

Админская паранойя в быту

или страшная
криптографическая сказка для
самых маленьких параноиков

Disclaimer

"...Мопед не мой..."

Если вы все это знаете или думаете что это бред сумасшедшего параноика - well...

Как мы храним пароли?

Back to Happy 90s !

Никто не заморачивается и хранит пароли в plain text :)

Welcome to Mox.Perl.COM!

More than 300 megabytes amongst 14,000 files available!

Mox nox in rem!

With a PC, I always felt limited by the software available.

On Unix, I am limited only by my knowledge.

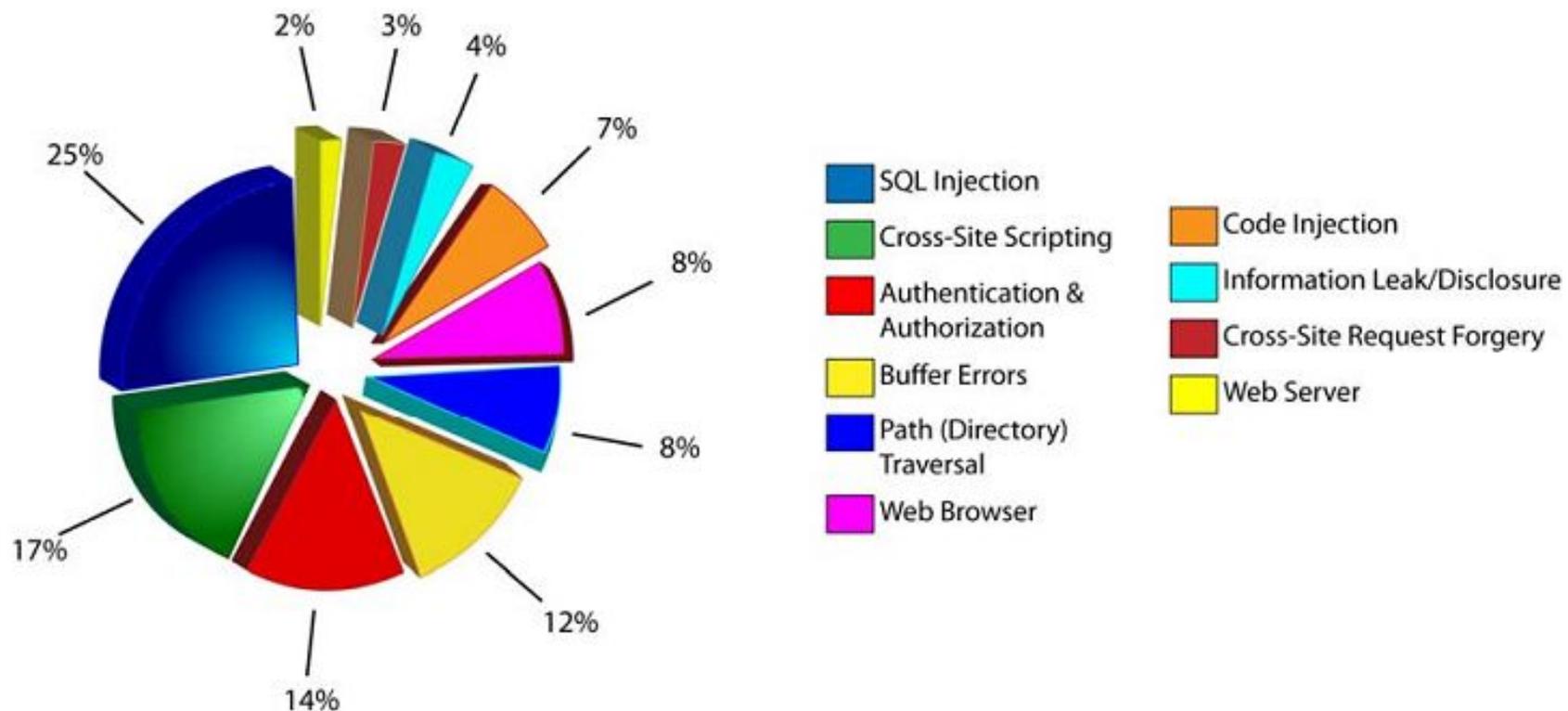
--Peter J. Schoenster <pschon@baste.magibox.net>

Humor item: Ever wonder what [real programmers](#) think of [your system](#)?

Scary 00's

Все плохо :(

Web Vulnerabilities by Class
Q1-Q2 2009



Пример: perlmonks & ZFO

"There is a really simple reason we owned PerlMonks:
we couldn't resist more than 50,000 unencrypted programmer passwords."--- ZFO

979 Volma379 Tim Vroom vroom@blockstackers.com

171588 adv59416 Nigel Sandever vev6s4702@sneakemail.com

381608 archforc ELB ikegami@adaelis.com

9073 4p3rlm0n Randal L. Schwartz merlyn.perlmonk@stonehenge.com

5348 a5q!po9 Max Maischein corion@corion.net

17000 pineappl Curtis Poe curtis_ovid_poe@yahoo.com

85580 kieran Rob Kinyon rob.kinyon@gmail.com

22609 ij7dlcmy Tye McQueen nothingisobvious@gmail.com

26179 rtkJhiG2 Ben Tilly btilly@gmail.com

82147 bZ9jFSgN Tom Leete tleete@zoominternet.net

1382 p3rlm0 chromatic chromatic@wgz.org

461912 william Peter Jaquiery peter@adi.co.nz

29008 MonkEBiz david landgren david@landgren.net

169744 EahejY7f Abigail abigail@foad.org

22308 davesmit Dave Smith dws@davewsmith.co

Что же делать?

Hash functions to the rescue!

MD5, SHA1, SHA256,

$X = \text{MD5}(\text{PW}) \rightarrow$ храним X

Радужные таблицы?

$X = \text{MD5}(\text{salt} + \text{PW}) \rightarrow$ храним X и salt

Все хорошо?

Не совсем.

Improvements

1. Использовать случайную "соль" размером как и результат - 160 бит для MD5

"Bytes Are Cheap Now"

--- Bruce Schneier

2. Вместо $\text{HASH}(\text{PW} + \text{Salt})$ нужно использовать $\text{HASH}(\text{HASH}(\text{PW}) + \text{Salt})$

$X = \text{MD5}(\text{MD5}(\text{PW}) + \text{Salt}) \rightarrow$ храним X и Salt

Все теперь хорошо?

GPU Bruteforce

2008г. - Nvidia 8600GT - 64 млн MD5 / сек

2010г. - **IGHASHGPU** - Fastest SHA1/MD5 hash cracker on ATI and NVIDIA GPUs (c) golubev.com -

HD5770+HD4770+8600GT = 2 731 млн MD5 / сек

(в пересчете на карту - ускорение в 10 раз за 2 года!)

Много это или мало?

8-ми символьные буквенно-цифровые пароли =

368 = 2 821 109 907 456 / 2 731 000 = 20 минут (!)

(добавление спецсимволов не спасает -

528 = 53 459 728 531 456 ~ 6 часов !)

Что делать?

Увеличивать вычислительную трудность пароля!

Вместо $X = \text{MD5}(\text{MD5}(\text{Pw}) + \text{Salt})$ используем

$$X_1 = 0$$

$$X_i = \text{MD5}(X_{i-1} + \text{Pw} + \text{Salt})$$

$$i = 1 \dots 2M$$

Умные люди советуют $M=20$ (20 бит энтропии к паролю)

Но что делать на Perl?

Цикл на 1000000 раз будет считаться довольно долго...

Сколько?

Проверим

```
use Digest::MD5 qw(md5 md5_hex);
use Benchmark;
use String::Random qw(random_string);

my $pass = random_string('.'x20);
my $salt = random_string('.'x20);

timethis(100, '
  my $data = 0;
  foreach (1..1024*1024) {
    $data = md5($data.$pass.$salt);
  }
  $data = md5_hex($data.$pass.$salt);
');
```

Результаты

Для 2^{20} итераций -

timethis 100: 98 wallclock secs (97.45 usr + 0.07 sys = 97.52 CPU) @ **1.03/s** (n=100)

Для 2^{10} итераций чуть лучше -

timethis 100000: 93 wallclock secs (92.79 usr + 0.05 sys = 92.84 CPU) @ **1077.12/s** (n=100000)

Более менее-оптимально - при 2^{13} итераций -

timethis 10000: 76 wallclock secs (75.32 usr + 0.02 sys = 75.34 CPU) @ **132.73/s** (n=10000)

Достаточно быстро? в принципе да, но не для параноиков :) ...

Vcrypt to the rescue!

<http://www.usenix.org/events/usenix99/provos.html>



- Используется в OpenBSD с 1999г.
- Пароли по умолчанию в OwlLinux и AltLinux, поддерживается в OpenSuSE и ASPLinux
- PHP Suhoshin Patch, PostgreSQL etc.

Features

- Основан на шифре Blowfish by Bruce Schneier
- Настраиваемый параметр вычислительной сложности
 - M = 5 - вычисляется за 100 мс
 - M = 8 - вычисляется за 5 сек

Реализация

- Crypt::Eksblowfish::Vcrypt на CPAN

Сравним Bcrypt, MD5 и SHA512

M=8

Benchmark: timing 1000 iterations of bcrypt...

bcrypt: 25 wallclock secs (24.91 usr + 0.00 sys = 24.91 CPU) @ **40 .14/s** (n=1000)

...

M=13

Benchmark: timing 1000 iterations of md5, sha512...

md5: 8 wallclock secs (7.31 usr + 0.00 sys
= 7.31 CPU) @ 136.80/s (n=1000)

sha512: 18 wallclock secs (18.22 usr + 0.00 sys =
18.22 CPU) @ 54.88/s (n=1000)

Спорить сюда -

denis.zhdanov@gmail.com