AUTOMATED PASSWORD GENERATOR

Lukman Hakim – NIM : 13503114
Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13114@students.if.itb.ac.id

Abstrak
Password merupakan karakter string yang digunakan untuk autentikasi identitas user dalam sistem komputer atau digunakan untuk autorisasi pengaksesan sistem. Ketika user diperkenankan untuk memilih passwordnya sendiri, mereka sering memilih password dengan mudahnya tanpa melalui pertimbangan. Automated password generator membuat random password yang tidak berkaitan apapun dengan user. Automated password generator adalah algoritma untuk menghasilkan password untuk melindungi data-data komputer. Algoritmanya menggunakan angka acak untuk memilih karakter yang membentuk password acak yang mudah dieja dan diingat oleh user.

Kata kunci: password, random number.

Password merupakan cara autentikasi paling umum yang digunakan untuk mengendalikan pengaksesan informasi mulai dari *personal identification number* atau PIN yang digunakan dalam ATM (*automatic teller machiners)*, kartu kredit, kartu panggilan telepon, sistem mail suara sampai password *alphanumeric* yang lebih kompleks untuk melindungi pengaksesan file, komputer dan jaringan server. Password secara luas digunakan karena mudah dan tidak mahal dalam mekanisme penggunaan dan implementasinya.

Pada saat yang sama, password juga diakui sangat lemah pertahanannya. *Computer Emergency Response Team* (CERT) memperkirakan sekitar 80 persen insiden keamanan yang dilaporkan adalah berkaitan dengan lemahnya pemilihan password. Permasalahan password sangat sulit untuk dikelola karena satu jaringan komputer lokal dapat memiliki ratusan atau ribuan account password (yang dilindungi) dan hanya satu kebutuhan keamanan yang dipertimbangkan yaitu memberikan ruang bagi penyerang (*attacker*) memasuki sistem atau jaringan lokal. Dengan adanya jaringan internet hari ini, permasalahannya menjadi lebih berpotensi untuk penghancuran dalam skala besar; intruder yang mahir dapat dengan mudah masuk ke dalam sistem, bukan untuk merusak tetapi menggunakannya sebagai alat untuk menyerang jutaan target populasi.

Berikut ini merupakan garis pedoman dalam pemilihan dan pengelolaan password dan juga penjelasan mengenai jenis-jenis penyerangan password dan kemungkinan pencegahan serta perbaikannya.

Pedoman Pemilihan Password

Request for Comments (RFC) 1244 menyarankan beberapa pedoman dalam pemilihan dan pemeliharaan password. Pedoman ini telah digunakan dalam berbagai situs keamanan. Pedoman itu antara lain:

- Jangan menggunakan login atau username anda. (as-is, terbalik, kapital, ganda, dan sebagainya)
- Jangan menggunakan nama pertama, tengah atau terakhir.
- Jangan menggunakan nama pasangan (suami atau istri) atau orang terdeket lainnya seperti anak, teman, atau hewan peliharaan
- Jangan menggunakan informasi lain tentang anda yang dimudah ditebak seperti tanggal lahir, nomer plat kendaraan bermotor, nomer telepon, nomer KTP, nomer rumah, dan sebagainya.
- Jangan mengunakan password yang semuanya dari angka saja atau huruf saja.
- Jangan menggunakan kata dalam kamus bahasa, daftar ejaan, daftar akronim atau singkatan, daftar kata lainnya.
- Jangan menggunakan password yang kurang dari enam karakter.

- Jangan memberikan password anda kepada orang lain dengan alasan apapun.
- Jangan menggunakan password yang berisi karakter non-alfabetik (digit dan atau tanda baca)
- Gunakan password yang dengan mudah dingat, sehingga anda tidak perlu untuk menulisnya.
- Gunakan password yang dapat ditulis dengan cepat, tanpa harus melihat ke keyboard.

Terdapat banyak daftar saran tentang password untuk melakukan dan tidak melakukan. Karena terdapat kemiripan, hampir kedua daftar tersebut membuat rekomendasi yang sama. Berikut ini merupakan daftar hal yang harus dihindari dikombinasikan dengan semua rekomendasi umum. Daftar ini telah dimodifikasi untuk beberapa tingkat yang telah memiliki alat untuk memecahkan password. Perulangannya mengindikasikan aturan dengan contoh yang lebih spesifik dari aturan umum sebelumnya. Berikut ini merupakan hal-hal yang harus dipertimbangkan dalam pembuatan password:

Hindari :

1. penggunaan nama account anda atau data apapun yang muncul pada file penyimpanan password.
2. penggunaan kata atau nama apaun yang terdapat di kamus kata atau daftar referensi khususnya hindari penggunaan karakter yang muncul dalam daftar kata yang digunakan dalam penyerangan password atau dalam daftar password buruk.
3. Frase dan logat/ucapan populer dengan atau tanpa spasi.
4. penggunaan metodologis, legendaris, religius atau fiksional baik berupa nama karakter, objek tempat, ras kebangsaan, atau peristiwa.
5. penggunaan akronim.
6. penggunaan alfabetik, angka atau urutan keyboard; banyak urutan huruf tersebut dimasukkan dalam daftar kata yang dapat diserang.
7. judul buku, film, puisi, essay, lagu, CD atau komposisi musikal.
8. merubah-rubah urutan karakter yang didapat dari item apapun sebelumnya dengan metode sebagai berikut:
    a. tambahkan simbol, tanda baca atau dan angka pada sebuah kata.
    b. gunakan kata dengan beberapa atau semua huruf dengan terbalik.
    c. gunakan konjugasi atau plural dari sebuah kata.
    d. gunakan kata-kata dengan menghapus huruf hidupnya.
    e. ganti huruf-huruf dengan simbol atau angka.
        - A → 4
        - a → 2
        - C → (
        - E → [
        - E → {
        - e → 3
        - G → @
        - h → 4
        - I → 1
        - I → !
        - I → |
        - l → 1
        - l → !
        - l → |
        - O → 0
        - S → 5
        - S → $
        - Z → 5
    f. ganti angka-angka dengan huruf atau simbol
        - 0 → O
        - 3 →]
        - 3 → }
    g. Gunakan hanya karakter pertama atau terakhir dalam huruf kecil.
    h. gunakan hanya huruf hidup dalam huruf kecil.
    i. gunakan hanya konsonan dalam huruf kecil.
9. penggunaan informasi yang berkaitan dengan pribadi.
10. penggunaan apapun yang anda dapat membayangkan untuk dikumpulkan dalam sebuah daftar
11. penggunaan password yang diumumkan ke umum dan menyakini bahwa itu merupakan password terbaik.
12. penerjemahan kata-kata dari bahasa lain
13. pengulangan karakter apapun lebih dari sekali.

Lakukan:

- penggunaan paling sedikit delapan karakter
- penggunaan angka atau tanda baca
- penggunaan huruf kecil dan besar.
- Pemilihan frase atau kombinasi kata untuk membuat password lebih mudah diingat.
- Dua kata yang dipisahkan oleh karakter bukan huruf dan angka.
- Tidak memiliki karakter *non-priting*.
- Penggunaan password berbeda pada mesin yang berbeda
- Perubahan password secara regular dan menghindari penggunaan kembali password atau membuat sedikit variasi dengan menambahkan sebuah angka.

Saran yang berulang berasal dari sumber yang berbeda. Kebanyakan user dan beberapa sistem akan memiliki kesulitan yang nyata dengan karakter *non-printing*.

Informasi yang berkaitan dengan pribadi

Kebanyakan orang memilih password yang mudah untuk diingat. Salah satu cara untuk membuat password yang mudah dinngat adalah memilih password yang langsung berkaitan dengan dirinya sendiri. Umumnya yang dihasilkan adalah pilihan password yang lemah. Berikut ini merupakan daftar semua informasi yang berkaitan dengan pribadi yang telah ditemui dalam berbagai password atau dalam daftar kata yang tidak boleh digunakan sebagai password. Daftar ini sangat berguna untuk mempertimbangkan dalam pembentukan password.

- Nama dan inisial
- Nama *account*
- Nama anggota keluarga terdekat
- Nama khusus hewan peliharaan
- Tanggal lahir
- Tanggal lahir anggota keluarga
- Tahun dan model pembuatan kendaraan
- Hobby, ketertarikan, dan kata yang berkaitan dengan hal tersebut.
- Nama pekerjaan
- Nama pekerja
- Kata yang berkaitan dengan pekerjaan
- Nama teman
- nomer atau nama jalan, kota, ibukota, kode pos rumah, kantor atau teman.
- Nomer telepon rumah, kantor, keluarga, atau teman.
- Nomer kartu tanda penduduk dan nomer KTP keluarga terdekat.
- Nomer plat kendaraan
- Tempat tanggal lahir termasuk nama jalan
- Nama universitas atau sekolah.
- Nama kepala sekolah
- Nama sekolah SMU.
- Nomer ID mahasiswa atau pekerja.
- Nomer serial dari produk konsumen.

"Dan permutasi serta kombinasi" dapat ditambahkan untuk setiap *point* yang disebutkan.

Terdapat berbagai macam mekanisme yang dapat digunakan seseorang untuk membuat password tanpa melanggar pedoman ini. Salah satu contohnya adalah mekanisme pemilihan beberapa puisi, lirik lagu, atau dialog terkenal kemudian membuat password dari huruf pertama dari setiap katanya. Contohnya frase *Keadilan sosial bagi seluruh bangsa Indonesia* dapat dibuat menjadi password 5*kes:sbain!*. Password ini terdiri dari campuran tipe karakter seperti huruf, angka dan tanda baca, dan panjanganya lebih dari enam karakter. Skema lainnya adalah mengganti antara satu huruf konsonan dengan satu dua huruf hidup, sampai tujuh atau delapan karakter; ini membuat kata yang aneh tetapi mudah dieja dan juga mudah untuk diingat. Alternatif lainnya adalah memilih dua atau lebih kata pendek kemudian menggabungkannya, dengan menyisipkan tanda baca diantara kedua kata tersebut.

Beberapa contoh penggunaan kata dengan tanda baca. Contoh berikut merupakan pasangan bagus penggunaan kata sebagai password:

- gOt%L0st! - *got lost!*
- heLP4me$ - *help for me (money)*

dan ini merupakan password yang buruk:

- T0gether - *to get her*

Contoh berikut merupakan contoh password baik yang menggunakan frase:

- rsKf0myH - *Raindrops keep falling on my head.*
- wru2rxy? - *Who are you to ask why.*
- bWiIso3! - *Beware the ides of March!*

Berikut merupakan contoh password yang buruk:

- Aaaaaaaa - *Always assert an ambiguous axiom and argue aggressively.*

Beberapa sistem menggunakan program yang dapat secara otomatis menghasilkan password; pada beberapa kasus, program tersebut juga menghasilkan username. Banyak opini pada sistem seperti ini oleh pakar keamanan komputer. Di satu sisi, *random password generator* menghasilkan password yang tidak mudah ditebak atau diserang melalui pendekatan kamus kata. Di sisi lain, password tersebut sulit untuk diingat sehingga user harus menulisnya pada kertas dan dapat menimbulkan permasalahan baru. Sistem ini juga tidak memperkenankan user untuk mengganti passwordnya, tetapi secara periodik akan diberikan password baru secara acak.

Beberapa situs menggunakan program untuk menghasilkan username dan password sekaligus. Username dibentuk dari beberapa fungsi aritmatik dengan input nama lengkap user, nomer pekerja, tanggal kelahiran, dan atau nomer identitas lainnya sehingga user yang memiliki nama lengkap Budi Santoso diberikan username zx2Haqqt. Bentuk ini hanya masalah lain; Budi dipastikan akan menulis baik username dan password pada secarik kertas dan menempelkannya di depan komputer. Dengan begitu sungguh tidak diperlukan penyimanan secara rahasia username karena dapat diketahui dengan segera.

Masalah lain yang berkaitan dengan hal ini adalah setiap orang terlalu banyak memiliki password. Salah satu alasannya karena *World Wide Web* (WWW) yaitu peningkatan jumlah situs yang memiliki username dan password dalam registrasinya. Ketika kebanyakan registrasi ini gratis dan hanya digunakan untuk tujuan pemasaran dibandingkan keamanan, hasilnya adalah pengembangbiakan password. Beberapa user menggunakan password yang sama untuk setiap lokasi; tetapi jika satu password disetujui bersama, begitu pula semuanya. User yang lain memilih password yang berbeda setiap situs, dan akhirnya harus menulis pada media. Hal ini bukan merupakan solusi yang baik.

Untuk itu dalam makalah ini akan mencoba merancang dan mengimplemetasikan sebuah aplikasi password generator yang menghasilkan password yang mudah dieja, diingat dan dikelola sehingga user tidak perlu menuliskan pada media apapun. Sebelum sampai pada pembahasan tersebut, penulis akan menjelaskan kelemahan password dan serangannya.

Kelemahan Password dan Serangannya

Password merupakan bentuk lemah dalam perlindungan dengan berbagai macam alasan. Salah satu alasan utamanya adalah password bergantung pada link terlemah dalam komputer dan jaringan keamanan; yaitu bernama manusia. Kebanyakan user berpikir bahwa prosedur keamanan merupakan bahan tertawaan administrator sistem dan jaringan dan atau disebabkan oleh paranoia. Hasilnya, user tersebut tidak memperhatikan dan mempertimbangkan dalam pemilihan password yang akan melindungi mereka.

Terdapat beberapa cara intruder untuk menyerang password (yang dilindungi) sistem. Cara serangan yang paling umum adalah *password guessing* (menebak-nebak). Seseorang sering memilih nama , username, nomer telepon, atau beberapa variasinya sebagai password mereka. Kemudian, mereka memilih nama anggota keluarga atua teman, hewan peliharaan, benda kesukaan atau beberapa variasinya. Dan bagaimana penyerang (attacker) menemukan informasi ini? Dalam banyak kasus, hal ini mudah. Kegunaan finger, sistem keamanan terkenal memiliki keamanan lemah yang menunggu untuk dimanfaatkan, yang menampilkan status semua user yang sedang online lengkap dengan username, salah satu item informasi yang seorang penyerang tidak dapat berbuat tanpanya. Daftar finger juga menampilkan nama lengkap user; file PLAN.TXT dan PROJECT.TXT sering menambahkan informasi personal yang dengannya seorang intruder dapat melakukan password guessing attack, yaitu informasi tetang login terakhir.

Hal ini juga mengejutkan bagaimana banyak situs memilih password untuk beberapa account atau tidak merubah setting pembuatanya pada beberapa *account*. Dalam sistem digital VAX/VMS, contohnya, account SYSTEM dan FIELD didapat dari password yang terdefisini sebelumnya yaitu MANAGER dan SERVICE. Pengelola sistem dan pelayanan pribadi memberitahu administrator sistem untuk mengubah passsword dan men-*disable*-nya ketika tidak digunakan.; tindakan pencegahan sederhana ini sering diabaikan. Banyak sistem menambahkan account GUEST dengan tanpa password tetapi tidak secara keras membatasi kapabilitas *account* tersebut.

Pertahanan yang normal melawan seranggan password guessing adalah sebuah fitur yang disebut *blacklisting,* yang membatasi jumlah usaha login berturut-turut yang tidak berhasil. Dalam implementasinya, counter login diset menjadi nol setelah login sukses dan ditambahkan setelah usaha login tidak sukses. Jika counter mencapai blacklist threshold (biasanya antara 3 sampai 7), account login akan *disable* meskipun password yang benar dipenuhi.

Intruder dapat menggunakan blacklisting sebagai basis bentuk penyerangan lainnya. Meskipun jika intruder tidak dapat memasuki ke dalam sistem, penyerang dapat secara efektif menyangkal layanan ke user dengan blacklist attack, dimana attacker dapat secara efekti men-*disable* banyak (atau semua) user dengan tujuan agar memblacklisting mereka. Untuk pencegahan level sistem account dari diblacklist oleh attacker, kebanyakan sistem operasi memperbolehkan login ke account sistem dari konsole operator tanpa memperhatikan status blacklist account.

Kemungkinan penyerangan kedua adalah mencuri file password sistem, sesuatu yang sangat luar biasa dilakukan jika file tidak diberikan perlindungan akses secara benar. Ketika kebanyakan password selalu disimpan dalam beberapa file yang terenkripsi atau dalam bentuk hash, maka masih rentan terhadap penyerangan melalui *dictionary attack*, dimana jumlah kata dalam jumlah besar dienkripsi menggunakan skema enkripsi sistem operasi sebagai usaha untuk menemukan kecocokan dalam file password. Beberapa penyelidikan menyarankan bahwa terdapat 99 persen kesempatan sukses *cracking* minimal satu password dalam sebuah file yang berisi paling tidak 16 password. Dengan prosesesor berkecepatan tinggi hari ini yang tersedia di desktop pada harga wajar, maka setiap orang dengan *spell checker* dapat melakukan *dictionary attack*.

Oleh karena itu, hal ini menjadi sesuatu yang penting untuk dicatat bahwa panjang password bukan faktor utama dalam menentukan apakah password itu bagus atau tidak. Kebanyakan user hari ini masih memilih password yang berisi hanya huruf kecil, kebanyakan pula sering membentuk kata atau string dari kata-kata. Tipe password tersebut adalah paling banyak mudah diserang dengan *dictionary attack*.

Bentuk serangan lainnya disebut *login spoofing*, dan dapat dengan sukses terjadi ruangan terminal umum pada insititusi pendidikan. Pada skenarionya, *attacker* menjalankan sebuah program yang menampilkan kemunculan pesan login. Ketika user lain mecoba untuk login, program membuat queri seperti biasa untuk username dan password, menulis informasi tersebut pada sebuah file, menampilkan pesan "invalid login", dan kemudian mencatatnya. User hanya berpikir bahwa program tersebut membuat kesalahan typographical, sehingga user mencoba kembali login sampai berhasil. Penyerangan ini sering dapat bekerja dengan baik dan jika beruntung attacker menemukan user yang memiliki level tertinggi dalam sistem *privilige*nya.

Serangan keempat adalah memantau aktivitas antara user dan komputer. Jika penyerangan ini digunakan, attacker dapat menemukan username dan password dalam plain teks. Dalam jaringan lokal, bentuk penyerangan ini membutuhkan akses fisik untuk membuka komunikasi atau *wiring*, pada internet, *intruder* hanya memantau paket yang digunakan dalam Telnet, WWW, atau *account* password lainnya.

Setelah mendapatkan legitimasi username dan password, penyerang dapat melawan dengan *replay attack*, dimana penyerang mengirimkan kembali informasi autentikasi yang valid kepada sistem target untuk memasukinya. Sistem apapun yang menggunakan identifikasi tetap dan atau informasi autentikasi dapat dengan mudah diserang dengan cara seperti itu.

Sistem *Bellcore's S/KEY*™ dirancang untuk meng-counter seperti halnya *replay attack*. Dalam S/KEY, seorang user memilih *passphrase* dari algoritma terkenal yang menghasilkan sejumlah password sederhana. Setiap password yang dihasilkan merupakan sebuah kata yang panjangnya satu sampai empat huruf dan setiap password tergantung pada penciptaan password sebelumnya. Ketika user mencoba untuk login ke host, host tersebut mengeluarkan penolakan berdasarkan password terakhir yang digunakan oleh user; klien membalas dengan password dalam rentetan. Dengan skema ini, intruder dapat menebak atau menghitung password berikutnya. Implementasi beberapa kompatibel S/KEY mulai digunakan secara luas di internet.

Pendekatan termudah untuk mendapatkan password adalah perbuatan yang termudah pula.

Attacker belajar password user lainnya yang dengan mudah menanyakan kepada mereka, baik melalui e-mail, telepon atau dalam ruangan *chatting online*. Dengan mengaku sebagai "*network security officer*", seorang penyerang akan menanyakan user untuk passwordnya untuk tujuan verifikasi. Meskipun semua adminstrator sistem telah memberitahu user bahwa mereka tidak akan pernah menanyakan password mereka dengan cara seperti ini, tetapi beberapa user masih ada yang membocorkan dan memberitahukan passwordnya tanpa berpikir dua kali. Intruder juga dapat menghubungi pengelola sistem sebagai user yang lupa passwordnya dan menanyakan kepada pengelola sistem untuk medapatkan password yang baru; permintaan seperti itu tidak pernah dipenuhi tanpa mengidentifikasi pemanggil.

Alternatifnya, intruder mengirimkan email pada target user untuk memberitahukan mereka bahwa terdapat penerobosan keamanan dan mereka harus mengubah password mereka untuk alasan keamanan. Banyak user, berpikir bahwa mereka harus melakukan terbaik bagi sistem sehingga mereka memenuhi permintaan tersebut.

Dengan mengetahui bentuk-bentuk penyerangan ini dapat diketahui bahwa bentuk password yang bagus tidak menentukan apakah apakah perlindungan itu bagus atau tidak tetapi minimal dapat mencegah dari penyerang yang menggunakan *dictionary attack*. Pembahasan berikutnya adalah tentang berapa banyak karakter yang dibutuhkan untuk sebuah password yang aman.

Ketika hampir semua sistem komputer menyimpan password dalam bentuk terenkripsi, maka password menjadi kunci menuju sistem kriptografi. Sistem kriptografi menyediakan beberapa keamanan peningkatan ukuran kunci, yang menyarankan password akan lebih aman apabila lebih panjang ukurannya. Terdapat beberapa kebenaran pada penelitian ini. Bagaimanapun juga password yang lebih panjang itu tidak sekuat ketika dibandingkan dengan password yang lebih pendek. Hal ini karena pembatasan yang ditentukan oleh beberapa sistem komputer dan cara tersebut tidak dipakai oleh user untuk memilih password mereka.

Pertimbangkan contoh berikut. Kebanyakan sistem Unix membatasi password dengan panjang sampai delapan karakter, atau 64 bits. Tetapi Unix hanya menggunakan tujuh signifikan bits pada setiap karakter sebagai kunci enkripsi, mengurangi ukuran kunci sampai menjadi 56 bits. Tetapi meskipun hal ini tidak sebaik dengan yang dapat terlihat karena 128 kemungkinan kombinasi dari tujuh bit tiap karakter tidak sama; user biasanya tidak menggunakan pengendali karakter atau karakter non-alfanumerik dalam password mereka. Kenyataannya, kebanyakan user hanya menggunakan huruf kecil dalam password mereka. (dan beberapa sistem pasword menerapkan case-sensitif pada beberapa kasus). Pada kamus kata bahasa inggris dengan delapan huruf memiliki informasi tentang 2.3 bits tiap huruf, menghasilkan 18.4 bit panjang kunci untuk delapan karakter yang dibentuk dari kamus kata bahasa inggris. Banyak orang memilih namanya sebagai password dan ini menghasilkan informasi yang lebih rendah sekitar 7.8 bits untuk input dengan panjang nama delapan huruf. Apabila lebih panjang dari itu, maka setiap pertambahan huruf hanya menambah sekitar 1.2 sampai 1.5 bits tiap informasi, artinya bahwa password menggunakan 16 huruf dari frase kata bahasa inggris hanya menghasilkan 19 sampai 24 bit kunci, tidak mendekati pada apa yang diharapkan.

Tabel 1 Sejumlah kemungkinan kunci dengan berbagai macam panjang password dan kumpulan karakter.

Kumpulan karakter	Panjang password				
	4-octet	5-octet	6-octet	7-octet	8-octet
huruf kecil (26)	$4.6x10^5$	$1.2x10^7$	$3.1x10^8$	$8.0x10^9$	$2.1x10^{11}$
huruf kecil / angka (36)	$1.7x10^6$	$6.0x10^7$	$2.2x10^9$	$7.8x10^{10}$	$2.8x10^{12}$
karakter alfanumerik (62)	$1.5x10^7$	$9.2x10^8$	$5.7x10^{10}$	$3.5x10^{12}$	$2.2x10^{14}$
Karakter cetak (95)	$8.1x10^7$	$7.7x10^9$	$7.4x10^{11}$	$7.0x10^{13}$	$6.6x10^{15}$

karakter ASCII 7-bit (128)	$2.7x10^8$	$3.4x10^{10}$	$4.4x10^{12}$	$5.6x10^{14}$	$7.2x10^{16}$
karakter ASCII 8-bit (256)	$4.3x10^9$	$1.1x10^{12}$	$2.8x10^{14}$	$7.2x10^{16}$	$1.8x10^{19}$

Tabel 1 menunjukkan sejumlah kemungkinan kunci yang dihasilkan dengan 4, 5, 6, 7, dan 8 octet password berbeda yang konstrain dengan inputnya. Tabel dua menunjukkan sejumlah waktu yang dibutuhkan untuk melakukan pencarian exhaustive untuk semua kemungkinan kunci dengan prosesor yang dapat menguji satu juta kunci tiap detiknya. Dapat disimpulkan bahwa meskipun password yang lebih panjang memiliki perlindungan yang lebih baik dibandingkan yang lebih pendek, password yang menggunakan kombinasi yang lebih luas dari kemungkinan kombinasi bit lebih baik dan memiliki konstrain yang tinggi.

Tabel 2 Jumlah waktu untuk pencarian semua kemungkinan kunci (dengan satu juta kunci setiap detik)

Kumpulan karakter	Panjang password				
	4-octet	5-octet	6-octet	7-octet	8-octet
huruf kecil (26)	0.5 det	12 det	5.2 mnt.	2.2 jam	2.4 hari
huruf kecil / angka (36)	1.7 det.	1 mnt.	36.7 mnt.	21.7 jam	32.4 hari
semua karakter alfanumerik (62)	15 det.	15 mnt.	15.8 jam	40.5 hari	7 tahun
karakter cetak (95)	1.4 mnt.	2.1 jam	8.6 hari	2.2 tahun	209 tahun
karakter ASCII 7 bit (128)	4.5 mnt.	9.4 jam	50.9 hari	17.8 tahun	2283 tahun
karakter ASCII 8-bit (256)	1.2 jam	12.7 hari	8.9 tahun	2283 tahun	570,776 tahun

Tabel ini menunjukkan mengapa sebuah rahasia yang memiliki 64 bits acak umumnya lebih aman karena tidak mungkin melakukan komputasi untuk mencari 2^{64} kemungkinan kunci. Dan berapa banyak karakter yang dibutuhkan untuk menghasilkan kunci 64 bit ?

- Jika karakter acak dari kumpulan alphanumerik digunakan, password dengan panjang 11 karakter akan dibutuhkan. Sayangnya, user tidak menyukai untuk mengingat karakter string acak dengan panjang 11 karakter
- Jika password dapat dieja dipilih maka setiap karakter berkontribusi sekitar 4 bits pada ukuran kunci, sehingga panjang passwordnya adalah 16 karakter. Hal ini juga terlalu panjang bagi seseorang untuk diingat.
- Jika seseorang memperkenankan memilih password mereka sendiri, *conventional wisdom* berpendapat bahwa setiap karakter berkontribusi hanya 2 bits, sehingga password harus memiliki panjang 32 karakter. Hal ini juga terlalu panjang.

Sehingga apa yang bisa disimpulkan? Apakah ada keberhasilan password pada kasus ini – kebanyakan orang akan mengingat dan menulis pada basis regular tidak sebaik angka acak 64 bit. Oleh karena itu password akan dapat mudah untuk diserang dengan cara menerkanya.

Meskipun password memiliki bentuk lemah dalam perlindungannya tetapi kesederhanaanya membuat mudah untuk digunakan dan dikelola. Jika user mempercayakan komitmennya, pendidikan yang pantas disediakan, dan sedikit kepedulian, password dapat memiliki perlindungan yang memadai. Sebagai catatan pula bahwa password merupakan sebuah bentuk "apa yang anda ketahui" keamanan, meskipun dapat mudah diserang ketika digunakan sendiri, password tersebut juga kuat ketika menggunakan kombinasi dengan "apa yang kamu miliki"

(misalnya kartu identifikasi) atau sistem "apa kamu itu" (misalnya pindai tangan atau suara) .

Administrator sistem dan jaringan harus membuat kebijakan dan prosedur untuk keamanan situsnya, termasuk administrasi password. User harus diingatkan tentang kebijakan tersebut, kemudian memotivasi mereka, dan menerima konsekuensi dari semua itu.

Hal ini penting karena mengingat kesuksesan tersebut tidak membutuh perhatian terhadap serangan password, dengan ratusan bahkan ribuan komputer di internet yang masing-masing memiliki ratusan atau ribuan *user account*, intruder yang berpengalaman hanya butuh sedikit titik masuk sukses untuk menyebabkan kerusakan yang signifikan.

Karakter apa yang harus muncul pada password yang bagus

Pembahasan sebelumnya mengasumsikan bahawa password terdiri dari huruf besar dan kecil serta angka. Apa yang terjadi jika kumpulan karakter ini diturunkan atau dinaikkan? Tabel berikut ini menunjukkan beberapa pilihan dari password dengan panjang delapan karakter:

Table 3 Password dengan panjang delapan karakter

Jenis Password	Jumlah karakter	Jumlah kemungkinan password	Waktu cracking
7-bit ASCII	128	72057594037927936	350 tahun
karakter cetak	95	6634204312890625	33 tahun
huruf dan angka	62	218340105584896	satu tahun
hanya huruf	52	53459728531456	96 dari
huruf kecil dengan satu huruf besar	26/special	1670616516608	tiga hari
huruf kecil saja	26	208827064576	sembilan jam
kata bahasa inggris: delapan karakter atau lebih	special	250000	kurang dari satu detik

Hal ini jelas bahwa semakin banyak kumpulan karakter digunakan, akan semakin sulit untuk memecahkan password tersebut. User harus berusaha untuk memasukan minimal huruf besar dan kecil dan jika memungkinkan, user harus juga memasukkan beberapa angka, simbol tanda baca dan atau kode kontrel dalam passwordnya.

Goal administrator

Tujuan jelas dari seorang administrator adalah menjaga sistemnya dari para intruder. Jika hal ini tidak memungkinkan, maka paling itdak membuat intruder bekerja sangat keras untuk memasuki sistem.

Pembahasan berikut ini berkaitan dengan usaha yang dibutuhkan untuk memecahkan password baik dihasilkan secara random atau menggunakan suatu pola atau kumpulan pola yang terdifinisi jelas dan intruder memiliki informasi yang dibutuhkan untuk membangun kamus kata bentukan yang mencocokkan dengan pola password pada situs yang menjadi tergat penyerangan.

Potensi intruder tidak pernah mengetahui kebijakan situs dalam membuat password, dan khususnya pola passwordnya jika sebuah pola didasarkan pada password generator digunakan untuk menghasilkan atau menyarankan password user; jika potensial intruder memiliki informasi seseorang yang bocor maka apa yang harus menjadi informasi yang rahasia. Hal ini sangat banyak ditemukan di banyak situs yang tidak memperkenankan siapapun mengetahui apapun mengenai kebijikan passwordnya. Sebuah situs yang memiliki password kuat di lapangan, akan kehilangan beberapa keuntungan dari kebijakan ini, jika rincian pengetahuan kebijakan tersebut diperbolehkan organisasi luar.
Sebuah situs yang memiliki kebijakan password, harus menginformasikan semua user tentang

kebijakannya, salah satu hal yang user tidak boleh mendiskusikan sistem password atau kebijakan keamanan dengan siapapun di luar organisasi. User harus diinformasikan tentang pelanggaran dari kebijakan tersebut dapat menghasilkan dalam aksi disiplin mulai dari pengurangan privasi sistem sampai terminasi jika tepat.

Tidak ada keuntungan lebih bagi sebuah situs dibandingkan memiliki seorang crakcer yang bekerja dengan asumsi yang salah berkaitan tentang panjang password atau kumpulan karakter, karena berbagai rintangan menghadang crakter menjadi lebih besar dibandingkan dengan apa yang mereka harapkan. Dengan kata lain, ketika berbagai rintangan tersebut dari cracking password normal adalah baik dan sangat kecil, jika cracker mencoba password yang terlalu pendek atau berisi variasi karakter yang sedikit, mereka tidak pernah mendapat password apapun.

Password yang kuat sebenarnya akan memiliki panjang dan tipe karakter yang beragam serta struktur yang beragam pula yang jika intuder memiliki ide cantik bagaimana sebuah situs membuat password, intruder tidak harus berusaha keras dengan menambahkan kecepatan komputasi untuk memecahkan password dalam waktu yang cepat.

Fungsi random

Software kriptografi telah banyak digunaka contohnya seperti Kerberos, PEM, PGP, dan sebagainya telah dewasa dan menjadi bagian dari jaringan. Sistem tersebut menyediakan perlindungan yang substansial dari pengintai (snooping) dan spoofing. Bagaimanapun juga, masih terdapat potensi kelemahan. Pada jantung dari semua sistem kriptografi adalah pembuatan sesuatu yang rahasia angka yang tidak dapat ditebak.
Kekurangan dari fasilitas yang tersedia umumnya dalam pembuatan angka yang tidak dapat diprediksi merupakan kesempatan untuk diperbaikan dalam rancangan software kriptografi. Untuk developer yang ingin membangun sebuah prosedur pembuatan kunci atau password yang berjalan pada hardware yang luas, strategi yang paling aman sejauh ini telah dilakukan pada instalasi lokal untuk menambahkan prosedur yang cocok untuk menghasilkan angka acak.

Hal ini penting untuk diingat bahwa kebutuhannya untuk data yang musuh memiliki kemungkinan sangat kecil untuk menebak atau menemukannya. Hal ini akan gagal jika data pseudo-random digunakan yang hanya menemukan pengujian statistik tradisional atau yang didasarkan pada sumber terbatas seperti jam. Seringkali kuantitas random dapat ditentukan oleh musuh yang mencari melalui kemungkinan ruang-ruang kecil.

Penggunanan paling banyak dalam bilangan acak adalah password user yang merupakan karakter string sederhana. Jelas, jika sebuah password dapat ditebak, maka tidak menyediakan keamanan. (untuk password yang digunakan kembali, ini yang diperlukan user untuk mengingat password. Hal ini dapat menggunakan karakter string yang dapat dieja atau frase kata yang digabung pada kata-kata sehari-hari. Tetapi ini hanya mempengaruhi format informasi password, tidak ada kebutuhan yang sangat banyak password tersebut dapat ditebak.)

Banyak kebutuhan lainnya datang dari arena kriptografi. Teknik kriptografi dapat digunakan untuk menyerdiakan berbagai macam layanan termasuk *confidentiality* dan autentikasi. Layanan tersebut didasarkan pada kuantitas, secara tradisional disebut “keys”, yang tidak diketahui dan tidak mudah ditebak oleh seorang musuh.

Pada beberapa kasus, seperti penggunaan enkripsi simetrik dengan *one time pads* atau US Data Encryption Standar (DES), sekelompok yang menginginkan komunikasi yang *confidentialilly* dan atau dengan autentikasi harus semuanya mengetahui kunci rahasia yang sama. Pada kasus lain, penggunaan teknik kriptografi asymetrik atau “public key”, kunci menjadi berpasang-pasang. Satu kunci merupakan pasangan privat dan harus disimpan oleh satu kelompok, kunci lainnnya sebagai publik dan dapat dipublikasikan ke dunia. Perhitungan komputasi tidak mudah dilakukan untuk menentukan kunci privat dari kunci publik.

Frekuensi dan volume kebutuhan kuantitas random berbeda-beda untuk sistem kriptografi yang berbeda. Penggunaan RSA, kuantitas random dibutuhkan ketika pasangan kunci dihasilkan, tetapi kemudian angka apapun dari pesan dapat ditandai tanpa ada kebutuhan randomisasi. Kunci publik Digital Signature Algorithm yang diusulkan oleh US National

Institute of Standars and Technology (NIST) menyerdiakan angka random baik untuk setiap tanda tangan. Dan enkripsi dengan *one time pad*, yang secara prinsip merupakan teknik terkuat enkripsi, membutuhkan volume randomisasi yang sama untuk semua pesan untuk diproses.

Pada kebanyakan kasus ini, musuh dapat mencoba untuk menentukan kunci rahasia dan cara *trial and error*. (Hal ini mungki sepanjang kunci cukup kecil dibandingkan pesannya sehingga kunci dapat diidentifikasi). Kemungkinan musuh berhasil harus dibuat serendah-rendahnya, tergantung dari aplikasinya. Ukuran ruang pencarian musuh berkaitan dengan jumlah kunci "informasi" yang ditampilkan dalam teori informasi. Hal ini tergantung pada jumlah nilai perbedaan rahasia yang mungkin dan kemungkinan setiap nilai sebagai berikut:

Bits-of-info = $\sum -p_i * \log_2 (p_i)$

Dimana i memiliki nilai dari 1 sampai kemungkinan nilai rahasia dan p_i adalah probabilitas nilai ke i. (karena p_i lebih kecil, maka log akan negatif setiap penjumlahan akan di non-negatif-kan.

Jika terdapat 2^n nilai berbeda dari probabilitas, maka n bits informasi ditampilkan dan musuh harus mencoba setengah nilai atau 2^{n-1} , sebelum menebak quantitas rahasia. Jika probabilitas nilai berbeda tidak sama, maka terdapat informasi yang berkurang dan akan ada sedikit penebakan, yang dibutuhkan oleh musuh. Nilai apapun yang musuh dapat memungkinkan diketahui, atau memiliki probabilitas rendah, dapat diabaikan oleh musuh, yang akan mencari melalui kemungkinan nilai lainnya.

Contohnya, pertimbangan sistem kriptografi yang menggunakan kunci 56 bit. Jika kunci 56 bit diturunkan dari penggunaan generator angka pseudorandom diumpankan dengan 8 bit umpan, maka musuh mudah untuk mencari hanya melalui 256 kunci (dengan menjalankan generator angka pseudo-random dengan setiap kemungkinan umpan), bukan kunci 2^{56} yang muncul pertama pada kasus ini. Hanya delapan bits dari informasi pada kunci 56 bit ini.

Standar *Key Generation*

Beberapa standar umum ditempatkan untuk menghasilkan kunci. Dua diantaranya dijelaskan sebagai berikut yaitu US DoD dan X9.17. Keduanya menggunakan DES tetapi memiliki kekuatan yang sama bahkan lebih kuat apabila digabungkan.

1. Rekomendasi US DoD untuk pembuatan kunci

   Departemen Pertahanan Amerika Serikat memiliki rincian rekomendasi bagi pembuatan password [DoD]. Departemen tersebut menyarangkan menggunakan Data Encription Standard [DES] dalam Output Feedback Mode [DES MODES] sebagai berikut:

   gunakan initialization vector ditentukan dari
   jam sistem,
   ID sistem,
   ID user,
   tanggal dan waktu;
   gunakan sebuah kunci ditentukan dari
   interupsi register sistem
   status register sistem, dan
   counter sistem, dan
   sebagai plain teks, gunakan secara acak untuk menghasilkan 64 bit atau 8 karakter.

   Password dapat dihitung dari 64 bit "chiper text" dihasilkan dalam 64-bit Output Feedback Mode. Sebanyak bits yang dibutuhkan dapat diambil dari 64 bits ini dan diperluas kedalam kata yang dapat dieja, frase kata, format lain jika user menginginkan untuk mengingat password.

2. X9.17 Pembuatan kunci

   Institut Standar Nasional Amerika telah menetapkan sebuah metode untuk menghasilkan urutan kunci sebagai berikut:
   s_0 : inisial 64 bit umpan
   g_n : urutan hasil 64 bit kunci
   k : kunci acak yang disediakan untuk menghasilkan urutan kunci ini
   t : waktu ketika sebuah kunci dihasilkan dalam resolusi baik (sampai 64 bits)
   DES (K,Q) : enkripsi DES dari Q dengan kunci K.

   $g_n = DES (k, DES (k, t) \ .xor. \ s_n)$

   $s_{n+1} = DES (k, DES (k, t) \ .xor. \ g_n)$

   Jika g menggantikan n digunakan sebagai sebagai kunci DES, maka setiap kedepalan bit harus diatur untuk parity yang digunakan tetapi 64 bit masukan tidak memodifikasi g

yang harus digunakan untuk menghitung s berikutnya.

Perancangan Automated Password Generator

APG dibangun dengan menggunakan bahasa pemograman C. Program ini juga telah digunakan dalam sistem Unix untuk menghasilkan angka acak yang dibutuhkan oleh password generator tetapi fungsi acaknya digantikan dengan fungsi acak berbasis DES dalam mode ECB (Electronic Code Book). DES memiliki input yaitu password lama atau karakter string dari user dan sebuah kunci pseudorandom. Perubahan sekecil apapun terhadap kunci atau input data string akan menyebabkan DES menghasilkan angka acak yang berbeda. Setiap kali perubahan ini terjadi, password generator menghasilkan password acak yang baru.

Implementasi Automated Password Generator

Automated Password Generator terdiri dari prosedur utama yang memiliki tiga komponen utama yaitu tabel unit, tabel digram, dan prosedur angka acak. Random password generator bekerja untuk membentuk suka kata ejaan dan menggabungkannya sehingga membentuk sebuah kata. Aturan-aturan pengejaan disimpan dalam sebuah tabel pada setiap unit dan setiap pasangan unit-unit (digram). Aturan tersebut digunakan untuk menentukan apakah unit yang diberikan legal atau ilegal yang didasarkan pada posisi unit terhadap suku kata dan unit yang berdekatan. Kebanyakan aturan tersebut dan pemeriksaanya berorientasi ejaan dan tidak bergantung pada ejaan selainnya.

Kode implementasi automated password generator terdiri dari prosedur angka acak DES, prosedur aktual DES, dan kode untuk menghasilkan kunci psedorandom. Dalam implementasi password generator, nilai yang dipilih untuk dua kunci DES dan input bagi random number generator dapat dibaca di dalam kode. Sebenarnya implementasi nilai kunci dan masukannya harus rahasia yang secara acak dihasilkan oleh aplikasi.

Tabel Unit

Tabel unit menentukan unit (karakter alfabetik) dan menjelaskan aturan penyinggungan pada individu unit yang digunakan dalam menghasilkan kata secara acak. Contohnya lokasi huruf mati (vowel) pada kata yang dihasilkan ditentukan oleh aturan ini.

Kode dalam bahasa C sebagai berikut

```
static struct unit  rules[] =
{{"a", VOWEL},
 {"b", NO_SPECIAL_RULE},
 {"c", NO_SPECIAL_RULE},
 {"d", NO_SPECIAL_RULE},
 {"e", NO_FINAL_SPLIT | VOWEL},
 {"f", NO_SPECIAL_RULE},
 {"g", NO_SPECIAL_RULE},
 {"h", NO_SPECIAL_RULE},
 {"i", VOWEL},
 {"j", NO_SPECIAL_RULE},
 {"k", NO_SPECIAL_RULE},
 {"l", NO_SPECIAL_RULE},
 {"m", NO_SPECIAL_RULE},
 {"n", NO_SPECIAL_RULE},
 {"o", VOWEL},
 {"p", NO_SPECIAL_RULE},
 {"r", NO_SPECIAL_RULE},
 {"s", NO_SPECIAL_RULE},
 {"t", NO_SPECIAL_RULE},
 {"u", VOWEL},
 {"v", NO_SPECIAL_RULE},
 {"w", NO_SPECIAL_RULE},
 {"x", NOT_BEGIN_SYLLABLE},
 {"y", ALTERNATE_VOWEL | VOWEL},
 {"z", NO_SPECIAL_RULE},
 {"ch", NO_SPECIAL_RULE},
 {"gh", NO_SPECIAL_RULE},
 {"ph", NO_SPECIAL_RULE},
 {"rh", NO_SPECIAL_RULE},
  {"sh", NO_SPECIAL_RULE},
  {"th", NO_SPECIAL_RULE},
  {"wh", NO_SPECIAL_RULE},
  {"qu", NO_SPECIAL_RULE},
  {"ck", NOT_BEGIN_SYLLABLE}
};
```

Tabel Digram

Tabel digram menentukan aturan-aturan mengenai semua kemungkinan pasangan unit dan pendekatan unit yang digunakan. Tabel digram berisi satu input untuk setiap pasangan

unit (digram) dan diuji, apakah pasangan tersebut diperkenankan atau tidak.
Random word generator memastikan aturan aturan yang ditetapkan dalam tabel digram dipenuhi untuk setiap dua unit berturutan dalam pembentukan kata.

Sebagian kode dalam bahasa C sebagai berikut:

```
static  int   digram[][RULE_SIZE]
=
{
  {/* aa */ ILLEGAL_PAIR,
   /* ab */ ANY_COMBINATION,
   /* ac */ ANY_COMBINATION,
   /* ad */ ANY_COMBINATION,
   /* ae */ ILLEGAL_PAIR,
   /* af */ ANY_COMBINATION,
   /* ag */ ANY_COMBINATION,
   /* ah */ NOT_BEGIN | BREAK |
NOT_END,
   /* ai */ ANY_COMBINATION,
   /* aj */ ANY_COMBINATION,
   /* ak */ ANY_COMBINATION,
   /* al */ ANY_COMBINATION,
   /* am */ ANY_COMBINATION,
   /* an */ ANY_COMBINATION,
   /* ao */ ILLEGAL_PAIR,
   /* ap */ ANY_COMBINATION,
   /* ar */ ANY_COMBINATION,
   /* as */ ANY_COMBINATION,
   /* at */ ANY_COMBINATION,
   /* au */ ANY_COMBINATION,
   /* av */ ANY_COMBINATION,
   /* aw */ ANY_COMBINATION,
   /* ax */ ANY_COMBINATION,
   /* ay */ ANY_COMBINATION,
   /* az */ ANY_COMBINATION,
   /* ach */ ANY_COMBINATION,
   /* agh */ ILLEGAL_PAIR,
   /* aph */ ANY_COMBINATION,
   /* arh */ ILLEGAL_PAIR,
   /* ash */ ANY_COMBINATION,
   /* ath */ ANY_COMBINATION,
   /* awh */ ILLEGAL_PAIR,
   /* aqu */ BREAK | NOT_END,
   /* ack */ ANY_COMBINATION},
  {/* ba */ ANY_COMBINATION,
   /* bb */ NOT_BEGIN | BREAK |
NOT_END,
   /* bc */ NOT_BEGIN | BREAK |
NOT_END,
   /* bd */ NOT_BEGIN | BREAK |
NOT_END,
   /* be */ ANY_COMBINATION,
   ……..
```

Prosedur Algoritma

1. periksa apakah variabel minlen > maxlen. Jika iya maka terjadi kesalahan.
2. periksa kata yang memiliki panjang nol. Secara teknik ini bukan kesalahan sehingga hanya mengembalikan kata null dengan panjang nol.
3. Temukan password:
   a. Inisialisasi array yang menyimpan unit kata. Ketika diketahui panjang kata, hanya dibutuhkan salah satu panjang. Metode ini lebih baik digunakan pada array yang statik, karena memperkenankan fleksibilitas pemilihan panjang kata yang berubah-ubah. Karena sebuah kata dapat berisi satu ejaan, harus dibuat unit ejaan, array tersebut memiliki unit analog untuk tiap ejaan dengan panjang sama. Tidak ada aturan eksplisit yang membatasi panjang ejaan, tetapi aturan digram dan heurististik melakukannya secara tidak lansung.
   b. Temukan ejaan sampai kata masukan terbangun.
      i. Dapatkan ejaan dan tentukan panjangnya.
      ii. Gabungkan unit ejaan dengan unit kata
      iii. Jika kata tersebut tidak tepat dibentuk, keluarkan ejaan tersebut. Pemeriksaan yang dilakukan disini adalah kata yang harus dibentuk dalam basis kata. Pengujian lainnya dilakukan sepenuhnya didalam ejaan tersebut. Sebaliknya, gabungkan ejaan tersebut pada kata dan gabungkan ejaan tersebut pada hasil kata.
      iv. Periksa kata tersebut tidak berisi kombinasi ilegal yang dapat menjangkau ejaan. Khususnya, yaitu:
         1. sebuah pasangan ilegal dari unit-unit antara ejaan.
         2. tiga unit vowel berturutan.
         3. tiga unit konsonan berturutan.
      v. Modifikasi silabel untuk angka atau simbol kapital yang dibutuhkan. Dilakukan setelah kualitas kata diperiksa.
      vi. Tetap menjaga tracking waktu yang telah dicoba untuk mendapatkan ejaan. Jika melebihi threshold, maka inisialisasi kembali variabel pwlen dan word_size, bersihkan array kata, dan mulai dari awal.

Prosedur utamanya sebagai berikut:

```
int gen_pron_pass (char *word, char
*hyphenated_word,     USHORT     minlen,
USHORT maxlen, unsigned int pass_mode)
{
    int     pwlen;
    if (minlen > maxlen || minlen >
        APG_MAX_PASSWORD_LENGTH ||
         maxlen > APG_MAX_PASSWORD_LENGTH)
       return (-1);

    if (maxlen == 0)
    {
     word[0] = '\0';
     hyphenated_word[0] = '\0';
     return (0);
    }

    pwlen     =     gen_word     (word,
        hyphenated_word,        get_random
        (minlen, maxlen), pass_mode);
    return (pwlen);
}
```

Random Number Generator

Random number generator menggunakan prosedur DES untuk menghasilkan nilai presisi antara 0 dan 1. Angka ini digabungkan oleh variabel sebuah program n yang bertipe integer. Operasi ini menghasilkan integer acak antara 0 dan (n-1). Angka acak yang dibuat oleh rutin DES menghasilkan output yang nantinya digunakan sebagai input bagi random word generator. Prosedur ini menghasilkan sejumlah angka yang akan dipanggil oleh word generator setiap waktu sebuah karakter (unit) dibutuhkan.

Tidak semua karakter yang dihasilkan akan diterima oleh word generator pada setiap posisi dalam kata. Setiap karakter diperiksa untuk ketepatan dan kesesuainnya dengan aturan yang didefisinikan oleh tabel unit dan digram. Oleh karenanya prosedur random number generator akan dipanggil berulang sampai menghasilkan karakter yang dapat diterima. Pemanggilan dibatasi sampai 100 kali. Jika telah mencapai nilai 100 maka kata tersebut ditolak dan program memulai kembali dari awal.

Distribusi unit yang legal berbeda untuk setiap posisi bagian kata, untuk setiap unit, tergantung pada unit-unit sebelumnya menurut tabel unit dan digram. Prosedur random number sendiri tidak melakukan pengunjian unit yang telah legal.

DES menerima input dua 64 bits blok data. Salah satunya terdiri dari password lama atau string data; lainnya merupakan 64 bit (56 bits + 8 parity bits) kunci pseudorandom yang diturunkan dengan menggunakan prosedur standar. Password lama dimasukkan secara manual dari keyboard. Input array dibuat dari bytes pertama dari delapan karakter password atau input string. Program akan menerima string null (carraige return). Karakter kedelapan diabaikan. Jika input blok kurang dari delapan karakter maka akan ditambahkan elemen ekstra dengan ASCII 0. DES mode ECB digunakan untuk mengenkripsi input data. Outputnya adalah angka acak 64 bit yang dienkripsi bentuk inputnya.

Fungsi pertama dalam struktur DES adalah setkey(), yang mengkonversi kunci psedorandom menjadi sebuah format yang digunakan DES dalam melakukan enkripsi. Input bagi fungsi setkey pada command line adalah (0,0, key). Nol yang pertama diset sehingga setkey() tidak menghasilkan pariti; nol yang kedua memberitahui setkey() bahwa enkripsi (bukan dekripsi) dibutuhkan.

Kunci yang digunakan merupakan sebuah pointer pada awal array kunci. Setelah setkey() berakhir, fungsi des() dipanggil. Untuk inputnya menggunakan alamat input dan array output. Kedua input dan output didefinisikan sebagai array karakter asli dengan panjang 8 bytes.

Hasil keluaran array, dikirim ke sebuah fungsi, answer(), yang mengembalikan angka final yang dibutuhkan. Fungsi answer() berada pada alamat dari array output sebagai pointer char unsignetdan integer n yang memiliki nilai 0 sampai (n-1) yang dibutuhkan oleh program kata acak. Fungsi ini membuat variabel sum, didefinisikan sebagain integer asli. Untuk mendapatkan nilai angka dari keluar array karakter, tambahkan nilai ASCII pada tiga elemen pertama dalam keluaran array dan menyimpan penjumlahannya pada variabel sum. Sum = out[0] + out[1] + out[2], bertipe integer. Untuk mendapatkan sebuah angka dengan rentang dari 0 sampai n-1 dari penjumlahan, fungsi tersebut mencari modulus sum dan n (sum%n). Nilai ini kemudian dikembalikan pada fungsi pemanggil dalam program random word.

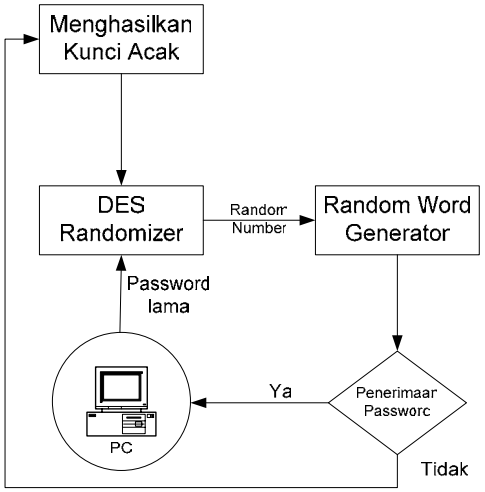
Algoritma Kata Acak

Algoritma tersebut yang digunakan untuk menghasilkan kata acak sudah tetap dan tidak dapat dimodifikasi tanpa perubahan logik dari program. Fungsi dari algoritma tersebut digunakan untuk menentukan apakah unit yang diberikan, dihasilkan oleh prosedur unit acak, dapat digabungkan pada akhir kata yang dibentuk sejauh ini. Aturan pengejaan disimpan dalam tabel unit dan digram dijelaskan sebelumnya. Aturan tersebut digunakan untuk memeriksa jika unit yang diberikan legal atau illega. Jika illegal, unit ditolak dan prosedur unit acak dipanggil kembali. Ketika unit diterima, berbagai macam status variabel diupdate dan sebuah unit untuk posisi berikutnya dalam kata dicoba.

Password yang dibuat oleh automated password generator ini terdiri dari 26 karakter alfabet inggris. Meskipun angka-angka dan karkater spesial tidak dimasukkan, ruang password, yang merupakan sebuah fungsi jumlah karakter dalam password sangatlah besar. Diperkirakan 18 juta 6-karakter, 5.7 miliyar 8-karakter, dan 1.6 triliun 10-karakter password dapat dibuat oleh program tersebut. User harus memilh sebuah ruang password sepadan dengan level keamanan yang dibutuhkan bagi informasi agar terlindungi.

Algoritma password tersebut tidak menghalangi menghasilkan kata-kata yang ditemukan dalam kamus kata bahasa inggris. Jika dibutuhkan, kamus kata terkomputerisasi harus digunakan untuk memeriksa kata bahasa inggris dan implementasinya harus memasukan pengujian test untuk mencegah dari tawaran kepada user sebagai password.

Gambar 1 merupakan diagram blok implementasi algoritma automated password generation. Program ini menggantikan fungsi acak angka Unix dalam versi aslinya dengan ‘DES Randomizer” dan fungsi “Generate Random Key”. DES randomiser menerima password lama dan kunci pseudorandom dan menghasilkan sebuah angka acak. Angka ini digunakan oleh generator huruf acak untuk membentuk sebuah password. Setelah password telah dihasilkan oleh setiap kelompok huruf kemudian diuji grammar dan semantiknya untuk menentukan jika kata yang diterima telah dibuat. Jika semua ujian dilalui, password baru dikeluarkan ke PC.



Gambar 1 Diagram blok implementasi automated password generate

Dalam implementasi program, nilai minlen dan maxlen (yaitu panjang password), diset menjadi 5 dan 8. User yang membutuhkan password dengan panjang tetap harus memset variabel ini pada nilai yang spesifik.

Daftar Pustaka

[1] Australian Computer Emergency Response Team (AusCERT). Choosing good passwords. AusCERT Reference # GoodPasswords, February 1, 2001. http://www.auscert.org.au/render.html?it=2260 (diakses Desember 2006).

[2] Ganesan, R. and Davies, C. A New Attack on Random Pronounceable Password Generators. Proceedings of the 17th {NIST}-{NCSC} National Computer Security Conference, 1994.

[3] Gasser, M. A Random Word Generator for Pronounceable Passwords. Technical Report ESD-TR-75-97, Electronic Systems Division, Hanscom Air Force Base, 1975.

[4] Schneier, Brute. Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons, 1994.