Carteiras em Bitcoin

Carteiras são recipientes para chaves privadas. Estas podem ser geradas de maneira aleatória (usando uma moeda), de forma pseudo-aleatória (usando pedaços de software certificados) ou através de geração determinística de chave. Neste terceiro método cada chave privada nova é derivada a partir de uma chave privada prévia usando-se uma função de hash que liga as diversas chaves em sequência. Para recriar a série de chaves você só necessita a primeira (chamada de semente, seed, ou chave mestre).

O nome carteira não é lá grande coisa. Simplesmente por que as carteiras não contém bitcoins e sim pares de chaves (privada e pública).

Nos primeiros clientes bitcoin, as carteiras eram coleções de chaves privadas geradas aleatoriamente. Este tipo de carteira é apelidada de JBOK (*just a bunch of keys*=apenas um monte de chaves) e tais carteiras estão caindo de desuso, substituídas por carteiras determinísticas. A razão é que carteiras JBOK são difíceis de administrar, fazer backups e manusear. Se quiser não reutilizar chaves (uma boa política) vai precisar gerar e manter um nonte delas, precisando fazer backups frequentes de todas elas. Lembre-se que fundos geridos por uma chave serão irremediavelmente perdidos se a carteira se tornar inacessível. Em dezembro de 2017 apareceu a notícia:

> Um técnico de informática britânico planeja escavar um aterro sanitário para recuperar um disco rígido de computador que contém o equivalente a 276,6 milhões de reais em bitcoins. O equipamento foi jogado fora por engano há quatro anos. Com o aumento astronômico da cotação da moeda virtual nos últimos tempos, a fortuna guardada nele aumentou consideravelmente... (veja, 5/12/17)

Para evitar este problema pode-se reutilizar endereços, mas isto reduz a privacidade na medida em que associa múltiplas transações e endereços uns com os outros. Em resumo, não é uma boa idéia usar carteiras JBOK, também conhecidas como carteiras do tipo-0.

A evolução veio no BIP0032/BIP0044 (propostas de melhoria do ambiente Bitcoin). Agora, todas as chaves privadas são geradas a partir de uma semente e esta (a semente) é a única coisa que precisa ser *backupeada*. Com ela, todas as chaves podem ser regeradas. Para variar usa-se aqui uma função hash a partir da semente e de um número índice também conhecido como código de corrente.

A semente também é a única coisa necessária para importar e exportar a carteira permitindo a fácil migração de todas as chaves entre diferentes implementações da carteira.

Essa proposta de melhoria (BIP0039) veio com outra idéia luminosa: uma lista de 2048 palavras fixas que descrevem a senha: Eis a razão: uma lista de dígitos maiusculos e minúsculos misturados a números, pode gerar uma dificuldade na hora de copiar/transmitir/guardar a senha.

Agora a senha é convertida em binário e os dígitos binários são truncados em números de 11 bits ($0 \leq d \leq 2047$) e cada número assim formado endereça uma palavra padrão e fixa em inglês. Dessa maneira, a senha vira uma lista de 12 a 24 palavras, facilmente copiáveis, ditáveis ou coisas assim.

O início da lista é

```
abandon, ability, able, about, above, absent, absorb,
abstract, absurd, abuse, access, accident, account,
accuse, achieve, acid, acoustic, acquire, ...
```

Essa proposta de codificação propõe o seguinte algoritmo:

- Cria uma sequência aleatória de 128 a 256 bits
- Cria um checksum, pegando alguns bits iniciais do SHA256
- Adiciona o checksum à direita da sequência aleatória
- Divide a sequência em partes de 11 bits
- recupera 12 a 24 palavras da lista padrão

Veja-se a propósito as regras de tamanho

bits	checksum	ambos	palavras
128	4	132	12
160	5	165	15
192	6	198	18
224	7	231	21
256	8	264	24

Este código é usado para derivar uma semente maior de 512 bits através da função de extensão de chave PBKDF2. Tendo interesse localize esta referência na Internet.

As carteiras determinísticas são conhecidas como HD (carteira determinística hierárquica) e elas têm formato de árvore. Na raiz está a semente original e dela derivam as senhas filho. Cada senha filho pode gerar infinitas senhas neto e assim por diante, sem limite.

Uma vantagem adicional desta estrutura de árvore é que ela pode ser usada para exprimir significado organizacional adicional, como por exemplo quando um ramo específico é usado para receitas de um estado e outro ramo para receitas da matriz ou ainda para trocos de pagamentos feitos.

Outra vantagem da carteira HD é que usuários podem criar sequências de chaves públicas sem precisar acessar as chaves privadas correspondentes. Isto permite que carteiras HD estejam em um servidor inseguro ou que alguém possa gerar muitas chaves públicas.

O processo de criação ... pág 38 e seguintes...

Exemplo Seja o conjunto

```
boleto,viaduto,rato,vitrine,borracha,almofada,premio,programa,
vidro,sol,feio,professor,banana,jornal,laranja,figo,
praia,caneta,zebra,bonita,duque,pinheiro,padaria,prova,
jiboia,montanha,loja,viagem,bebado,empadao,tabela,vedete,
saturno,salario,numero,azar,tribunal,comida,panela,xuxu,
trinco,abobora,ponte,desconto,morango,chave,logica,arvore,
pulseira,alcatra,avestruz,vereador,atraso,cigarro,miseria,teclado,
curitiba,gato,radio,picareta,encosto,cadeira,hotel,sorte
```

A senha 11.06.D1.C4.D8.EC ficaria:

```
borracha,praia,viagem,caneta,alcatra,jornal,azar,morango
```

☞ Para você fazer

Suponha uma tabela com 64 palavras ($64 = 2^6$).

```
advogado,vedete,anel,professor,tribunal,pinheiro,beringela,jornal
gato,ceu,panela,baronesa,feio,peralta,basquete,logica
gigante,hotel,cigarro,sortudo,radio,rainha,arvore,rei
desconto,vendedor,futebol,quebrar,chave,capacete,filosofia,morango
teclado,programa,caneta,comida,jiboia,numero,pagamento,encosto
atraso,borracha,abobora,coxinha,loja,viagem,quirera,duque
azar,xuxu,fisica,laranja,salario,bonita,praia,azia
empadao,prova,saturno,vidro,palmeira,cavalo,papel,vitrine
```

1. Suponha uma senha hexadecimal, formada por 6 bytes (12 caracteres hexadecimais) e cuja representação em palavras é

```
cavalo,capacete,capacete,numero,radio,beringela,vedete,baronesa
```

Ache a representação hexadecimal dessa senha.

2. Agora suponha uma senha de 6 bytes, cuja representação hexadecimal é

```
C4.06.C7.7F.81.CF
```

Escreva a primeira e a última palavras da representação textual da senha.

senha hexadec	prim e últ palavras