RSNI3

Rancangan Standar Nasional Indonesia 3

RSNI3 ISO/IEC 10118-1:2016
(Ditetapkan oleh BSN tahun 2024)

Teknologi informasi — Teknik keamanan — Fungsi-*hash* — Bagian 1: Umum

Information technology — Security techniques — Hash-function — Part 1: General

(ISO/IEC 10118-1:2016 dan ISO/IEC 10118-1:2016/Amd.1:2021, IDT)

ICS 35.030

BSN BADAN STANDARDISASI NASIONAL

BSN
Email: dokinfo@bsn.go.id
www.bsn.go.id

Diterbitkan di Jakarta

Daftar isi

Prakata

SNI ISO/IEC 10118-1:2016, *Teknologi informasi, Teknik keamanan — Fungsi-hash — Bagian 1: Umum*, merupakan standar yang disusun dengan jalur adopsi tingkat keselarasan identik dari ISO/IEC 10118-1:2016, *Information technology — Security techniques — Hash-functions — Part* 1*: General*, dan ISO/IEC 10118-1:2016/Amd.1:2021 *Information technology — Security techniques — Hash-functions — Part* 1*: General Amendment 1: Padding methods for sponge functions,* dengan metode adopsi terjemahan dua bahasa dan ditetapkan oleh BSN Tahun 2024.

Standar ini disusun oleh Komite Teknis 35-04, Keamanan Informasi, Keamanan Siber, dan Perlindungan Privasi. Standar ini telah dibahas melalui rapat teknis dan disepakati dalam rapat konsensus pada tanggal 21 Juni 2024 di Depok, yang dihadiri oleh para pemangku kepentingan (*stakeholders*) terkait, yaitu perwakilan dari pemerintah, pelaku usaha, konsumen, dan pakar. Standar ini telah melalui tahap jajak pendapat pada tanggal 18 Juli 2024 sampai dengan 1 Agustus 2024 dengan hasil akhir disetujui menjadi SNI.

Kosakata yang digunakan dalam standar ini mengikuti bentuk baku yang dicantumkan dalam Kamus Besar Bahasa Indonesia (KBBI), tetapi ada beberapa kosakata yang belum ada di dalam KBBI.

Kata/istilah "*hash"*, "*imprint", "digest", "round",* dan "*preimage*" tidak diterjemahkan dalam Standar ini karena Komite Teknis 35-04 belum menemukan padanan kata/istilah yang sesuai dengan konteks dalam Bahasa Indonesia.

Apabila pengguna menemukan adanya keraguan dalam Standar ini, disarankan untuk melihat standar aslinya yaitu ISO/IEC 10118-1:2016 dan/atau dokumen terkait lain yang menyertainya.

Perlu diperhatikan bahwa kemungkinan beberapa unsur dari Standar ini dapat berupa hak kekayaan intelektual (HAKI). Namun selama proses perumusan SNI, Badan Standardisasi Nasional telah memperhatikan penyelesaian terhadap kemungkinan adanya HAKI terkait substansi SNI. Apabila setelah penetapan SNI masih terdapat permasalahan terkait HAKI, Badan Standardisasi Nasional tidak bertanggung jawab mengenai bukti, validitas, dan ruang lingkup dari HAKI tersebut.

Teknologi informasi — Teknik keamanan — Fungsi-*hash* — Bagian 1: Umum

1 Ruang lingkup

ISO/IEC 10118 (semua bagian) menspesifikasikan fungsi-*hash* dan oleh karena itu berlaku untuk penyediaan layanan autentikasi, integritas, dan nonrepudiasi. Fungsi-*hash* memetakan rangkaian bit dengan panjang variabel (tetapi biasanya diberi batas atas) ke rangkaian bit dengan panjang tetap, menggunakan algoritma tertentu. Fungsi-*hash* dapat digunakan untuk

— mereduksi suatu pesan menjadi *imprint* untuk input ke mekanisme tanda tangan digital, dan

— mengikat pengguna ke suatu rangkaian bit tertentu tanpa mengungkapkan rangkaian ini.

CATATAN Fungsi-*hash* yang dispesifikasikan dalam ISO/IEC 10118 (semua bagian) tidak melibatkan penggunaan kunci rahasia. Namun demikian, fungsi-*hash* ini boleh digunakan, bersamaan dengan kunci rahasia, untuk membuat kode autentikasi pesan. Kode Autentikasi Pesan (*Message Authentication Codes* – MAC) menyediakan autentikasi asal data serta integritas pesan. Teknik untuk komputasi MAC menggunakan fungsi-*hash* yang dispesifikasikan dalam ISO/IEC 9797-2 [1].

Dokumen ini berisi definisi, simbol, singkatan dan persyaratan yang umum untuk semua bagian ISO/IEC 10118 lainnya. Kriteria yang digunakan untuk memilih algoritma yang dispesifikasikan dalam bagian berikutnya dari ISO/IEC 10118 didefinisikan dalam Lampiran B dokumen ini.

2 Acuan normatif

Tidak ada acuan normatif dalam dokumen ini

3 Istilah dan definisi

Untuk tujuan dokumen ini, istilah dan definisi berikut berlaku.

ISO dan IEC memelihara basis data peristilahan untuk digunakan dalam standardisasi di alamat berikut:

— IEC Electropedia: tersedia di https://www.electropedia.org/

— ISO *Online browsing platform*: tersedia di https://www.iso.org/obp

3.1
Fungsi-*hash* tahan kolisi
Fungsi-*hash* yang memenuhi properti berikut: secara komputasi tidak fisibel menemukan dua masukan berbeda yang dipetakan ke output yang sama.

Catatan 1 untuk entri: Fisibilitas komputasi bergantung pada persyaratan keamanan dan lingkungan spesifik. Mengacu ke Lampiran C.

3.2
string data
data
string bit yang merupakan input ke fungsi-*hash*

3.3
kode-*hash*
string bit yang merupakan output dari fungsi-*hash*

Catatan 1 untuk entri: Literatur mengenai subjek ini berisi berbagai istilah yang memiliki arti yang sama atau serupa dengan kode-*hash.* Kode Deteksi Modifikasi, Kode Deteksi Manipulasi, *digest,* hasil-*hash*, nilai-*hash* dan *imprint* adalah beberapa contohnya.

3.4
fungsi-*hash*
fungsi yang memetakan string bit dengan panjang variabel (tetapi biasanya diberi batas atas) ke string bit dengan panjang tetap, memenuhi dua properti berikut:

— untuk keluaran tertentu, secara komputasi tidak mungkin menemukan input yang memetakan ke output ini;

— untuk input tertentu, secara komputasi tidak mungkin menemukan input kedua yang dipetakan ke keluaran yang sama

Catatan 1 untuk entri: Fisibilitas komputasi bergantung pada persyaratan keamanan dan lingkungan spesifik. Rujuk ke Lampiran C.

3.5
nilai inisialisasi
nilai yang digunakan dalam mendefinisikan titik awal dari suatu fungsi-*hash*

Catatan 1 untuk entri: Literatur mengenai subjek ini berisi berbagai istilah yang memiliki arti yang sama atau serupa dengan nilai inisialisasi. Vektor inisialisasi dan nilai awal adalah contohnya.

3.6
transformasi output
transformasi atau pemetaan keluaran dari tahap iterasi untuk mendapatkan kode-*hash*

3.7
padding
menambahkan bit ekstra ke suatu string data

3.8
Fungsi-*round*
fungsi $\phi(.,.)$ yang mengubah dua string biner dengan panjang L_1 dan L_2 menjadi suatu string biner dengan panjang L_2 yang digunakan secara iteratif sebagai bagian dari fungsi-*hash*, yang menggabungkan string data dengan panjang L_1dengan output sebelumnya dengan panjang L_2 atau nilai inisialisasi.

Catatan 1 untuk entri: Literatur mengenai subjek ini mengandung berbagai istilah yang memiliki arti yang sama atau serupa dengan fungsi *round*. Fungsi kompresi dan fungsi iteratif adalah beberapa contohnya.

4 Simbol dan singkatan istilah

4.1 Simbol

Untuk ISO/IEC 10118 (semua bagian), simbol dan singkatan berikut digunakan:

B_i *a byte* (byte)

$\mathbb{D}$ *data* (data)

D_i *a block derived from the data string* $\mathbb{D}$ *after the padding process* (sebuah blok yang diderivasi dari string data $\mathbb{D}$ setelah proses padding)

h *hash-function* (fungsi-*hash*)

H *hash-code* (kode-*hash*)

H_i *a string of* L_2 *bits which is used in the hashing operation to store an intermediate result* (suatu string dari L_2 bit yang digunakan dalam operasi *hash* untuk menyimpan suatu nilai tengah *IV* nilai inisialisasi)

IV *Initializing value* (nilai inisialisasi)

L_1 *length (in bits) of the first of the two input strings to the round-function* (panjang (dalam bit) dari string pertama dari dua input untuk fungsi-*round*)

L_2 *length (in bits) of the second of the two input strings to the round-function, the output string from the round-function, and of the initializing value* (panjang (dalam bit) dari string kedua dari dua string input ke fungsi-*round*, string output dari fungsi-*round*, dan nilai inisialisasi)

L_X length (in bits) of a string of bits X (panjang (dalam bit) dari suatu string bit X)

ϕ *round-function* (phi) (fungsi-*round*)

T an output transformation function, e.g. truncation (fungsi transformasi keluaran, misalnya *truncation*)

$X \,||\, Y$ *concatenation of strings of bits* X *and* Y *in the indicated order* (penggabungan dari string dari bit X dan Y dalam orde yang diindikasikan)

$X \oplus Y$ *exclusive-or of strings of bits* X *and* Y (*where* $L_X = L_Y$) (*exclusive-or* dari string bit X dan Y (dengan $L_X = L_Y$))

4.2 Simbol spesifik untuk dokumen ini

Untuk tujuan dokumen ini, simbol berikut ini berlaku:

q jumlah blok dalam string data setelah proses padding dan pemisahan

4.3 Konvensi pengodean

Dalam konteks istilah "*most significant bit/byte*" dan "*least significant bit/byte*" mempunyai arti (misalnya ketika string bit/byte diperlakukan sebagai nilai numerik), bit/byte paling kiri dari sebuah blok harus yang paling signifikan.

5 Persyaratan

Penggunaan fungsi-*hash* mensyaratkan para pihak yang terlibat harus beroperasi pada string bit yang sama, meskipun representasi data mungkin berbeda di lingkungan masing-masing entitas. Hal ini mungkin memerlukan satu atau lebih entitas untuk mengonversi data tersebut menjadi representasi string bit yang disepakati sebelum menerapkan fungsi-*hash*.

Beberapa fungsi-*hash* yang dispesifikasikan dalam ISO/IEC 10118 (semua bagian) memerlukan padding, sehingga string data memiliki panjang yang disyaratkan. Beberapa metode padding disajikan dalam Lampiran A dokumen ini; metode padding tambahan dapat dispesifikasikan di setiap bagian dari ISO/IEC 10118 yang memerlukan padding.

6 Model umum untuk fungsi-*hash*

6.1 Umum

Fungsi-*hash* yang dispesifikasikan dalam ISO/IEC 10118 (semua bagian) mensyaratkan penggunaan fungsi *round* ϕ. Pada bagian selanjutnya dari ISO/IEC 10118, beberapa alternatif untuk fungsi dispesifikasikan.

Fungsi-*hash* yang dispesifikasikan pada bagian selanjutnya dari ISO/IEC 10118 menyediakan kode-*hash* dengan panjang L_H di mana L_H kurang dari atau sama dengan nilai L_2 untuk fungsi *round* ϕ yang digunakan.

6.2 Operasi *hashing*

6.2.1 Umum

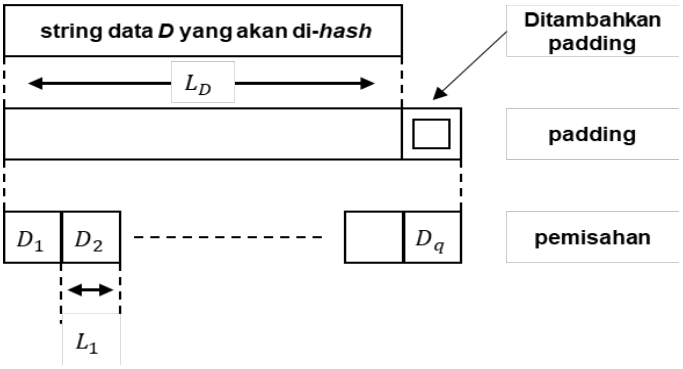
Misalkan ϕ merupakan fungsi-*round* dan IV merupakan nilai inisialisasi dengan panjang L_2. Untuk fungsi-*hash* yang dispesifikasikan pada bagian selanjutnya dari ISO/IEC 10118, nilai *IV* harus ditetapkan untuk fungsi-*hash* tertentu. Kode-*hash* H dari data D harus dikalkulasi menggunakan empat langkah berikut.

6.2.2 Langkah 1 (padding)

String data D dipadding untuk memastikan bahwa panjangnya merupakan kelipatan bilangan bulat dari L_1. Lihat Lampiran A untuk informasi lebih lanjut.

6.2.3 Langkah 2 (pemisahan)

Versi yang sudah dipadding dari string data D yang sudah terisi dipecah menjadi blok-blok L_1 bit $D_1, D_2, \ldots, D_q$ di mana D_1 merepresentasikan L_1, bit pertama dari versi yang sudah dipadding D, D_2 merepresentasikan L_1 bit berikutnya, dan seterusnya. Proses padding dan pemisahan diilustrasikan pada Gambar 1.



Gambar 1 – Proses padding dan pemisahan

CATATAN: Terkadang, akan lebih efisien jika pemisahan dilakukan sebelum padding. Selanjutnya padding dilakukan pada blok terakhir.

6.2.4 Langkah 3 (iterasi)

Misalkan $D_1, D_2, \dots, D_q$ menjadi blok L_1-bit dari data setelah padding dan pemisahan. Misalkan H_0 adalah string bit yang sama dengan IV. String L_2 -bit $H_1, H_2, \dots, H_q$ dikalkulasi secara iteratif dengan cara berikut.

untuk i dari 1 sampai q:

$$H_1 = \phi(D_i, H_{i-1}).$$

6.2.5 Langkah 4 (transformasi keluaran)

Kode-*hash* H diperoleh dengan melakukan transformasi T pada H_q, output dari langkah 3, untuk mendapatkan L_H bit dari kode-*hash* final.

CONTOH Transformasi T dapat berupa operasi *truncation*.

6.3 Model siklus hidup kunci generik

Pada bagian selanjutnya dari ISO/IEC 10118, dispesifikasikan contoh fungsi-*hash* berbasiskan model umum. Spesifikasi fungsi-*hash* individual dalam setiap kasus akan mensyaratkan hal berikut untuk didefinisikan:

— parameter L_1, L_2 ;

— metode padding;

— nilai inisialisasi IV;

— fungsi-*round* ϕ;

— transformasi output T.

Penggunaan praktis fungsi-*hash* yang didefinisikan menggunakan model umum juga akan mensyaratkan pemilihan parameter L_H.

Lampiran A
(informatif)
Metode padding

A.1 Umum

Kalkulasi kode-*hash*, sebagaimana dispesifikasikan dalam bagian lain ISO/IEC 10118, mungkin mensyaratkan penyeleksian metode padding. Metode padding akan selalu mengeluarkan string data terisi yang panjangnya (dalam bit) merupakan kelipatan L_1. Tiga metode disajikan dalam lampiran ini.

Bit padding (jika ada) tidak perlu disimpan atau dikirim bersama data. Pemverifikasi harus mengetahui apakah bit padding telah disimpan atau dikirim atau belum, dan metode padding mana yang sedang digunakan.

A.2 Metode 1

Data yang akan dikalkulasi kode *hash*-nya ditambahkan dengan suatu bit "1" tunggal. Data yang dihasilkan kemudian ditambahkan dengan bit "0" sesedikit mungkin seperlunya (mungkin nol) untuk mendapatkan panjang yang disyaratkan.

CATATAN Metode 1 selalu mensyaratkan penambahan setidaknya satu bit padding

A.3 Metode 2

Metode padding ini mensyaratkan pemilihan parameter r (di mana $r \leq L_1$), misalkan $r = 64$, dan metode untuk mengkodekan panjang bit data D, yaitu L_D , sebagai string bit dengan panjang r. Pilihan untuk r akan membatasi panjang D, dalam $L_D < 2^r$.

Data D yang kode *hash*-nya akan dikalkulasi diisi menggunakan prosedur berikut.

a) D digabungkan dengan satu bit "1" tunggal.

b) Hasil dari langkah sebelumnya digabungkan dengan bit "0" antara nol dan $L_1 - 1$, sedemikian hingga panjang string yang dihasilkan kongruen dengan $L_1 - r$ modulo L_1. Hasilnya akan berupa string bit yang panjangnya *r* bit lebih pendek dari kelipatan bilangan bulat L_1 bit, atau dalam kasus $r = L_1$, akan berupa string bit yang panjangnya merupakan kelipatan tepat dari bit L_1.

c) Tambahkan pengodean r-bit L_D, menggunakan metode pengodean yang dipilih, menghasilkan versi D yang sudah dipadding.

A.4 Metode 3 — Pad10*1

Data D yang kode *hash*-nya akan dikalkulasi diisi menggunakan prosedur berikut.

a) D digabungkan dengan satu bit "1" tunggal.

b) Hasil dari langkah sebelumnya digabungkan dengan $(-m - 2) \text{ modulo } L_1$ bit '0', dengan m adalah panjang asli dari D.

c) Tambahkan bit tunggal "1", menghasilkan versi D yang sudah dipadding.

Lampiran B
(normatif)
Kriteria pengajuan fungsi-*hash* untuk kemungkinan inklusi dalam ISO/IEC 10118 (semua bagian)

B.1 Pedoman yang digunakan untuk menyeleksi fungsi-*hash*

Fungsi-*hash* yang disertakan dalam bagian selanjutnya dari ISO/IEC 10118 telah diseleksi dari berbagai macam teknik yang dipublikasikan dan digunakan. Pengecualian fungsi-*hash* tertentu tidak semestinya berimplikasi bahwa teknik-teknik tersebut tidak aman. Fungsi-*hash* yang dispesifikasikan merepresentasikan sekumpulan kecil teknik yang terpilih berdasarkan kriteria berikut (di mana kondisi urutan penyajian kriteria tidak penting). SC 27/WG 2 *Standing Document* 5 (WG2 SD5) menjelaskan proses yang dapat diikuti oleh ISO/IEC JTC 1/SC 27 ketika memutuskan inklusi fungsi-*hash* baru dalam ISO/IEC 10118 (semua bagian).

Seleksi dilakukan berkenaan dengan aspek fungsi-*hash* berikut.

a) Keamanan fungsi-*hash,* yaitu algoritma terseleksi harus tahan terhadap serangan kriptanalitis. Gol serangan, serangan generik, dan dampak serangan kriptanalitis didefinisikan dalam Lampiran C dari dokumen ini. Keberadaan bukti keamanan atau reduksi keamanan dianggap sebagai argumen signifikan yang mendukung fungsi-*hash*, bergantung pada model keamanan dan asumsi pembuktian. Sifat dari evaluasi apapun juga sangat penting, terutama yang dilakukan oleh organisasi evaluasi yang diakui secara luas.

b) Performa fungsi-*has*h pada berbagai platform pada umumnya. Hal ini tidak hanya mencakup isu-isu seperti efisiensi waktu dan ruang, tetapi juga apakah teknik tersebut memiliki karakteristik yang memberikan keunggulan dibandingkan teknik lainnya.

c) Sifat permasalahan lisensi apapun yang memengaruhi fungsi-*hash.*

d) Maturitas fungsi-*hash*. Maturitas fungsi-*hash* dievaluasi dalam hal seberapa ekstensif penggunaannya, seberapa luas sebarang analisis telah dipublikasikan, dan seberapa banyak fungsi-*hash* telah diteliti dengan cermat. Standar nasional dipandang mempunyai maturitas tinggi.

e) Sejauh mana fungsi-*hash* didukung oleh organisasi yang diakui (misalnya badan standar, badan keamanan pemerintah, dll.), atau sedang diselidiki dan/atau dianalisis untuk disetujui oleh badan tersebut.

f) Tingkat adopsi fungsi-*hash* saat ini. Kecuali ada pertimbangan lain yang mengesampingkan keputusan tersebut, fungsi-*hash* yang merupakan standar *de facto* lebih disukai daripada teknik yang kurang sering digunakan dengan baik.

g) Secara umum, jumlah fungsi-*hash* yang akan distandarisasi di setiap bagian ISO/IEC 10118 sebaiknya sesedikit mungkin. Ada tiga pengecualian terhadap prinsip ini.

   1) Jika dua fungsi-*hash* memiliki karakteristik berbeda, misalnya fungsi-*hash* dengan panjang kode-*hash* yang berbeda atau fungsi-*hash* dengan persyaratan komputasi dan implementasi ruang yang sangat berbeda, dan kedua rangkaian karakteristik tersebut memiliki signifikansi praktis, fungsi-*hash* dari kedua jenis tersebut kemungkinan besar akan distandarisasi.

2) Secara umum diinginkan untuk memiliki beberapa fungsi-*hash* terstandarisasi yang berbasis pada prinsip-prinsip fundamental yang berbeda, sehingga jika satu fungsi-*hash* menjadi rentan terhadap serangan kriptanalitis, fungsi-*hash* lainnya mempunyai peluang bagus untuk tetap aman.

3) Secara umum diinginkan untuk memiliki fungsi-*hash* yang terstandarisasi dengan margin keamanan yang besar untuk tingkat keamanan aplikasi yang berubah-ubah.

B.2 Kriteria kualifikasi minimum untuk pengajuan fungsi-*hash* baru

Kriteria yang ditetapkan dalam Pasal B.2 dimaksudkan untuk pengajuan fungsi-*hash* yang belum termasuk dalam bagian selanjutnya dari ISO/IEC 10118. Agar fungsi-*hash* dipertimbangkan untuk inklusi dalam bagian berikutnya dari ISO/IEC 10118, fungsi-*hash* harus memenuhi persyaratan berikut.

a) Kriptanalisis dilakukan dan hasilnya diketahui: Harus tidak ada serangan kriptanalitis yang diketahui yang memecahkan fungsi-*hash*. Lampiran C memberikan informasi lebih lanjut mengenai serangan yang terkait dengan fungsi-*hash.*

b) Domain publik: Deskripsi fungsi-*hash* harus sudah terpublikasi minimal 3 tahun di domain publik. Contoh publikasi dan konferensi yang dapat diterima untuk presentasi fungsi-*hash* mencakup namun tidak terbatas pada yang berikut:

   1) Konferensi dan lokakarya IACR:

      i) *International Conference on the Theory and Application of Cryptology and Information Security* (Asiacrypt)

      ii) *International Cryptology Conference* (Crypto)

      iii) *International Conference on the Theory and Applications of Cryptographic Techniques* (Eurocrypt)

      iv) *International Workshop on Fast Software Encryption* (FSE)

      v) *International Workshop on Cryptographic Hardware and Embedded Systems* (CHES)

   2) Konferensi tahunan IEEE:

      i) *Symposium on Security and Privacy*

      ii) *Symposium on the Foundations of Computer Science* (FOCS)

   3) Konferensi tahunan ACM

      i) *Symposium on Theory of Computing* (ACM-STOC)

      ii) *Computer and Communication Security* (ACM-CCS)

   4) Konferensi internasional terkenal yang memiliki sejarah lebih dari 15 tahun dengan prosiding yang tersedia:

      i) USENIX *Security*

      ii) *European Symposium on Research in Computer Security* (ESORICS)

      iii) *Australasian Conference on Information Security and Privacy* (ACISP)

      iv) *Financial Cryptography and Data Security* (FC)

      v) *International Conference on Information Security and Cryptography* (ICISC)

      vi) *Conference on Selected Areas in Cryptography* (SAC)

5) Jurnal terkenal:
   i) ACM
      I) *Journal of the* ACM
      II) *Communications of the* ACM
   ii) Elsevier
      I) *Computer Communications*
      II) *Information and Computation*
      III) *Journal of Computer and System Sciences* (JCSS)
      IV) *Journal of Discrete Algorithms*
   iii) IEEE
      I) IEEE *Transactions on Information Theory*
      II) IEEE *Transactions on Computers*
      III) IEEE *Security & Privacy*
   iv) IEICE
      I) IEICE *Transactions on Fundamentals of Electronics, Communications and Computer Sciences*
      II) IEICE *Transactions on Information and Systems*
   v) SIAM *Journal on Computing*
   vi) Springer
      I) *Combinatorica*
      II) *Cryptography and Communications*
      III) *Designs, Codes and Cryptography*
      IV) *Journal of Cryptology*
      V) *International Journal of Information Security*

6) Publikasi resmi sebagai suatu standar dalam bahasa Inggris yang tersedia bagi masyarakat umum oleh organisasi standardisasi yang diakui.

7) Kompetisi internasional dengan tujuan utama untuk memilih fungsi-*hash* mutakhir yang dijalankan minimal 2 tahun, dan yang analisis serta publikasinya terbuka untuk masyarakat umum. Pengiriman ke kompetisi internasional ini dapat dianggap dipublikasikan.

c) Ada dokumentasi kriptanalisis: Sebelum diinklusi, fungsi-*hash* harus memiliki makalah kriptanalisis yang dipublikasikan di jurnal atau konferensi reviu-sejawat seperti yang tercantum dalam b).

d) Adopsi industri: Bukti kuat harus diberikan mengenai aplikasi komersial yang menggunakan fungsi-*hash* dan kemungkinan penyebaran aplikasi tersebut di seluruh dunia.

e) Performa: Untuk panjang kode-*hash* yang telah ditentukan sebelumnya, pengukuran performa harus disediakan untuk banyak vektor berbeda seperti bit/*cycle* atau bit/*watt*. Bukti kuat harus diberikan bahwa fungsi-*hash* menawarkan performa yang dapat diterima pada vektor performa yang dioptimalkan untuk aplikasi yang dimaksudkan dibandingkan dengan fungsi-*hash* yang sudah ada dalam standar.

Lampiran C
(normatif)
Pertimbangan keamanan

C.1 Gol serangan

Terdapat berbagai gol serangan terkait dengan fungsi-*hash* (lihat, sebagai contoh, Referensi [2]). Hal-hal berikut ini sangat penting.

Pencarian kolisi — Gol serangan adalah menemukan dua string data berbeda M_1, M_2 sedemikian rupa sehingga $h(M_1) = h(M_2)$.

Pencarian *preimage* — Diberikan string bit H dengan panjang yang sesuai, gol serangan adalah menemukan string data M sedemikian rupa sehingga $h(M) = H$.

Pencarian *preimage* kedua — Diberikan string data M, gol serangan adalah menemukan string data lain M' sedemikian rupa sehingga $h(M') = h(M)$ dan $M'^1 \neq M$.

Serangan ekstensi panjang — Diberikan string bit $h(M)$ untuk beberapa string data tidak kosong yang tidak diketahui *M*, gol serangan adalah menemukan string data M' dan nilai $h(M||M')$.

CATATAN Kecanggihan mutakhir dalam kriptanalisis fungsi-*hash* melibatkan penggunaan berbagai macam gol serangan termasuk beberapa yang tidak tercantum di atas. Gol serangan ini dipertimbangkan selama proses standardisasi, namun memiliki kepentingan tambahan. Selain itu, dalam aplikasi, biasanya tidak dipersyaratkan bahwa fungsi-*hash* tahan terhadap semua gol serangan. Secara khusus, beberapa bagian tertentu dari persyaratan tersebut hanya dipertimbangkan.

C.2 Serangan generik

Serangan generik adalah serangan yang berlaku untuk semua fungsi-*hash* dan tidak bergantung pada desain fungsi-*hash.*

CONTOH Salah satu contoh serangan generik adalah pencarian *preimage* secara *brute force*. Diberikan nilai kode-*hash*, penyerang mengevaluasi nilai $h(M)$, mencoba berbagai kemungkinan string data M dan membandingkan kode-*hash* yang dihasilkan dengan yang diberikan. Jika kedua kode *hash* cocok, tujuan pencarian *preimage* tercapai.

C.3 Dampak serangan kriptanalitis

Dalam ISO/IEC 10118 (semua bagian), resistensi fungsi-*hash* terhadap kemungkinan sasaran serangan dispesifikasikan dalam istilah “ketidaklayakan komputasi” untuk memenuhi sasaran. Sebagaimana tercantum dalam definisi, arti ketidaklayakan komputasi bergantung pada persyaratan spesifik keamanan dan lingkungan. Salah satu arti yang umum digunakan oleh praktisi keamanan adalah bahwa ketidaklayakan komputasi merupakan properti untuk suatu tugas yang menuntut jumlah sumber daya komputasi melebihi apa yang tersedia secara umum.

Pendekatan yang lebih ketat adalah dengan membandingkan efisiensi serangan tertentu terhadap serangan generik dengan tujuan serangan yang sama. Jika semua serangan kriptanalisis yang diketahui untuk sasaran serangan tertentu tidak lebih efisien dibandingkan serangan generik terkait, maka fungsi-*hash* dikatakan tahan terhadap sasaran serangan

tersebut. Meskipun demikian, jika ada serangan kriptanalitis yang secara substansial lebih efisien dibandingkan serangan generik terkait, maka fungsi-*hash* dikatakan terpecahkan.

Efisiensi serangan kriptanalitis ditentukan oleh tiga parameter: kompleksitas serangan, jumlah memori yang disyaratkan, dan kemungkinan keberhasilan.

Kompleksitas serangan kriptanalitis dinormalisasi dengan jumlah panggilan ke fungsi-*round* untuk menentukan kompleksitasnya dibandingkan dengan serangan generik. Kompleksitas normalisasi ini dapat bervariasi tergantung pada sifat serangannya. Dalam kebanyakan kasus, kompleksitas algoritma serangan hanya dapat diestimasi.

CONTOH 1 Pertimbangkan beberapa sasaran serangan tetap. Misalkan untuk fungsi-*hash* tertentu, terdapat serangan dengan kompleksitas W, probabilitas keberhasilan P, jumlah memori yang disyaratkan N, dan kondisi berikut berlaku:

— N tidak melebihi jumlah memori yang disyaratkan untuk serangan generik apa pun untuk gol serangan ini;

— P tidak kurang dari probabilitas keberhasilan serangan generik apa pun untuk gol serangan ini;

— W secara signifikan lebih kecil dibandingkan kompleksitas serangan generik apa pun untuk gol serangan ini.

Dalam hal ini, fungsi-*hash* dianggap terpecahkan sehubungan dengan gol serangan tersebut.

CONTOH 2 Pertimbangkan fungsi-*hash* dengan kode-*hash* 256-bit. Jika ada serangan pencarian *preimage* dengan kompleksitas sekitar 2^{192} panggilan fungsi-*round*, kebutuhan memori praktis sekitar 2^{20} byte dan probabilitas keberhasilan mendekati 1, maka fungsi-*hash* terpecahkan sehubungan dengan pencarian *preimage.*

Bibliografi

[1] ISO/IEC 9797-2, Information technology — Security techniques — Message Authentication Codes (MACs) — Part 2: Mechanisms using a dedicated hash-function

[2] PRENEEL B. Analysis and Design of Cryptographic Hash Functions, Doctoral Dissertation, Katholieke Universiteit Leuven, 1993ISO/IEC 9796 (all parts), *Information technology — Security techniques — Digital signature schemes giving message recovery*

Information technology — Security techniques — Key management — Part 1: Framework

1 Scope

ISO/IEC 10118 (all parts) specifies hash-functions and is therefore applicable to the provision of authentication, integrity and non-repudiation services. Hash-functions map strings of bits of variable (but usually upper bounded) length to fixed-length strings of bits, using a specified algorithm. They can be used for

— reducing a message to a short imprint for input to a digital signature mechanism, and

— committing the user to a given string of bits without revealing this string.

NOTE The hash-functions specified in ISO/IEC 10118 (all parts) do not involve the use of secret keys. However, these hash-functions may be used, in conjunction with secret keys, to build message authentication codes. Message Authentication Codes (MACs) provide data origin authentication as well as message integrity. Techniques for computing a MAC using a hash-function are specified in ISO/IEC 9797-2 [1].

This document contains definitions, symbols, abbreviations and requirements that are common to all the other parts of ISO/IEC 10118. The criteria used to select the algorithms specified in subsequent parts of ISO/IEC 10118 are defined in Annex B of this document.

2 Terms and definitions

There are no normative references in this document

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at http://www.iso.org/obp

3.1
collision-resistant hash-function
hash-function satisfying the following property: it is computationally infeasible to find any two distinct inputs which map to the same output

Note 1 to entry: Computational feasibility depends on the specific security requirements and environment. Refer to Annex C.

3.2
data string
data
string of bits which is the input to a hash-function

3.3
hash-code
string of bits which is the output of a hash-function

Note 1 to entry: The literature on this subject contains a variety of terms that have the same or similar meaning as hash-code. Modification Detection Code, Manipulation Detection Code, digest, hash-result, hash-value and imprint are some examples.

3.4
hash-function
function which maps strings of bits of variable (but usually upper bounded) length to fixed-length strings of bits, satisfying the following two properties:

— for a given output, it is computationally infeasible to find an input which maps to this output;

— for a given input, it is computationally infeasible to find a second input which maps to the same output

Note 1 to entry: Computational feasibility depends on the specific security requirements and environment. Refer to Annex C.

3.5
initializing value
value used in defining the starting point of a hash-function

Note 1 to entry: The literature on this subject contains a variety of terms that have the same or similar meaning as initializing value. Initialization vector and starting value are examples.

3.6
output transformation
transformation or mapping of the output of the iteration stage to obtain the hash-code

3.7
padding
appending extra bits to a data string

3.8
round-function
function $\phi(.,.)$ that transforms two binary strings of lengths L_1 and L_2 to a binary string of length L_2 that is used iteratively as part of a hash-function, where it combines a data string of length L_1 with the previous output of length L_2 or the initializing value

Note 1 to entry: The literature on this subject contains a variety of terms that have the same or similar meaning as round-function. Compression function and iterative function are some examples.

4 Symbols and abbreviated terms

4.1 General Symbols

For ISO/IEC 10118 (all parts), the following symbols and abbreviations are used:

B_i a byte

D	data
D_i	a block derived from the data string D after the padding process
h	hash-function
H	hash-code
H_i	a string of L_2 bits which is used in the hashing operation to store an intermediate result
IV	initializing value
L_1	length (in bits) of the first of the two input strings to the round-function
L_2	length (in bits) of the second of the two input strings to the round-function, the output string from the round-function, and of the initializing value
L_X	length (in bits) of a string of bits X
ϕ	round-function (phi)
T	an output transformation function, e.g. truncation
$X \| \| Y$	concatenation of strings of bits X and Y in the indicated order
$X \oplus Y$	exclusive-or of strings of bits X and Y (where $L_X = L_Y$)

4.2 Symbols specific to this document

For the purpose of this document, the following symbol applies:

q	number of blocks in the data string after the padding and splitting process

4.3 Symbols specific to this document

In contexts where the terms “most significant bit/byte” and “least significant bit/byte” have a meaning (e.g. where strings of bits/bytes are treated as numerical values), the leftmost bits/bytes of a block shall be the most significant.

5 Requirements

The use of a hash-function requires that the parties involved shall operate upon precisely the same bit string, even though the representation of the data may be different in each entity’s environment. This may require one or more of the entities to convert the data into an agreed bit-string representation prior to applying a hash-function.

Some of the hash-functions specified in ISO/IEC 10118 (all parts) require padding, so that the data string is of the required length. Several padding methods are presented in Annex A of this document; additional padding methods may be specified in each part of ISO/IEC 10118 where padding is needed.

6 General model for hash-functions

6.1 General

The hash-functions specified in ISO/IEC 10118 (all parts) require the use of a round-function ϕ. In subsequent parts of ISO/IEC 10118, several alternatives for the function ϕ are specified. The hash-functions which are specified in subsequent parts of ISO/IEC 10118 provide hash-codes of length L_H, where L_H is less than or equal to the value of L_2 for the round-function ϕ being used.

6.2 Hashing operation

6.2.1 General

Let ϕ be a round-function and IV be an initializing value of length L_2. For the hash-functions specified in subsequent parts of ISO/IEC 10118, the value of the IV shall be fixed for a given hash-function ϕ. The hash-code H of the data $\mathbb{D}$ shall be calculated using the following four steps.

6.2.2 Step 1 (padding)

The data string $\mathbb{D}$ is padded in order to ensure that its length is an integer multiple of L_1. See Annex A for more information.

6.2.3 Step 2 (splitting)

The padded version of the data string $\mathbb{D}$ is split into L_1-bit blocks $D_1, D_2, ..., D_q$, where D_1 represents the first L_1 bits of the padded version of $\mathbb{D}$, D_2 represents the next L_1 bits, and so on. The padding and splitting processes are illustrated in Figure 1.

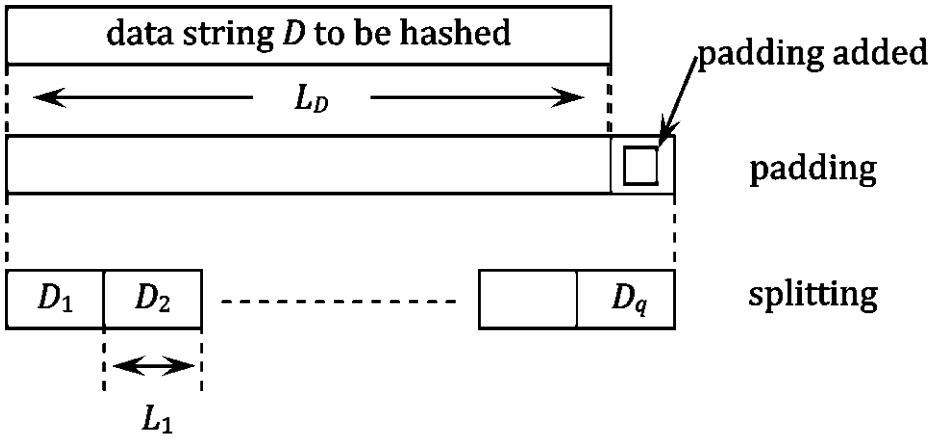


Figure 1 — Padding and splitting processes

NOTE Sometimes, it is more efficient to have the splitting occur before the padding. The padding is then done on the last block.

6.2.4 Step 3 (iteration)

Let $D_1, D_2, ..., D_q$ be the L_1-bit blocks of the data after padding and splitting. Let H_0 be a bit string equal to IV. The L_2-bit strings $H_1, H_2, ..., H_q$ are calculated iteratively in the following way.

for i from 1 to q:

$$H_i = \phi(D_i, H_{i-1}).$$

6.2.5 Step 4 (output transformation)

The hash-code H is derived by performing a transformation T on H_q, the output of step 3, to obtain the L_H bits of the final hash-code.

EXAMPLE The transformation T may be a truncation operation.

6.3 Use of the general model

In subsequent parts of ISO/IEC 10118, examples of hash-functions based on the general model are specified. Specification of an individual hash-function will in each case require the following to be defined:

— parameters L_1, L_2;

— the padding method;

— the initializing value IV;

— the round-function ϕ;

— the output transformation T.

Practical use of a hash-function defined using the general model will also require the choice of the parameter L_H.

Annex A
(informative)
Padding methods

A.1 General

The calculation of a hash-code, as specified in other parts of ISO/IEC 10118, may require the selection of a padding method. The padding method will always output a padded data string whose length (in bits) is a multiple of L_1. Three methods are presented in this annex.

The padding bits (if any) need not be stored or transmitted with the data. The verifier shall know whether or not the padding bits have been stored or transmitted, and which padding method is in use.

A.2 Method 1

The data for which the hash-code is to be calculated is appended with a single “1” bit. The resulting data are then appended with as few (possibly zero) “0” bits as are necessary to obtain the required length.

NOTE Method 1 always requires the addition of at least one padding bit**.**

A.3 Method 2

This padding method requires the selection of a parameter r (where $r \leq L_1$), e.g. r = 64, and a method for encoding the bit length of the data $\mathbb{D}$, i.e. L_D, as a bit string of length r. The choice for r will limit the length of $\mathbb{D}$, in that $L_D < 2^r$.

The data $\mathbb{D}$ for which the hash-code is to be calculated is padded using the following procedure.

a) $\mathbb{D}$ is concatenated with a single “1” bit.

b) The result of the previous step is concatenated with between zero and $L_1 – 1$ “0” bits, such that the length of the resultant string is congruent to $L_1 – r$ modulo L_1. The result will be a bit string whose length will either be r bits short of an integer multiple of L_1 bits, or in the case of $r = L_1$, will be a bit string whose length is an exact multiple of L_1 bits.

c) Append an r-bit encoding of L_D, using the selected encoding method, yielding the padded version of $\mathbb{D}$.

A.4 Method 3 — Pad10*1

The data D for which the hash-code is to be calculated is padded using the following procedure.

a) D is concatenated with a single '1' bit.

b) The result of the previous step is concatenated with $(-m-2) \text{ modulo } L_1$ '0' bits, where m is the original length of D.

c) Append the single bit '1', yielding the padded version of D.

Annex B
(normative)
Criteria for submission of hash-functions for possible inclusion in ISO/IEC 10118 (all parts)

B.1 Guidelines used for selecting hash-functions

The hash-functions included in subsequent parts of ISO/IEC 10118 have been selected from the large variety of such techniques published and in use. The exclusion of particular hash-functions does not necessarily imply that these techniques are insecure. The hash-functions specified represent a small set of techniques chosen according to the following criteria (where the order of presentation of the criteria is not of significance). SC 27/WG 2 Standing Document 5 (WG2 SD5) describes a process that ISO/IEC JTC 1/SC 27 can follow when deciding on the inclusion of new hash-functions in ISO/IEC 10118 (all parts).

Selection is made with respect to the following aspects of the hash-function.

a) The *security* of the hash-function, i.e. selected algorithms shall be resistant to cryptanalytic attacks. Attack goals, generic attacks and impact of cryptanalytic attacks are defined in Annex C of this document. The existence of a proof of security or security reduction is regarded as a significant argument in favour of a hash-function, depending on the security model and the proof assumptions. The nature of any evaluations is also of great importance, especially those conducted by widely recognized evaluation organizations.

b) The *performance* of the hash-function on a variety of typical platforms. This includes not only issues such as time and space efficiency, but also whether or not it has characteristics that give it advantages over other techniques.

c) The nature of any *licensing issues* affecting the hash-function.

d) The *maturity* of the hash-function. The *maturity* of the hash-function is evaluated in terms of how extensively it is used, how widely any analysis has been published, and how much the hash-function has been scrutinized. National standards are considered to have high maturity.

e) The degree to which the hash-function is *endorsed* by a recognized organization (e.g. a standards body, government security agency, etc.), or is under investigation and/or analysis for endorsement by such a body.

f) The existing *level of adoption* of the hash-function. Unless other considerations over-ride such a decision, hash-functions that are de facto standards are to be favoured over less well-used techniques.

g) In general, the *number* of hash-functions to be standardized in each part of ISO/IEC 10118 should be as small as possible. Three exceptions to this principle exist.

    1) Where two hash-functions have different characteristics, e.g. hash-functions with different hash-code lengths or hash-functions with widely differing computational and space implementation requirements, and both sets of characteristics have practical significance, hash-functions of both types are likely to be standardized.

    2) It is generally desirable to have multiple standardized hash-functions which are based on different fundamental principles, so that if one hash-function becomes vulnerable to cryptanalytic attack, another hash-function has a good chance of remaining secure.

3) It is generally desirable to have standardized hash-functions with large security margins for an arbitrary application's security level.

B.2 Minimum qualification criteria for the submission of new hash-functions

The criteria set out in Clause B.2 are meant for the submission of hash-functions not already included in subsequent parts of ISO/IEC 10118. In order for a hash-function to be considered for inclusion in subsequent parts of ISO/IEC 10118, the hash-function shall comply with the following requirements.

a) Cryptanalysis is performed and results are known: There shall be no known cryptanalytic attacks that break the hash-function. Annex C provides further information on attacks as related to hash-functions.

b) Public domain: The hash-function description shall have been published for a minimum period of 3 years in the public domain. Examples of acceptable publications and conferences for hash-function presentation include but are not limited to the following:

   1) IACR conferences and workshops:

      i) International Conference on the Theory and Application of Cryptology and Information Security (Asiacrypt)

      ii) International Cryptology Conference (Crypto)

      iii) International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt)

      iv) International Workshop on Fast Software Encryption (FSE)

      v) International Workshop on Cryptographic Hardware and Embedded Systems (CHES)

   2) IEEE annual conferences:

      i) Symposium on Security and Privacy

      ii) Symposium on the Foundations of Computer Science (FOCS)

   3) ACM annual conferences

      i) Symposium on Theory of Computing (ACM-STOC)

      ii) Computer and Communication Security (ACM-CCS)

   4) Well-known international conferences which have a more than 15-year history with available proceedings:

      i) USENIX Security

      ii) European Symposium on Research in Computer Security (ESORICS)

      iii) Australasian Conference on Information Security and Privacy (ACISP)

      iv) Financial Cryptography and Data Security (FC)

v) International Conference on Information Security and Cryptography (ICISC)

vi) Conference on Selected Areas in Cryptography (SAC)

5) Well-known journals:

i) ACM

I) Journal of the ACM

II) Communications of the ACM

ii) Elsevier

I) Computer Communications

II) Information and Computation

III) Journal of Computer and System Sciences (JCSS)

IV) Journal of Discrete Algorithms

iii) IEEE

I) IEEE Transactions on Information Theory

II) IEEE Transactions on Computers

III) IEEE Security & Privacy

iv) IEICE

I) IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences

II) IEICE Transactions on Information and Systems

v) SIAM Journal on Computing

vi) Springer

I) Combinatorica

II) Cryptography and Communications

III) Designs, Codes and Cryptography

IV) Journal of Cryptology

V) International Journal of Information Security

6) Official publication as a standard in English that is made available to the general public by a recognized standardization organization.

7) An international competition with the sole purpose of choosing a new state-of-the-art hash-function which is run for a minimum of 2 years, and where analysis and

publications are open to the general public. Submissions to this international competition can be considered as published.

c) Cryptanalysis documentation exists: Prior to inclusion, a hash-function shall have cryptanalysis papers published in peer-reviewed journals or conferences such as those listed in b).

d) Industry adoption: Robust evidence shall be provided of commercial applications using the hash-function and possible world-wide deployments of the applications.

e) Performance: For a pre-determined length of the hash-code, performance measurements shall be provided for many different vectors such as bits/cycle or bits/watt. Robust evidence shall be provided that the hash-function offers acceptable performance on the performance vectors that are optimized for the intended applications compared to existing hash-functions already in the standard.

Annex C
(informative)
Security considerations

C.1 Attack goals

Various attack goals exist related to hash-functions (see, for example, Reference [2]). The following are of particular importance.

Collision search — The attack goal is to find two different data strings M_1, M_2 such that $h(M_1) = h(M_2)$.

Preimage search — Given a bit string H of appropriate length, the attack goal is to find a data string M such that $h(M) = H$.

Second preimage search — Given a data string M, the attack goal is to find another data string M' such that $h(M') = h(M)$ and $M' \neq M$.

Length-extension attack — Given a bit string $h(M)$ for some unknown non-empty data string M, the attack goal is to find any data string M' and the value of $h(M \,||\, M')$.

NOTE The current state-of-the-art in cryptanalysis of hash-functions involves the use of a large variety of attack goals including some not listed above. These attack goals are taken into consideration during the standardization process, but have auxiliary importance. Moreover, in applications, it is not usually required that the hash-function is resistant to all attack goals. Typically, some particular subset of those requirements is only considered.

C.2 Generic attacks

A generic attack is an attack that is applicable to all hash-functions and does not rely on the design of the hash-function.

EXAMPLE One example of a generic attack is a brute force search for a preimage. Given a value of a hash-code, the attacker evaluates the values of $h(M)$, trying different possible data strings M and comparing the resulting hash-code with the given one. If the two hash-codes match, the preimage search goal is achieved.

C.3 Impact of cryptanalytic attack

In ISO/IEC 10118 (all parts), the resistance of a hash-function to possible attack goals is specified in terms of “computational infeasibility” of meeting the goal. As noted in the definitions, the meaning of computational infeasibility depends on the specific security requirements and environment. One meaning commonly used by security practitioners is that the computational infeasibility is a property for a task demanding an amount of computing resource beyond what is generally available.

A more rigorous approach is to compare the efficiency of a particular attack against generic attacks with the same attack goal. If all known cryptanalytic attacks for a given attack goal are not more efficient than the corresponding generic attacks, then the hash-function is said to be resistant to that attack goal. However, if there is a cryptanalytic attack substantially more efficient than the corresponding generic attacks, then the hash-function is said to be broken.

The efficiency of a cryptanalytic attack is determined by three parameters: attack complexity, amount of required memory and probability of success.

The complexities of cryptanalytic attacks are normalized to the number of calls to the round-function in order to determine their complexity relative to generic attacks. The complexity of this normalization can vary due to the nature of the attack. In most cases, the complexity of the attack algorithm can only be estimated.

EXAMPLE 1 Consider some fixed attack goal. Suppose for a particular hash-function, there is an attack with complexity W, success probability p, required amount of memory N, and the following conditions hold:

— N does not exceed the amount of memory required for any generic attack for this attack goal;

— p is not less than the success probability of any generic attack for this attack goal;

— W is significantly less than the complexity of any generic attack for this attack goal.

In this case, the hash-function is considered to be broken with respect to that attack goal.

EXAMPLE 2 Consider a hash-function with a 256-bit hash-code. If there is a preimage search attack with complexity of about 2^{192} round-function calls, practical memory requirements of about 2^{20} bytes and success probability close to 1, then the hash-function is broken with respect to preimage search.

Bibliography

[1] ISO/IEC 9797-2, *Information technology — Security techniques — Message Authentication Codes (MACs) — Part 2: Mechanisms using a dedicated hash-function*

[2] PRENEEL B. *Analysis and Design of Cryptographic Hash Functions*, Doctoral Dissertation, Katholieke Universiteit Leuven, 199

Informasi pendukung terkait perumus standar

[1] Komite Teknis Perumusan SNI
Komite Teknis 35-04 Keamanan Informasi, Keamanan Siber, dan Perlindungan Privasi

[2] Susunan keanggotaan Komtek perumus SNI Tahun 2024

Ketua	:	Soetedjo Joewono
Sekretaris	:	Didik Utomo
Anggota	:	1. Chandra Yulistia
		2. Pedro Libratu Putu Wirya
		3. Zaenal Arifin
		4. Sugi Guritman
		5. Wisnoe Prasetyo Pribadi
		6. Bisyron Wahyudi
		7. Satriyo Wibowo
		8. Sarwono Sutikno
		9. Pratama Dahlian Persadha
		10. Bety Hayat Susanti
		11. Sari Agustini Hafman

[3] Konseptor rancangan SNI
Gugus Kerja 2 Kriptografi dan Mekanisme Keamanan – Komtek 35-04 Tahun 2024:

Ketua	:	Sari Agustini Hafman
Wakil Ketua	:	Sugi Guritman
Sekretaris	:	Novita Angraini
Anggota	:	1. Pinuji Prasetyaningtyas
		2. Bety Hayat Susanti
		3. Dory Marselly
		4. Afifah
		5. Fadilla Paradise
		6. Freddy Ajax Pratama

[4] Sekretariat pengelola Komite Teknis Perumusan SNI
Direktorat Kebijakan Teknologi Keamanan Siber dan Sandi
Badan Siber dan Sandi Negara (BSSN)