Metode Autentikasi melalui Saluran Komunikasi yang Tidak Aman

Arie Karhendana
NIM 13503092

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jalan Ganesha 10, Bandung
arie@students.if.itb.ac.id

Abstrak. Saat ini, autentikasi dengan menggunakan *password* adalah teknik yang paling umum digunakan pada berbagai layanan untuk membuktikan atau mengkonfirmasi bahwa identitas seseorang adalah benar-benar milik orang yang sah. Namun, jika metode ini digunakan pada saluran komunikasi yang tidak aman, metode ini rentan terhadap berbagai serangan, misalnya penyadapan *password* oleh pihak yang tidak berhak.

Oleh karena itu, diperlukan metode untuk mengatasi kerentanan tersebut. Pada makalah ini, dikaji beberapa metode untuk mengatasi masalah tersebut, dengan menggunakan teknik kriptografi yang sudah ada, yaitu fungsi hash. Kemudian akan dipaparkan pula serangan yang mungkin terhadap metode-metode tersebut.

Kemudian diusulkan pula suatu skema autentikasi berbasis *password* yang tidak rentan terhadap penyadapan maupunpencurian *password*. Skema ini memanfaatkan proses fungsi hash sebanyak dua kali.

Kata kunci: autentikasi, *password*, challenge-response, fungsi hash

1 Pendahuluan

Keamanan data merupakan bagian yang tak terhindarkan pada kehidupan sehari-hari saat ini. Untuk itu, setiap orang berupaya untuk melindungi datanya dengan berbagai cara.

Salah satu teknik perlindungan data adalah dengan menggunakan autentikasi terhadap pengguna. Dengan menggunakan autentikasi, maka identitas pengguna dapat diketahui, sehingga sistem dapat menentukan hak akses yang sesuai bagi pengguna tersebut.

Autentikasi adalah suatu langkah untuk menentukan atau mengonfirmasi bahwa seseorang (atau sesuatu) adalah autentik atau asli. Melakukan autentikasi terhadap sebuah objek adalah melakukan konfirmasi terhadap kebenarannya. Sedangkan melakukan autentikasi terhadap seseorang biasanya adalah untuk memverifikasi identitasnya.

Pada suatu sistem komputer, autentikasi biasanya terjadi pada saat login atau permintaan akses.

Dikenal pula istilah otorisasi, yaitu proses untuk memverifikasi bahwa seseorang atau sesuatu memiliki wewenang untuk melakukan suatu aksi atau kegiatan. Otorisasi biasanya didahului dengan autentikasi.

Contoh yang paling nyata dari autentikasi adalah untuk keperluan kontrol akses (*access control*). Sebuah sistem komputer biasanya hanya diizinkan untuk diakses oleh pihak yang berwenang, namun tidak diizinkan kepada pihak lain. Sehingga, akses kepada sistem biasanya diawali dengan prosedur autentikasi untuk menentukan identitas seorang pengguna. Kemudian baru dapat dilakukan pemberian hak akses kepada pengguna yang telah terotorisasi.

Contoh yang sering ditemukan pada kehidupan sehari-hari adalah sebagai berikut:

a. Login ke komputer lain melewati jaringan internet
b. Mengambil uang dari ATM
c. Menggunakan *internet banking*

Autentikasi yang dianggap kuat (*strong authentication*) didefinisikan sebagai pendekatan autentikasi yang berlapis-lapis dan tergantung kepada dua atau lebih faktor autentikasi untuk menentukan identitas penerima atau pengirim informasi.

Salah satu metode autentikasi yang paling sering digunakan adalah autentikasi berbasis *password*. Pada makalah ini akan dipaparkan proses pengiriman *password* pada media yang tidak aman dengan berbagai serangannya.

2 Faktor Autentikasi

Faktor autentikasi adalah sebuah informasi yang digunakan untuk memverifikasi identitas seseorang untuk kepentingan keamanan.

Tiga jenis faktor autentikasi yang umum digunakan adalah:

a. Sesuatu yang diketahui oleh pengguna
   Contoh: *password*, *passphrase*, dan PIN (*Personal Identification Number*)
b. Sesuatu yang dimiliki oleh pengguna
   Contoh: *ID card*, kartu kredit, telepon seluler, dan perangkat token
c. Sesuatu yang 'ada' pada pengguna
   Contoh: sidik jari, DNA, suara, pola retina, atau aspek biometrik lain.

Sedangkan, beberapa faktor autentikasi lain yang lebih jarang digunakan adalah:

a. Berbasis pengenalan (*recognition*) atau autentikasi cognometric, yaitu sesuatu yang dikenal oleh pengguna
   Contoh: Pengguna harus mengenali dari beberapa wajah yang dirahasiakan.
b. Berbasis *cybermetric*, yaitu sesuai yang ada pada komputer
   Contoh: Membatasi akses hanya dari komputer yang memiliki kombinasi unik hardware dan software tertentu.
c. Berbasis lokasi
   Contoh: Membatasi penggunaan ATM atau kartu kredit hanya pada cabang tertentu, membatasi login root hanya dari terminal tertentu.
d. Berbasis waktu
   Contoh: Membatasi penggunaan sebuah account hanya pada waktu tertentu, misalnya jam kerja.
e. Berbasis ukuran
   Contoh: Membatasi terjadinya transaksi hanya pada sejumlah tertentu saja.

3 Autentikasi berbasis *Challenge-Response*

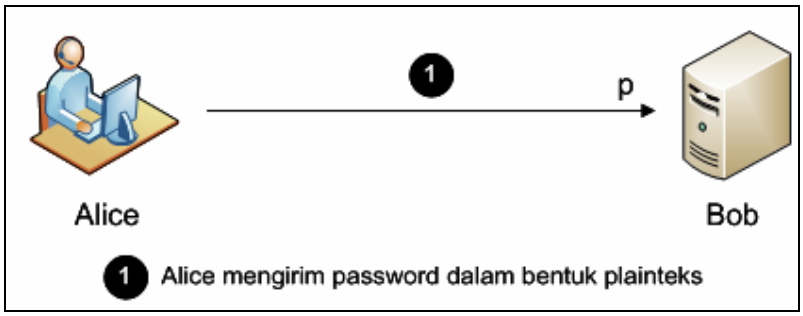
Autentikasi berbasis *challenge-response* (*challenge-response authentication*) adalah salah satu jenis protokol autentikasi. Pada autentikasi berbasis *challenge-response*, salah satu pihak memberikan *challenge* (tantangan), dapat berupa pertanyaan atau nilai acak. Kemudian pihak lain harus memberikan *response* (jawaban) agar dapat diautentikasi.

Bentuk paling sederhana dari protokol *challenge-response* adalah autentikasi dengan *password*. Pada jenis ini, salah satu pihak memberikan pertanyaan untuk meminta *password*, sedangkan pihak lain menjawab dengan memberikan *password* yang valid.

4 Pengiriman *Password* melalui Media Komunikasi dan Permasalahannya

Pada mekanisme autentikasi yang menggunakan *password* sebagai alat identifikasi, maka pada saat proses autentikasi, *password* harus dikirimkan dari pengguna ke sistem. Setelah itu, sistem baru dapat memverifikasi apakah *password* yang dikirimkan valid atau tidak, berdasarkan informasi yang dimiliki oleh sistem.

Dalam skema yang paling sederhana, *password* dikirimkan dari pengguna ke sistem dalam bentuk plainteks. Proses ini dapat dilihat pada Gambar 1. Pada gambar tersebut, Alice sebagai pengguna yang akan diautentikasi dan Bob sebagai server pengautentikasi.



Gambar 1: Pengiriman password (p) dalam bentuk plainteks

Skema seperti ini sangat rentan terhadap berbagai serangan, misalnya penyadapan. Karena *password* dikirim dalam bentuk plainteks, maka penyadap dapat langsung mengetahui *password* Alice.

Serangan ini dapat menjadi lebih berbahaya jika ternyata *password* Alice pada sistem yang lain sama dengan *password*-nya pada sistem ini. Kasus ini umum terjadi karena biasanya pengguna menggunakan satu *password* yang sama untuk mengakses berbagai sistem, dengan alasan kemudahan untuk diingat dan kenyamanan.

Dengan berhasil dicurinya *password* pada sistem ini, maka penyadap dapat menggunakan *password* tersebut untuk mengakses sistem lain atas nama pengguna. Kejadian ini dapat berujung pada berbagai kasus, misalnya transaksi tidak sah, pencurian identitas, maupun kejahatan kriminal dalam dunia *cyber* (*cybercrime*).

Untuk mengatasi pencurian *password* seperti yang terjadi pada skema ini, maka diusahakanlah sebuah skema pengembangan dengan sedikit perbaikan pada data yang dikirim. Skema ini dikembangkan dengan memanfaatkan fungsi hash kriptografis yang sudah ada.

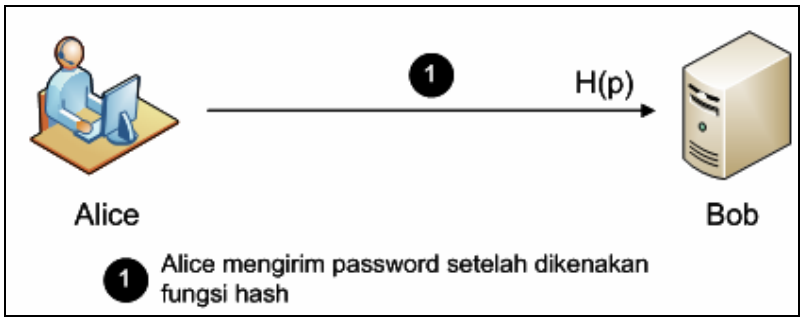
Fungsi hash kriptografis adalah fungsi yang mnerima sebuah string atau pesan dengan panjang yang bervariasi dan kemudian mengembalikan string dengan panjang yang tetap. Hasil keluaran fungsi hash kriptografis sering disebut *message digest* atau *digital fingerprint*.

Karakteristik fungsi hash kriptografis adalah:

a. Diberikan x, maka mudah untuk menghitung H(x)
b. Diberikan H(x), maka sulit secara komputasi untuk menentukan x
c. Diberikan x, maka sulit secara komputasi untuk mencari y, dengan H(x) = H(y)
d. Perubahan pada x menjadi x' walaupun sedikit akan menyebabkan H(x) tidak sama / berbeda jauh dengan H(x')

Saat ini, fungsi hash kriptografis yang paling umum dipakai adalah MD5 dan SHA-1. MD5 (*Message Digest 5*) adalah fungsi hash kriptografis yang menghasilkan keluaran dengan panjang yang tetap sepanjang 128 bit. Sedangkan SHA-1 (*Secure Hash Algorithm 1*) adalah fungsi hash kriptografis yang menghasilkan keluaran dengan panjang yang tetap sepanjang 160 bit.

Pada skema ini, data yang dikirim adalah hasil keluaran dari fungsi hash / *message digest* terhadap *password*. Proses yang terjadi seperti pada Gambar 2.

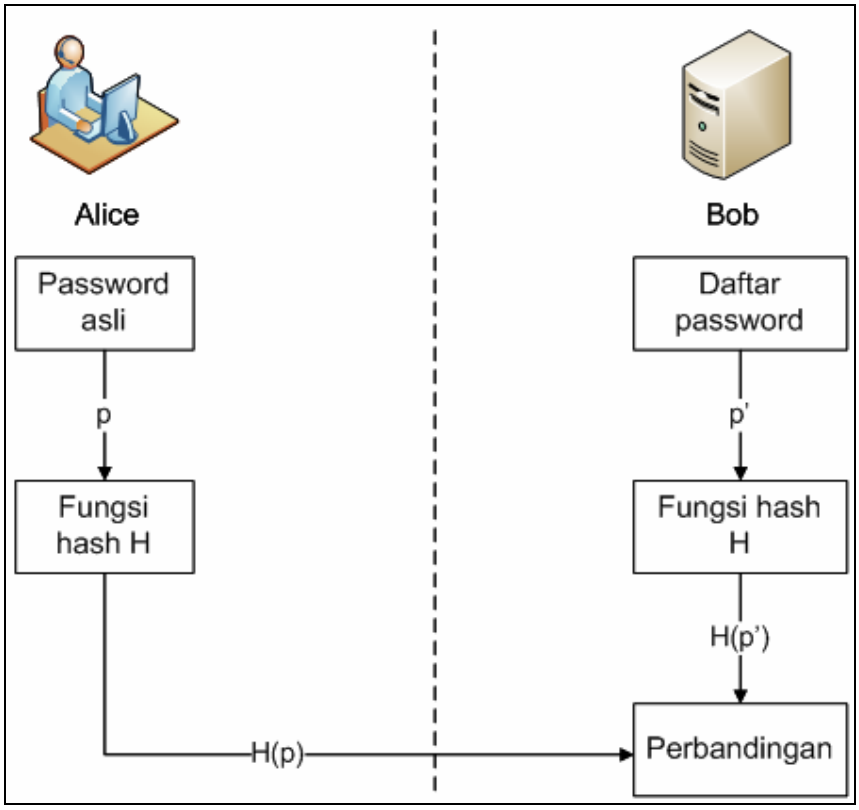


Gambar 2: Pengiriman password (p) dalam bentuk keluaran fungsi hash

Tujuan dari skema ini adalah untuk menyamarkan *password* yang dikirim ke sistem, sehingga jika terjadi penyadapan, tidak akan mudah untuk mendapatkan *password* aslinya. Dengan demikian, secara efektif mengurangi kemungkinan terjadinya pencurian *password*.

Karena Bob hanya menerima *message digest* dari Alice dan bukan *password* aslinya, maka untuk memverifikasinya, Bob harus melakukan hash terhadap *password* yang terdapat pada daftar *password* validnya. Jika hasil keluaran dari fungsi hash ini sama dengan digest yang dikirim oleh Alice, maka dapat disimpulkan bahwa *password*-nya benar dan autentikasinya berhasil.

Proses verifikasi *password* pada skema ini dapat dilihat pada Gambar 3.

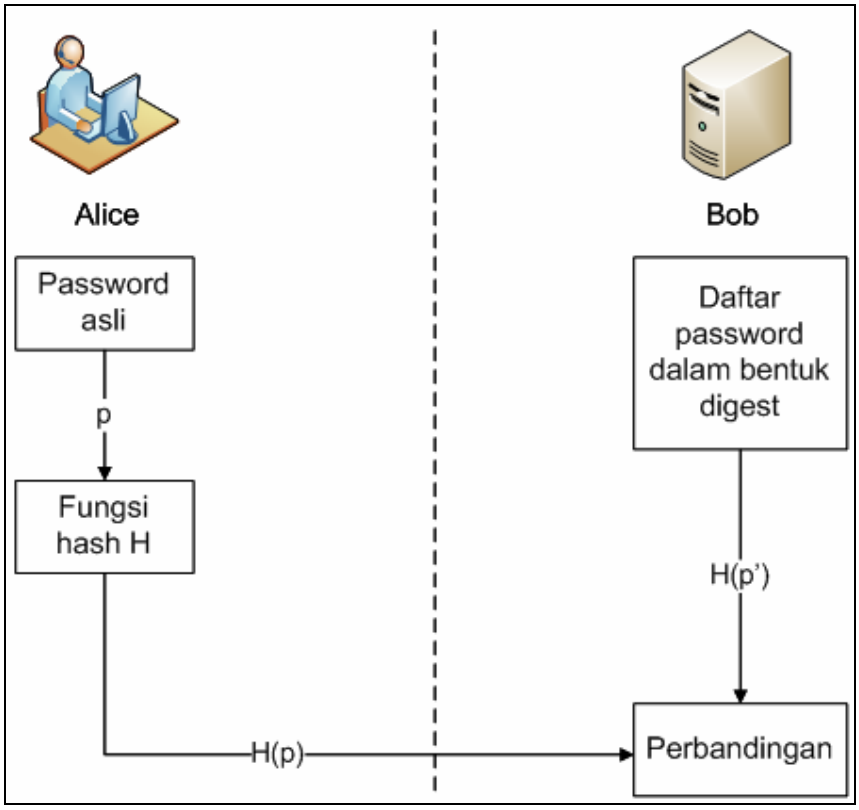


Gambar 3: Verifikasi password yang dikirim dalam bentuk digest

Namun, pada skema ini, Bob harus memiliki daftar *password* asli dari pengguna. Kelemahannya adalah, jika seseorang berhasil menyusup atau mengambil alih Bob, maka penyusup tersebut kemungkinan dapat mengakses daftar *password* milik semua orang.

Oleh karena itu, skema ini dapat diperbaiki dengan cara Bob hanya menyimpan *password* dalam bentuk message digest dan tidak menyimpan dalam bentuk asal.

Proses verifikasi *password* pada skema perbaikan ini dapat dilihat pada Gambar 4.

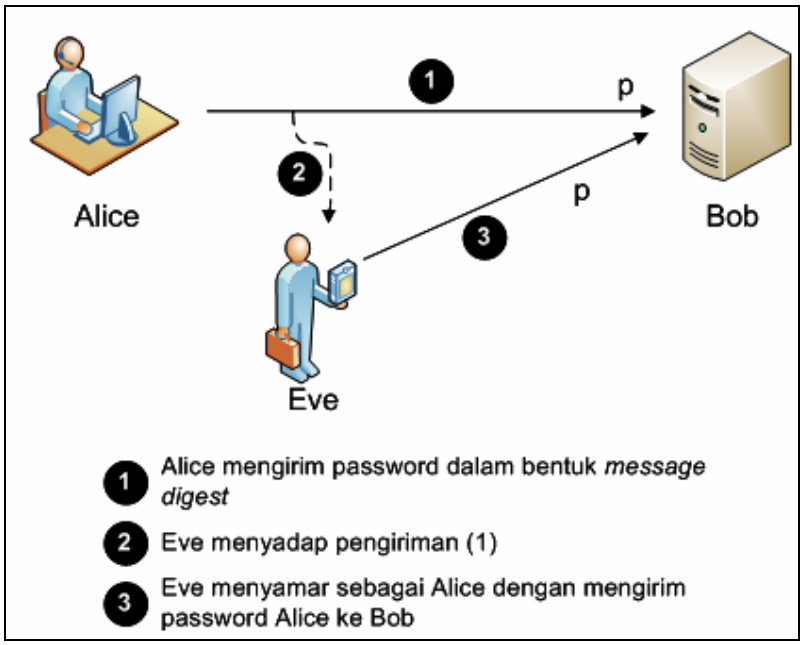


Gambar 4: Verifikasi password yang dikirim dalam bentuk digest; Bob hanya menyimpan digest dari password

Dengan menggunakan skema ini, jika terjadi penyadapan, maka penyadap hanya akan mendapatkan *message digest* dari *password* dan tidak akan mendapat *password* aslinya.

Namun skema ini masih rentan terhadap jenis serangan *replay attack*. *Replay attack* adalah suatu bentuk serangan pada jaringan komunikasi di mana pengiriman data yang valid disadap oleh pihak yang tidak sah, kemudian data tersebut dikirimkan ulang atau ditunda pengirimannya.

Ilustrasi *replay attack* pada skema ini adalah seperti pada Gambar 5.



Gambar 5: Replay attack pada pengiriman password dalam bentuk plainteks

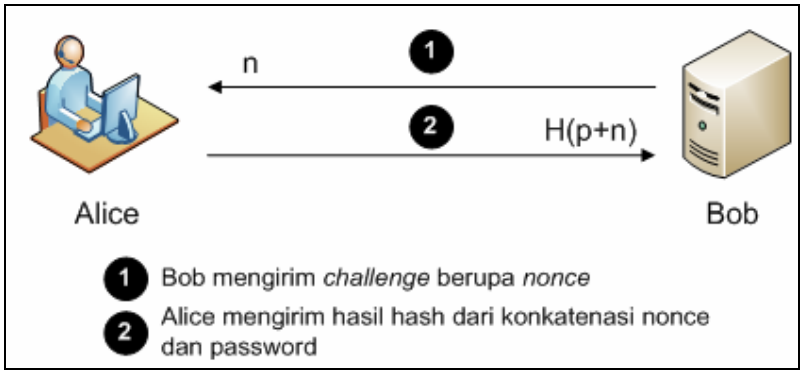
Pada Gambar 5, dapat dilihat bahwa walaupun Eve hanya berhasil mendapatkan *message digest* dari *password* Alice, ia masih dapat mendapatkan akses dari Bob dengan mengirimkan kembali *message digest* tersebut. Bob akan mengira Eve adalah Alice dan verifikasi *password* dianggap berhasil.

Oleh karena itu, dikembangkanlah sebuah skema perbaikan dari skema sebelumnya. Skema ini memanfaatkan *nonce* (*number used once*). *Nonce* adalah bilangan yang hanya digunakan sekali, biasanya berupa bilangan acak (*random*) atau acak-semu (*pseudo-random*) yang dikirimkan sebagai *challenge* oleh pihak yang akan mengautentikasi.

Nonce yang digunakan setiap kali akan berbeda dengan sebelumnya. Ketika menerima *nonce*, pihak yang ingin diautentikasi harus mengirim jawaban sesuai *nonce* yang diterima. Dengan demikian, jawaban dari Alice akan berbeda-beda tergantung *nonce* yang diterimanya. Sehingga, penggunaan nonce akan menghilangkan kerentanan terhadap *replay attack*.

Untuk memastikan agar *nonce* hanya digunakan sekali, maka harus terdapat suatu mekanisme untuk menghasilkan bilangan yang acak. Sehingga, probabilitas untuk menghasilkan bilangan yang sudah pernah dihasilkan sebelumnya sangat rendah. Ini dapat dicapai misalnya dengan memanfaatkan waktu pada sistem.

Skema perbaikan ini dapat dilihat seperti pada Gambar 6.



Gambar 6: Skema autentikasi dengan memanfaatkan nonce

Pada skema ini, Bob mengirimkan *nonce* sebagai *challenge*. Alice kemudian menjawab dengan mengirimkan hasil hash dari *password* Alice yang sudah disambung atau dikonkatenasi dengan *nonce* yang dikirim oleh Bob.

Karena sifat fungsi hash kriptografis yang keluarannya akan berubah drastis begitu masukannya berubah, maka setiap nonce yang berbeda akan menghasilkan keluaran yang berbeda pula ketika disambung dengan *password* asli.

Proses verifikasi *password* pada skema ini dapat dilihat pada Gambar 7.

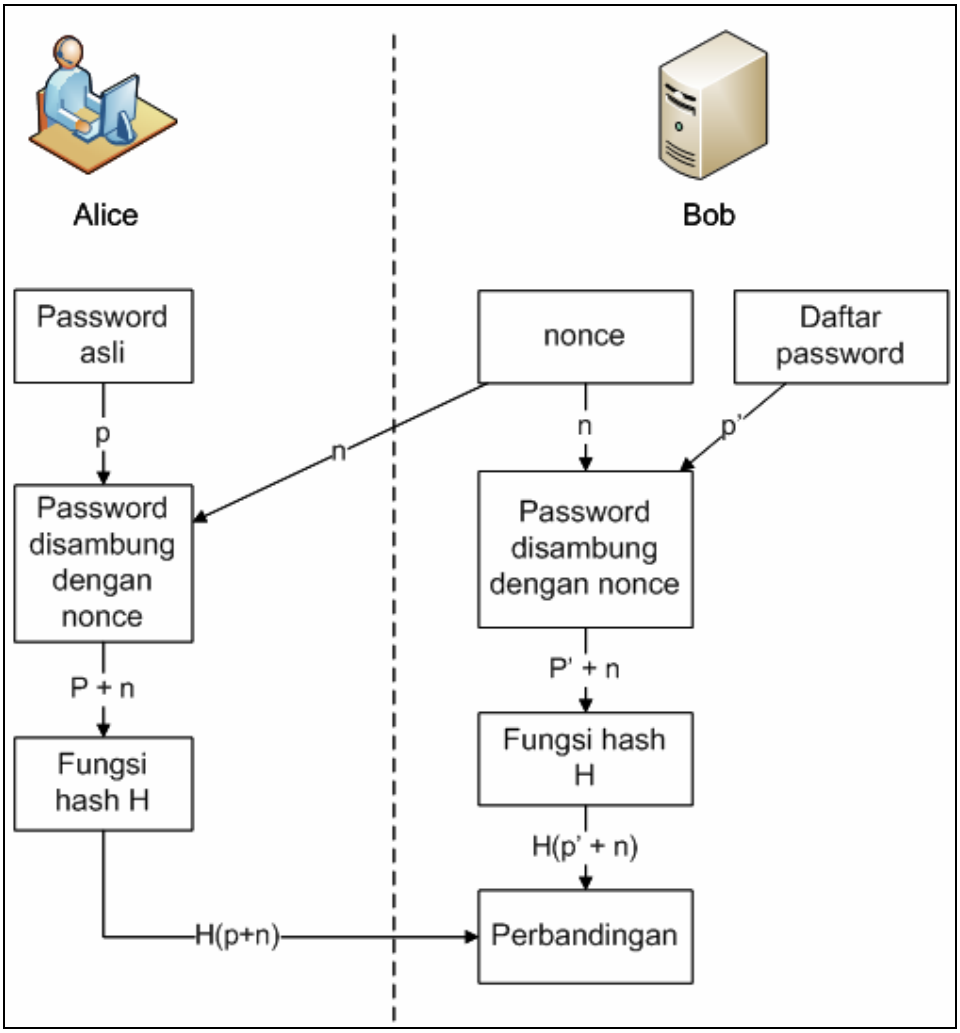
Jika terjadi penyadapan, maka penyadap hanya akan mendapatkan *digest* yang khusus untuk sebuah *nonce* saja dan tidak dapat digunakan untuk nilai *nonce* yang lain.

Skema ini dapat menghilangkan kerentanan terhadap *replay attack*. Namun, skema ini memiliki kelemahan, yaitu Bob harus menyimpan daftar *password* dalam bentuk asli atau plainteks.

Seperti telah dijelaskan sebelumnya bahwa menyimpan *password* dalam bentuk plainteks sangat beresiko jika terjadi penyusupan ke dalam sistem. Jika seseorang berhasil menyusup atau mengambil alih Bob, maka penyusup tersebut kemungkinan dapat mengakses daftar *password* milik semua orang.

Namun, di sisi lain, jika Bob menyimpan daftar *password* dalam bentuk *message digest*, maka skema ini tidak dapat digunakan, karena Bob harus mengetahui *password* dalam bentuk plainteks untuk disambungkan dengan *nonce*.

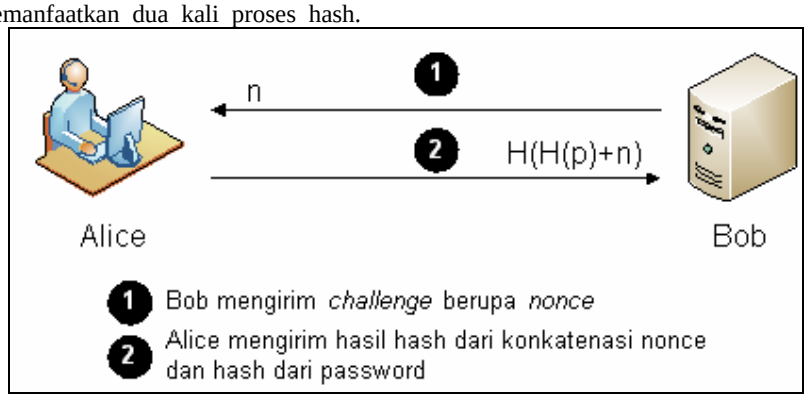
Untuk itu, penulis mengusulkan perbaikan terhadap skema ini dengan skema pada bagian berikutnya.



Gambar 7: Verifikasi password dengan memanfaatkan nonce

5 Usulan Perbaikan pada Skema Pengiriman *Password* dengan *Nonce*

Pada skema perbaikan ini, proses dilakukan dengan memanfaatkan dua kali proses hash. Proses pengiriman yang terjadi dapat dilihat pada Gambar 8.

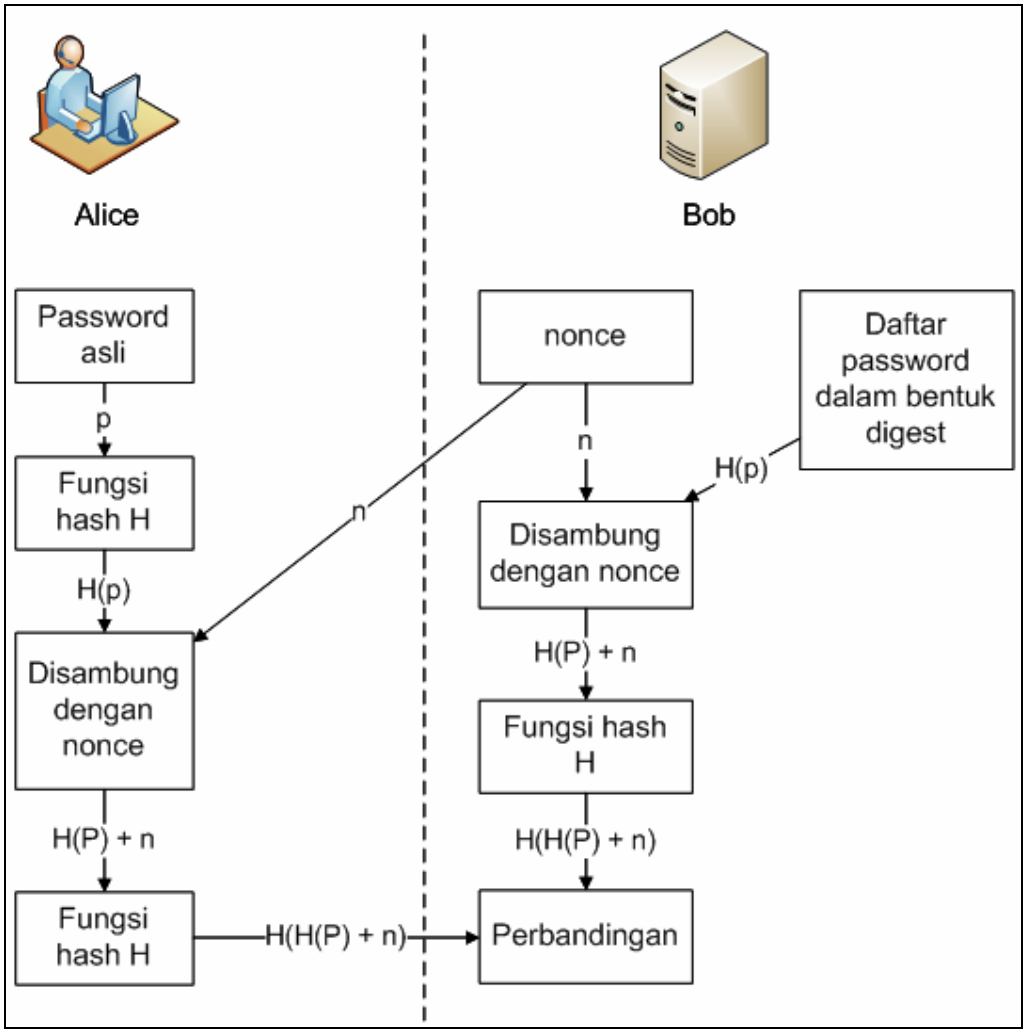


Gambar 8: Skema pengiriman password dengan dua kali proses hash

Pada skema ini, sebelum Alice menyambungkan *password* dengan nonce, *password* tersebut dilewatkan terhadap fungsi hash terlebih dahulu.

Dengan demikian, Bob dapat menyimpan daftar *password* dalam bentuk *message digest*. Ini dapat mengurangi resiko pencurian *password* jika penyusup berhasil memasuki Bob.

Proses verifikasi *password* pada skema ini dapat dilihat pada Gambar 9.



Gambar 9: Verifikasi password pada skema dengan dua kali hash

Pada skema ini, Bob dapat menyimpan daftar *password* dalam bentuk *message digest*. Sehingga mengurangi resiko pencurian *password* apabila Bob disusupi.

Skema ini juga tidak rentan terhadap *replay attack*, karena memanfaatkan nonce.

Secara garis besar, proses yang terjadi pada sisi Alice (pengguna) adalah sebagai berikut:

a. Alice menerima *nonce* dari Bob.
b. *Password* Alice dalam bentuk plainteks diproses dengan menggunakan fungsi hash H, sehingga diperoleh H(p).
c. Hasil keluaran fungsi hash tersebut disambungkan atau dikonkatenasi dengan nonce dari Bob, sehingga diperoleh H(p)+n.

d. Hasil penyambungan tersebut diproses dengan menggunakan fungsi hash, sehingga diperoleh H(H(p)+n).
e. Alice kemudian mengirimkan H(H(p)+n) kepada Bob.

Sedangkan proses verifikasi yang terjadi pada Bob adalah sebagai berikut:

a. Bob melakukan pencarian pada daftar *password* yang tersimpan dalam bentuk digest, kemudian diperoleh H(p).
b. Digest dari *password* ini kemudian disambungkan dengan nonce yang sudah dikirim pada Alice, sehingga diperoleh H(p)+n.
c. Hasil penyambungan tersebut diproses dengan menggunakan fungsi hash, sehingga diperoleh H(H(p)+n).
d. Bob membandingkan hasil perhitungannya dengan H(H(p)+n) yang diperoleh dari Alice. Jika keduanya cocok, berarti *password* yang dikirimkan oleh Alice benar

6 Kesimpulan

Autentikasi berbasis *password* adalah salah satu jenis autentikasi yang paling banyak digunakan.

Pengiriman *password* pada saluran komunikasi yang tidak aman membutuhkan penanganan tersendiri. Ini karena *password* tersebut dapat disadap atau dicuri oleh pihak yang tidak sah.

Oleh karena itu, diusulkanlah suatu skema pengiriman *password* yang aman terhadap serangan penyadapan dan pencurian *password*. Skema ini didesain dengan menggunakan fungsi hash kriptografis yang sudah ada sebelumnya. Proses *hashing* dilakukan sebanyak dua kali terhadap *password* pengguna.

Daftar Pustaka

[1] Schneier, Bruce. *The Failure of Two-Factor Authentication.* http://www.schenier.com. 2005 (diakses pada Desember 2006)

[2] Lamport, Leslie. *Password authentication with insecure communication*. Communications of the ACM: 1981

[3] Wikipedia. *http://en.wikipedia.org*. (diakses pada Desember 2006)

[4] Evans, Arthur Jr. *A User Authentication Scheme Not Requiring Secrecy in the Computer*. Communications of the ACM: 1974