Universidade do Minho - Dep.to Informática
1º Semestre, 2022/2023
Mestrado em Engenharia de Telecomunicações e Informática
Módulo de Gestão de Redes

Ficha de Trabalho Prático

Proxy SNMPv2c

Objectivos:

- Consolidação dos conhecimentos sobre os protocolos, mecanismos e filosofias da arquitetura de gestão *Internet-standard Network Management Framework* (INMF), dando especial relevo aos aspetos de segurança e controlo de acesso.
- Consolidação dos conceitos principais sobre ameaças de segurança em aplicações/serviços distribuídos, métodos, tecnologias e estratégicas que permitem implementar garantias de segurança e efetivar níveis adequados de mitigação.

Observações:

- O trabalho deverá ser realizado ao longo de 20 a 30 horas efetivas de trabalho em grupo.
- Os grupos podem ter até dois elementos.

Requisitos:

- Acesso a sistema com, pelo menos, um pacote *freeware* instalado com suporte a SNMP (versão 2, no mínimo): **Net-SNMP**, CMU-SNMP, SCOTTY, etc.
- Utilização opcional de APIs de programação que facilitem a implementação de primitivas SNMPv2c ou SNMPv1.

AVISOS:

- Não serão tolerados atropelos aos direitos de autor de qualquer tipo de *software*...

Bibliografia específica e material de apoio

Material de apoio:

- Manuais/Tutoriais do *net-snmp*;
- MIBs em `/usr/share/snmp/mibs` (ou diretoria equivalente da instalação);
- Recurso `http://net-snmp.sourceforge.net/wiki/index.php/Tutorials/`;
- Recurso `http://www.simpleweb.org/`;
- Recurso `http://www.snmplinks.org/`.

Bibliografia:

- M. Rose, *The Simple Book*, Second Edition, Prentice Hall, 1996.
- B. Dias, *Gestão de Redes*, PAPCC, Universidade do Minho, 1996.
- W. Stallings, *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2*, Addison-Wesley, 2000.
- D. Mauro, K. Schmidt, *Essential SNMP*, O'Reilly, 2001.
- Charles P. Pfleeger, Shari Lawrence Pfleeger, Jonathan Margulies, *Security in Computing*, Prentice Hall, 2015.
- Security in Telecommunications and Information Technology, ITU-T, 2015.
- Man Y. Rhee, *Internet Security – Cryptographic Principles, Algorithms and Protocols*, Wiley, 2003.
- William Stallings, *Cryptography and Network Security Principles and Practices*, Prentice Hall, 2015.

Ver outros recursos na área de "Conteúdo" no BB da UC e no material fornecido no início do semestre.

A. Implementação dum proxy SNMPv2cSec

A inclusão de mecanismos de segurança para garantia de privacidade, autenticação e verificação da integridade dos dados foram definidos na segunda versão do INMF. No entanto, a sua implementação era opcional pelo que as duas primeiras versões do modelo, que passaram a ser conhecidas como SNMPv1 e SNMPv2c, não oferecem mecanismos de segurança nativos. Esses mecanismos são apenas garantidos pela terceira versão do INMF, comumente conhecida como SNMPv3, sendo que, na prática, não existe razão para os equipamentos e aplicações implementarem uma eventual versão SNMPv2. Ou implementam uma versão que não suporta mecanismos de segurança (v1 ou v2c) ou implementam uma versão que os suporta (v3).

Quando o suporte nativo para os mecanismos de segurança do SNMPv3 não está disponível, ainda é possível garantir algum nível de segurança mesmo que se utilize o SNMPv1 ou SNMPv2c. Nesse caso, a segurança não está embebida no protocolo normalizado, mas tem de ser implementada indiretamente à custa da semântica de objetos duma MIB especial implementada num agente proxy SNMPv2c. Esta estratégia é usada nalguns ambientes industriais onde a mudança do protocolo para a versão 3 era mais complicada de se conseguir do que a adoção dessa MIB especial que permite isolar os agentes SNMP da rede privada interna que realmente implementam as MIB da monitorização e controlo da produção.

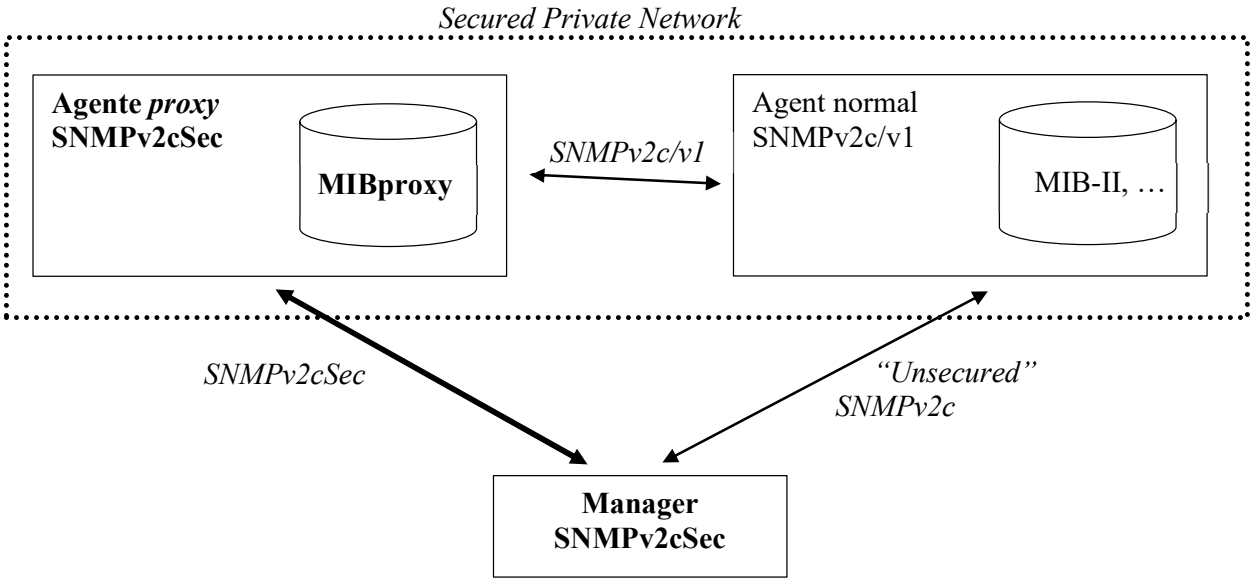

Figura 1: Interligação do agente *proxy* com um agente normal.

Assim, a forma mais simples de implementar esta segurança é acrescentar um agente *proxy* que implementa esta MIB especial de segurança e que serve de interface seguro para um agente SNMPv1 ou SNMPv2c. Passaremos a designar o agente *proxy* como um agente SNMPv2cSec porque para aceder a este agente de fora da rede privada é preciso usar um protocolo SNMPv2c especial que implemente indiretamente mecanismos de segurança (inclusive mecanismos que suportem garantias adicionais às previstas no SNMPv3). Este protocolo especial deve suportar as mesmas primitivas e usar os mesmos PDU do SNMPv2c, mas em que os valores do *VarBindList* são especiais sem o protocolo saber.

O agente proxy SNMPv2cSec fará de *interface* seguro para um agente SNMPv1 ou SNMPv2 normal, não seguro, conforme o esquema da Figura 1. Este agente normal pode ser independente e pode estar implementado e instalado numa máquina da rede interna/privada ou ser implementado na mesma máquina do agente *proxy*. Também pode ser um processo interno numa implementação integrada com o agente *proxy*, sendo esta opção menos modular.

Os objetivos principais deste trabalho é definir a MIBproxy, construir um protótipo e de um agente SNMPv2Sec que implemente essa MIBproxy e construir um protótipo dum gestor SNMPv2cSec que permita aceder a alguns objetos da MIB-II dum agente normal para o qual o agente SNMPv2cSec faça de *proxy*. Não é necessário que o gestor e o agente SNMPv2cSec implementem todas as primitivas da norma SNMPv2c, mas é necessário, pelo menos, o suporte às primitivas `get-request` (ou `get-next-request`) e `set-request` (para o gestor usar) e a primitiva `response` (para o agente *proxy* responder às primitivas do gestor).

O agente normal SNMPv2c a usar no projeto pode ser um qualquer agente SNMPv2c já existente, como por exemplo, a implementação do Net-SNMP ou a implementação incluída nalguns sistemas operativos.

Protocolo SNMPv2cSec

A vantagem deste modelo é que não precisa qualquer alteração ao protocolo de comunicação SNMPv2c (ou SNMPv1). As alterações necessárias não são na sintaxe do PDU nem nas primitivas possíveis, mas sim na forma como os objetos da MIBproxy são definidos e codificados. A sua codificação especial é o que permite implementar as garantias de segurança. No entanto, no contexto deste trabalho não é necessário definir nem implementar os mecanismos necessários ao suporte dessas garantias de segurança.

Assim, o protocolo SNMPv2Sec entre o gestor e o agente *proxy* pode ser feito utilizando as mesmas API do SNMPv2c já disponibilizadas por vários pacotes de software disponíveis publicamente (e que utilizam a codificação normalizada BER) ou pode ser implementado de raiz pelos alunos para os protótipos do agente *proxy* e do gestor como um protocolo aplicacional com uma codificação também a definir no contexto deste trabalho (e que seja até mais simples do que o BER), bastando para isso indicar a definição dos PDU usados (que podem ser uma simplificação dos PDU do SNMPv2c) e o encapsulamento usado (preferencialmente UDP).

MIBproxy

Esta MIBproxy deveria permitir definir mecanismos de segurança indiretos para manipulação segura dos objetos que são implementados nas MIB normais dos agentes normais.

Assim, a MIBproxy deveria conter objetos que suportassem a autenticação dos gestores, a encriptação dos valores dos objetos (e da sua identificação) acedidos indiretamente nas MIB normais e a verificação da integridade dos dados trocados entre os agentes e o gestor. A MIBproxy deveria suportar também um mecanismo equivalente às vistas de controlo de acesso do SNMPv3. Opcionalmente, esta MIB deveria permitir ao agente proxy SNMPv2cSec implementar mecanismos de segurança adicionais, comportamentais ou puramente tecnológicos.

No entanto, no contexto deste trabalho prático, os protótipos devem suportar, através do agente *proxy*, apenas operações normais de leitura/monitorização e de configuração dos valores das instâncias dos objetos das MIB normais implementadas nos agentes normais.

A Figura 2 sugere uma definição num modelo de informação pouco preciso e de alto nível da tabela principal a ser implementada na MIBproxy (é conveniente a inclusão de objetos adicionais nessa MIB que ajudem na manipulação dos valores das instâncias da tabela).

idOper	typeOper	idSource	idDestination	oidArg	valueArg	typeArg	sizeArg	ttl

- `idOper` – identificador da operação recebida pelo agente *proxy* (número aleatório e unívoco, gerado pelo gestor e verificado pelo *proxy*); este objeto serve de chave da tabela
- `typeOper` – tipo de operação SNMP (*get*, *getnext*); a identificação pode ser feita através dum número inteiro/*tag*
- `idSource` – identificador da fonte do pedido (um alias/nome que identifica um gestor)
- `idDestination` – identificador do destino onde a operação será executada (um alias/nome que identifica um gestor SNMPv2c)
- `oidArg` – OID do objeto da MIB que é argumento da operação a ser executada no agente SNMPv2c remoto
- `valueArg` – valor do objeto referido por `oidArg` que é o resultado recebido no agente *proxy* vindo do agente SNMPv2c remoto (deve ser do tipo OPAQUE – conjunto de bytes que devem ser interpretados tendo em conta o tipo definido em `typeArg`)
- `typeArg` – tipo de dados do `valueArg` (INTEGER, STRING, etc.); a identificação pode ser feita através dum número inteiro/*tag*
- `sizeArg` – tamanho em bytes do `valueArg`; enquanto o valor de `valueArg` não é devolvido pelo agente SNMPv2c remoto, `sizeArg` deve ser igual a zero; quando o agente *proxy* recebe a resposta do agente SNMPv2c remoto grava o valor da instância do objeto em `valueArg` e coloca `sizeArg` com o tamanho desse valor (em bytes)
- `ttl` – tempo de vida restante da entrada na tabela; quando este valor atingir zero a entrada é retirada e o gestor não poderá aceder aos seus valores

Figura 2: Colunas da tabela de informação a ser implementada na MIBproxy do agente *proxy*.

Relatório e outras recomendações

O código dos protótipos do agente e do gestor devem utilizar, preferencialmente, apenas funções normalizadas da linguagem C, de preferência da norma 217 ISO (STD 17), e ou funções desenvolvidas pelo grupo de trabalho. Podem usar-se API, funções ou excertos de código de terceiros para implementar aspetos tecnológicos. Em caso de dúvida consulte o docente para verificar a adequação da sua utilização no contexto do trabalho.

Em alternativa pode usar-se qualquer linguagem de mais alto nível como Python ou Java. De qualquer forma, o código deve ser claro e usar convenções internacionais de nomeação de variáveis, tipos, funções e constantes. O código deve ser estruturado duma forma o mais modular possível, sem complexidades desnecessárias, e permitir reutilização sempre que possível. A qualidade e correção do código não se mede pelo seu tamanho! Os alunos devem documentar/explicar o código criado através de comentários nas secções mais relevantes e no relatório. Se possível, os componentes desenvolvidos devem ter um modo de funcionamento de "*debug*" (para além do modo de funcionamento normal) onde devem ser geradas mensagens de atividade durante a execução. Isto ajuda na deteção e correção de erros durante o desenvolvimento e ajuda na sessão de defesa do trabalho se o docente pedir a sua utilização.

Todos os ficheiros do código devem ter um cabeçalho com a informação relevante que identifique os seus autores e explique as principais funções/classes criadas, tipos de dados e variáveis usadas, etc.

O relatório pode ser escrito no formato e no editor que for mais conveniente e deve incluir, no mínimo:

- Uma primeira página com o título do trabalho e a identificação do autor (incluindo fotografia), universidade, curso, unidade curricular e data de entrega;

- Um índice do conteúdo;
- Uma secção com a discussão das estratégias escolhidas, as opções tomadas, os mecanismos e tecnologias adotados, incluindo eventuais otimizações;
- Uma secção com a definição e explicação detalhada da MIBproxy;
- Uma secção com a definição e explicação detalhada da forma de se atingir uma manipulação indireta dos objetos das MIB dos agentes normais;
- Uma secção com a explicação e análise crítica das principais funções/classes implementadas, os seus principais méritos e a suas limitações mais importantes;
- Uma secção de conclusões que inclua uma eventual discussão sobre o que poderia ter sido feito melhor ou poderia ter sido acrescentado;
- Uma lista de eventuais referências bibliográficas, artigos científicos ou recursos informais na web e que tenham sido úteis.

O relatório é para ser avaliado pelo docente por isso não inclua informação genérica e irrelevante que o docente já conheça. Tente ser conciso e claro. Sempre que incluir uma afirmação importante no contexto do relatório e que seja de autoria de terceiros, ou que seja baseada diretamente em afirmações de terceiros ou concluída de informação retirada de recursos alheios, deve referenciar corretamente essas autorias ou proveniências e acrescenta-las na lista das referências.

O relatório pode incluir algumas partes relevantes do código quando estas ajudem à análise e justificação apresentadas. Não inclua código desnecessário no texto do relatório. Sempre que possível, comente antes o próprio código.

No material entregue inclua apenas dois ficheiros: um ficheiro PDF com o relatório e um ficheiro zip com todos os ficheiros do código do projeto dentro duma diretoria com o seguinte nome `GR-22_23-Números_Alunos.zip`, como por exemplo, `GR-22_23-83974-65535.zip`.

Por fim, é recomendável que, durante a defesa do trabalho, tente responder honesta e concisamente apenas às questões colocadas por forma a que as sessões de apresentação não se arrastem muito para além dos 30-40 minutos.