Instituto Superior de Engenharia de Lisboa
Licenciatura/Mestrado em Engenharia Informática e de Computadores
Segurança Informática
Segunda série de exercícios, Semestre de Inverno de 16/17
Data de entrega: 9 de Dezembro de 2016

---

1. No contexto do protocolo TLS:

   1.1. Quando o *handshake* é baseado na primitiva RSA, de que forma é protegido o envio do *pre-master secret* entre o cliente e o servidor? Esta proteção garante a sua confidencialidade e integridade?

   1.2. Qual a importância da propriedade *perfect forward secrecy*? O estabelecimento de chaves usando cifra assimétrica fornece esta propriedade?

   1.3. Quantas chaves usa o *record protocol*? Porque motivo são usados conjuntos de chaves diferentes em cada sentido da comunicação?

2. Considere uma aplicação *web* que autentica os seus utilizadores com base em *passwords*.

   2.1. Admita que foi utilizado um *salt* de 4 bits para produzir a informação de validação. Se o atacante quiser pré-calcular o dicionário para este cenário, quantos cálculos terão de ser feitos para cada entrada do dicionário?

   2.2. Considere que a aplicação web mantém o estado de autenticação num *cookie*. Qual destes dois esquemas deve ser sempre usado: esquema de cifra simétrico ou esquema MAC?

3. Explique duas diferenças entre os esquema de autenticação HTTP *Basic* [1] e *Digest* [2]. Estes esquemas de autenticação garantem a confidencialidade da *password*?

4. No contexto da *framework* de autorização OAuth 2.0 :

   4.1. No fluxo *authorization code*, durante um pedido de autorização (*authorization request*), se o dono de recursos já estiver autenticado no cliente e no servidor de autorização, o formulário de consentimento é apresentado?

   4.2. Qual a vantagem do fluxo *authorization code* em comparação com o fluxo implícito?

5. No contexto do fluxo *authorization code* do protocolo OpenID Connect:

   5.1. Para que serve o ID Token?

   5.2. Qual destas duas entidades desempenha o papel de relying party: a aplicação cliente ou o *resource server*?

6. Adicione ao programa desenvolvido na primeira série de exercícios a possibilidade de usar uma chave derivada de uma *password* para transportar a chave que cifra a mensagem. Utilize a norma PKCS#5 [4] para derivar a chave a partir da *password*, tal como descrito na Secção 4.8 do RFC 7518 [5].

7. Realize uma aplicação Web com as seguintes funcionalidades:

   - Os utilizadores são autenticados através do fornecedor de identidade social Google, usando o protocolo OpenID Connect [6].
   - Os utilizadores autenticados podem consultar a listagem de *issues* de um projecto GitHub [7].
   - Os utilizadores autenticados podem criar uma *task* Google a partir de *issues* do GitHub [8].

Referências

[1] `https://tools.ietf.org/html/rfc7617`, visitado a 8 de novembro de 2016.

[2] `https://tools.ietf.org/html/rfc7616`, visitado a 8 de novembro de 2016.

[3] `https://github.com/hueniverse/hawk`, visistado a 8 de novembro de 2016.

[4] `https://tools.ietf.org/html/rfc2898`, visitado a 31 de outubro de 2016

[5] `https://tools.ietf.org/html/rfc7518#page-20`, visitado a 31 de outubro de 2016

[6] https://developers.google.com/identity/protocols/OpenIDConnect, visitado a 31 de outubro de 2016

[7] https://developer.github.com/v3/issues/, visitado a 31 de outubro 2016.

[8] https://developers.google.com/google-apps/tasks/v1/reference/, visitado a 31 de outubro 2016.