Indice TESINA DI GIOVANNI DELL'ESAME DI STATO

Introduzione

La mia passione per i computers e l'informatica nacque molto presto, fin da piccolo riuscivano a scaturire in me un grande interesse e una grande curiosità, ma mi limitai ad usarli come un semplice utente.
L'interesse e la curiosità verso queste macchine iniziò ad accrescere con l'adolescenza. Ricordo ancora quando sentivo al telegiornale, alla radio, o leggevo sui giornali di storie di questi
hackers, definiti pirati informatici, che si intrufolavano nei computers di governi e agenzie segrete, tutte azioni illegali che però mi intrigavano e affiscinavano moltissimo.
Un giorno poi decisi di capire meglio la struttura, il pensiero di questa sottocultura, iniziai così a navigare su internet alla ricerca di qualche "sito da hacker".
Colgo l'occasione per sfatare il luogo comune degli hackers. L'hacker viene definito e visto erratamente come un pirata, una persona dannosa per la società, immagine che si è diffusa largamente a causa dei Mass Media.
Analizziamo invece l'importanza fondamentale che hanno avuto queste persone per lo sviluppo tecnologico degli ultimi decenni. E' grazie a loro infatti che oggi possiamo usare un computer, navigare su internet, e usufruire di tutte le comodità di cui disponiamo oggi; andiamo dunque a vedere la loro nascita storica, la loro filosofia, il loro pensiero.

L'origine del termine Hacker

A prescindere dall'ampiezza della definizione, la maggioranza degli odierni hacker ne fa risalire l'etimologia al MIT, dove il termine fece la sua comparsa nel gergo studentesco all'inizio degli anni '50. Stendere una vecchia carcassa fuori dalla finestra del dormitorio veniva considerato un "hack", ma altre azioni più pesanti o dolose - ad esempio, tirare delle uova contro le finestre del dormitorio rivale, oppure deturpare una statua nel campus - superavano quei limiti. Era implicito nella definizione di "hack" lo spirito di un divertimento creativo e innocuo.
È a tale spirito che s'ispirava il gerundio del termine: "hacking". Uno studente degli anni '50 che trascorreva gran parte del pomeriggio chiacchierando al telefono o smontando una radio, poteva descrivere quelle attività come "hacking".
Più avanti negli anni '50, il termine "hack" acquistò una connotazione più netta e ribelle. Al MIT degli anni '50 vigeva un elevato livello di competizione e l'attività di hacking emerse sia come reazione sia come estensione di una tale cultura competitiva. Goliardate e burle varie divennero tutt'a un tratto un modo per scaricare la tensione accumulata, per prendere in giro l'amministrazione del campus, per dare spazio a quei pensieri e comportamenti creativi repressi dal rigoroso percorso di studio dell'istituto. Va poi aggiunto che quest'ultimo, con la miriade di corridoi e tunnel sotterranei, offriva ampie opportunità esplorative per quegli studenti che non si facevano intimorire da porte chiuse e da cartelli tipo "Vietato l'ingresso". Fu così che "tunnel hacking" divenne l'accezione usata dagli stessi studenti per indicare queste incursioni sotterranee non autorizzate. Grazie ad esperimenti casuali ma accurati, gli studenti impararono a fare scherzi divertenti.
La combinazione tra divertimento creativo ed esplorazioni senza limiti costituirà la base per le future mutazioni del termine hacking. I primi ad auto-qualificarsi "computer hacker" nel campus del MIT negli anni '60 traevano origine da un gruppo di studenti appassionati di modellismo ferroviario, che negli ultimi anni '50 si erano riuniti nel Tech Model Railroad Club. Una ristretta enclave all'interno di quest'ultimo era il comitato Signals and Power (segnali ed elettricità), gli addetti alla gestione del sistema del circuito elettrico dei trenini del club. Un sistema costituito da un sofisticato assortimento di relè e interruttori analogo a quello che regolava il sistema telefonico del campus. Per gestirlo era sufficiente che un membro del gruppo inviasse semplicemente i vari comandi tramite un telefono collegato al sistema, osservando poi il comportamento dei trenini.
Adottando il termine hacking, iniziarono così a raffinarne ulteriormente la portata. In maniera sottile, il termine hacking si trasformò da sinonimo di gioco ozioso, a un gioco in grado di migliorare le prestazioni o l'efficienza complessiva del sistema ferroviario del club. Quanto prima i membri di quel comitato cominciarono a indicare con orgoglio l'attività di ricostruzione e miglioramento del circuito per il funzionamento delle rotaie con il termine "hacking", mentre "hacker" erano quanti si dedicavano a tali attività.
Considerata la loro affinità per i sistemi elettronici sofisticati - per non parlare della tradizionale avversione degli studenti del MIT verso porte chiuse e divieti d'ingresso - non ci volle molto prima che gli hacker mettessero le mani su una macchina appena arrivata al campus. Noto come TX-0, si trattava di uno dei primi modelli di computer lanciati sul mercato. Sul finire degli anni '50, l'intero comitato Signals and Power era emigrato in massa nella sala di controllo del TX-0, portandosi dietro lo stesso spirito di gioco creativo. Il vasto reame della programmazione informatica avrebbe portato a un ulteriore mutamento etimologico. "To hack" non indicava più l'attività di saldare circuiti dalle strane sembianze, bensì quella di comporre insieme vari programmi, con poco rispetto per quei metodi o procedure usati nella scrittura del software "ufficiale". Significava inoltre migliorare l'efficienza e la velocità del software già esistente che tendeva a ingolfare le risorse della macchina. Ed è qui che successivamente si colloca una diversa radice del termine hacker, la forma sostantiva del verbo inglese "to hack" che significa "tagliare", "sfrondare", "sminuzzare", "ridurre", "aprirsi un varco", appunto fra le righe di codice che istruiscono i programmi software. Un hacker era quindi uno che riduceva la complessità e la lunghezza del codice sorgente, con un "hack", appunto, una procedura grossolana ma efficace, che potrebbe essere tradotta in italiano come "zappata" o "accettata" (tagliata con l'accetta) o altrimenti con una "furbata".
Il fatto che la successiva generazione di programmatori, incluso Richard Stallman, aspirasse a seguire le orme dei primi hacker, non fa altro che testimoniare le prodigiose capacità di questi ultimi. Nella seconda metà degli anni '70 il termine "hacker" aveva assunto la connotazione di élite. In senso generale, computer hacker era chiunque scrivesse il codice software per il solo gusto di riuscirci. In senso specifico, indicava abilità nella programmazione. Definire hacker un collega programmatore costituiva un segno di rispetto.
Con il restringimento della definizione, l'attività di computer hacking acquistò nuove connotazioni semantiche. Per potersi definire hacker, una persona doveva compiere qualcosa di più che scrivere programmi interessanti; doveva far parte dell'omonima cultura e onorarne le tradizioni allo stesso modo in cui un contadino del Medio Evo giurava fedeltà alla corporazione dei vinai. Pur se con una struttura sociale non così rigida come in quest'ultimo esempio, gli hacker di istituzioni elitarie come il MIT, Stanford e Carnegie Mellon iniziarono a parlare apertamente di "etica hacker": le norme non ancora scritte che governavano il comportamento quotidiano dell'hacker. Nel libro del 1984 "Hackers. Gli eroi della

rivoluzione informatica", l'autore Steven Levy, dopo un lungo lavoro di ricerca e consultazione, codificò tale etica in cinque principi fondamentali.
A partire dai primi anni '80 i computer presero a spuntare un po' ovunque, e i programmatori che una volta dovevano recarsi presso grandi istituzioni o aziende soltanto per aver accesso alla macchina, improvvisamente si trovarono a stretto contatto con hacker di grande livello. Grazie a questa vicinanza, i comuni programmatori presero ad appropriarsi delle filosofie anarchiche tipiche della cultura hacker di ambiti come quello del MIT. Tuttavia, nel corso di un simile trasferimento di valori andò perduto il tabù culturale originato al MIT contro ogni comportamento malevolo, doloso. Mentre i programmatori più giovani iniziavano a sperimentare le proprie capacità con finalità dannose, creando e disseminando virus, facendo irruzione nei sistemi informatici militari, provocando deliberatamente il blocco di macchine quali lo stesso Oz del MIT, il termine "hacker" assunse connotati punk, nichilisti. Quando polizia e imprenditori iniziarono a far risalire quei crimini a un pugno di programmatori rinnegati che citavano a propria difesa frasi di comodo tratte dall'etica hacker, quest'ultimo termine prese ad apparire su quotidiani e riviste in articoli di taglio negativo. Nonostante libri come quello di Levy avessero fatto parecchio per documentare lo spirito originale di esplorazione da cui nacque la cultura dell'hacking, per la maggioranza dei giornalisti "computer hacker" divenne sinonimo di "rapinatore elettronico". Contro l'originale definizione da questo momento si insinua nella conoscenza popolare l'uguaglianza Hacker-Malvivente.
Anche di fronte alla presenza, durante gli ultimi due decenni, delle forti lamentele degli stessi hacker contro questi presunti abusi, le valenze ribelli del termine risalenti agli anni '50 rendono difficile distinguere tra un quindicenne che scrive programmi capaci di infrangere le attuali protezioni cifrate, dallo studente degli anni '60 che rompe i lucchetti e sfonda le porte per avere accesso a un terminale chiuso in qualche ufficio. D'altra parte, la sovversione creativa dell'autorità per qualcuno non è altro che un problema di sicurezza per qualcun altro. In ogni caso, l'essenziale tabù contro comportamenti dolosi o deliberatamente dannosi trova conferma a tal punto da spingere la maggioranza degli hacker ad utilizzare il termine cracker - qualcuno che volontariamente decide di infrangere un sistema di sicurezza informatico per rubare o manomettere dei dati - per indicare quegli hacker che abusano delle proprie capacità.

La mia crescita - Prima Parte

Quando iniziai altro non ero che un newbie, termine in gergo informatico col quale si indicano i novizi, i novellini. Ero ancora un semplice ed innocente utente, una persona che usava il computer per svolgere attività banali e quotidiane, nulla di più.
Il primo sito che frequentai, fu molto importante per me, in quanto mi indirizzò verso la strada giusta, era un forum gestito da ragazzi esperti di informatica, programmatori, loro si definivano degli hackers.
Impossibile non ricordare l'ammirazione che provavo per queste persone, io non ero ancora nessuno, e ognuno di loro, anche il meno bravo, mi sembrava un Dio.
E' brutto da dire, ma è così, quando si inizia ad addentrarsi in questo mondo, si pensa di aver capito la sua filosofia, la sua visione, ma è solo un illusione. Iniziai così super interessato a leggermi tutorial, semplici guide che illustravano metodologie di attacco a computer Windows. Il tutto mi affascinava moltissimo, tanto da tenermi incollato allo schermo per interi pomeriggi alla ricerca di un qualcosa, nella speranza di apprendere, capire, diventare uno di loro.
Ben presto la semplice curiosità, il semplice interesse iniziò a trasformarsi in una ossessione, un qualcosa che era in grado di rapirmi, farmi perdere la cognizione del tempo, estraniarmi dal mondo, da tutto ciò che mi circondava. Iniziai così a passare le prime notti al computer, le prime eccitanti notti dell'inizio di una seconda vita: la mia camera diventava quasi magica, un luogo di enorme concentrazione, ispirazione; il buio incombeva sulla stanza, una luce soffusa usciva dal monitor, silenzio, l'unico rumore presente era quello della ventola del mio computer, e delle
mie mani che battevano freneticamente sulla tastiera. Una atmosfera magica, spirituale.
La prima guida che lessi, parlava della vulnerabilità del NetBios, servizio offerto dai computer di casa Microsoft. Con questo servizio, se abilitato su una macchina, era possibile accedervi da remoto, ed avere accesso alle periferiche condivise. Con una serie di comandi, era possibile entrare nel computer con il NetBios attivato, e se si aveva accesso in scrittura, si potevano anche creare e modificare file sul suo disco rigido.
Era la prima guida che leggevo, e già pensavo di essere un hacker. Solo ora capisco con quale stupidità agivo e pensavo, una stupidità però dettata dall'ignoranza e dalla presunzione.
Col passare del tempo, anche il mio stile di vita cambiò, quasi drasticamente. Fino alle medie, ero sempre stato un ragazzo preciso, ordinato e diligente. Iniziai a diventare disordinato, disordine che tuttavia mi risultava ordinato, sembra un gioco di parole, ma era e tuttora è proprio così. Iniziai a studiare sempre meno per la scuola, perchè dedicavo tutto il tempo che avevo al computer, a studiare, a programmare, a capire. Una semplice e innocua passione si era trasformata in una ossessione tale da cambiarmi la vita. Le notti le passavo fino a tardi sulla mia macchina, la mattina a scuola ero quindi sempre meno concentrato a causa della mia vita sregolata.
Non me ne stavo accorgendo, ma stavo diventando uno di loro. Il primo linguaggio di programmazione che studiai, fu il Perl, un famoso e molto utilizziato linguaggio di scripting. Leggevo la guida con grande diligenza, tanto che mia madre una sera pensò a voce alta: "se studiasse così per la scuola, sarebbe il primo della classe..".
Leggevo quel manuale con velocità, tutto mi pareva molto logico, le strutture portanti del Perl le imparai così molto rapidamente.
Nacquero così i miei primi programmini, ricordo ancora l'entusiasmo che provai nel vedere il mio primo programma funzionare. Era molto banale, il codice molto semplice. Non faceva altro che registrare in memoria l'ora precisa all'avvio del programma, e dopo due minuti diceva quanto tempo era passato. Ovviamente il tutto era molto semplice, ma mi diede una soddisfazione mai provata prima.

Il successivo passo fu quello di stendere del codice sempre più complesso. In effetti, feci passi più grandi di me, perchè passai dallo scrivere un programma banale come quello sopra citato, allo scrivere un programma molto più complesso, composto da un migliaio di righe di codice.
Era uno scanner, che io affettuosamente chiamai v5. Il programma una volta eseguito, si connetteva ad un server e canale IRC scelto dall'utente. IRC, è l'acronimo di Internet Relay Chat, è stata la prima forma di comunicazione istantanea su Internet; una rete dunque composta da canali, sui quali si collegano utenti per comunicare tra loro. Il v5 faceva da bot, si collegava cioè ad un server IRC, e da qui veniva comandato come se fosse una vera persona.
Dal canale dunque, si poteva controllare questo programma, che a seconda dei comandi che riceveva, era in grado di fare delle scansioni sui più grossi motori di ricerca e trovare quindi siti web vulnerabili.
Con vulnerabile intendo un sito afflitto da bug, meglio conosciuto come un errore di programmazione lato sicurezza. Bastava dunque iniziare la scansione, e attendere sul canale IRC i risultati del v5. Grazie a questo programma era dunque possibile ottenere tutti i siti affetti dalla vulnerabilità scelta dall'utente. Questo programma ebbe una notevole diffusione, tanto che rimpiazzò tutti gli scanner precedentemente scritti.
Fu una grande soddisfazione, perchè vidi che il mio duro lavoro era stato apprezzato. Lo sviluppo di questo programma, il mio primo vero programma, mi portò via circa un mese di tempo, tempo che fu tuttavia importantissimo per la mia crescita informatica.
Successivamente scrissi da capo una nuova versione del v5, il v6, caratterizzato da notevoli migliorie. Questo diventò in assoluto lo scanner più famoso, viene tutt'ora usato nei canali IRC frequentati da hacker. Inizia così a farmi conoscere, ad assumere importanza nell'underground, italiano e non solo. In rete non ero Giovanni Buzzin, ma Osirys.
Scelsi questo nickname anni fa a causa della mia passione per l'Antico Egitto, e le sue divinità. Osiride, dio egiziano della morte e dell'oltretomba, diventò il mio nomignolo.
La scrittura del codice sorgente del v6 mi portò via oltre un mese e mezzo, ricordo ancora i pomeriggi e le sere estive passate sul computer. Era una continua sfida con me stesso, programmando tutt'ora mi metto alla prova, sfido le mie capacità e faccio lavorare il cervello. Per mia madre è una malattia, per me è invece un gratificante e stimolante allenamento cerebrale.
Sempre la scorsa estate, durante le lunghe notti passate in Australia grazie a uno scambio, scrissi un tool in grado di cracckare accessi su i più famosi servizi presente su normali computer e server. Chiamai questo programma "Remote Bruter", in quanto effettua degli attacchi di tipo forza bruta da remoto sul servizio desiderato. I servizi sui quali lavora sono l'ftp, il servizio per il trasferimento dei file tra computer, telnet, servizio di connessione remota, ssh, servizio di connessione remota tra computer rappresentante la versione più sicura e stabile del vecchio telnet. Una volta lanciato, il programma tenta di accedere al servizio con una lunga serie di possibili username e password di cui è in dotazione, e una volta trovata la giusta combinazione, è in grado di garantire l'accesso alla macchina attaccata.
L'estate scorsa iniziai poi a studiare approfonditamente come funziona il web e i linguaggi di programmazione su cui esso si basa.
Soffermiamoci un attimo sul significato di questa parola, per capire bene che cosa sia lo strumento di comunicazione più usato al mondo.

Web - Origine e significati del termine

Il World Wide Web (www), è un insieme vastissimo di contenuti multimediali, e di servizi, di Internet, contenuti e servizi che possono essere resi disponibili dagli stessi utenti di Internet. Il Web rappresenta infatti uno spazio per la pubblicazione di contenuti multimediali pubblicabili da chiunque lo desideri, nonché un mezzo per fornire particolari servizi implementabili di nuovo da chiunque desideri avvalersi di tale possibilità.
Più in generale il Web rappresenta uno dei servizi di Internet, in particolare, assieme alla posta elettronica, il servizio di Internet più utilizzato e conosciuto.
Il Web è stato inizialmente implementato da Tim Berners-Lee, mentre era ricercatore al CERN di Ginevra, sulla base di idee dello stesso Berners-Lee e di un suo collega, Robert Cailliau, e oggi gli standard su cui è basato, in continuo sviluppo, sono mantenuti dal World Wide Web Consortium (W3C).
La nascita del Web risale al 6 agosto 1991, giorno in cui Berners-Lee mise on-line su Internet il primo sito web. Inizialmente utilizzato solo dalla comunità scientifica, il 30 aprile 1993 il CERN decise di rendere pubblica la tecnologia web, decisione a cui seguì un immediato successo del Web che portò ad una crescita esponenzialmente di Internet ancora oggi in atto e alla nascita della cosiddetta "era del Web".
Chiunque disponga di un computer, di un accesso ad Internet, degli opportuni programmi e del cosiddetto spazio web, porzione di memoria di un server web destinata alla memorizzazione di contenuti web e all'implementazione di servizi web, può, nel rispetto delle leggi vigenti nel Paese in cui risiede il server web, pubblicare contenuti multimediali sul Web e fornire particolari servizi attraverso esso. I contenuti del Web sono infatti costantemente on-line quindi costantemente fruibili da chiunque disponga di un computer, di un accesso a Internet, e degli opportuni programmi. In particolare del cosiddetto browser web, il programma che permette, come si dice in gergo, di "navigare" nel Web, cioè di fruire dei contenuti e dei servizi del Web.

La mia crescita - Seconda Parte

Quando noi andiamo su internet, altro non facciamo che dare qualche click al mouse, aprire dunque il browser installato sul sistema, e navigare sui siti desiderati. Tutte azioni compiute automaticamente allo scuro del vero funzionamento di esso. Vi siete mai chiesti cosa c'è sotto, cosa vi si nasconde dietro a quelle amichevoli pagine e qual'è il funzionamento

di un sito internet? Tutte le cose che ci appaiono sul monitor quando navighiamo, sono il frutto di un intenso lavoro di programmatori, persone che hanno scritto il codice sorgente del sito, che una volta interpretato e inviato al nostro browser, ci mostra tutte le cose che vediamo comunemente durante la navigazione.
I linguaggi principali su cui il web si basa sono il l'HTML, il PHP, l'ASP e il Java. Mentre per il salvataggio delle informazioni, vi sono i database, basati sul linguaggio MySQL.
Iniziai dunque a studiarmi questi linguaggi, partendo dall'HTML, relativamente semplice, per poi passare al PHP. Mi recai dunque alla libreria universitaria di Udine, per acquistare il manuale "PHP 5", libro di riferimento per la programmazione in PHP.
Essendo il PHP nato dal Perl, lo appresi con rapidità e facilità, inizia dunque ad occuparmi del Web lato sicurezza.
Essendo ormai internet uno strumento di comunicazione di massa, estramamente diffuso, ci sono sempre più persone che possiedono un proprio sito web, e poichè questi non sono tutti dei programmatori, anzi, nella maggior parte non lo sono, si devono appoggiare a software scritti da altre persone.
Vi sono infatti dei pacchetti pre-scritti, che vanno semplicemente installati sul sito, per renderlo funzionale e trasformarlo in un vero e proprio sito. Dopo l'installazione del pacchetto dunque il sito avrà dei contenuti, degli effetti grafici, e svariate funzioni.
Questi pacchetti possono essere a pagamento, dunque privati, oppure Open-Source.
Con il termine Open-Source viene indicato l'insieme dei codici liberi e pubblici, distribuiti gratuitamente dalla comunità dei programmatori, codici distributi su internet, che possono essere liberamente modificati dall'utente.
L'Open Source oltre a essere tutto ciò, costituisce una vera e proprio filosofia, una filosofia del software, a cui aderiscono totalmente gli Hackers.
L'hacker è una persona che scrive codice per il gusto di farlo, per migliorare il codice di un programma già scritto, per aiutare la comunità, la sua indole quindi è quella di scrivere programmi visibile e modificabili da chiunque, aderisce quindi alla filosofia dell'Open Source, fondata dal famoso programmatore Richard Stallmann. Su questa concezione sono nati i primi veri sistemi operativi, i sistemi Unix, dai quali poi sono nati i famosi sisetemi FreeBsd, e il gioellino Linux, ideato e creato da Linus Torvalds.
Iniziai così a cimentarmi nello studio del PHP, prendendo codici scritti da altri programmatori, studiandoli e migliorandoli, ben presto cominciai a specializzarmi nel campo della sicurezza, prendendo e analizzando i CMS (Content Managent Script), quei pacchetti pre-scritti di cui ho parlato prima. Passai così pomeriggi e pomeriggi interi nella ricerca di possibili falle di sicurezza nel codice sorgente di questi, in modo tale che poi il produttore ne venisse a conoscenza così da migliorare il codice del suo programma rendendolo così sicuro.
Per divertimento, una volta trovata la falla su queste Web Application, iniziavo rapidamente a scriverne i relativi Exploit, programmi cioè volti a sfruttare una falla nelle sicurezza per consentirne l'accesso alla macchina.
In pochi mesi imparai a trovare qualsiasi tipo di falla in applicazioni Web, dalle più immediate, a quelle più complesse, unendo diversi tipi di vulnerabilità con ingnegno e fantasia, riuscendo così a scrivere un Exploit tale che mi consentisse di ottenere l'accesso alla macchina avente la Web Application vulnerabile. Ho scritto dall'inizio di quest'anno circa una cinquantina di exploit e advisories, alcuni dei quali anche per applicazioni molto usate; tutti questi exploit sono reperibili su MilwOrm nella mia pagina personale; MilwOrm.com è il sito più grande al mondo dedicato alla sicurezza informatica, destinato alla pubblicazione di Exploit e Advisories di qualsiasi tipo di Software, punto di riferimento di ogni programmatore e Hacker.
Col passare del tempo, e grazie alla mia dedizione, sono diventato un esperto di sicurezza Web, tanto che son stato commissionato da una società di sicurezza informatica a scrivere un corso, che a breve verrà lanciato, con l'unico fine di insegnare a programmatori e amministratori di rete a tenere i loro programmi e sistemi nella più totale sicurezza.
Con il progredire delle mie conoscenze, iniziai non più ad analizzare codici per siti web, cioè le varie Web Application, ma incominciai a specializzarmi nella sicurezza web da live, ovvero nell'azione di trovare falle nella sicurezza senza aver sotto mano il codice sorgente di un sito, ma semplicemente analizzandolo dal browser. Iniziai così il mio cammino da buon samaritano, assumendo un atteggiamento da Hacker White Hat, "Hacker col cappello bianco".
Incominciai così ad effettuare dei test su siti web anche molto importanti, trovando su di essi delle pericolose vulnerabilità tali da consentire l'accesso senza credenziali a un malintenzionato. Ovviamente, dopo aver trovato la falla e aver testato il funzionamento dell'exploit da me scritto, mi son sempre preoccupato di avvisare gli amministratori dei siti, riferendo loro la vulnerabilità e il modo in cui sistemarle, offrendo così le mie conoscenze per rendere i loro siti più sicuri.
Tra i siti da me testati sui quali ho trovato pericolose falle troviamo siti governativi, italiani e non, siti educativi e scolastici, siti di partiti politici, aereoporti e università italiane e straniere.
Circa due mesi fa ho inoltre trovato una pericolosa falla di tipo SQL Injection su una Applicazione Web privata destinata a essere montata escusivamente su siti comunali. Quando infatti il fondatore di una società di sicurezza informatica con la quale collaboro mi chiese di fare dei test su questa applicazione, e notai con stupore che era afflitta da una grave falla nella sicurezza, mi ritrovai spiazzato, in quanto non riuscivo a credere che una applicazione molto costosa usata da migliaia di siti comunali italiani potesse avere dei buchi.
Dopo aver fatto pratica nel live web auditing, decisi così di scrivermi un tool che mi consentisse di sfruttare e trovare falle di tipo SQL Injection sui siti sui quali dovevo fare dei test. Scrissi così in pochi giorni il "SQL Magic Dumper", un software in grado di prelevare qualsiasi informazione presente nel database di un sito affetto da una vulnerabilità di tipo SQL Injection.
Tutt'ora vi sto applicando delle migliorie per implementare nuove funzionalità.

Expressions Xolver

Siamo giunti quasi alla fine, dove voglio parlare del programma che ho voluto scrivere per la tesina. Avendo sempre scritto software di vulnerability assistment, o di exploiting remoto, devo ammettere che mi son trovato in difficoltà a scegliere cosa portare agli esami. Dopo un po' decisi di portare un programma in grado di risolvere espressioni matematiche, dotate di parentesi o no. Il mio intento iniziale era quello di riuscire a scrivere un tool che risolvesse qualsiasi equazione, ma a causa di molti imprevisti e per mancanza di tempo mi son dovuto limitare nella scrittura di un programma risolutore di espressioni.
Devo ammettere che è stato il programma che più mi ha fatto dannare, in quanto si basa quasi esclusivamente su espressioni regolari, che sono la parte più problematica del Perl, ho dovuto applicare un sacco di brevetti affinché tutto funzionasse. E' stato senza dubbio il programma non più lungo, ma più complesso e complicato fin'ora da me scritto, in quanto ho dovuto ricorrere a tutta la mia fantasia e ingegnosità. E' stato un cammino lungo e duro, ma che alla fine mi ha dato una grande soddisfazione.
Il programma è molto complesso, e difficile da spiegare, cercherò però di illustrare a parole ciò che ho dovuto fare, a cosa ho dovuto ricorrere, per dire al programma di fare delle semplici cose, giusto per mostrarvi cosa significhi programmare.
Ecco di seguito illustrato il funzionamento di due piccoli pezzi del programma: la scomposizione di una frazione, e moltiplicazione/divisone di una serie di numeri.

A) Scomposizione di una frazione

```
sub scomp_fraz() {
   my($fraz,$count) = ($_[0],0);
   my($big,$little,$done);
   if ($fraz =~ /([0-9]{1,})V([0-9]{1,})/) {
      my($numA,$numB) = ($1,$2);
      @todiv = qw(2 3 4 5 6 7 8 9 11);
      while ($count < 9) {
         foreach my $e(@todiv) {
            $count++;
            my $y = $numA/$e;
            my $k = $numB/$e;
            if (($y !~ /\./)&&($k !~ /\./)) {
               $count = $count-9;
               ($numA,$numB) = ($y,$k);
               if ($numB == 1) {
                  $count .= +50;
               }
            }
         }
      }
      if ($numA > $numB) {
         ($big,$little) = ($numA,$numB);
      }
      else {
         ($big,$little) = ($numB,$numA);
      }
      my $y = $big/$little;
      if ($y !~ /\./) {
         $big =~ s/.+/$y/;
         $little =~ s/.+/1/;
      }
      if ($numA > $numB) {
         $numA =~ s/.+/$big/;
         $numB =~ s/.+/$little/;
      }
      elsif ($numB > $numA) {
         $numA =~ s/.+/$little/;
         $numB =~ s/.+/$big/;
      }
      if ($numB == 1) {
         $done = $numA;
      }
      else {
         $done = $numA."/".$numB;
      }
      return($done);
   }
```

}

Spiegazione:

Immaginiamo di avere come frazione la seguente: [48/60], ovviamente questa va scomposta.
Si procederà quindi nel seguente modo:
48:2 -> 24:2 -> 12:3 -> 4
60:2 -> 30:2 -> 15:3 -> 5

Risultato: [4/5]

Il pezzo di codice sopra mostrato agisce in questo modo:
1) Separa la frazione in numeratore e denominatore salvandone i valori nelle rispettive variabili:
    $numA ha come valore 48
    $numB ha come valore 60
2) è presente un array, una specie di archivio dove sono presenti i numeri a cui vanno divisi numeratore e denominatore:
    @todiv contiente: 2 3 4 5 6 7 8 9 11

Il tutto inizia con un ciclio, che viene eseguito fino a quando la variabile $count ha valore numerico strettamente minore di 9. All'inizio del ciclo essa ha valore = 0.
All'interno di questo clico while, vi è un altro:
Per ogni elemento del contenitore @todiv inzia il ciclo foreach, la variabile $count viene incrementata di una unità, $numA e $numB vengono poi divisi per l'elemento del ciclo in azione.
Essendo @todiv composto dai seguenti elementi : 2 3 4 5 6 7 8 9 11
Il ciclo foreach verrà svolto per ognuno di essi, dunque $numA e $numB verranno a ogni ciclo divisi per l'emento preso in considerazione.
A questo punto il risultato della divisone tra $numA e $numB e l'elemento verrà salvato in due variabili: $y e $k.
Il ciclo procederà così dunque:

Y = 48 : 2
K = 60 : 2

Al ripetersi del ciclo verra poi preso il secondo elemento di @todiv, dunque:

Y = 48 : 3
K = 60 : 3

E così via dicendo.
Ad ogni ciclo, per ogni elemento di @todiv quindi verrà fatta la divisone tra numeratore ed elemento e tra denominatore ed elemento dell'archivio.
Se il risultato della divisione, quindi se sia Y che K saranno numeri interi senza virgola, vuol dire che sono entrambi divisibili per l'elemento di @todiv, la variabile $count a questo punto verrà diminuità di 9 unità, $numA e $numB diventaranno il risultato della divisone, quindi $numA diventerà Y e $numB diventerà K. A questo punto, se il denominatore, quindi $numB è diventato 1, vorrà dire che la frazione sarà già scomposta nei suoi minimi termini, quindi non necessita di ulteriori semplificazioni, quindi $count verrà aumentata di 50 unità, in modo tale da non essere più strettamente < 9, causando quindi l'uscita dal ciclo while iniziale che viene eseguito solo quando la condizione che richiede è rispettata, essendo questa che $count deve essere < 9, e poichè ora è stata aumenta di 50, la condizione non è più rispettata e quindi il ciclo termina.
Vi chiederete perché ho posto che $count deve diminuire di 9 unità quando Y e K sono numeri interi..
Il ciclo while viene eseguito fino a quando $count non diventa > 9 : vediamo il seguente caso:
$numA = 43;
$numB = 47;

$count = 0;

Ciclo 1:

43 : 2 = 21,5
47 : 2 = 23,5

Entrambi i risultati hanno la virgola, $count diventa 1

Ciclo 2:

43 : 3 = 14,3
47 : 3 = 15,6

Entrambi i risultati hanno la virgola, $count diventa 2

Andando così avanti fino all'ultimo elemento:

Ciclo 9:

43 : 11 = 3,9
47 : 11 = 4,2

Entrambi i risultati hanno la virgola, $count diventa 9, poiché è stat incrementata di una unità per ogni ciclo dal momento che i risultati hanno sempre avuto la virgola.

$count è 9, non è più < 9, non rispetta più dunque la condizione necessaria allo svolgimento del ciclo while, quindi esso termina. 43/47 non è divisibile.

Ad ogni ciclo $count veniva incrementata di una unità perché numeratore e denominatore non erano mai divisibili per l'elemento per il quale venivano divisi, se invece lo sarebbero stati, $count venendo dimunuita di 9, tornava alla suo valore iniziale, consentendo così i cicli successivi.

A questo punto ci troviamo con la frazione scomposta, dobbiamo però verificare se numeratore e denominatore sono divisibili tra se stessi.

Esempio: Facciamo finta che la frazione scomposta sia diventata: [37/74]
Numeratore e denominatore non hanno numeri divisibili in comune, tranne se stessi.
Il programma a questo punto verifica quale dei due sia maggiore dell'altro, assegnando quello maggiore e quello minore alle rispettive variabili.

74 > 37 => 74 = big; 37 = little

Y = big/little => 74:37

Se Y non ha la virgola, è un numero intero significa che i due numeri sono ancora divisibili tra loro.
big viene sostituito con il risultato, little con 1.

74:37 = 2 -> 74 diventa 2 (big sostituito col risultato); 37 diventa 1 (little)

Ora se era il num a essere > del den, il num diventa big, mentre il den little, in caso contrario la sostituzione sarà opposta.
Se ora il den è 1, il risultato finale sarà: num
Altrimenti sarà: num/den

Vediamo lo script avviato:

```
osirys[~]>$ perl test.txt

[+] Frazione da scomporre: 48/60

[+] Risultato: 4/5

osirys[~]>$ perl test.txt

[+] Frazione da scomporre: 37/74

[+] Risultato: 1/2

 osirys[~]>$ perl test.txt

[+] Frazione da scomporre: 74/37

[+] Risultato: 2

osirys[~]>$
```

Analizziamo ora una piccola parte, volta al fine di moltiplicare tra loro una serie di numeri.
Immaginiamo di avere una stringa di questo tipo:
[2*4*-2*7]
A parole sembra molto facile da fare, voi direte: "basta dire di moltiplicare i numeri tra loro". Ora illustrerò cosa ho dovuto fare per dire al programma di fare una cosa che sembra così banale, giusto per rendervi l'idea.

Innanzitutto ho fatto in modo che gli elementi della stringa venissero separati tra loro, mettendo tutti i numeri in un contenitore, che chiamero @numeri, tutti i segni in un altro contenitore, che sarà @signs, e i per e i diviso in un altro ancora, chiamato @ops.
A questo punto viene avviato un ciclo che viene eseguito per ogni numero presente in @numeri.
La variabile $cop serve a contare gli elementi dell'archivio @ops, inizialmente ha valore 0.
A questo punto se l'elemento preso in considerazione dal ciclo corrisponde al primo elemento di @numeri, significa che questo deve essere moltiplicato/diviso per il secondo elemento di @numeri. La scelta di divisione o moltiplicazione avviene prendendo il primo elemento di @ops, grazie all'indice di $cop, che al primo ciclo sarà 0. In Perl infatti il primo elemento di un archivio non ha indice 1, ma 0.
A questo punto il risultato della prima operazione viene messo al primo posto in un nuovo archivio, @risultati. Il segno finale del risultato verrà scelto grazie a una funzione, la funzione signs(), alla quale verranno inviati come parametri i primi due elementi di @signs.
Se saranno due più, il segno risultante sarà un più, se sranno due meno sarà sempre più, in tutti gli altri casi il segno finale sarà un meno.
A questo punto, che abbiamo il risultato dell'operazione, questo verrà sostituito nella stringa.
Sempre all'interno di questo ciclo, $cop verrà incrementato di una unità, così da scorrere al prossimo elemento di @ops.
Il primo ciclo controllava dunque se si trattava del primo elemento del ciclo, ora inizia il secondo ciclo, quello in cui l'elemento preso in considerazione di @numeri non è il primo, ma dal secondo fino all'ultimo.

2*4*-2*7

@numeri = 2 4 2 7
@signs = + + - +
@ops = * * *

Nel primo ciclo, si prenderà in esamine 2, cosituente l'elemento numero uno di @numeri.
Dunque verrà moltiplicato al secondo -> 2 * 4

Il primo blocco è stato eseguito, e serve per moltiplicare/dividere i primi due numeri.

2 * 4 = 8
8 è il risultato, messo come primo elemento di @risultati.
A questo punto, il secondo elemento di @risultati sarà dato dal primo risultato ottenuto, moltiplicato per il terzo elemento di @numeri.

8 * 2 = 16
8 -> primo elemento di @risultatu
2 -> terzo elemento di @numeri

16 diventa il secondo elemento di @risultati

Ora 16 verrà moltiplicato per 7, 4o ed ultimo elemento di @numeri.

Il procedimento è stato spiegato, ma bisogna renderlo in una funzione.

```
while (my $a = <@numeri>) { # Ciclo che scorre gli elementi di @numeri
        if ($a__ == $numeri[0]) { #Se l'elemento è il primo elemento di @numeri
                esegue il primo blocco, moltiplica/divide il primo elemento di @numeri
                con il secondo. Sostituisce poi il risultato nella stringa
                Mette il risultato in @risultati
        }
        Moltiplica/Divide l'elemento di indice K di @risultati



        Con l'elemtno di indice X di @numeri
        Il risultato diventa l'elemento di indice K di @risultati
        Sostituisce il risultato nella stringa

        if ($count == Num degli elem totali di @numeri - 2) {
                Il ciclo While termina
        }
        X,Y e K vengono incrementate di una unità.
```

}

All'inizio, X = 2
Y = 1
K = 0

1° Ciclo:

In questo modo,al primo ciclo, quindi al numero 2 (1o el di @numeri), lo moltiplica con 4,
(2° elemento di @numeri), il risultato diventa il 1° el di @risultati

@risultati = 8

K vale ancora 0, deve moltiplicare/dividere l'elemento di indice K (0) di @risultati, con l'elemento
di indice X (X non è ancora stato alterato, vale quindi 2) di @numeri
Sarà dunque: @risultati[0] * @numeri[2]
8 * 2
In quanto il primo elemento di @risultati è 8, il terzo elemento di @numeri è 2. Questa volta è il terzo in quanto i primi due numeri sono già stati moltiplicati tra loro al primo ciclo.

Finito il ciclo, X Y K vengono aumentati di 1, così il ciclo continua con i sucessivi elementi degli arichivi.

Il ciclo termina quando il ciclo sarà eseguito sul penultimo elemento di @numeri, poichè questo verrà moltiplicato/diviso per l'ultimo numero.

Spero di essere stato abbastanza chiaro nella spiegazione.

Expressions Xolver - Codice Sorgente

```
#!/usr/bin/perl

# Expression Solver
# Programma per la tesina di V Superiore
# Liceo Scientifico Paolo Diacono
# Giungo 2009

# Coded by Osirys


print "\n".
    "-------------------------\n".
    "    Expressions Xolver   \n".
    "    by Giovanni Buzzin   \n".
    "        \"Osirys\"      \n\n";

print "[+] Type now the Expression: ";

chomp(my $equazionez = <STDIN>);


#$equazionez = "+2-[(+3-1:+5):(+2+4:+5)+1:+3]*(+1-5:+8)+1:+2+1:+4";                    ES 1 pag 117
#$equazionez = "(+19:+3-2*+5:+6)+3:+4-2:+3:(+1:+6):(+8:+5)+(+7:+2+3:+4*+5:+6:+15)*+12:+17"; ES 2 pag 117
#$equazionez = "+1:+2+[+1:+3+(+2:+5)*(+1+2:+3)+2:(+1-1:+3)]*(+6:+5-1):(+1:+5)";          ES 9 pag 117

print "\n[+] Risolvendo: $equazionez\n\n";
```

```
$equazione = conv($equazionez,2,2);

risolvi($equazione);

exit;

sub risolvi() {
   my $equazione = $_[0];
   conv($equazione,1,1);
   if ($equazione  =~ /\(/) {
      my($tcnt,$tcntt) = (0,0);
      while ($equazione =~ /\(/g) {
         $tcnt++;
      }
      while (($equazione =~ /\(([^)]+)\)/g)&&($tcntt < $tcnt)) {
         my $piece = $1;
         my $tcntt++;
         my $mpiece = &risolvi_($piece);
         $equazione =~ s/\($piece\)/$mpiece/;
         $equazione = &clean($equazione);
         conv($equazione,1,1);
      }
   }
   if ($equazione  =~ /\[/) {
      my($tcnt,$tcntt) = (0,0);
      while ($equazione =~ /\[/g) {
         $tcnt++;
      }
      while (($equazione =~ /\[([^\]]+)\]/g)&&($tcntt < $tcnt)) {
         my $piece = $1;
         my $tcntt++;
         my $mpiece = &risolvi_($piece);
         $equazione =~ s/\[$piece\]/$mpiece/;
         $equazione = &clean($equazione);
         conv($equazione,1,1);
      }
   }
   if ($equazione  =~ /\{/) {
      my($tcnt,$tcntt) = (0,0);
      while ($equazione =~ /\{/g) {
         $tcnt++;
      }
      while (($equazione =~ /\{([^}])\}/g)&&($tcntt < $tcnt)) {
         my $piece = $1;
         my $tcntt++;
         my $mpiece = &risolvi_($piece);
         $equazione =~ s/\{$piece\}/$mpiece/;
         $equazione = &clean($equazione);
         conv($equazione,1,1);
      }
   }
   $equazione_u = risolvi_($equazione);
}

sub risolvi_() {
   my($equazione,$count) = ($_[0],-1);
   my(@numerii,@signss,@opss,$countz,@tmp_nnum,$cnm,@total,$string);
   while ($equazione =~ /(^|[0-9.])(meno|più)([0-9.]+)(diviso|per)(meno|più)([0-9.]+)(meno|più|$)/g) {
      my($num,$num2) = ($2.$3,$5.$6);
      my $num_ = $num."$4".$num2;
      my $g_num = $num."-".$num2;
      if ($4 =~ /per/) {
         $cnm = &sign($g_num,1);
      }
      elsif ($4 =~ /diviso/) {
         $cnm = &sign($g_num,2);
      }
      $equazione =~ s/$num_/$cnm/;
      conv($equazione,1,1);
   }
   while ($equazione =~ /(più|meno|)([0-9.]+)(per|diviso|più|meno|)/g) {
      my($sign,$num,$boh) = ($1,$2,$3);
      $count += 3;
      $countz = $count - 3;
      if ($tmp_nnum[$countz] !~ /per|diviso/) {
         push(@total," ");
      }
      if ($sign !~ /./) {
         $sign =~ s/(.*)/vuoto/;
      }
```

```
        push(@tmp_nnum,$sign,$num,$boh);
        if ($sign =~ /vuoto/) {
            $sign =~ s/(.*)/$tmp_nnum[$countz]/;
        }
        if ($boh =~ /per|diviso/) {
            push(@total,$sign,$num,$boh);
        }
        else {
            if ($tmp_nnum[$countz] =~ /per|diviso/) {
                push(@total,$sign,$num);
            }
        }
    }
    $string = join '', @total;
    $string =~ s/.+/$string /;
    $string =~ s/( +)/ /g;
    $string =~ s/^ //;
    while ($string =~ /([^ ]+) /g) {
        $equazione = &calcola($1,$equazione,"1");
    }
    if ($equazione =~ /(meno|più)([0-9.]+)(meno|più)/) {
        $equazione = &calcola($equazione,$equazione,"2");
    }
    #conv($equu,1,1);
    return($equazione);
}

sub calcola() {
    my($string,$equazione,$way,$stop,$cntm) = ($_[0],$_[1],$_[2],0,0);
    my(@numeri,@signs,@ops,@risultati,$sign,$st0p);
    if ($string =~ /\./) {
        $string =~ s/\./0011001100/g;
    }
    while ($string =~ /([0-9.]+)/g) {
        my $anum = $1;
        if ($anum =~ /0011001100/) {
            $anum =~ s/0011001100/./;
        }
        push(@numeri,$anum);
    }
    while ($string =~ /(diviso|per)/g) {
        push(@ops,$1);
    }
    while ($string =~ /(meno|più)/g) {
        push(@signs,$1);
    }
    $string =~ s/0011001100/./g;
    my($count,$count_,$count__,$cop) = (2,1,0,0);
    my $num_n = scalar(@numeri);
    $num_n++;
    while (my $a__ = <@numeri>) {
        if ($stop != 1) {
            if ($a__ == $numeri[0]) {
                if ($way == 1) {
                    if ($ops[$cop] =~ /per/) {
                        $risultati[$count__] = $a__*$numeri[1];
                    }
                    elsif ($ops[$cop] =~ /diviso/) {
                        $risultati[$count__] = $a__/$numeri[1];
                    }
                    $signss[$count__] = signs($signs[0],$signs[1]);
                    my($ris,$sign,$num2,$bstring) = ($risultati[$count__],$signss[$count__],$numeri[1],$string);
                    if ($string =~ /(meno|più)$a__(per|diviso)(meno|più)$num2/) {
                        $string =~ s/$1$a__$2$3$num2/$sign$ris/;
                        $equazione =~ s/$bstring/$string/;
                        conv($equazione,1,1);
                    }
                }
                elsif ($way == 2) {
                    if (($signs[0] =~ /più/)&&($signs[1] =~ /più/)) {
                        $risultati[$count__] = $a__+$numeri[1];
                    }
                    elsif (($signs[0] =~ /più/)&&($signs[1] =~ /meno/)) {
                        $risultati[$count__] = $a__-$numeri[1];
                    }
                    elsif (($signs[0] =~ /meno/)&&($signs[1] =~ /più/)) {
                        $risultati[$count__] = -$a__+$numeri[1];
                    }
                    elsif (($signs[0] =~ /meno/)&&($signs[1] =~ /meno/)) {
                        $risultati[$count__] = -$a__-$numeri[1];
                    }
                    my($ris,$fsign,$ssign,$num2,$bstring) = ($risultati[$count__],$signs[0],$signs[1],$numeri[1],$string);
                    if ($string =~ /$fsign$a__$ssign$num2/) {
```

```
                $string =~ s/$fsign$a__$ssign$num2/$ris/;
                $equazione =~ s/$bstring/$string/;
                $equazione =~ s/-/meno/g;
                $equazione =~ s/\+/più/g;
                $equazione = &clean($equazione);
                if ($equazione !~ /^(meno|più)/) {
                   $equazione =~ s/.+/più$equazione/;
                }
                if ($equazione !~ /(più|meno)([0-9.]+)(più|meno)/) {
                   $st0p = 1;
                }
                conv($equazione,1,1);#print "uuu $equazione\n";
             }
          }
          $cop++;
       }
       if ($way == 1) {
          if ($ops[$cop] =~ /per/) {
             $risultati[$count_] = $risultati[$count__]*$numeri[$count];
          }
          elsif ($ops[$cop] =~ /diviso/) {
             $risultati[$count_] = $risultati[$count__]/$numeri[$count];
          }
          $signss[$count_] = signs($signss[$count__],$signs[$count]);
          my($ris,$sign,$num,$num2,$bstring) = ($risultati[$count_],$signss[$count_],$risultati[$count__],$numeri[$count],$string);
          if ($string =~ /(meno|più)$num(per|diviso)(meno|più)$num2/) {
             $string =~ s/$1$num$2$3$num2/$sign$ris/;
             $equazione =~ s/$bstring/$string/;
             conv($equazione,1,1);
          }
       }
       elsif (($way == 2)&&($st0p != 1)) {
          if ($signs[$count] =~ /più/) {
             $risultati[$count_] = $risultati[$count__]+$numeri[$count];
          }
          elsif ($signs[$count] =~ /meno/) {
             $risultati[$count_] = $risultati[$count__]-$numeri[$count];
          }
          $string =~ s/-/meno/g;
          $string =~ s/\+/più/g;
          my($ris,$num,$sign,$num2,$bstring) = ($risultati[$count_],$risultati[$count__],$signs[$count],$numeri[$count],$string);
          $num =~ s/-/meno/g;
          $num =~ s/\+/più/g;
          if ($string =~ /$num$sign$num2/) {
             $string =~ s/$num$sign$num2/$ris/;
             $equazione =~ s/$bstring/$string/;
             $equazione =~ s/-/meno/g;
             $equazione =~ s/\+/più/g;
             $equazione = &clean($equazione);
             if ($equazione !~ /^(meno|più)/) {
                $equazione =~ s/.+/più$equazione/;
             }
             conv($equazione,1,1);
          }
       }
       if ($count == ($num_n-2)) {
          $stop = 1;
       }
       $count++; $count_++;$count__++;
    }
    $cop++;
 }
 if ($equazione !~ /^(più|meno)/) {
    $equazione =~ s/.+/più$equazione/;
 }
 return($equazione);
}

sub signs() {
 my($sign,$sign2) = @_;
 my $fsign;
 if (($sign =~ /più/)&&($sign2 =~ /più/)) {
    $fsign = "più";
 }
 elsif (($sign =~ /meno/)&&($sign2 =~ /meno/)) {
    $fsign = "più";
 }
 elsif (($sign =~ /meno/)&&($sign2 =~ /più/)) {
    $fsign = "meno";
 }
 elsif (($sign =~ /più/)&&($sign2 =~ /meno/)) {
    $fsign = "meno";
 }
```

```
    return($fsign);
}

sub sign() {
    my($numz,$op) = @_;
    my($num,$num2);
    if ($numz =~ /(.+)-(.+)/) {
        ($num,$num2) = ($1,$2);
    }
    my(@signsm,@signsp,$num_a_,$num_b_,$sign,$ris);
    if ($num =~ /(più|meno)([0-9.]+)/) {
        my $sign = $1;
        $num_a_ = $2;
        if ($sign =~ /più/) {
            push(@signsp,$1);
        }
        else {
            push(@signsm,$1);
        }
    }
    if ($num2 =~ /(più|meno)([0-9.]+)/) {
        my $sign = $1;
        $num_b_ = $2;
        if ($sign =~ /più/) {
            push(@signsp,$1);
        }
        else {
            push(@signsm,$1);
        }
    }
    if ((scalar(@signsp) == 0)||(scalar(@signsm) == 0)) {
        $sign = "più";
    }
    else {
        $sign = "meno";
    }
    if ($op == 1) {
        $ris = $num_a_*$num_b_;
    }
    elsif ($op == 2) {
        $ris = $num_a_/$num_b_;
    }
    my $cnm = $sign.$ris;
    return($cnm);
}

sub conv() {
    my($string,$mode,$opt) = @_;
    if ($mode == 1) {
        $string =~ s/ //g;
        $string =~ s/più/+/g;
        $string =~ s/meno/-/g;
        $string =~ s/per/*/g;
        $string =~ s/diviso/:/g;
    }
    elsif ($mode == 2) {
        $string =~ s/ //g;
        $string =~ s/\+/più/g;
        $string =~ s/-/meno/g;
        $string =~ s/\*/per/g;
        $string =~ s/:/diviso/g;
    }
    if ($opt == 1) {
        if ($string =~ /\./) {
            $string = conv_fraz($string);
        }
        print "$string =\n";

    }
    elsif ($opt == 2) {
        return($string);
    }
}

sub conv_fraz() {
    my($eq,@numeri) = ($_[0]);
    if ($eq =~ /\./) {
        while ($eq =~ /([0-9]+)\.([0-9]+)/g) {
            push(@numeri,$1."/".$2);
        }
        foreach my $e(@numeri) {
            my $match = 0;
```

```
if ($e =~ /(.+)\.(.+)/) {
   my($num1,$num2) = ($1,$2);
   if ($num2 =~ /([0-9][0-9][0-9]){8,}/) {
      my($f_num,$s_num,$t_num,$q_num,$ss_num);
      if ($num2 =~ /^([0-9]{1})([0-9]{0,1})([0-9]{0,1})([0-9]{0,1})([0-9]{0,1})/) {
         ($f_num,$s_num,$t_num,$q_num,$ss_num) = ($1,$2,$3,$4,$5);
      }
      if ($num2 =~ /^($f_num$s_num$t_num){8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num.$t_num,$num1);
         my $y = $a-$b;
         my $fraz = $y."/999";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
      elsif ($num2 =~ /^$f_num{1}($s_num$t_num$q_num){8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num.$t_num.$q_num,$num1.$f_num);
         my $y = $a-$b;
         my $fraz = $y."/9990";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
      elsif ($num2 =~ /^$f_num{1}$s_num{1}($t_num$q_num$q_num){8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num.$t_num.$q_num.$ss_num,$num1.$f_num.$s_num);
         my $y = $a-$b;
         my $fraz = $y."/99900";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
   }
   elsif ($num2 =~ /([0-9][0-9]){8,}/) {
      my($f_num,$s_num,$t_num,$q_num);
      if ($num2 =~ /^([0-9]{1})([0-9]{0,1})([0-9]{0,1})([0-9]{0,1})/) {
         ($f_num,$s_num,$t_num,$q_num) = ($1,$2,$3,$4);
      }
      if ($num2 =~ /^($f_num$s_num){8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num,$num1);
         my $y = $a-$b;
         my $fraz = $y."/99";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
      elsif ($num2 =~ /^$f_num{1}($s_num$t_num){8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num.$t_num,$num1.$f_num);
         my $y = $a-$b;
         my $fraz = $y."/990";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
      elsif ($num2 =~ /^$f_num{1}$s_num{1}($t_num$q_num){8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num.$t_num.$q_num,$num1.$f_num.$s_num);
         my $y = $a-$b;
         my $fraz = $y."/9900";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
   }
   elsif ($num2 =~ /([0-9]{1}){8,}/) {
      my($f_num,$s_num,$t_num);
      if ($num2 =~ /^([0-9]{1})([0-9]{0,1})([0-9]{0,1})([0-9]{0,1})/) {
         ($f_num,$s_num,$t_num,$x_num) = ($1,$2,$3,$4);
      }
      if ($num2 =~ /^$f_num{8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num,$num1);
         my $y = $a-$b;
         my $fraz = $y."/9";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
      elsif ($num2 =~ /^$f_num{1}$s_num{8,}/) {
         $match = 1;
         my($a,$b) = ($num1.$f_num.$s_num,$num1.$f_num);
         my $y = $a-$b;
         my $fraz = $y."/90";
         my $fraz = scomp_fraz($fraz);
         $eq =~ s/$e/$fraz/;
      }
      elsif ($num2 =~ /^$f_num{1}$s_num{1}$t_num{8,}/) {
```

```
                $match = 1;
                my($a,$b) = ($num1.$f_num.$s_num.$t_num,$num1.$f_num.$s_num);
                my $y = $a-$b;
                my $fraz = $y."/900";
                my $fraz = scomp_fraz($fraz);
                $eq =~ s/$e/$fraz/;
            }
            elsif ($num2 =~ /^$f_num{1}$s_num{1}$t_num{1}$x_num{8,}/) {
                $match = 1;
                my($a,$b) = ($num1.$f_num.$s_num.$t_num.$x_num,$num1.$f_num.$s_num.$t_num);
                my $y = $a-$b;
                my $fraz = $y."/9000";
                my $fraz = scomp_fraz($fraz);
                $eq =~ s/$e/$fraz/;
            }
        }
        if ($match != 1) {
            my($count,$zero) = (0,);
            while ($num2 =~ /[0-9]/g) {
                $count++;
            }
            for (1..$count) {
                $zero .= "0";
            }
            my $mul = "1".$zero;
            my $y = $e*$mul;
            my $fraz = $y."/".$mul;
            my $fraz = scomp_fraz($fraz);
            $eq =~ s/$e/$fraz/;
        }
      }
    }
    return($eq);
  }
}

sub scomp_fraz() {
  my($fraz,$count) = ($_[0],0);
  my($big,$little,$done);
  if ($fraz =~ /([0-9]{1,})\/([0-9]{1,})/) {
    my($numA,$numB) = ($1,$2);
    @todiv = qw(2 3 4 5 6 7 8 9 11);
    while ($count < 9) {
      foreach my $e(@todiv) {
        $count++;
        my $y = $numA/$e;
        my $k = $numB/$e;
        if (($y !~ /\./)&&($k !~ /\./)) {
          $count = $count-9;
          ($numA,$numB) = ($y,$k);
          if ($numB == 1) {
            $count .= +50;
          }
        }
      }
    }
    if ($numA > $numB) {
      ($big,$little) = ($numA,$numB);
    }
    else {
      ($big,$little) = ($numB,$numA);
    }
    my $y = $big/$little;
    if ($y !~ /\./) {
      $big =~ s/.+/$y/;
      $little =~ s/.+/1/;
    }
    if ($numA > $numB) {
      $numA =~ s/.+/$big/;
      $numB =~ s/.+/$little/;
    }
    elsif ($numB > $numA) {
      $numA =~ s/.+/$little/;
      $numB =~ s/.+/$big/;
    }
    if ($numB == 1) {
      $done = $numA;
    }
    else {
      $done = $numA."/".$numB;
    }
    return($done);
  }
```

```
}

sub clean() {
   my $equazione = $_[0];
   if ($equazione =~ /(piùpiù|menomeno|piùmeno|menopiù)/) {
      my $double = $1;
      if ($double =~ /piùpiù/) {
         $equazione =~ s/piùpiù/più/g;
      }
      elsif ($double =~ /menomeno/) {
         $equazione =~ s/menomeno/più/g;
      }
      elsif ($double =~ /piùmeno|menopiù/) {
         $equazione =~ s/piùmeno/meno/g;
         $equazione =~ s/menopiù/meno/g;
      }
   }
   return($equazione);
}
```

Conclusione

Con questa tesi spero di essere riuscito a farvi capire quanto affiscinante sia il mondo degli Hackers, dei programmatori, e della programmazione stessa. In fin dei conti, sono questi gli elementi si cui si basa il mondo al giorno d'oggi.

Diciamo che non sono stato io a scegliere di fare questa tesi, ma è stata lei a scegliere me, sentivo infatti davvero un grande bisogno di raccontare la mia seconda vita, quella che passo al computer sotto il nomignolo di Osirys, condiviendo così con tutti i lettori le grandi emozioni che questo mondo riesce ad offrirmi, emozioni uniche, grandi soddisfazioni che continuano a farmi crescere, nella speranza di essere utile un domani per la comunità, e di fare qualcosa di importante.