MD5 no Android como mecanismo de autenticação na API do Google Maps

Claudio André
claudio.andre@correios.net.br

2011

MD5 no Android como mecanismo de autenticação na API do Google Maps

Sumário

Primeira parte

- Contextualização;
- Autenticação;
- Função Hash; MD5;
- O MapView (vendo mapas sem navegador);

Segunda parte

- Como funciona a autenticação no MapView;

Terceira parte

- Conclusão;
- Palavras Chave.

MD5 no Android como mecanismo de autenticação na API do Google Maps

Contexto do Problema

- O Google deseja criar um controle de tela Android que permita que os desenvolvedores usem mapas em seus aplicativos.
- Como controlar o acesso dos dados do Maps?
- Como evitar que os desenvolvedores copiem o banco de dados de mapas (até para criar uma ferramenta de mapas concorrente).
- Mas, o usuário final deve acessar os mapas livremente.
- Logo, o Google precisa controlar o acesso e uso feito pelos desenvolvedores sem impactar os usuários finais dos aplicativos criados por estes mesmos desenvolvedores.

Autenticação

- Autenticação é o ato de confirmar se algo é verdadeiro;
- No nosso contexto, a verificação da identidade de uma pessoa, processo, serviço, etc.
- Mas, como identificar um telefone? Ou uma aplicação? Podemos identificar uma aplicação? Colocar nela um marcador?
- Sim. E o Google ao aliar assinatura digital com chaves de acesso permite que se realize a autenticação e, consequentemente, o controle do uso de seus recursos na nuvem.

Função de hash

- Função publicamente conhecida; Da forma: *h = H(M);*
- Recebe um texto de tamanho variável e produz uma saída de tamanho fixo;
- Pequena mudança na entrada => saída muito diferente.
- Utilizaremos neste trabalho o MD5 (Message-Digest Algorithm ou algoritmo de resumo de mensagem). Largamente conhecido e utilizado. Produz uma saída de 128-bits (16-bytes).
- MD5 é empregado em uma variedade de aplicações de segurança e na verificação da integridade de dados e arquivos.
- Desenvolvido em 1991 por Ron Rivest para substituir o MD4 (MD6 já está em desenvolvimento).

Função de hash

Requisitos de uma função hash H:

1. H deve ser aplicável a um bloco M de qualquer tamanho.
2. H deve produzir uma saída h de tamanho fixo.
3. H(x) deve ser relativamente fácil de ser calculado para qualquer mensagem x. Implementação prática, tanto em software quanto em hardware.
4. A função H não deve ser reversível. Dado o valor h = H(x) deve ser computacionalmente impraticável calcular o valor de x.
5. Dado qualquer bloco x deve ser computacionalmente impraticável calcular outro bloco y tal que H(x) = H(y).
6. Deve ser computacionalmente impraticável encontrar um par qualquer (x; y) tal que H(x) = H(y).

O MapView

- MapView é uma classe das bibliotecas do Android.
- Permite que se integre o Google Maps a uma aplicação qualquer desenvolvida para Android.
- Por exemplo, usando o GPS do aparelho, minha aplicação pode localizar a agência mais próxima e mostrar o mapa do caminho até lá.
- A classe provê uma série de funções para mostrar, salvar, colocar no cache e controlar os mapas.

Como Funciona?

- Para usar o MapView e obter acesso aos dados do Google Maps, o Google exige que os desenvolvedores registrem e aceitem as regras do serviço. *Isto permite o controle do uso gratuito, atende aos interesses comerciais do Google e de comercialização de serviços;
- Reproduzindo os termos do Google sobre as restrições: ”*To ensure that applications use Maps data in an appropriate manner*”.
- O registro é feito usando o fingerprint MD5 do certificado que será usado para assinar a aplicação [1,2].

1 – o Android exige que as aplicações instaladas sejam assinadas digitalmente, usando o conhecido mecanismo de assinatura digital.

2 – não são exigidos uso de autoridades certificadoras. Certificados ”particulares” servem.

Como Funciona?

Portanto, para utilizar o MapView e conseguir acessar os dados do Maps (na nuvem), o desenvolvedor deve:

- Obter o MD5 fingerprint do certificado que será usado para assinar o aplicativo que usará o MapView. Usando, por exemplo, o software keytool;
- Ir ao site de registro do Maps API;
- Criar uma conta no Google ou usar a sua (pré existente);
- Cadastrar o MD5 fingerprint do certificado no site Android Maps API usando (relacionando o fingerprint a) sua conta Google.
- O servidor, então, irá relacionar o fingerprint MD5 e sua conta de desenvolvedor e irá retornar uma chave (map key).
- Usando esta chave, o desenvolvedor deve configurar a aplicação, ajustando o campo android:apiKey dos arquivo XML de layout do MapView.

Como Funciona?

Logo:

- É usado o MD5 fingerprint do certificado digital;
- O MD5 (logo, indiretamente, o certificado) é relacionado a uma conta do Google.
- O certificado é usado para assinar digitalmente o(s) aplicativo(s) que executarão no Android.
- Usando MD5 fingerprint e os dados do usuário é gerada uma string (a map key) que deve ser usada para nos arquivos XML de layout para validar (autenticar em tempo de execução) se o aplicativo pode acessar os dados do Maps.
- Desta forma, mau uso dos dados do Maps pode ser detectado e relacionado a um desenvolvedor.
- O desenvolvedor pode, então, ser notificado e seu acesso bloqueado, se necessário.

Como Funciona?

Garantias oferecidas:

- Como se usam certificados e assinatura digitais (o aplicativo será assinado digitalmente usando a chave privada e verificado com a chave pública), e o MD5 do certificado é relacionado com a conta do usuário, não é possível usar o Maps sem a chave correta, pois verifica-se:
- A map key do MapView está relacionada a um certificado/desenvolvedor?
- A map key em uso pelo MapView da aplicação está relacionada ao certificado que assinou a aplicação. Ou seja, foi o certificado que assinou a aplicação o mesmo usado para gerar a map key?

- Se estes testes falham, o acesso ao dados do Maps é negado.
- Lembre-se (em tese) não é possível acessar a chave privada de outra pessoa.
- Chutar ou copiar uma map key de outra pessoa é detectado pelos dois testes listados acima.

Conclusão

- Google encontrou uma forma interessante de utilizar assinaturas digitais e MD5 para resolver um problema novo;
- O problema do Google era como permitir o acesso e uso “justo” dos dados do Google Maps. Ou seja, garantir que o usuário final obtenha acesso aos dados que lhe sejam úteis, prevenindo, porém, que os desenvolvedores usem robôs para “chupar” os dados do Maps.
- Em outras palavras, garantir o acesso controlado aos dados.

Conclusão

Basicamente:

- A forma de resolver um problema conhecido, uma técnica, uma ferramenta, um algoritmo podem ser aplicados com sucesso como (parte) solução em outros problemas.
- Para isto, é necessário ser criativo e manter a mente aberta para as possibilidades.

Conclusão

- Existem usos para hash na computação ubíqua, inclusive em celulares;
- Mais importante que a tecnologia em si, são as possíveis aplicações que se pode dar a estas tecnologias.
- A computação em nuvem, a massificação dos celulares, a vulnerabilidade intrínseca a estas novas tecnologias exigem que a segurança seja pensada e adaptada a novas necessidades.

Conclusão

- A técnica foi pensada para resolver um problema do Google, não os de seus usuários;
- O uso do MapView e seu processo de autenticação é transparente para o usuário final;
- Para o desenvolvedor o uso é simples e descomplicado. Apenas um pequeno trabalho burocrático é inserido. E este não precisa ser repetido ou revalidado em futuras versões do software.

Palavras Chave

- Funções Hash, MD5;
- Assinatura digital;
- Autenticação;
- A revolução causada pelos smartphones;
- Computação ubíqua, computação pervasiva;
- Computação na nuvem;
- Android.

Obrigado

Mais Informações

http://code.google.com/intl/pt-BR/android/add-ons/google-apis/mapkey.html

http://en.wikipedia.org/wiki/MD5
http://www.claudioandre.drivehq.com

Claudio André
claudio.andre@correios.net.br