib dan OpenSSH keduanya menggunakan OpenSSL. OpenSSL adalah implementasi open source dari SSL/TLS yang didasarkan pada SSLeay, implementasi SSL oleh Eric Young. Meskipun *open source*, OpenSSL menggunakan licensing yang tidak kompatibel dengan GPL (GNU Public License). Untuk pembaca yang menginginkan implementasi SSL yang lebih kompatibel dengan GPL, GnuTLS adalah implementasi serupa yang menggunakan LGPL (Lesser GNU Public License).

2.2.15 X.509 [14]

X.509 adalah bagian dari X.500, suatu percobaan yang sangat ambisius dari ITU (International Telecommunication Union) untuk standardisasi *directory services*. Judul dari standard X.509 adalah *The Directory*: *Public-key and attribute certificate frameworks*. X.500 sendiri dianggap terlalu rumit, namun format *certificate* X.509 dijadikan dasar untuk standard defacto oleh industri, dan kini standard *public key infrastructure* untuk internet yaitu PKIX (*Public Key Infrastructure* X.509) berdasarkan pada format X.509.

Berbeda dengan “cara” PGP yang cenderung demokratis, “cara” X.509 lebih bersifat struktural hirarkis. Ini adalah warisan dari X.500 yang semula ditujukan untuk membuat direktori pengguna jaringan komputer seperti direktori telpon yang akan dikelola oleh berbagai perusahaan telpon di berbagai negara. Sifat-sifat struktural dalam X.509 antara lain:

a. Konsep *relative distinguished name* (RDN) yang cukup rumit dalam membuat *distinguished name* (DN) (identitas).
b. Konsep *certificate authority* yang semula dikaitkan dengan RDN

Konsep *relative distinguished* name (RDN) bertujuan untuk membuat identitas berdasarkan pada posisi dalam suatu hirarki. Hirarki ini diatur berdasarkan negara, propinsi, organisasi, sub-unit organisasi dan seterusnya. Berbagai negara dan organisasi melakukan lokalisasi standar yang kerap disebut *profile* yang menambah rumit konsep RDN. Semula, *certificate authority* (CA) juga dikaitkan dengan RDN, dimana satu CA mempunyai otoritas untuk satu subhirarki. Namun dalam prakteknya ini tidak berjalan.

Karena konsep RDN terlalu rumit dan dapat bertentangan dengan berbagai kepentingan seperti:

a. *privacy* (contohnya informasi pribadi), dan
b. *confidentiality* (contohnya banyak perusahaan yang tidak menginginkan informasi mengenai struktur organisasinya dipublikasikan),

belum lagi pada kenyataannya dunia tidak sepenuhnya tersusun rapi secara hirarkis tanpa tumpang tindih, akibatnya boleh dikatakan tidak ada yang mengetahui bagaimana cara membuat DN yang "benar." Meskipun demikian, DN tetap merupakan bagian tak terpisahkan dari *certificate* X.509, dan X.509 sudah merupakan defacto standard. Berikut komponen DN yang kerap digunakan:

Tabel 2.16 Komponen DN

Komponen	Penjelasan
CountryName(C)	Kode negara 2 huruf berdasarkan ISO 3166 (contohnya ID).
Organization(O)	Nama organisasi (contohnya PT Sendiri).
OrganizationalUnit(OU)	Nama unit organisasi (contohnya Sales).
StateOrProvince(SP)	Nama negara bagian atau propinsi (contohnya DKI).
Locality(L)	Nama daerah (contohnya Kebayoran Baru)
CommonName(CN)	Nama pemegang *certificate*, bisa perorangan atau bagian dari organisasi, bahkan nama komputer (contohnya Budi Santoso).

Setiap komponen kecuali CountryName besarnya maksimum 64 karakter, sedangkan CountryName besarnya 2 karakter. Masih banyak komponen DN lainnya yang dapat digunakan, namun komponen yang terpenting sudah masuk dalam daftar diatas. Minimal suatu DN harus mempunyai komponen C dan komponen CN. DN dapat mengidentifikasi orang, unit organisasi atau perangkat, dan digunakan baik untuk identifikasi pemegang *certificate* maupun untuk identifikasi pembuat *certificate*.

Sejak penggunaan internet menjadi dominan, ada alternatif dari DN yang disebut *general name* (GN) yang lebih berorientasi pada internet dibandingkan dengan DN. Komponen GN termasuk antara lain DNS name, IP *address, email address* dan *uniform resource identifier*. Untuk identifikasi perangkat, penggunaan GN mungkin lebih cocok dibandingkan DN.

Sebetulnya ada bermacam jenis struktur untuk *certificate* X.509, termasuk

1. *certificate* biasa,
2. *attribute certificate*,
3. *certification request*
4. *certificate revocation list* (CRL).

Berikut adalah daftar komponen *certificate biasa,* termasuk :

Tabel 2.17 Komponen *certificate* biasa

Komponen	Penjelasan
Version	Versi X.509 yang digunakan.
SerialNumber	Nomor seri. Setiap *certificate* yang dibuat oleh suatu CA harus mempunyai nomor seri yang berbeda.
SignatureAlgorithm	Algoritma kriptografi yang digunakan untuk menanda-tangan *certificate*.
IssuerName	DN/GN untuk pembuat *certificate* (biasanya CA).
Validity	Masa berlaku *certificate*.
SubjectName	DN/GN untuk pemegang *certificate*.
Komponen	**Penjelasan**
SubjectPublicKeyInfo	Kunci publik pemegang *certificate*.
Extensions	Untuk berbagai macam informasi tambahan.

Attribute certificate fungsinya adalah sebagai *certificate* untuk berbagai atribut, jadi komponen SubjectPublicKeyInfo diganti oleh Attributes. *Certificate request* fungsinya adalah permintaan *certificate* untuk suatu kunci publik, jadi hanya berisi komponen berikut:

Tabel 2.18 Komponen *attribute certificate*

Komponen	Penjelasan
Version	Versi X.509 yang digunakan.
SerialNumber	DN/GN untuk pemegang kunci.
SignatureAlgorithm	Kunci publik yang diminta dibuatkan *certificate*.
IssuerName	Untuk berbagai macam informasi tambahan.

Certificate revocation list (CRL) dimaksudkan sebagai daftar *certificate* yang dibatalkan oleh suatu CA. Untuk berbagai macam alasan, termasuk kunci yang dicuri, suatu *certificate* dapat dibatalkan oleh CA yang membuatnya. Secara berkala, suatu CA mempublikasikan daftar *certificate* yang dibatalkannya dalam

bentuk CRL. CRL dapat digunakan untuk menentukan validitas suatu *certificate*. Komponen dari CRL adalah sebagai berikut:

Tabel 2.19 Komponen CRL

Komponen	Penjelasan
Version	Versi X.509 yang digunakan.
SignatureAlgorithm	Algoritma kriptografi yang digunakan untuk menanda-tangan *certificate*.
IssuerName	DN/GN untuk CA.
ThisUpdate	Waktu CRL ini dibuat.
NextUpdate	Waktu CRL berikutnya akan dibuat.
User*Certificate*	Daftar *certificate* yang dibatalkan.
*Revocation*Date	Tanggal *certificate* dibatalkan.

Meskipun dalam X.500 *certificate authority* (CA) dikaitkan dengan RDN, kini tidak ada lagi kaitan formal antara CA dengan RDN. *Certificate* bahkan tidak harus dibuat oleh CA. Namun konsep CA masih berguna, meskipun siapa yang patut menjadi CA tergantung pada aplikasi dan/atau situasi. Validitas dari suatu kunci publik ditentukan oleh *certificate* untuk kunci tersebut: jika *certificate* dibuat oleh orang, badan atau CA yang dipercaya oleh seorang pengguna, maka kunci publik dianggap valid oleh pengguna tersebut. Perbedaan utama antara "cara" X.509 dan "cara" PGP dalam menentukan validitas *certificate* adalah dengan X.509, CA biasanya tidak ditentukan oleh pengguna, sedangkan dengan PGP pengguna berperan lebih besar dalam menentukan siapa yang dipercaya sebagai pembuat *certificate* menggunakan mekanisme *web of trust*.

Dari segi teknis, masalah yang dihadapi oleh X.509 dalam manajemen validitas *certificate* jauh lebih rumit dibandingkan masalah yang dihadapi PGP. Ini antara lain karena:

a. Berbagai CA dan aktor lainnya berperan aktif dalam manajemen validitas *certificate*.
b. Bermacam skenario penggunaan *certificate* X.509 dapat melibatkan aktor yang beragam dan dalam skala yang besar.
c. Konsep *certificate revocation* sukar untuk dipraktekkan secara efektif.

Sebagai contoh banyaknya aktor yang berperan dalam manajemen validitas *certificate*, kita bisa lihat begitu banyaknya *certificate authority* yang ada, baik yang bersifat komersial seperti Verisign, maupun yang non-komersial. Contoh

penggunaan *certificate* X.509 yang beragam termasuk untuk keperluan *website authentication*, untuk *authentication* kunci publik suatu perangkat dalam jaringan yang menggunakan IPsec untuk VPN, dan untuk secure email menggunakan S/MIME. Sukarnya menggunakan konsep *certificate revocation* secara efektif pernah dijelaskan oleh Peter Gutmann (arsitek dari Cryptlib) dengan membuat analogi dengan basis data relasional dimana mempublikasikan *certificate* ibarat operasi *prepare* dalam suatu basis data relasional, sedangkan *certificate revocation* ibarat operasi *commit* dalam basis data relasional:

a. Dalam basis data relasional, status setelah *prepare* tetapi sebelum *commit* tidak bisa dijamin konsisten.
b. Analoginya untuk *certificate*, jika *certificate* telah dipublikasi tetapi *revocation* untuk *certificate* tersebut belum terlihat, maka status dari *certificate* tidak pasti, karena bisa saja *certificate* telah dibatalkan.

Akibat dari berbagai masalah tersebut, tidak ada satu standard protokol untuk manajemen *certificate*, tetapi ada beberapa protokol yang fungsinya beragam.

Salah satu standard protokol untuk manajemen *certificate* adalah standard *Lightweight Directory Access Protocol* (LDAP). Protokol ini digunakan untuk mengakses *certificate* store (tempat penyimpanan *certificate*) yang menggunakan direktori LDAP. Suatu LDAP server melayani client dengan operasi penyimpanan *certificate* di direktori, penghapusan *certificate* dari direktori, dan pencarian *certificate* dalam direktori. Karena LDAP berorientasi pada X.500 yang terkenal rumit, implementasi LDAP tidak bisa efisien sehingga tidak bisa digunakan dalam skala besar. Untuk skala besar, relational database banyak digunakan untuk *certificate* store.

Selain LDAP, berbagai protokol berjenis request/response yang digunakan untuk manajemen *certificate* antara lain:

a. CMP (*certificate management protocol*).
b. OCSP (*online certificate status protocol*).
c. TSP (*time-stamp protocol*).

CMP melayani *certificate* requests yaitu pembuatan *certificate* dan pembatalan (*revocation*) *certificate*. Komunikasi dengan CMP server dapat melalui

http, email, atau cara lain. Karena CMP cukup rumit, sukar untuk menggunakannya dalam skala besar. Akibatnya, beberapa pembuat perangkat jaringan seperti Cisco menggunakan protokol yang lebih sederhana untuk digunakan dalam skala besar yaitu SCEP (*simple certificate enrollment protocol*).

OCSP adalah protokol untuk melayani pemeriksaan *certificate* yang berdasarkan pada *concept revocation*. Jadi OCSP *server* hanya dapat menjawab bahwa *certificate* telah dibatalkan (*revoked*) atau tidak, dan informasi yang digunakan biasanya tidak up-to-date. Seperti telah dibahas diatas, Peter Gutmann menganggap solusi ini tidak memuaskan. Peter Gutmann menyarankan penggunaan RTCS (*realtime certificate* status protocol) sebagai extension dari OCSP yang bisa menjawab apakah *certificate* valid atau tidak secara real-time. Buku ini merekomendasikan penggunaan RTCS seperti yang disarankan oleh Peter Gutmann.

TSP adalah protokol yang digunakan untuk melayani *time-stamping requests*. Suatu *time-stamping authority* (TSA) dapat diimplementasi menggunakan TSP server yang melayani *time-stamping requests* dengan membuat *time-stamps* yang ditanda-tangan menggunakan kunci TSA.

Kecuali untuk TSP, *server* untuk berbagai protokol *request/response* perlu menggunakan suatu *certificate store*. Tren saat ini adalah menggunakan basis data relasional sebagai *back end* untuk *certificate store* agar dapat digunakan dalam skala besar. PKI untuk komunitas terbuka (dimana tidak ada ikatan formal untuk anggota) biasanya tidak dapat berfungsi efektif. Ini antara lain karena:
Keperluan dan karakteristik anggota komunitas terbuka beragam.

a. Sukar untuk melakukan kontrol terhadap setiap anggota komunitas terbuka dan membuatnya bertanggung jawab atas penggunaan PKI.
b. Sukar untuk mendapatkan CA yang dapat dipercaya oleh semua pihak dalam komunitas terbuka.

PKI biasanya hanya dapat berfungsi efektif jika digunakan oleh komunitas tertutup dimana ada ikatan formal untuk anggota, misalnya

a. PKI untuk suatu perusahaan.
b. PKI untuk suatu pemerintahan atau badan pemerintahan.

c. PKI untuk komunitas jaringan bisnis tertentu misalnya jaringan bisnis suku cadang otomotif, dimana setiap anggota terdaftar secara formal dan mempunyai tanggung jawab yang jelas.
d. PKI untuk VPN.

2.2.16 Base 64 Encoding [15]

Base 64 encoding dirancang untuk mewakili urutan sewenang-wenang oktet dalam bentuk yang memungkinkan penggunaan huruf besar dan huruf kecil tetapi tidak perlu dapat dibaca manusia. Base 64 menggunakan 65 karakter *subset* dari US-ASCII yang memungkinkan setiap 6 bit data diwakili per karakter yang dapat dicetak (karakter tambahan ke 65 "=" digunakan untuk penanda pemrosesan khusus). Proses encoding mewakilkan kelompok 24-bit dari bit masukan menjadi 4 karakter terkodekan, proses pembacaan bit dimulai dari kiri ke kanan, kelompok 24-bit masukan dibentuk dengan menggabungkan tiga 8-bit kelompok masukan. Kemudian 24-bit masukan ini diubah menjadi kelompok 6-bit dimana setiap kelompok di di translasi menjadi satu karakter base 64. Setiap kelompok 6-bit digunakan sebagai *index* dari *array* karakter yang dapat dicetak, karakter yang direferensikan oleh index adalah output dari pengkodean. Tabel karakter yang digunakan oleh base 64 dapat dijelaskan pada Tabel 2.20.

Tabel 2.20 Karakter Base 64

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v	(pad)	=
14	O	31	f	48	w		
15	P	32	g	49	x		
16	Q	33	h	50	y		

Pemrosesan khusus dilakukan jika bit masukan kurang dari 24-bit diakhir data yang akan dikodekan, ketika bit masukan kurang dari 24-bit pada sebuah kelompok bit masukan, bit dengan nilai 0 ditambahkan pada (pada sisi kanan) sampai terbentuk kelompok 6-bit. Padding pada akhir data dilakukan dengan menggunakan karakter ‘”=”. Dikarenakan semua input base 64 adalah oktet dengan jumlah yang integral maka hanya kasus – kasus berikut yang dapat terjadi:

1. Kuantum akhir pengkodean masukan adalah kelipatan 24-bit disini output terbagai menjadi kelipatan 4 karakter sehingga tidak dibutuhkan padding “=”.
2. Kuantum akhir pengkodean masukan adalah tepat 8-bit, hasil akhir pengkodean akan memiliki dua karakter dengan dua karakter padding “=”.
3. Kuantum akhir pengkodean masukan adalah tepat 16-bit, hasil akhir pengkodean akan memiliki tiga karakter dengan satu karakter padding “=”

Contoh dari proses pengkodean base 64 dijelaskan pada Gambar 2.21.

```
Input data:  0x14fb9c03d97e
Hex:     1   4    f   b    9   c     | 0   3    d   9    7   e
8-bit:   00010100 11111011 10011100  | 00000011 11011001 0111111
6-bit:   000101 001111 101110 011100 | 000000 111101 100101 1111
Decimal: 5      15     46     28       0      61     37     62
Output:  F      P      u      c        A      9      l      +

Input data:  0x14fb9c03d9
Hex:     1   4    f   b    9   c     | 0   3    d   9
8-bit:   00010100 11111011 10011100  | 00000011 11011001
                                              pad with 00
6-bit:   000101 001111 101110 011100 | 000000 111101 100100
Decimal: 5      15     46     28       0      61     36
                                                  pad with =
Output:  F      P      u      c        A      9      k      =

Input data:  0x14fb9c03
Hex:     1   4    f   b    9   c     | 0   3
8-bit:   00010100 11111011 10011100  | 00000011
                                      pad with 0000
6-bit:   000101 001111 101110 011100 | 000000 110000
Decimal: 5      15     46     28       0      48
                                           pad with =      =
Output:  F      P      u      c        A      w      =      =
```

Gambar 2.21 Contoh proses pengkodean base 64

2.2.17 UTF-8

UTF-8 adalah pengkodean karakter yang mampu mengkodekan karakter atau *code point* yang ada dalam unicode, UTF-8 adalah pengkodean *variabel length* dan menggunakan 8 bit-code unit. UTF-8 didesain agar *backward compatible* dengan ASCII [16] . Berikut sebagian karakter dari UTF-8 dari U+0020 - U+007E berisikan angka, simbol , huruf latin kecil dan huruf latin besar dijelaskan pada Tabel 2.21.

Tabel 2.21 Tabel karakter utf-8 U+0020 - U+007E [17]

Unicode Code Point	Character	UTF-8 (HEX)	Name
U+0020		0x20	SPACE
U+0021	!	0x21	EXCLAMATION MARK
U+0022	"	0x22	QUOTATION MARK
U+0023	#	0x23	NUMBER SIGN
U+0024	$	0x24	DOLLAR SIGN
U+0025	%	0x25	PERCENT SIGN
U+0026	&	0x26	AMPERSAND
U+0027	'	0x27	APOSTROPHE
U+0028	(	0x28	LEFT PARENTHESIS
U+0029	)	0x29	RIGHT PARENTHESIS
U+002A	*	0x2a	ASTERISK
U+002B	+	0x2b	PLUS SIGN
U+002C	,	0x2c	COMMA
U+002D	-	0x2d	HYPHEN-MINUS
U+002E	.	0x2e	FULL STOP
U+002F	/	0x2f	SOLIDUS
U+0030	0	0x30	DIGIT ZERO
U+0031	1	0x31	DIGIT ONE
U+0032	2	0x32	DIGIT TWO
U+0033	3	0x33	DIGIT THREE
U+0034	4	0x34	DIGIT FOUR
U+0035	5	0x35	DIGIT FIVE
U+0036	6	0x36	DIGIT SIX
U+0037	7	0x37	DIGIT SEVEN
U+0038	8	0x38	DIGIT EIGHT
U+0039	9	0x39	DIGIT NINE
U+003A	:	0x3a	COLON
U+003B	;	0x3b	SEMICOLON
U+003C	<	0x3c	LESS-THAN SIGN
U+003D	=	0x3d	EQUALS SIGN
U+003E	>	0x3e	GREATER-THAN SIGN
U+003F	?	0x3f	QUESTION MARK
U+0040	@	0x40	COMMERCIAL AT
U+0041	A	0x41	LATIN CAPITAL LETTER A
U+0042	B	0x42	LATIN CAPITAL LETTER B

U+0043	C	0x43	LATIN CAPITAL LETTER C
U+0044	D	0x44	LATIN CAPITAL LETTER D
U+0045	E	0x45	LATIN CAPITAL LETTER E
U+0046	F	0x46	LATIN CAPITAL LETTER F
U+0047	G	0x47	LATIN CAPITAL LETTER G
U+0048	H	0x48	LATIN CAPITAL LETTER H
U+0049	I	0x49	LATIN CAPITAL LETTER I
U+004A	J	0x4a	LATIN CAPITAL LETTER J
U+004B	K	0x4b	LATIN CAPITAL LETTER K
U+004C	L	0x4c	LATIN CAPITAL LETTER L
U+004D	M	0x4d	LATIN CAPITAL LETTER M
U+004E	N	0x4e	LATIN CAPITAL LETTER N
U+004F	O	0x4f	LATIN CAPITAL LETTER O
U+0050	P	0x50	LATIN CAPITAL LETTER P
U+0051	Q	0x51	LATIN CAPITAL LETTER Q
U+0052	R	0x52	LATIN CAPITAL LETTER R
U+0053	S	0x53	LATIN CAPITAL LETTER S
U+0054	T	0x54	LATIN CAPITAL LETTER T
U+0055	U	0x55	LATIN CAPITAL LETTER U
U+0056	V	0x56	LATIN CAPITAL LETTER V
U+0057	W	0x57	LATIN CAPITAL LETTER W
U+0058	X	0x58	LATIN CAPITAL LETTER X
U+0059	Y	0x59	LATIN CAPITAL LETTER Y
U+005A	Z	0x5a	LATIN CAPITAL LETTER Z
U+005B	[	0x5b	LEFT SQUARE BRACKET
U+005C	\	0x5c	REVERSE SOLIDUS
U+005D	]	0x5d	RIGHT SQUARE BRACKET
U+005E	^	0x5e	CIRCUMFLEX ACCENT
U+005F	_	0x5f	LOW LINE
U+0060	`	0x60	GRAVE ACCENT
U+0061	a	0x61	LATIN SMALL LETTER A
U+0062	b	0x62	LATIN SMALL LETTER B
U+0063	c	0x63	LATIN SMALL LETTER C
U+0064	d	0x64	LATIN SMALL LETTER D
U+0065	e	0x65	LATIN SMALL LETTER E
U+0066	f	0x66	LATIN SMALL LETTER F
U+0067	g	0x67	LATIN SMALL LETTER G
U+0068	h	0x68	LATIN SMALL LETTER H
U+0069	i	0x69	LATIN SMALL LETTER I
U+006A	j	0x6a	LATIN SMALL LETTER J
U+006B	k	0x6b	LATIN SMALL LETTER K
U+006C	l	0x6c	LATIN SMALL LETTER L
U+006D	m	0x6d	LATIN SMALL LETTER M
U+006E	n	0x6e	LATIN SMALL LETTER N
U+006F	o	0x6f	LATIN SMALL LETTER O
U+0070	p	0x70	LATIN SMALL LETTER P

U+0071	q	0x71	LATIN SMALL LETTER Q
U+0072	r	0x72	LATIN SMALL LETTER R
U+0073	s	0x73	LATIN SMALL LETTER S
U+0074	t	0x74	LATIN SMALL LETTER T
U+0075	u	0x75	LATIN SMALL LETTER U
U+0076	v	0x76	LATIN SMALL LETTER V
U+0077	w	0x77	LATIN SMALL LETTER W
U+0078	x	0x78	LATIN SMALL LETTER X
U+0079	y	0x79	LATIN SMALL LETTER Y
U+007A	z	0x7a	LATIN SMALL LETTER Z
U+007B	{	0x7b	LEFT CURLY BRACKET
U+007C	|	0x7c	VERTICAL LINE
U+007D	}	0x7d	RIGHT CURLY BRACKET
U+007E	~	0x7e	TILDE

2.2.18 Basis Data [18]

Basis data adalah kumpulan data yang saling berelasi. Data sendiri merupakan fakta mengenai objek, orang dan lain-lain. Data dinyatakan dengan nilai (angka, deretan karakter atau , symbol).

Basis data dapat di definisikan dalam berbagai sudut pandang sebagai berikut:

1. Himpunan Kelompok data yang saling berhubungan yang di organisasi sedemikian rupa sehingga kelak dapat di manfaatkan dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa tanpa pengulangan (*redudancy*) yang tidak perlu, untuk memenuhi kebutuhan.
3. Kumpulan file/label/arsip yang saling berhubungan yang di simpan dalam media penyimpanan elektronik

2.2.19 Manfaat/Kelebihan Basis Data [19]

Banyak manfaat yang dapat kita perolah dengan menggunakan basis data. Manfaat/Kelebihan basis data diantaranya adalah:

1. Kecepatan dan kemudahan (*speed)*

   Dengan menggunakan basis data pengambilan informasi dapat di lakukan dengan cepat dan mudah. Basis data memiliki kemampuan dalam mengkelompokan, mengurutkan bahkan perhitungan dengan matematika.

Dengan perancangan yang benar, maka penyajian informasi akan dapat di lakukan dengan cepat dan mudah.

2. Kebersamaan pemakai (*sharebility*)

   Sebuah basis data dapat di gunakan oleh banyak *user* dan banyak aplikasi. Untuk data- data yang diperlukan oleh banyak bagian/orang, tidak perlu di lakukan pencatatan di masing-masing bagian, tetapi cukup dengan satu basis data untuk dipakai bersama.

3. Pemusatan kontrol data

   Karena cukup dengan satu basis data untuk banyak keperluan, pengontrolan terhadap data juga cukup di lakukan di satu tempat saja.

4. Efisiensi ruang penyimpanan (*space)*

   Dengan pemakaian bersama, kita tidak perlu menyediakan tempat penyimpanan di berbagai tempat, tetapi cukup satu saja sehingga ini akan menghemat ruang penyimpanan yang dimiliki oleh sebuah organisasi. Dengan teknik perancangan basis data yang benar, kita akan dapat menyederhanakan penyimpanan sehingga tidak semua data harus disimpan.

5. Keakuratan (*Accuracy)*

   Penerapan secara ketat aturan tipe data, domain data, keunikan data, hubungan antar data, dan lain-lain, dapat menekan ketidakakuratan dalam pemasukan/penyimpanan data.

6. Ketersediaan (*Availability)*

   Dengan basis data kita dapat mem-backup, memilah-milah data mana yang masih diperlukan dan data mana yang perlu kita simpan ke tempat lain. Hal ini mengingat pertumbuhan transaksi suatu organisasi dari waktu ke waktu membutuhkan media penyimpanan yang semakin besar.

7. Kelengkapan (*Completeness*)

8. Keamanan (Security)

   Kebanyakan DBMS dilengkapi dengan fasilitas manajemen pengguna. Pengguna di berikan hak akses yang berbeda-beda sesuai dengan kepentingan dan posisinya. Basis data di berikan *password* untuk membatasi orang yang mengaksesnya.

9. Kemudahan dalam pembuatan program aplikasi baru

   Penggunaan basis data merupakan bagian dari perkembangan teknologi. Dengan adanya basis data pembuatan aplikasi bisa memanfaatkan kemampuan dari DBMS, sehingga pembuat aplikasi tidak perlu mengurusi penyimpanan data, tetapi cukup mengatur *interface* untuk pengguna.

10. Pemakaian secara langsung

    Basis data memiliki fasilitas untuk melihat datanya secara langsung dengan *tool* yang disediakan oleh DBMS. Untuk melihat data, langsung ke tabel ataupun dengan menggunakan *query*. Biasanya yang menggunakan fasilitas ini adalah *user* yang sudah ahli, atau *database* administrator.

11. Kebebasan data (*Data independence)*

    Jika sebuah program telah selesai dibuat, dan ternyata ada perubahan isi/struktur data. Maka dengan basis data, perubahan ini hanya perlu di lakukan pada level DBMS tanpa harus membongkar kembali program aplikasinya.

12. *User view*

    Basis data menyediakan pandangan yang berbeda untuk tiap-tiap pengguna.

2.2.20 Object Oriented Programming [20]

Object-oriented programming (OOP) adalah sebuah pendekatan dalam pengembangan perangkat lunak dimana struktur sebuah *software* di dasarkan pada berbagai objek yang saling berinteraksi satu dengan lainnya untuk menyelesaikan sebuah tugas. Interaksi ini memiliki bentuk berupa pesan yang di kirim secara bolak balik antar objek. Menanggapi pesan sebuah objek dapat melakukan sebuah *action* atau *method*.

2.2.21 Software Framework [21]

Dalam pemrograman komputer, *software framework* adalah abstraksi di mana *software* menyediakan fungsionalitas umum yang dapat diubah dengan menambahkan kode yang di tulis oleh pengguna, sehingga menjadi *software* yang khusus. *Software Framework* bersifat universal, lingkungan software yang *reuseable* yang menyediakan fungsionalitas tertentu sebagai bagian dari *software platform* yang di gunakan untuk memfasilitasi pengembangan *software*, produk dan

solusi. *Framework* memiliki beberapa karakteristik utama yang membedakannya dari *library* perangkat lunak pada umumnya:

1. *Inversion of control*

   Tidak seperti pada *library* umumnya, alur program di tentukan oleh *framework* bukan oleh pengguna.
2. *Default behavior*

   Framework memiliki perilaku standar tersendiri.
3. *Extensibility*

   Framework dapat di *extend* oleh pengguna untuk menambahkan fungsionalitas khusus.
4. *Non-modifiable framework code*

   Pada umumnya kode dari *framework* tidak untuk di ubah oleh pengguna, dengan kata lain pengguna dapat meng *extend framework* namun bukan mengubah kode *framework*.

2.2.22 Model View Controller (MVC)

MVC adalah sebuah pola pemrograman yang bertujuan memisahkan logika bisnis, logika data dan logika tampilan (*interface*), atau secara sederhana memisahkan antara proses, data dan tampilan. MVC mengatur arsitektur sebuah aplikasi. Umumnya aplikasi yang dibangun dengan konsep MVC adalah aplikasi yang cukup besar, karena salah satu keuntungan dari MVC itu adalah kemudahan *maintenance*, dan pengembangan aplikasi tersebut [22]. Berikut penjelasan masing-masing komponen dari MVC [23] :

1. Model

   Merepresentasikan struktur data, biasanya *class* model berisikan fungsi yang di pergunakan untuk operasi pada basis data.
2. *View*

   Berisi antar muka dari aplikasi, berisi informasi yang di tampilkan ke pada penguna.
3. *Controller*

Berperan sebagai penengah antara model, *view* dan berbagai sumber daya yang di butuhkan untuk memproses permintaan HTTP dan membangkitkan halaman web.

2.2.23 YII [24]

Yii adalah *framework* (kerangka kerja) PHP berbasis-komponen, berkinerja tinggi untuk pengembangan aplikasi *Web* berskala-besar. Yii menyediakan *reusability* maksimum dalam pemrograman *Web* dan mampu meningkatkan kecepatan pengembangan secara signifikan. Nama Yii (dieja sebagai /i:/) singkatan dari "*Yes It Is!*".

1. Persyaratan

   Untuk menjalankan aplikasi *Web* berbasis Yii, Anda memerlukan *server Web* yang mendukung PHP 5.1.0. Bagi para pengembang yang ingin menggunakan Yii, mengerti pemrograman berorientasi-objek (OOP) akan sangat membantu, karena Yii merupakan *framework* OOP murni.

2. Yii Paling Cocok untuk Apa.

   Yii adalah *framework* pemrograman umum *Web* yang bisa dipakai untuk mengembangkan semua jenis aplikasi *Web*. Dikarenakan sangat ringan dan dilengkapi dengan mekanisme *caching* yang canggih, Yii sangat cocok untuk pengembangan aplikasi dengan lalu lintas-tinggi, seperti portal, forum, sistem manajemen konten (CMS), sistem *e-commerce*, dan lain-lain.

3. Bagaimana Yii Dibandingkan dengan *Framework* Lain.

   Seperti kebanyakan PHP *framework*, Yii adalah MVC *framework*. Yii melampaui *framework* PHP lain dalam hal efisiensi, kekayaan-fitur, dan kejelasan dokumentasi. Yii didesain dengan hati-hati dari awal agar sesuai untuk pengembangan aplikasi *Web* secara serius. Yii bukan berasal dari produk pada beberapa proyek maupun konglomerasi pekerjaan pihak-ketiga. Yii adalah hasil dari pengalaman kaya para pembuat pada pengembangan aplikasi *Web* dan investigasi *framework* pemrograman *Web* dan aplikasi yang paling populer.

2.2.24 Apache Cordova [25]

Apache Cordova adalah *framework* pengembangan perangkat lunak *mobile* yang berlisensi *open source*, cordova memungkinkan pengembangan perangkat lunak *mobile cross platform* dengan menggunakan teknologi standar web seperti HTML5, CSS3, dan *Javascript* tanpa menggunakan bahasa pemrograman *native* dari platform *mobile*. Aplikasi di jalankan dalam sebuah *wrapper* yang di targetkan berjalan pada setiap platform dan mengandalkan API yang taat standar untuk melakukan akses pada sensor perangkat, data dan status jaringan.

Aplikasi Apache Cordova mengandalkan *file config.xml* umum yang menyediakan informasi tentang aplikasi dan mencatat berbagai parameter bagaimana sebuah aplikasi bekerja seperti respon terhadap perubahan orientasi perangkat.

Aplikasi Cordova sendiri di implementasikan sebagai *web page* yang secara standar di namai index.html yang mereferensi berbagai CSS, Javascript, Gambar, *file* media serta berbagai macam sumber daya yang diperlukan oleh aplikasi. Aplikasi di jalankan sebagai *WebView* di dalam *wrapper* aplikasi *native*.

2.2.25 Open SSL [26]

OpenSSL merupakan produk *open source* dengan lisensi yang berbeda dan tidak kompatibel dengan GPL. OpenSSL dapat diunduh dari *website http:/www.openssl.org*. OpenSSL terdiri dari 3 komponen utama yaitu:

1. SSL/TLS *library*
2. *Crypto library*, dan
3. Openssl *command line tool*.

SSL/TLS *library* memberikan fasilitas untuk melakukan sesi SSL/TLS dalam bentuk *loadable module* (DLL untuk Microsoft Windows). Fungsi API untuk SSL/TLS *library* dibagi menjadi 5 golongan:

1. *Protocol methods* (*client, server*, atau, untuk menentukan versi SSL/TLS dan jenis layanan *client-server*).
2. *Ciphers*, untuk menentukan berbagai jenis enkripsi yang dapat digunakan
3. *Protocol contexts*, untuk menentukan *context* secara global selama program berjalan.

4. *Sessions*, untuk menentukan berbagai parameter per sesi.
5. *Connections*, untuk koneksi. Ini adalah fungsi utama SSL/TLS *library*,dan selama program berjalan fungsi dari golongan ini yang banyak digunakan.

Crypto library adalah implementasi dari berbagai algoritma *enkripsi* dan *hashing*. Selain digunakan oleh SSL/TLS *library* dan openssl *command line tool*, *crypto library* juga digunakan oleh OpenSSH, GnuPG dan implementasi standar kriptografi lainnya. *Command line tool* openssl adalah program yang berorientasi Unix yang dapat digunakan untuk:

1. Pembuatan dan manajemen kunci privat/publik.
2. Operasi kriptografi *public key* termasuk *key management*, *digital signing* dan *digital signature checking*.
3. Pembuatan dan manajemen *certificate* X.509.
4. Kalkulasi digest.
5. Enkripsi dan dekripsi.
6. Testing SSL/TLS *client* dan *server*.
7. S/MIME.
8. Membuat *time-stamp request*, membuat *time-stamp* dan melakukan verifikasi *timestamp*.

2.2.26 Apache HTTPD Server [27]

Server HTTP Apache atau *Server Web*/WWW *Apache* adalah *server web* yang dapat dijalankan di banyak sistem operasi (Unix,BSD, Linux, Microsoft Windows dan Novell Netware serta platform lainnya) yang berguna untuk melayani dan memfungsikan situs web. Protokol yang digunakan untuk melayani fasilitas *web*/*www* ini menggunakan HTTP.

Apache memiliki fitur-fitur canggih seperti pesan kesalahan yang dapat disesuaikan, autentikasi berbasis basis data dan lain-lain. Apache juga didukung oleh sejumlah antarmuka pengguna berbasis grafik (GUI) yang memungkinkan penanganan *server* menjadi mudah.

Apache merupakan perangkat lunak sumber terbuka dikembangkan oleh komunitas terbuka yang terdiri dari pengembang-pengembang di bawah naungan Apache Software Foundation.