



3ª Entrega: Sistema Binas

SISTEMAS DISTRIBUÍDOS

2º SEMESTRE – 2017/2018

|  |  |  |
| --- | --- | --- |
| Grupo 8  https://github.com/tecnico-distsys/T08-SD18Proj.git | | |
|  |  |  |
| André Fonseca  84698 | Diogo D’Andrade  84709 | Leonor Loureiro  84736 |

Grupo 8

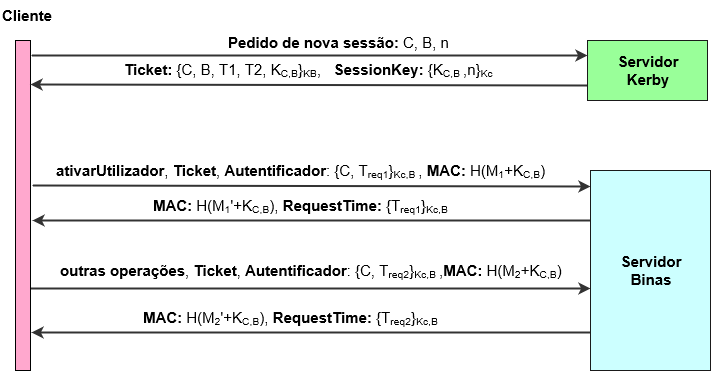
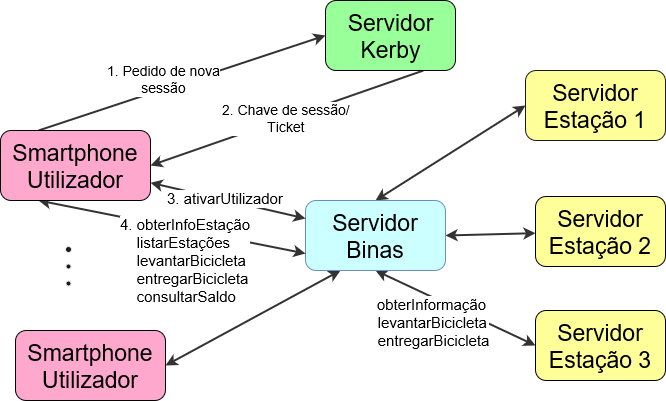
Protocolo Kerberos (versão V5) – Versão Simplificada

Figura 2: Trocas de Mensagens

Figura 1: Diagrama da Solução

|  |  |
| --- | --- |
| Login | 1. Cliente envia pedido de nova sessão ao servidor Kerby, incluindo no pedido um nonce, n 2. O Kerby retorna  * a chave de sessão, {KC,B, n}Kc, e * o ticket respetivo, T = {C, B, T1, T2, KC,B}KB |
| Acesso ao Binas | 1. Cliente decifra a chave de sessão, obtendo KC,B 2. Cliente gera um novo autenticador, A = {Treq}Kc,B 3. Cliente gera o resumo do pedido, S1 = H(M+KC,B) 4. Cliente invoca a operação ativarUtilizador do servidor Binas, incluindo no pedido:  * o ticket T, * o autenticador A, e * o resumo S1  1. Binas decifra o ticket e verifica a sua frescura. 2. Se o ticket ainda estiver no período de validade, 3. computa o resumo S1’, e 4. verifica se S1=S1’ 5. Binas decifra o autenticador A, obtendo o request time, Treq 6. Verifica se o Treq coincide com o *timestamp* atual, com uma margem mínima de erro, e não coincide com o *request time* de nenhum pedido anterior. 7. Binas executa o pedido |
| Autenticação do Binas | 1. Binas gera o resumo da mensagem de resposta, S2 = H(M’+KC,B) 2. Binas retorna, incluindo na resposta:  * o request time, Treq, encriptado com a chave de sessão KC,B, e * o resumo S2  1. Cliente computa o resumo S2’=H(M’+KC,B) 2. Verifica se S2=S2’ 3. Decifra Treq e verifica se coincide com o *timestamp* atual, com uma margem mínima de erro |

SOAP Handlers

Cliente:

* Mensagens *OUTBOUND*:
  + **Autenticação**

1. Verificar se tem um ticket válido
2. Caso não tenha, efetuar pedido de nova sessão ao Kerby
3. Adicionar à mensagem um *header* com o ticket
4. Decifrar a chave de sessão, usando a chave do cliente
5. Gerar um novo autenticador com *timestamp* atual
6. Encriptar o autenticador, usando a chave de sessão
7. Adicionar à mensagem um *header* com o autenticador
8. Guardar a chave de sessão e o timestamp usada no autenticador, para posteriormente tratar a mensagem de resposta
   * **Assinatura Digital (MAC)**
9. Computar o *digest* (MAC) da chave de sessão + conteúdo da mensagem
10. Adicionar à mensagem um *header* com o *digest* (MAC)

* Mensagens *INBOUND*:
  + **Autenticação**

1. Extrair o *request time* do *header*
2. Decifrar o *request time*
3. Verificar que o *request time* decifrado coincide com o *request time* do pedido
   * **Assinatura Digital (MAC)**
4. Extrair o *digest* (MAC) do *header*
5. Computar o *digest* da chave de sessão + conteúdo da mensagem
6. Verificar que coincide o *digest* extraído coincide com o computado

Binas:

* Mensagens *INBOUND*:
  + **Autenticação**

1. Extrair o ticket e decifra-lo, obtendo a chave de sessão
2. Verificar que o nome do servidor no ticket está correto
3. Verificar que o ticket ainda não expirou
4. Extrair o autenticador e decifra-lo
5. Verificar que o *username* no ticket coincide com o *username* no autenticador
6. Guardar o *request time* e o username no contexto da mensagem

* **Autorização**

1. Verificar que o *request time* decifrado coincide com o *timestamp* atual, com uma margem mínima de erro e que não corresponde ao *request time* de nenhum pedido anterior.
2. Verificar que o cliente com o *username* dado tem permissões para executar a operação pedida
   * **Assinatura Digital (MAC)**

* Mensagens *OUTBOUND*:
  + **Autenticação**

1. Encriptar o *request time* do pedido, usando a chave de sessão
2. Adicionar um *header* à mensagem com o *request time* do pedido
   * **Assinatura Digital (MAC)**