

2º Parte do Projeto de Sistemas Distribuídos

A47

*URL do repositório do GitHub:* <https://github.com/tecnico-distsys/A47-SD18Proj>

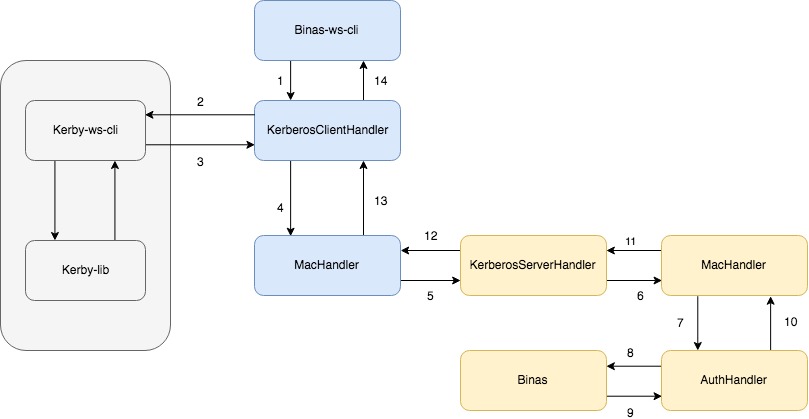




83521 83539 83540

Mariana Mendes Pedro Caldeira Pedro Lopes

**Figura da solução de Segurança**



**Descrição da Figura**

A figura consiste na troca de mensagens desde o Binas-ws-cli, ou seja, o cliente, até ao Binas, servidor de forma a garantir a integridade do pedido e o controlo de acessos. Para tal, como mostra a figura, foram implementados handlers que ficam encarregues da transmissão e receção de mensagens SOAP que contém os dados do pedido codificados por chaves. O Cliente envia um pedido para o servidor. Este pedido primeiramente passa pelo KerberosClientHandler que solicida o ticket, de acesso ao server, ao kerby-ws-cli. Após a obtenção do mesmo envia a mensagem para o MacHandler que verifica se o pedido não foi alterado do lado do cliente. Caso este se mantenha inalterado aquando da chegada segue para o KerberosServerHandler para verificar a validade do ticket obtido anteriormente. Em caso positivo, a mensagem é enviada para o MacHandler onde esta irá novamente ser inspecionada para detetar alterações, caso existam. De seguida a mensagem segue para o AuthorizationHandler que garantirá o controlo de acessos, querendo isto dizer que o utilizador X apenas pode aceder e realizar operações ao domínio a ele correspondente e não a um domínio alheio, assim como nenhum outro utilizador pode aceder ao seu domínio. Daqui o pedido é enviado para o servidor, Binas, que envia uma resposta. Resposta essa que percorrerá o percurso inverso até ao Cliente, sem passar pelo módulo kerby.

**Breve Explicação da Solução**

A solução aplicada ao problema consiste em utilizar o um módulo Kerby que tem como objetivo fornecer tickets de acesso ao servidor para garantir assim a autentificação dos utilizadores de forma a tornar os acessos mais seguros através do uso de chaves, este processo irá ser descrito em maior detalhe na secção seguinte. Para além da autentificação, a integridade também é um aspeto a ser protegido pelo que, na nossa solução foi implementado, como dito anteriormente, um MacHandler que encarregado de, quando recebe uma mensagem/pedido, verificar se este foi comprometido por terceiros. Para tal, o handler em questão, quando recebe uma mensagem, utiliza uma função de resumo (digest) que converte uma sequencia de texto numa sequência de bits de comprimento fixo para ser comparada. Alterações na mensagem provocam alterações no valor calculado pela função de resumo pelo que permite saber se a mensagem foi adulterada. Para garantir a ultima propriedade referida na secção anterior, o controlo de acessos, introduzimos um AuthorizationHandler (AuthHandler) na Handler Chain de modo a que, antes de um utilizador poder efetuar uma ação, esta tem de ser autorizada.

**Protocolo detalhado**

O protocolo adotado foi o Kerberos simplificado. Ao contrário do Kerberos completo, na versão simplificada, apenas usamos o componente Saut que é responsável por, ao receber uma mensagem com o email do cliente e do servidor, juntamente com o nounce, um numero gerado aleatoriamente, enviar uma mensagem SOAP composta por um header que contém a session key ( {Kcs , n}kc ) e o ticket de acesso ao servidor. O ticket entre o cliente e o servidor é constituído pelo email de ambos, dois timestamps que representam o intervalo de tempo de validade do ticket e pela chave kc,s gerada. Este é ainda cifrado pelo Ks, chave do servidor.

Quando o cliente recebe a session key e o ticket, envia uma mensagem para o servidor. Esta contém o ticket, o autenticador (auth) e a operação que o cliente deseja que o servidor realize. O autenticador é constituído pelo email do cliente e pelo tempo em que o pedido foi efetuado (Treq). Estes estão cifrados pela chave partilhada entre o cliente e o servidor. A mensagem irá atravessar a handler chain e após ter chegado ao servidor este efetua o pedido e devolve uma mensagem SOAP composta pela resposta ao pedido e pelo Trep cifrado pela chave partilhada.

**Conteúdos das mensagens SOAP**