



Autenticação Alternativa

Utilizando a biblioteca Linux-PAM

Bruno Ferreira

2015

brunoferreira@ufrj.br

Roteiro

- Introdução.
- A biblioteca PAM.
- Construindo sua própria biblioteca PAM.
- Proposta de autenticação alternativa.
- Considerações finais.



Parte 1 - Motivação



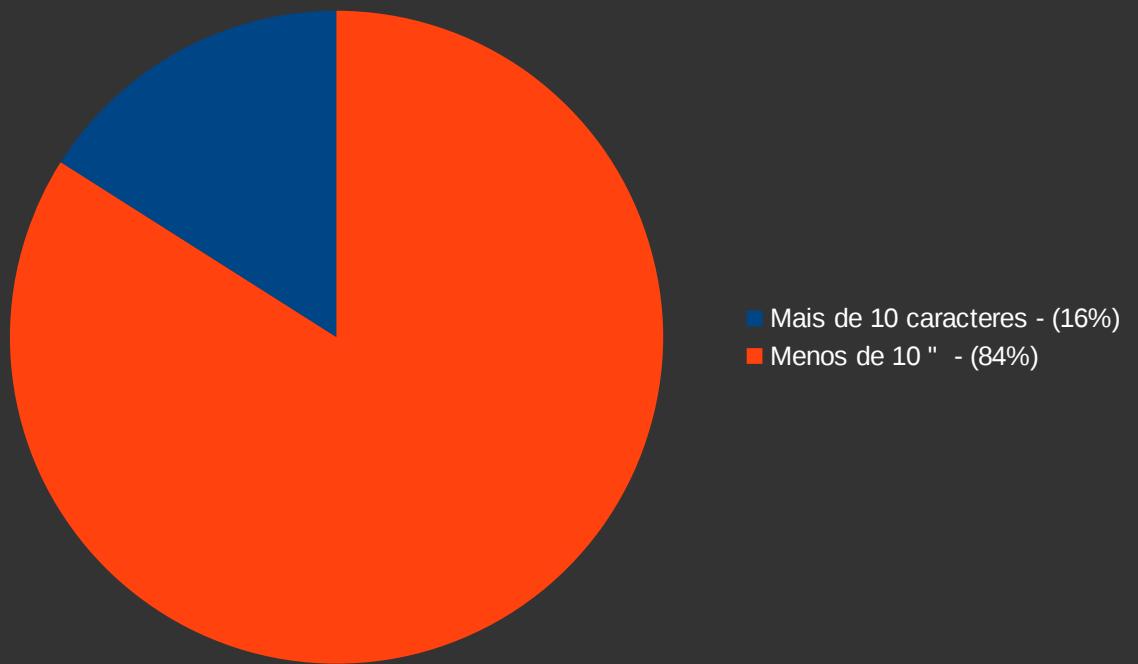
Motivação

- O uso de senhas complexas na rotina diária é algo cada vez mais complicado...



Motivação

- Em um estudo realizado em outubro de 2010 somente 16% dos entrevistados afirmaram usar senhas com mais de 10 caracteres.

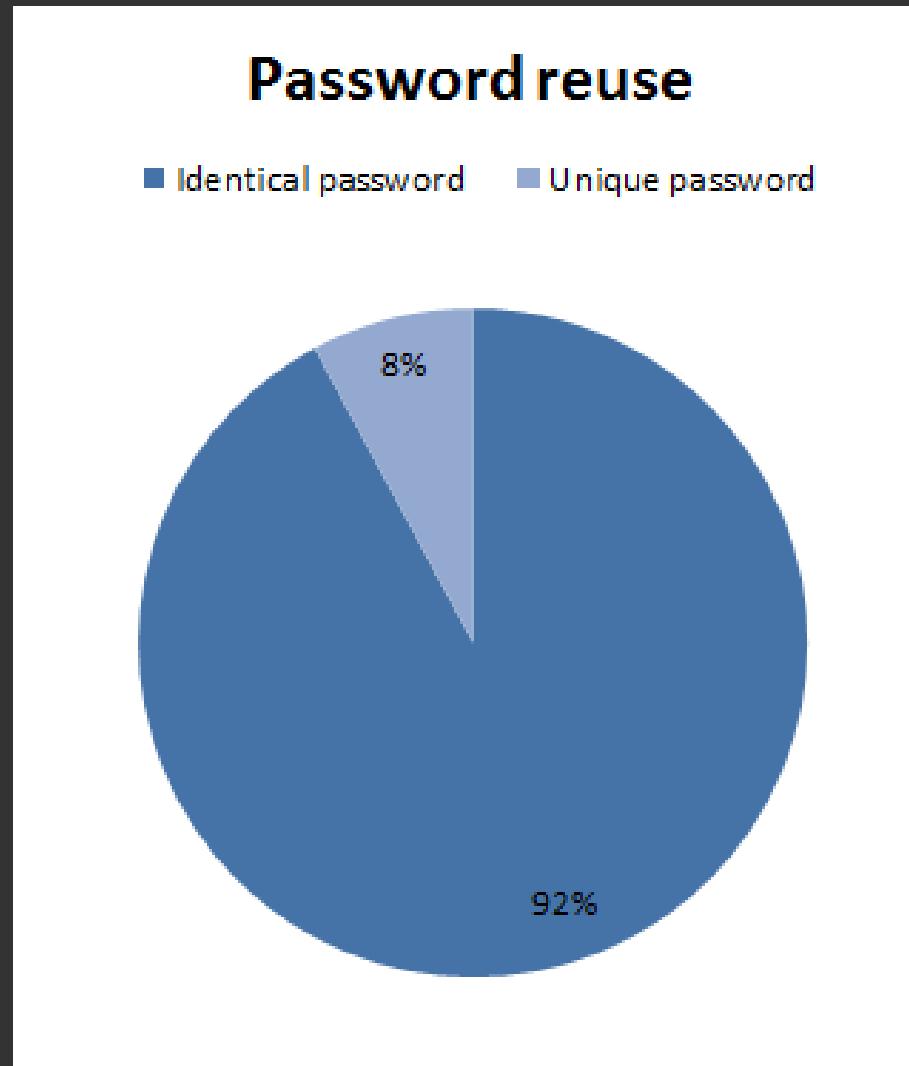


Fonte: PasswordResearch.com
Webroot Password Survey October 2010



Caso Sony

- O caso sony ocorreu em 2011.
- 77 milhões de senhas vazadas.
- Mais de 1milhão de senhas armazenadas sem criptografia.
- Somente 4% das senhas usavam 3 tipos ou mais de caracteres.

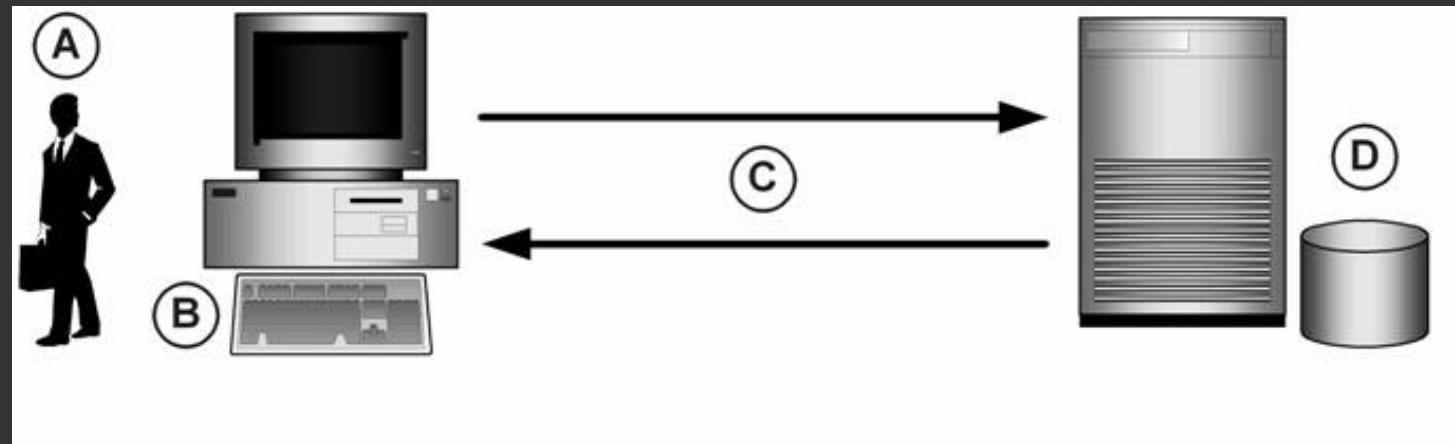


Fonte:Troy Hunt: A brief Sony password analysis

<http://www.troyhunt.com/2011/06/brief-sony-password-analysis.html>



Modelo básico de auth



- A) Fator de autenticação
- B) Entrada
- C) Transporte
- D) Verificação

Motivação

- Responda com sinceridade:
- Se você tivesse que quebrar uma senha, com qual das duas senhas abaixo, você gostaria que ela se parecesse?
 - **meunome1988**
ou
ASDBBUJO09877153\$*&*!@”)(

Relevância

- Dificuldade para o usuário de memorizar senhas complexas.
- Estudos mostram que maioria das senhas são pouco complexas e fáceis de quebrar.
- O reuso de senhas é recorrente.

Parte 2 - A biblioteca PAM



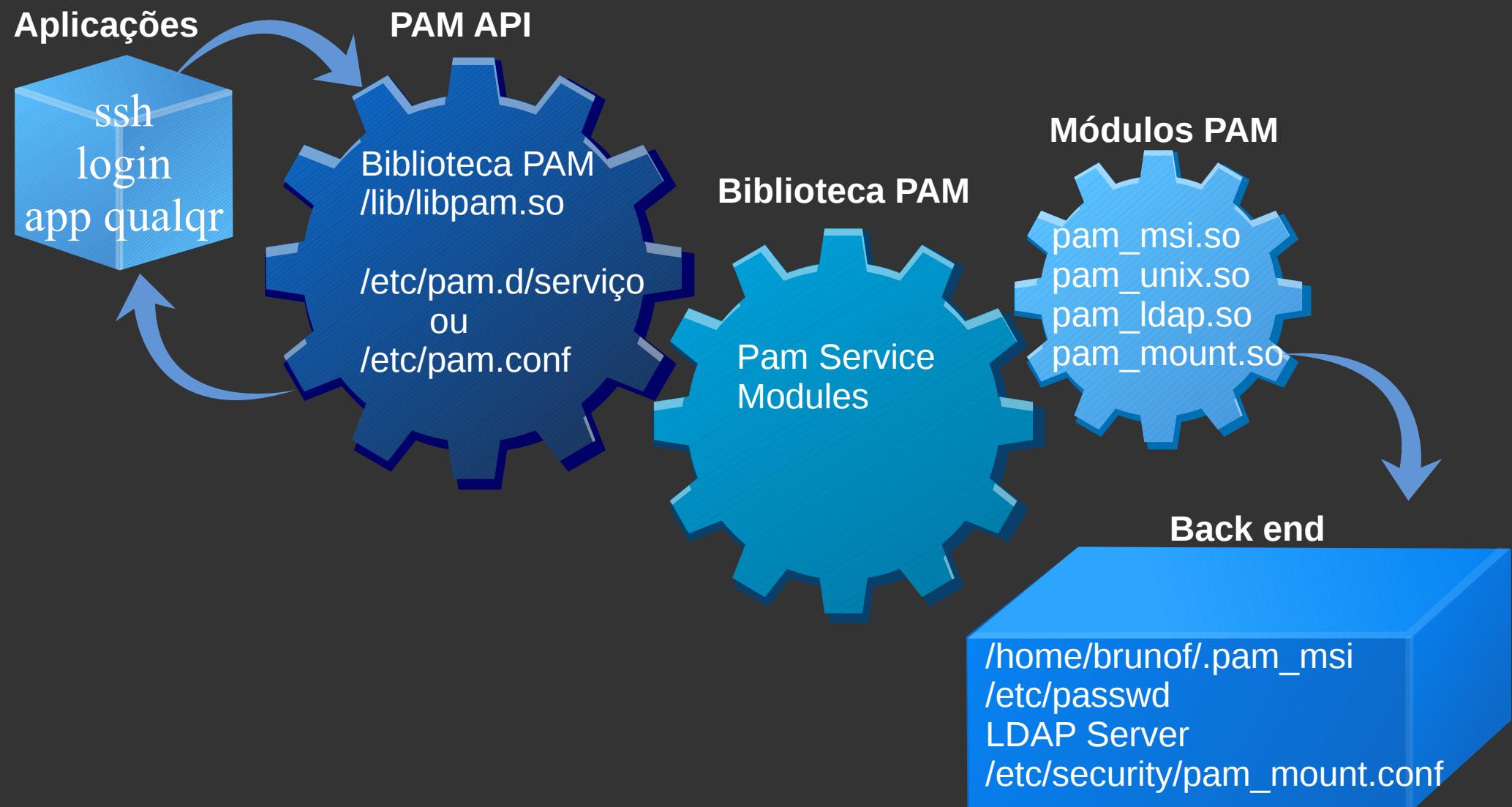
O que é?

- PAM(Pluggable authentication module) é uma API que cuida da autenticação de um usuário para um serviço.
- Os módulos do PAM (módulos de autenticação) são anexados aos aplicativos no tempo de execução para funcionar.
- Criado em 1995 por desenvolvedores da Sun Microsystems implementando um framework genérico para o Solaris.

Oque pode ser feito?

- Se voce usa linux voce usa pam(pam_unix.so).
- Autenticação 2 fatores(pam_opie.so).
- Preparar contexto seguro do SELinux(pam_selinux.so).
- Montar partições pam_mount.so

Esquema teórico



Sintaxe configuração PAM

type control module-path args

- Account

- Verifica os vários **aspectos da conta do usuário**, como limite de acesso em um período de tempo, ou de localização particular. Pode ser usado para limitar o acesso ao sistema baseado no sistema de recurso, ou expiração da conta.

- Auth

- Adquire da aplicação (prompt) dados do usuário para realizar a **identificação**, assim como uma senha.

- Password

- É normalmente empilhado no módulo auth. Ele é responsável pela **atualização** do token de autenticação do usuário, muita das vezes uma senha.

- Session

- É usado para prover funções antes e depois de **estabelecer a sessão**. Este contém um conjunto do ambiente, como montar home, tornar a mailbox disponível etc.



Sintaxe configuração PAM

type **control** module-path args

- required
 - Aqui a falha resultará na API retornando failure mas somente depois que os outros módulos forem invocados.
- requisite
 - Parecido com o required porém o controle e falha é imediatamente retornado ao serviço que o chamou.
- sufficient
 - O Sucesso de um módulo é suficiente para satisfazer os critérios de autenticação, a menos que um módulo anterior tenha falhado. A falha aqui não é fatal.
- optional
 - Sucesso ou falha deste módulo não é importante.



Sintaxe configuração PAM

type	control	module-path	args
------	---------	-------------	------

- Module-path
 - Pode ser tanto o caminho completo começando por "/" ou um caminho relativo para /lib/security ou /lib64/security

Sintaxe configuração PAM

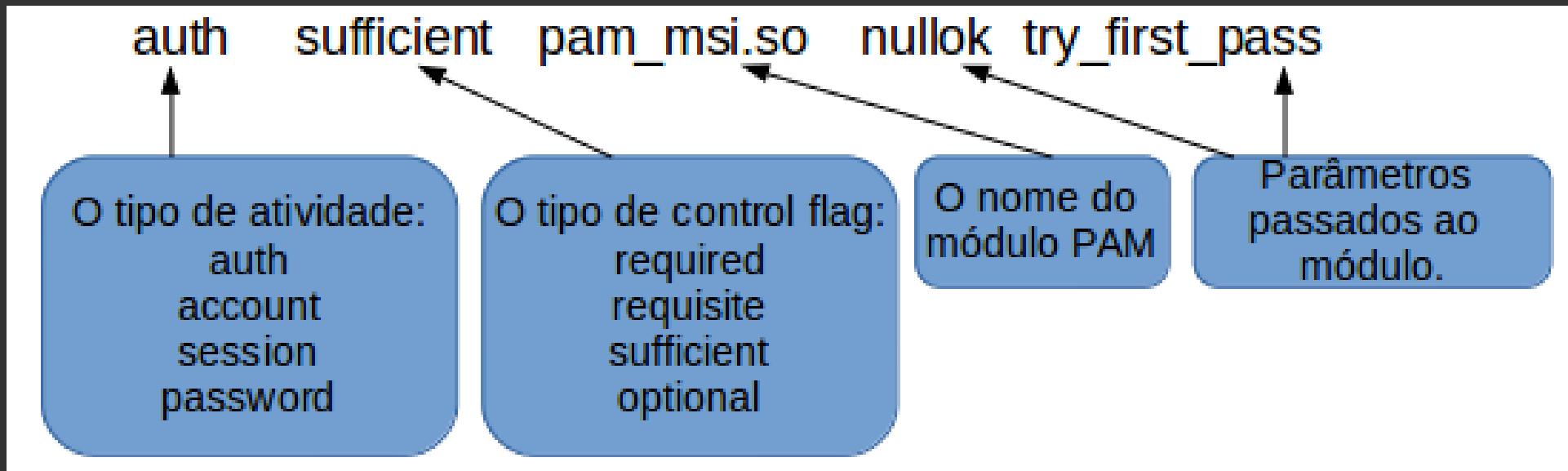
type	control	module-path	args
------	---------	-------------	------

- **Module-arguments**

- São opções delimitadas por espaço que podem modificar o comportamento do modulo em questão.
- Exemplo de argumentos padronizados:
debug no_warn use_first_pass try_first_pass
Obs:Nem todos os módulos trabalham com argumentos.

Exemplo configuração PAM

- /etc/pam.d/sshd



Parte 3 - Construindo sua biblioteca PAM



Ferramentas

- “pamdeveloper” starter pack.



Exemplo

```
1 #define PAM_SM_ACCOUNT ←
2 #define PAM_SM_AUTH ←
3 #define PAM_SM_PASSWORD ←
4 #define PAM_SM_SESSION ←
5 #include <security/pam_modules.h> ←
6
7
8 int pam_sm_open_session(pam_handle_t *pamh, int flags, int argc, const char **argv) {
9     return(PAM_SUCCESS);
10 }
11
12 int pam_sm_close_session(pam_handle_t *pamh, int flags, int argc, const char **argv) {
13     return(PAM_SUCCESS);
14 }
15
16 int pam_sm_acct_mgmt(pam_handle_t *pamh, int flags, int argc, const char **argv) {
17     return(PAM_SUCCESS);
18 }
19
20 int pam_sm_authenticate(pam_handle_t *pamh, int flags, int argc, const char **argv) {
21     return(PAM_SUCCESS);
22 }
23
24 int pam_sm_setcred(pam_handle_t *pamh, int flags, int argc, const char **argv) {
25     return(PAM_SUCCESS);
26 }
27
28 int pam_sm_chauthtok(pam_handle_t *pamh, int flags, int argc, const char **argv) {
29     return(PAM_SUCCESS);
30 }
```

Códigos de retorno mais comuns

Return code	Management group	Significado
PAM_SUCCESS	All(Auth, Password...)	Tudo ok
PAM_AUTH_ERR	Auth e Account	Erro de autenticação
PAM_SESSION_ERR	Session	Erro abrindo ou fechando a sessão



Testando

[PRÉ-REQUISITOS PARA COMPILEAR A BIBLIOTECA]

- libpam0g:amd64 1.1.8-1ubuntu2 amd64 Pluggable Authentication Modules library
- libpam0g-dev:amd64 1.1.8-1ubuntu2 amd64 Development files for PAM (/usr/include/security/pam_modules.h)
- pamtester 0.1.2-1 amd64 utility program to test the PAM facility
- gcc - GNU project C and C++ compiler
- crypt - password and data encryption

[COMPILEAR]

- gcc -fPIC -DPIC -shared -rdynamic -o pam_msi.so pam_msi.c -lcrypt



Parte 4 - Autenticação alternativa



Problema

- Reuso de senhas em vários serviços
- Senhas com baixa complexidade
- Dificuldade para memorizar senhas complexas



Solução

- Como usar senhas complexas como:
imnegucopm@08975@)(*%!@)+*
que mudam a cada login, sem perder a
sanidade decorando.



Exemplo

- Exemplo: Senha com 27 caracteres contendo letras, números e símbolos.

imnegucopm@08975@)(*%!@)+*

The screenshot shows a password strength checker interface. At the top, the question "HOW SECURE IS MY PASSWORD?" is displayed in large, white, sans-serif font against a dark green background. Below this, a large input field contains a password consisting of 27 characters, represented by a series of black dots. To the right of the input field is a "SHOW SETTINGS" button. The main content area has a light green background and displays the following text: "It would take a desktop PC about 180 duodecillion years to crack your password". Below this, there is a "[Tweet Result]" button. At the bottom of the main content area is a "HIDE DETAILS" button. The footer section, which has a dark background, contains the following information: "Length: 27 characters", "Character Combinations: 145", "Calculations Per Second: 4 billion", and "Possible Combinations: 22 octodecillion".



Solução

- Autenticação via módulo PAM customizável.
- O usuário configura o seu padrão de senha. A senha em si não é o que importa mas sim seu padrão.
- Exemplo de padrão: 5 caracteres minúsculos + 5 dígitos numéricos + 5 caracteres maiúsculos.



Autenticação alternativa

5 Uppercase 5 Lowercase 5 Digits No Symbols 15 Characters 

sword55691LLPQZ 

Enter and edit your test passwords in the field above while viewing the analysis below.

5 Uppercase 5 Lowercase 5 Digits No Symbols 15 Characters 

xxvbq18964ZAZNB 

Enter and edit your test passwords in the field above while viewing the analysis below.

5 Uppercase 5 Lowercase 5 Digits No Symbols 15 Characters 

mwpir99958LPOQV 

Enter and edit your test passwords in the field above while viewing the analysis below.



Video gerapam



Proteção

- Senhas com “*search space*” maior.
- Interceptações em geral, seja física ou lógica(Shouldersurfing, Keylogger, MITM.)



Parte 5 - Considerações finais



Vantagens do PAM

- Bem consolidado, décadas de serviço
- Flexibilidade
- Empilhamento de módulos distintos
- Agregar conhecimento
- Crie seu próprio padrão de senhas.



Considerações finais



GYUQ84vNeFBTPg3ZMdDznN5wsfGCplhdvWS9BwkKbupQM4vWeaBxPfyuqQ8z0I
gbCUQhd0WRF6wXKbCUIMd0rnFBxXgc7VRMIOXnFBxYgHD3IMI0Wnj62tpc8z0New
BoPLbuVQh4vWJFBxPLbCqRhA1rSa62kLcuqRi
XkCUHvhav,
NJ1soG73YhcEWmNJwXTP73Zpl95mOeFs
C3QMcuqRE51jK
NoOK3tpIDz0NeaXTPgytpl8z0WSFB1Pg
bQ8zviJEXSPCxZLHDVmN5wXfGCUQh4
3lg4RNz1OJ2YgxIhz
RNJ1sSG62PLHZqIE5nOf2YUH7yRid1
NeF6oP6xQ83mNAwjfxPLbZVQM40re
pjC3YLH8qR9A1OfGYpQ84uNIernjBxYge
FXKxQ8qQEre1O7
StUQ840ie9Xnjf2tpHDzmieFXokC3ugc9
IzREWewsKxk7V
fGYpID40WJaBTb7yIM40qmE51OKGYpl8z
fxupQ8zvNJaATjKGtUQ840NJF6oPKxtUmDz
iK2tTH84R
ypl8zvWe9AwOKbYpQh40VJa6oOgGtpH8zv
mi5wXKG7pQhz0qeFWSP73tgcuqmE5wjfxYpc
I1sTGBpYPMcuNd1WnF6xYLHDVmMdwWSaB2Pg
95wNKGBokgyZqdEASOKGYplD4Zre9BSjfb
IRiAwrbfbBpP84uhz1jexXgGupcDzmi6wOfxt
TGC3Qhl0WRa6TjKHYUI8zmNe1XTG7pQhz0Vda
G73Qhd0WeFBoPg3uUd9zmie1sob7UQMzvrJF
sTk8vZhdEWSOB2YLcDUQMz0F6GteUHDalwreas
TG7bdDam95nbBoH

É só uma questão de tempo...



Considerações finais

- 3 questões básicas

O que eu quero proteger?

De quem eu preciso proteger?

Qual o esforço e investimento necessário para a proteção adequada?



Observações

- Não é solução final, é apenas um reforço pois a segurança é feita em camadas!



Documentação

Site oficial do projeto -
<http://www.linux-pam.org/>



Open Software Foundation RFC 86.0 - V.
Samar (SunSoft), R. Schemers(SunSoft)
<http://www.opengroup.org/rfcref/rfc86.0.html>

- Repositório GITHUB:
 - <https://github.com/bruno-sf/ufrj-msi.git>

Perguntas?

