

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



FEUP

Gestão dinâmica dos acessos a uma rede local

Luís Pedro Campelos Moreira

Dissertação realizada no âmbito do Mestrado Integrado em Engenharia Electrotécnica e de Computadores Major de Telecomunicações

Orientador: João Manuel Couto das Neves

Junho de 2009

Resumo

Hoje em dia, quando se aborda o assunto do controlo de acessos a uma rede, destaca-se a sua implementação em redes sem fios, sem se dar a devida relevância a dispositivos que apenas têm uma interface 802.3, ou simplesmente não suportam o protocolo 802.1X. Existem aplicações de Autenticador que controlam eficazmente o acesso em redes sem fios, mas que têm dificuldades em fazê-lo de igual modo em redes por cabo. Por vezes, surgem dificuldades, por parte dos utilizadores, em configurar o software Suplicante, assim como por parte dos técnicos responsáveis, que se deparam com problemas de compatibilidade. Para além disso, é necessário pensar numa solução para dispositivos, que não computadores pessoais, sem suporte 802.1X. Neste documento, são analisados os protocolos associados ao controlo dos acessos a uma rede, assim como os protocolos suportados pelos dispositivos que se pretendem controlar. Através desta análise são apresentadas soluções de controlo de acesso baseadas no HTTPS, no SNMPv3 e no controlo de tráfego. Também é proposto um melhoramento ao software HostAP, por forma a melhorar o controlo de acesso de dispositivos com interface Ethernet e suporte 802.1X. Neste estudo, conclui-se que as ferramentas utilizadas para as soluções são frágeis, em termos de segurança, mas mesmo assim necessárias no controlo de acessos.

Abstract

Nowadays, when discussing the issue of Network Access Control, its implementation is emphasized in wireless networks, without giving much importance to devices that only have a 802.3 interface, or simply do not support the 802.1X protocol. There are Authenticator applications that effectively control the access to wireless networks, but have difficulties when controlling devices on wired networks. Sometimes, difficulties arise from users perspective as they try to configure the Supplicant software, as well as from technicians perspective, who face compatibility problems. Furthermore, it is necessary to think of a solution for devices, that are not PCs, which do not support 802.1X. Protocols associated with the Network Access Control are analyzed in this document, as well as the protocols supported by the controlled devices. From this analysis, solutions for access control are presented, based on HTTPS, SNMPv3 and traffic control. Also, it is proposed an improvement for the HostAP software in order to upgrade the access control of devices with an Ethernet interface and support for 802.1X. In this study, it is concluded that the tools used for the solutions are fragile, in terms of security, but still needed in access control.

Agradecimentos

Gostaria de agradecer, primeiramente, aos meus pais. Sem eles, não só este trabalho, mas toda a educação não seria possível, pois criaram todas as condições necessárias ao meu crescimento, a todos os níveis.

Gostaria também de agradecer ao meu orientador, professor João Manuel Couto das Neves, por me orientar ao longo do meu trabalho, fazendo sugestões e críticas úteis.

Agradeço também ao professor José Manuel Magalhães Cruz por me ter ajudado em várias alturas do meu trabalho, quando algumas dificuldades surgiram.

Por fim, agradeço a toda a gente que me ajudou e apoiou na realização deste estudo.

Luís Pedro

Conteúdo

1	Introdução	1
1.1	Tema	1
1.2	Objectivos	2
1.3	Estrutura da Dissertação	2
2	O Controlo de Acessos a uma Rede Local	5
2.1	Conceitos Base	5
2.1.1	Componentes Principais	6
2.1.2	Comunicações	6
2.2	EAPOL	8
2.2.1	Encapsulamento	8
2.2.2	Estrutura do pacote EAPOL	9
2.2.3	Tipos de pacotes EAPOL	9
2.3	Protocolos de Autenticação, Autorização e Contabilização	11
2.3.1	RADIUS	11
2.3.2	TACACS+	21
2.3.3	Diameter	26
2.4	EAP	36
2.4.1	Estrutura do pacote EAP	36
2.4.2	Géneros de pacotes EAP	37
2.4.3	Tipos de EAP Request/Response	37
2.4.4	Procedimentos	38
2.5	Métodos EAP	38
2.5.1	Outros tipos de métodos EAP	40
3	Análise Crítica e Comparativa	43
3.1	Análise Crítica	43
3.1.1	RADIUS	43
3.1.2	TACACS+	45
3.1.3	Diameter	45
3.1.4	EAP	46
3.2	Análise Comparativa	47
3.2.1	Vantagens do RADIUS	47
3.2.2	Vantagens do TACACS+ em relação ao RADIUS	48
3.2.3	Vantagens do Diameter em relação ao RADIUS	50
3.3	Resumo e Conclusão	51
3.3.1	Avaliação dos protocolos AAA	51
3.3.2	Conclusão	52

4	Exposição de protocolos acessórios	55
4.1	HTTP	55
4.1.1	Funcionamento global	55
4.1.2	Tipos de mensagens	56
4.1.3	Definição dos métodos	58
4.2	SNMP	62
4.2.1	Arquitetura	62
4.2.2	Informação de Gestão	62
4.2.3	Protocolo	63
4.2.4	SNMPv3	65
5	Problema e Solução	67
5.1	Problema	67
5.1.1	O problema do acesso	67
5.1.2	O problema da segurança	69
5.1.3	O problema em estudo	69
5.2	Solução	70
5.2.1	Funcionalidades da rede	70
5.2.2	Acesso de computadores pessoais	71
5.2.3	Acesso de outras máquinas	76
5.2.4	Alternativa	81
6	Conclusão e Trabalho Futuro	87
6.1	Conclusão	87
6.2	Trabalho Futuro	87
	Referências	89

Lista de Figuras

2.1	Esquema ilustrativo dos conceitos base	7
2.2	Empilhamento das camadas de comunicação	8
2.3	Estrutura de um pacote EAPOL	9
2.4	Descritor de chave do pacote EAPOL-Key	10
2.5	Estrutura de um pacote RADIUS	11
2.6	Formato dos atributos RADIUS	14
2.7	Formato dos atributo RADIUS Vendor-Specific	17
2.8	Encadeamento de proxies em RADIUS	21
2.9	Formato do cabeçalho TACACS+	22
2.10	Formato das mensagens Diameter	29
2.11	Formato dos atributos Diameter	31
2.12	Exemplo de troca de mensagens da aplicação Accounting do Diameter	34
2.13	Formato dos pacotes EAP	36
2.14	Formato dos dados EAP	37
4.1	Negociação TLS	61
4.2	Estrutura de informação de gestão (SMI)	63
4.3	Management Information Base (MIB)	64
4.4	Estrutura das mensagens SNMP	65
4.5	Troca de mensagens SNMP	66
5.1	Autenticação 802.1X	71
5.2	Controlo de acesso de PCs	72
5.3	Página de autenticação HTTPS	75
5.4	Controlo de acesso de dispositivos com suporte SNMPv3	77
5.5	Diagrama de rede da solução alternativa	82
5.6	Tentativa de acesso à página <i>www.google.com</i>	84
5.7	Resultado da consulta de <i>www.google.com</i>	85
5.8	Captura wireshark - sem resposta DNS	85
5.9	Acesso da página <i>www.google.com</i> através do endereço IP	86

Lista de Tabelas

2.1	Serviços suportados	16
2.2	Identificadores de aplicação Diameter	27
2.3	Exemplos de comandos Diameter	30
3.1	Tabela de comparação entre protocolos AAA	52

Abreviaturas e Símbolos

AAA	Authentication, Authorization and Accounting
AP	Access Point
ASN	Abstract Syntax Notation
CA	Certification Authority
EAP	Extensible Authentication Protocol
EAPOL	EAP Over LAN
GSM	Global System for Mobile communications
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IESG	Internet Engineering Steering Group
IETF	Internet Engineering Task Force
INESC	Instituto de Engenharia de Sistemas e Computadores
MAC	Media Access Control
MD5	Message-Digest algorithm 5
MIB	Management Information Base
MIME	Multipurpose Internet Mail Extensions
NAC	Network Access Control
NAI	Network Access Identifier
NAS	Network Access Server
NAT	Network Address Translation
NNTP	Network News Transfer Protocol
OID	Object IDentifier
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
RADIUS	Remote Authentication Dial-In User Service
RFC	Request For Comments
SHA	Secure Hash Algorithm
SMI	Structure of Management Information
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SSL	Secure Sockets Layer
TACACS	TAC Access Control System
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locators
URN	Uniform Resource Name
USM	User-based Security Model
VoIP	Voice over Internet Protocol
WAIS	Wide Area Information Servers

Capítulo 1

Introdução

1.1 Tema

Os sistemas de comunicação e informação são cada vez mais importantes, tanto para fins de lazer como para fins profissionais. Com esta evolução, as redes locais tornaram-se essenciais e, como suporte, existem normas estabelecidas pelo Institute of Electrical and Electronics Engineers (IEEE), sendo que as mais conhecidas são o IEEE 802.3 (Ethernet) e o IEEE 802.11 (redes sem fios).

Principalmente no ambiente profissional e comercial, muitas vezes torna-se necessário o controlo da acesso de dispositivos a redes que, pela informação que circula dentro destas, se podem tornar valiosas. Com isto em mente, foram criados protocolos de autenticação, autorização e contabilização (Authentication, Authorization, Accounting). O protocolo AAA mais conhecido e implementado é, sem dúvida, o Remote Authentication Dial In User Service (RADIUS). Para além deste, existem outros, entre os quais o TACACS+ e o Diameter. No entanto, para além de protocolos AAA, também são necessários diferentes métodos de autenticação. Com a necessidade de suportar vários desses métodos criou-se o Extensible Authentication Protocol (EAP), que é adaptado aos protocolos AAA para permitir uma grande flexibilidade na escolha do método de autenticação.

Por uma questão de facilidade de implementação e de centralização da informação, torna-se necessário a existência de um servidor de autenticação, ou um conjunto destes, que serve de controlo para uma rede inteira. Assim, nas fronteiras de uma rede encontra-se um dispositivo autenticador e não o próprio servidor de autenticação. Dadas as circunstâncias, o dispositivo que se pretende ligar à rede necessita de comunicar com o dispositivo autenticador, pelo que criou-se uma norma que permitisse que todos estes dispositivos e protocolos se articulassem e cooperassem, o IEEE 802.1X. Esta norma descreve a forma de interligar o EAP com dispositivo que se pretende ligar à rede, assim como com o protocolo AAA usado.

Porém, o IEEE 802.1X falha num aspecto: trata-se de uma só norma que tenta abranger o controlo de acesso para várias tecnologias e normas de comunicação. O IEEE 802.1X, parece estar bem adaptado a sistemas como redes sem fios (IEEE 802.11) e GSM, apesar de existirem outros

tipos de sistemas em que esta norma pode não funcionar como esperado. Um tipo de interface que geralmente encontra obstáculos com estes mecanismos é o IEEE 802.3, que apresenta complicações a nível de versões de software e compatibilidade entre sistemas. Para além disto, existem dispositivos que não suportam o software de implementação do IEEE 802.1X, não podendo, por isso, ser controlados os seus acessos.

Este estudo apresenta soluções para uma melhor adaptação do software de autenticador às redes locais Ethernet, simplificação do acesso de computadores pessoais para utilizadores menos experientes, assim como o controlo de acessos de máquinas que não suportam o 802.1X.

1.2 Objectivos

O objectivo desta dissertação é identificar e estudar os principais obstáculos no controlo dos acessos a uma rede Ethernet (IEEE 802.3). Um outro objectivo é o de apresentar soluções para os obstáculos encontrados, nomeadamente problemas relativos à arquitectura do software autenticador, à dificuldade de configuração e incompatibilidade do software suplicante, e à falta de controlo de máquinas que não suportam 802.1X.

Quanto à arquitectura do software autenticador, tenta-se perceber como este é implementado para que se possa encontrar uma solução que efectivamente controle os acessos. No que diz respeito à dificuldade de configuração e à incompatibilidade do software suplicante, tenta-se encontrar uma forma de garantir a compatibilidade do software, ao mesmo tempo que se minimiza o esforço de configuração deste. Por fim, estuda-se a forma de solucionar a falta de controlo de máquinas que não suportem o 802.1X.

Como forma de demonstrar algumas destas soluções, faz-se a apresentação de alguns testes práticos, que comprovam que as soluções podem ser postas em prática.

1.3 Estrutura da Dissertação

Esta dissertação encontra-se organizada em seis capítulos. O segundo capítulo faz o enquadramento teórico relativo ao controlo de acessos a uma rede. Primeiramente, é feita uma descrição do processo de autenticação, dando uma noção geral do controlo dos acessos. Em seguida, descrevem-se os vários protocolos que podem estar envolvidos no controlo de acessos.

No terceiro capítulo é feita uma análise de vários protocolos associados ao controlo de acessos. Primeiro, são feitas críticas aos protocolos de autenticação autorização e contabilização e ao EAP. Em segundo lugar, apresentam-se vantagens de uns protocolos em relação a outros. Finalmente, é feita uma comparação final, retirando daí uma conclusão.

O quarto capítulo expõe protocolos que servem de ferramentas para as soluções encontradas. É feita uma descrição do HTTP, do HTTPS, do SNMP e finalmente do SNMPv3.

O quinto capítulo apresenta os problemas existentes no controlo de acessos e propõe soluções. Em primeiro lugar, é feita uma exposição dos problemas existentes, tanto na perspectiva do controlo dos acessos como da segurança. Seguidamente, são apresentadas soluções para os problemas apresentados.

O sexto capítulo é a conclusão do trabalho, dando-se uma visão geral dos resultados obtidos e sugerindo eventuais melhorias.

Capítulo 2

O Controlo de Acessos a uma Rede Local

2.1 Conceitos Base

Um sistema que queira controlar os acessos a uma rede terá de, primeiramente, identificar quem a quer aceder. Apenas com base nessa identificação o sistema poderá permitir o acesso, ou não, dessa entidade (pessoa, dispositivo ou programa) à rede. Ao processo de identificar algo ou alguém dá-se o nome de Autenticação. Apenas o utilizador autenticado e com permissões para aceder a uma rede poderá ter autorização para o fazer de facto.

Num sistema de controlo de acessos a uma rede, o acesso é feito através de uma porta. Para as redes Ethernet, “porta” refere-se à porta num comutador de Ethernet (switch), sendo que a autenticação se completa a nível físico (cabos, placas, etc.) e a nível de ligação (2^a camada da pilha TCP/IP). Nas redes sem fios, esta a porta pode ser vista como uma associação entre o dispositivo cliente e o ponto de acesso (Access Point - AP).

Para se compreender melhor o mecanismo do controlo de acessos, será explicado todo o processo utilizando a analogia de uma festa privada de numerosos convidados. Imagine-se que Pereira, um senhor rico e popular, decide organizar uma festa privada com inúmeros convidados. Como seria de esperar, ele apenas recebe as pessoas dentro de sua própria casa, ou melhor mansão, pelo que contratou seguranças e incumbiu-lhes a tarefa de controlar quem entra. Apesar de haver muitos convidados, Pereira não gostaria de ter um intruso na sua festa. Assim, Pereira colocou Silva à porta de sua casa e Brandão com um documento que tinha a lista de convidados e descrição dos mesmos. Cada vez que alguém se aproximava da porta da casa de Pereira para entrar, Silva impedia a pessoa de continuar e pedia-lhe a sua identificação enquanto estava atento as suas características físicas mais evidentes. Depois de ter a identificação da pessoa na mão, Silva comunicava com Brandão descrevendo a pessoa e a sua identidade. Brandão responderia, dizendo se a pessoa fazia ou não parte da lista de convidados e se a identidade e as características batiam certo. Se a pessoa fizesse parte da lista de convidados e os seus dados batessem certo, estaria autorizada a entrar.

Dada esta analogia, será mais fácil explicar os elementos constituintes de um sistema de controlo de acessos.

2.1.1 Componentes Principais

2.1.1.1 Suplicante

Um Suplicante é um dispositivo cliente que necessita de ser autenticado antes de lhe ser dada permissão para aceder a uma rede. No caso da história da festa privada, cada pessoa que pretende participar na festa é um Suplicante. O Suplicante poderá ser visto como um utilizador desconhecido, ao qual se põem dúvidas quanto à sua identidade até a altura em que produz credenciais válidas ao Servidor de Autenticação.

2.1.1.2 Autenticador

Um Autenticador é um dispositivo de nível 2, como um switch ethernet ou um AP. O autenticador actua como um portal de segurança entre os suplicantes e a rede protegida. O portal permanece fechado até o sistema de autenticação verificar as credenciais do suplicante e declarar que o Suplicante é autorizado a aceder a rede protegida. Assim que o sistema autentica o Suplicante, o Autenticador abre a porta para que o suplicante possa aceder a rede protegida. O Autenticador é o tradutor entre o Suplicante e o Servidor de Autenticação. No exemplo da festa, o Autenticador é o segurança Silva.

2.1.1.3 Servidor de Autenticação

O Servidor de Autenticação é aquele que detém toda a informação acerca dos Suplicantes autorizados a aceder a rede protegida. É ele que determina se o Suplicante é autorizado ou não a aceder a rede protegida. Na festa do senhor Pereira, o Servidor de Autenticação era o segurança Brandão. As especificações do sistema de autenticação não obrigam a utilização particular de um tipo de servidor, mas em praticamente todas as implementações é utilizado um servidor RADIUS (Remote Authentication Dial In User Service).

2.1.2 Comunicações

O protocolo 802.1X foi pensado para englobar o protocolo EAP (Extensible Authentication Protocol), principalmente. Mas, EAP foi desenhado para ligações ponto-a-ponto, pelo que precisou de ser adaptado o EAPOL (EAP Over LANs), que encapsula EAP como dados. Este tipo de comunicação é feita entre o Suplicante e o Autenticador. Do lado protegido da rede, o Autenticador troca pacotes EAP com o Servidor de Autenticação sobre um outro protocolo. Se se seguir a maior parte das implementações, este protocolo será o RADIUS. Os protocolos envolvidos no sistema de autenticação são representados na figura 2.1.

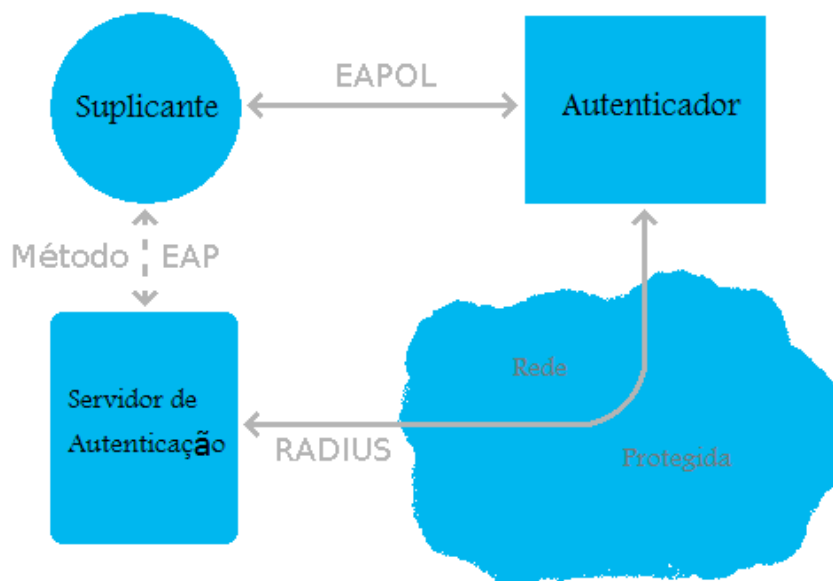


Figura 2.1: Esquema ilustrativo dos conceitos base

2.1.2.1 Suplicante para Servidor de Autenticação: Métodos EAP

A verdadeira troca de informação com o objectivo de autenticação dá-se entre o Suplicante e o Servidor de Autenticação. A conversa entre eles inclui dados do Método EAP, o que representa vários elementos, tal como as credenciais do Suplicante. Esta conversa também inclui várias trocas de informação EAP, dependendo do tipo do Método EAP.

Nos sistemas de autenticação 802.1X, os Métodos EAP fazem uso de diferentes tipos de credenciais, como “nome de utilizador/palavra-passe”, chaves de encriptação e certificados digitais. Os standards requerem a implementação dos seguintes Métodos EAP:

- Desafio MD5;
- One-Time Password (OTP);
- Generic token card.

Para além disso, há vários Métodos EAP proprietários e baseados em RFC, tais como EAP-TLS, EAP-TTLS, EAP FAST e EAP-LEAP.

2.1.2.2 Suplicante para Autenticador: 802.1X / EAPOL

802.1X apenas se aplica entre o Suplicante e o Autenticador. Um sistema de autenticação 802.1X completo faz uso de outros protocolos, como RADIUS. 802.1X é apenas parte do sistema.

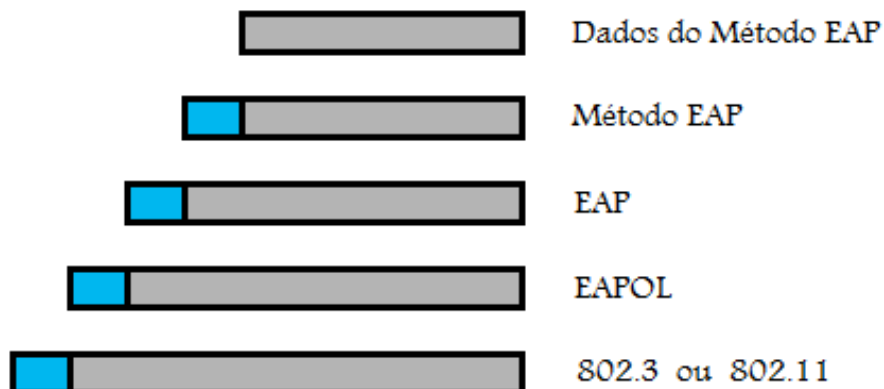


Figura 2.2: Empilhamento das camadas de comunicação

EAP foi desenhado como um protocolo Ponto-a-Ponto para comunicações sobre uma ligação série. EAPOL é definido no 802.1X para adaptar EAP para operar sobre LANs.

2.1.2.3 Autenticador para Servidor de Autenticação

O Autenticador envia dados dos Métodos EAP para o Servidor de Autenticação via RADIUS.

2.2 EAPOL

EAPOL é definido no 802.1X para adaptar comunicações EAP às LANs. Para o fazer, EAPOL acrescenta o seu cabeçalho aos pacotes EAP, para além de transportar os pacotes EAP como dados. EAPOL opera na segunda camada para prevenir que o suplicante se ligue à rede antes de estar autenticado.

2.2.1 Encapsulamento

O encapsulamento do EAPOL é feito de forma a suportar as comunicações entre o Suplicante e o Autenticador. A disposição dos protocolos nas camadas da pilha de comunicações é idêntico, em termos lógicos, à pilha TCP/IP ou OSI, tal como se pode vêr na Figura 2.2. A interligação destes protocolos completa um sistema de autenticação 802.1X.

O principal objectivo das comunicações da autenticação é transportar dados dos Métodos EAP, os quais implementam o verdadeiro processo de autenticação. Os pacotes EAP contêm os Métodos EAP e os seus dados. EAPOL transporta EAP e 802.3, ou 802.11, transporta EAPOL. Cada protocolo da pilha 802.1X tem a sua função e define pacotes que podem ou não transportar pacotes de camadas superiores.

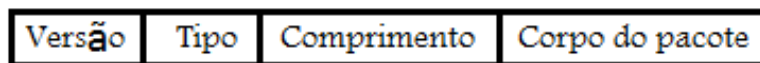


Figura 2.3: Estrutura de um pacote EAPOL

2.2.2 Estrutura do pacote EAPOL

A estrutura de um pacote EAPOL é apresentada na figura [2.3](#)

2.2.2.1 Versão

O campo “versão”, com o comprimento de um octeto, identifica a versão do protocolo EAPOL que é suportada por quem envia o pacote. Tal como noutros protocolos de comunicação, a implementação do campo versão permite o desenvolvimento de componentes e sistemas compatíveis com outros mais antigos.

2.2.2.2 Tipo

O campo “tipo”, de comprimento um octeto, identifica o tipo de pacote EAPOL que está a ser enviado. Os vários tipos de pacotes serão apresentados mais à frente.

2.2.2.3 Comprimento

O campo “comprimento” tem o comprimento de 2 octetos e especifica o número de octetos dos dados do pacote EAPOL. O valor nulo significa que o pacote não tem dados.

2.2.2.4 Corpo do pacote

O campo “corpo do pacote”, de comprimento variável, contém os dados enviados no pacote EAPOL.

2.2.3 Tipos de pacotes EAPOL

2.2.3.1 EAP-Packet (Pacote EAP)

Também conhecido por pacote EAPOL do tipo “0” (zero), ou pacote EAPOL de dados, é o pacote que transporta pacotes EAP. Assim sendo, quem quer que o receba, seja o Suplicante

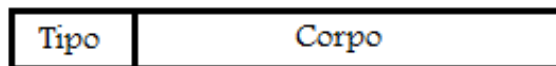


Figura 2.4: Descritor de chave do pacote EAPOL-Key

ou o Autenticador, simplesmente retira o cabeçalho do pacote EAPOL e trata o pacote EAP daí resultante. Os dados deste pacote são o pacote EAP.

2.2.3.2 EAPOL-Start

Este pacote, tipo “1”, é enviado pelo Suplicante ao Autenticador quando este se pretende autenticar, para mais tarde poder obter permissão para aceder à rede protegida. Este pacote não transporta dados.

2.2.3.3 EAPOL-Logoff

Este é o pacote, tipo “2”, enviado pelo Suplicante quando este pretende que a porta que lhe dá acesso à rede passe do estado “autorizado” para o estado “não autorizado”, ou seja o Suplicante termina a ligação com a rede a que antes estava ligado. Este pacote não transporta dados.

2.2.3.4 EAPOL-Key

EAPOL-Key é um pacote, tipo “3”, opcional em todo o processo de autenticação, que tanto pode ser enviado pelo Suplicante como pelo Autenticador. Se a implementação 802.1X necessitar de transportar palavras-chave entre o Suplicante e o Autenticador, o pacote EAPOL-Key contém o descritor da chave, como ilustrado na figura 2.4.

O campo “tipo”, com um comprimento de um octeto, representa o tipo de descritor de chave a ser transportado pelo pacote EAPOL-Key. Existem dois tipos de descritores de chave: RC4, e IEEE 802.11.

2.2.3.5 EAPOL-Encapsulated-ASF-Alert

Este pacote, tipo “4”, é útil quando o Suplicante necessita enviar informação para a rede protegida antes da autenticação estar completa. Por exemplo, o Suplicante pode precisar de enviar uma mensagem sobre o seu estado ao servidor. O conteúdo deste pacote é geralmente proprietário.

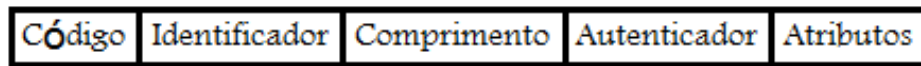


Figura 2.5: Estrutura de um pacote RADIUS

Esta subsecção do EAPOL foi baseada em [1].

2.3 Protocolos de Autenticação, Autorização e Contabilização

2.3.1 RADIUS

RADIUS é o mecanismo de comunicação primário entre o Autenticador e o Servidor de Autenticação num sistema de autenticação. O protocolo RADIUS entre o Autenticador e o Servidor de Autenticação transporta os dados dos Métodos EAP de forma encriptada. O standard 802.1X as especificações de EAP não impõem a implementação do RADIUS, apesar deste ser usado muitas das vezes.

O Servidor de Autenticação estabelece duas conversações durante o processo de autenticação, uma em RADIUS com o Autenticador e outra em Métodos EAP com o Suplicante. Muitas das comunicações entre Servidor de Autenticação e o Autenticador consistem em pacotes Access-Request e Access Challenge. O Autenticador envia dados dos Métodos EAP para o Servidor de Autenticação via Access-Request do RADIUS, e o Servidor de Autenticação dados dos Métodos EAP para o Autenticador via Access-Challenge do RADIUS.

2.3.1.1 Estrutura do pacote RADIUS

A estrutura dos pacotes RADIUS é apresentada pela figura 2.5.

Código

O campo “código”, de comprimento um octeto, identifica o tipo de pacote RADIUS. Mais à frente serão apresentados os vários pacotes RADIUS.

Identificador

Com o comprimento de um octeto, o campo “identificador” é responsável por fazer corresponder os pacotes enviados pelo Autenticador com os enviados pelo Servidor de Autenticação, mais precisamente os pacotes Access-Request e Access-Challenge. Este campo permite que cada passo da autenticação seja controlado, não permitindo que nenhum dos intervenientes se adiante e fazendo, também, com que pacotes duplicados sejam rejeitados.

Comprimento

O campo “comprimento”, com dois octetos de comprimento, identifica o número de octetos do pacote inteiro de RADIUS.

Autenticador

O campo “autenticador” tem o comprimento de 16 octetos e contém o valor correspondente ao pacote RADIUS a ser enviado.

Request Authenticator

Nos pacotes RADIUS Access-Request, o conteúdo do campo autenticador é o Request Authenticator, que não é mais do que um número criado aleatoriamente.

Response Authenticator

Response Authenticator reside no campo autenticador dos pacotes RADIUS Access-Accept, Access-Reject e Access-Challenge. O valor de Response Authenticator é o resultado de uma encriptação MD5 do pacote RADIUS Access-Request inteiro adicionado da chave secreta da rede.

Atributos

Este campo é de comprimento variável e contém elementos de dados específicos transmitidos entre o Autenticador e o Servidor de Autenticação. Estes atributos serão analisados em maior detalhe mais à frente.

2.3.1.2 Tipos de pacotes RADIUS**Access-Request**

É um pacote, de código “1”, enviado apenas pelo Autenticador ao Servidor de Autenticação, para transportar dados do Método EAP a ser usado. Access-Request poderia conter, por exemplo,

as credenciais do Suplicante.

Access-Challenge

De código “11”, Access-Challenge é um pacote que é enviado do Servidor de Autenticação para o Autenticador em resposta ao Access-Request. É o pacote responsável pela troca de informação, principalmente atributos RADIUS, essenciais ao processo de autenticação.

Access-Accept

De código “2”, Access-Accept é um pacote que é enviado do Servidor de Autenticação para o Autenticador em resposta ao Access-Request. O pacote significa a aceitação ou informação de configuração para um determinado Suplicante.

Access-Reject

De código “2”, Access-Accept é um pacote que é enviado do Servidor de Autenticação para o Autenticador em resposta ao Access-Request. O pacote é enviado quando algum dos atributos do Access-Request não podem ser aceites.

Accounting-Request

Este pacote, de código “4”, é enviado pelo Autenticador ao Servidor de Autenticação para pedir informação sobre a conta do Suplicante.

Accounting-Response

Accounting-Response, de código “5”, é enviado pelo Servidor de Autenticação ao Autenticador para fornecer informação sobre a conta do Suplicante.

2.3.1.3 Formato dos Atributos

Cada um dos atributos contidos no campo “atributos” do pacote RADIUS tem o formato apresentado na figura 2.6. Os pacotes RADIUS podem ter vários atributos, nesse caso a estrutura apresentada na 2.6 repete-se para cada um dos atributos.

Tipo

O campo “tipo”, de comprimento um octeto, identifica o atributo. Valores dos tipos 192-223 são para uso experimental, valores 224-240 são para uso de uma implementação específica, e valores 241-255 estão reservados e não devem ser usados. Os servidores RADIUS são programados



Figura 2.6: Formato dos atributos RADIUS

para ignorar atributos não reconhecidos.

Comprimento

Este campo tem um octeto de comprimento e indica o comprimento total do atributo definido. Se o Servidor de Autenticação receber um pacote com um comprimento inválido este responderá com um Access-Reject. Se o Autenticador receber um atributo dum pacote Access-Accept, Access-Challenge ou Access-Reject com um comprimento inválido, ele agirá como se tratasse de um Access-Reject, independentemente do tipo de pacote que recebeu.

Valor

Este campo é comprimento variável e contém os dados do atributo propriamente dito.

2.3.1.4 Atributos

Atributo EAP-Message

Este atributo, do tipo “79”, é o atributo primário para o 802.1X pois ele encapsula dados do Método EAP entre o Autenticador e o Servidor de Autenticação. Portanto, EAP-Message contém o Método EAP, sendo que estes constituem a conversa entre o Suplicante e o Servidor de Autenticação. Os EAP-Message que viajam do Autenticador para o Servidor de Autenticação são transportados por pacotes RADIUS Access-Request, enquanto que os que viajam no sentido oposto são transportados por pacotes RADIUS Access-Challenge. Em ambas as situações é permitido a troca de vários pacotes EAP, desde que estes estejam ordenados e que agrupados. Nos pacotes Access-Accept e Access-Reject normalmente apenas existe um pacote EAP, EAP-Success e EAP-Failure, respectivamente. Se o Servidor RADIUS não entender algum pacote EAP, responderá com um Access-Reject. Ao implementar o atributo EAP-Message, o atributo

Message-Authenticator protege todos os pacotes que contenham dados do Método EAP. Aliás, se o Servidor RADIUS receber um pacote RADIUS Access-Request com o atributo EAP-Message, mas com um Message-Autheticator inválido, este descartará o pacote sem que avise o Authenticador de que o fez.

Atributo Message-Authenticator

O atributo Message-Authenticator, do tipo “80”, é usado pelo Servidor RADIUS para autenticar os pacotes RADIUS Access-Request que lhe chegam. Para além disso, Message-Authenticator poderá verificar a integridade do pacote Access-Request para prevenir spoofing. Ao suportar o atributo EAP-Message, todos os pacotes RADIUS Access-Request, Access-Accept, Access-Reject e Access-Challenge deverão incluir o atributo Message-Autheticator. Este atributo tem um tamanho fixo de 18 octetos, em que o seu valor é o resultado de uma encriptação HMAC-MD5 do Access-Request, usando o “shared secret” como chave.

Atributo Password-Retry

Este atributo, do tipo “75”, tem um tamanho fixo de 6 octetos e pode ser incluído num pacote RADIUS para indicar o número de tentativas de autenticação que um utilizador pode realizar até lhe ser impedido o acesso.

Atributo User-Name

O atributo User-Name, do tipo “1”, identifica o nome do utilizador a ser autenticado e é enviado do Autenticador para o Servidor RADIUS num Access-Request.

Atributo User-Password

Este atributo, do tipo “2”, fornece a palavra-passe do utilizador a ser autenticado e é enviado num Access-Request do Autenticador para o Servidor RADIUS. A palavra-passe não está desprotegida. Em vez disso, é feita uma encriptação MD5 através do “shared secret” e do Request Authenticator. Ao resultado deste cálculo é ainda feito um outro cálculo lógico do tipo XOR com os 16 primeiros octetos da password. Este resultado é posto no valor do atributo, juntamente com o resultado do anterior MD5, fazendo-se um “padding” com “0” entre eles, para que o valor tenha um comprimento múltiplo de 16 octetos.

Atributo NAS-IP-Address

O atributo NAS-IP-Address, do tipo “4”, indica o endereço IP do Autenticador que está a requerer autenticação do Suplicante. Este atributo só é válido em pacotes Access-Request e tem um

Valor	Serviço
1	Login
2	Framed
3	Callback Login
4	Callback Framed
5	Outbound
6	Administrative
7	NAS Prompt
8	Authenticate Only
9	Callback NAS Prompt
10	Call Check
11	Callback Administrative

Tabela 2.1: Serviços suportados

comprimento fixo de 6 octetos.

Atributo NAS-Port

Este atributo, do tipo “5”, indica a porta física (não de TCP ou UDP) do Autenticador que está a pedir autenticação. Este atributo é apenas usado em pacotes Access-Request.

Atributo Service-Type

Do tipo “6”, Service-Type é o atributo que identifica o tipo de serviço que o Suplicante está a pedir ou que o Servidor RADIUS fornece. Este atributo só é usado nos pacotes Access-Request e Access-Accept. O atributo tem um comprimento fixo de 6 octetos e o Autenticador trata qualquer tipo de serviço não suportado como um Access-Reject. Na tabela 2.1 são apresentados os tipos de serviços possíveis.

Atributo Vendor-Specific

O atributo Vendor-Specific, do tipo “26”, permite a implementação de atributos proprietários que não são adequados à generalidade das utilizações. A inclusão de atributos adicionais não interfere com o funcionamento normal do protocolo RADIUS. Se o Servidor RADIUS recebe um pacote RADIUS contendo um atributo Vendor-Specific que ele não entende, descarta-se do atributo e provavelmente envia uma mensagem ao Autenticador. A figura 2.7 mostra o formato do atributo:

Vendor-ID

O campo Vendor-ID, de comprimento quatro octetos, tem o seu octeto mais significativo nulo,

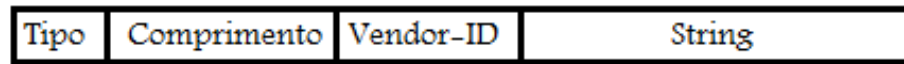


Figura 2.7: Formato dos atributo RADIUS Vendor-Specific

tudo a “zero”, e os seus outros três octetos com o valor de SMI Network Management Private Enterprise Code.

String

Este campo é de tamanho variável e contém os dados do atributo.

Atributo Session-Timeout

Este atributo, do tipo “27”, estabelece o número máximo de segundos que o Suplicante terá serviço. Session-Timeout é apenas enviado do Servidor RADIUS para o Autenticador num pacote Access-Accept ou Access-Challenge.

Atributo Idle-Timeout

O atributo Idle-Timeout, do tipo “28”, estabelece o tempo máximo, sem segundos, que o Suplicante pode estar num estado “idle” até ser desligado da rede protegida. Tal como o atributo Session-Timeout, este atributo só é enviado do Servidor RADIUS para o Autenticador em pacotes Access-Accept ou Access-Challenge.

Atributo Termination-Action

Este atributo, do tipo “29”, indica qual a ação que o Autenticador deve tomar quando o serviço prestado ao Suplicante é completado. Termination-Action é apenas usado em pacotes Access-Accept.

Esta subsecção sobre RADIUS, até a este ponto foi baseada em [2].

2.3.1.5 Segurança em RADIUS

Analisando a segurança de RADIUS constata-se que esta é rudimentar. Duas funções de segurança são implementadas, uma delas é a ocultação de atributos, a outra é a autenticação de certas mensagens. Ambas as funções assentam numa encriptação MD5 com uma chave que é o segredo partilhado pelo cliente RADIUS (Autenticador) e o servidor RADIUS, sendo que este tipo de protecção é considerado bastante fraca e primitiva.

Protecção da Integridade das Mensagens RADIUS

Todas as mensagens RADIUS contêm um campo autenticador. Para os Access Request este campo dá-se o nome de Request Authenticator, enquanto que para Access Challenge, Accept e Reject dá-se o nome de Response Authenticator, mas a principal diferença reside no nível de segurança que cada um tem. Request Authenticator trata-se apenas de um número de 16 octetos gerado aleatoriamente. Como não existe qualquer forma de encriptação nenhuma garantia de integridade é adicionada. Request Authenticator tem a função de oferecer à mensagem RADIUS um carácter único em termos globais e temporais. O carácter único em termos globais é necessário para que o Autenticador possa comunicar com vários servidores distintos, e em termos temporais para evitar ataques que repitam uma troca de pacotes feita no passado. O campo autenticador não só não oferece qualquer protecção de integridade, mas também garantir o carácter único global e temporal é algo que estava para além das possibilidades de um Autenticador (NAS) primitivo e com um processador obsoleto. Tal carácter único é impossível de conseguir com este método. Se o Identificador do NAS (NAI) e o seu endereço IP fossem incluídos na geração do campo autenticador, pelo menos o carácter único global poder-se-ia atingir até um certo nível. A implementação original de RADIUS justificava esta fraqueza oferecendo duas funções de segurança adicionais. Primeiro, os servidores RADIUS verificam o endereço IP de origem da mensagem recebida, pensando que verificando isso a mensagem vem do NAS esperado. Para além disso, como a palavra-passe do utilizador é protegida pelo segredo partilhado pelo NAS e o servidor RADIUS, ter-se-ia um mecanismo adicional de segurança. Tal forma de pensar não é aceitável hoje em dia, visto que pacotes IP podem facilmente ser falsificados e ferramentas para descobrir palavras-passe são facilmente acessíveis. Mais tarde, a comunidade RADIUS começou a pensar num atributo especializado, chamado “Message Authenticator”, para oferecer protecção de integridade às mensagens Access Request. Ao proteger o Access Request com o Message Authenticator, o cliente (NAS) calcula o MD5 sobre toda a mensagem, usando o segredo que partilha com o servidor RADIUS e coloca o resultado no atributo Message Authenticator. Ao receber a mensagem, o servidor RADIUS calcula de igual forma o campo do autenticador de resposta e, se não houver uma correspondência entre o que recebeu e o que calculou, descarta a mensagem. Apesar do atributo Message Authenticator

estar disponível, muitas vezes não é utilizado. O RFC 2869 não obriga o uso deste atributo nas mensagens Access Request que transportam a palavra-passe do utilizador/dispositivo e confia no mecanismo de ocultação de palavra-passe usando o segredo partilhado. É de referir que, nas mensagens RADIUS enviadas pelo Autenticador, o campo autenticador nem sempre é praticamente inútil, servindo como exemplo as mensagens que tratam de contas de utilizadores. Nestes casos, no Accounting Request, o campo autenticador é calculado através duma encriptação MD5 sobre vários campos da mensagem RADIUS. Em contraste com o Access Request, as mensagens enviadas pelo servidor RADIUS para o cliente (Access Challenge, Accept e Reject) têm uma melhor protecção de integridade. O valor do campo autenticador é o resultado de uma encriptação MD5 de praticamente toda a mensagem RADIUS com o segredo partilhado.

Ocultação de atributo

Muitas das funcionalidades oferecidas por RADIUS são conseguidas através dos seus atributos. Exigências de segurança e privacidade requerem que pelo menos alguns destes atributos tenham formas de evitar a exposição da informação enquanto as mensagens são trocadas. No protocolo RADIUS, isto é conseguido através de um processo chamado ocultação de atributos (attribute hiding). A ocultação de atributo em User Password pode ser classificado como ocultação de palavra-passe e é realizado da seguinte maneira: Se a palavra-passe do utilizador tem um comprimento de menos de 16 octetos, o NAS gera um número aleatório para o Request Authenticator e concatena o segredo partilhado com o servidor RADIUS com este; De seguida, calcula o resultado MD5 desta concatenação e realiza uma operação XOR com a palavra-passe do utilizador; O resultado é colocado no atributo de User Password para ser transportado numa mensagem RADIUS; Se, por outro lado, a palavra-passe tiver um comprimento superior a 16 octetos, esta é cortada em partes de 16 octetos e o processo repete-se para cada uma dessas partes. Apesar deste processo parecer complexo, esta ocultação de palavra-passe tem várias falhas. Em primeiro lugar, o carácter único global e temporal do resultado desta ocultação depende do Request Authenticator que, por si, já tem vários problemas. Dado que os NAS muitas das vezes não têm grande poder de processamento, o Request Authenticator pode não ser suficientemente aleatório, isto é, o campo autenticador pode-se repetir de forma mais breve do que seria suposto. Em segundo lugar, a ocultação de atributo baseia-se num segredo partilhado que é demasiadamente estático. Um ataque conhecido contra este método de ocultação de palavra-passe é quando alguém submete um pedido de autenticação com uma palavra-passe conhecida e guarda os resultados da ocultação. Sabendo a palavra-passe e os resultados da ocultação, o atacante pode calcular o resultado da encriptação MD5 da concatenação do segredo com o campo autenticador. Quando o Request Authenticator se repete frequentemente, o atacante começa a poder adivinhar qual será a palavra-passe dos outros utilizadores. Por isto, as especificações do RADIUS recomendam que o segredo seja, pelo menos, tão longo quanto as palavras-passe que tenta proteger. Especificações mais recentes do RADIUS oferecem a possibilidade de se criarem túneis da segunda camada, utilizado um outro número aleatório, o “sal” (salt), que dá um carácter mais único à mensagem. Porém, este número aleatório

não ajuda muito na segurança visto que é enviado em claro nas mensagens e, portanto, qualquer um o pode “ver”.

Esta parte sobre a segurança em RADIUS é baseada em [3].

2.3.1.6 RADIUS sobre IPsec

Como forma de superar as vulnerabilidades de segurança do uso do segredo partilhado, a comunidade RADIUS começou a recomendar o uso do IPsec em algumas especificações, para as quais a segurança era bastante importante. Uma dessas especificações é o RFC 3579 que define o suporte de RADIUS para EAP. EAP oferece uma estrutura de gestão de chaves que permite ao servidor assistir o utilizador/dispositivo e o NAS a estabelecer um canal de comunicação seguro antes de começar a troca de tráfego. Para oferecer suporte para autenticação e encriptação, é recomendado o uso de IPsec como forma de proteger mensagens RADIUS. Tudo isto é baseado em [4]

2.3.1.7 Contabilização em RADIUS

O documento base do RADIUS não contém qualquer especificação para suporte de contas de utilizadores. Em vez disso, a contabilização em RADIUS é definida num RFC separado (RFC 2866). Pelo facto do RADIUS ser transportado sobre UDP, não é garantida robustez na transferência de dados, pois existe a possibilidade de se perderem pacotes com informação sobre as contas dos Suplicantes. O RFC de contabilização em RADIUS resolveu a robustez de forma fácil, recomendando simplesmente que o cliente RADIUS continue a enviar Accounting Requests até que lhe chegue uma confirmação. Em termos de segurança, a parte da contabilização de RADIUS é um pouco melhor que as especificações base. Em vez de ser apenas a resposta do servidor que é autenticada, como nas especificações base, ambos Accounting Request e Accounting Response são autenticados através do campo autenticador que é resultado de uma encriptação MD5 de vários valores, incluindo o segredo partilhado. Isto pode ser confirmado em [5], [6] e [7].

2.3.1.8 Suporte de RADIUS para Roaming e Mobilidade

No que diz respeito à mobilidade, a especificação RADIUS fornecida pelo IETF não oferece qualquer suporte para funcionalidades Mobile IP. Muito do suporte à mobilidade do utilizador está mais relacionado com aplicações roaming e é baseado no trabalho feito pelo grupo de trabalho “Roaming Operations”, do IETF. Uma das especificações mais relevantes é o RFC 2607, o qual define o encadeamento de proxies. Um encadeamento de proxies pode ser definido como os procedimentos necessários para encaminhar pacotes RADIUS entre o NAS e o servidor RADIUS através de uma série de proxies quando um utilizador se desloca para um domínio que não o seu.

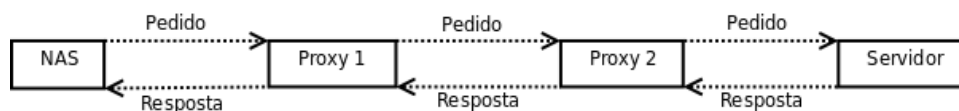


Figura 2.8: Encadeamento de proxies em RADIUS

Operação da cadeia de proxies

A figura 2.8 mostra o conceito do encadeamento de proxies. O pedido original pelo NAS ao Proxy1. Proxy1 examina o pedido e encaminha o pedido para o proxy seguinte, neste caso Proxy2. Proxy2 reencaminha o pedido para o servidor. Ambos Proxy1 e Proxy2 poderão achar necessário modificar os atributos do pacote ao aplicar algumas políticas de segurança. É possível que um dos proxies rejeite o pedido e envie uma mensagem de rejeição ao proxy anterior ou ao próprio NAS. O proxy pode até decidir fazer um desafio ao anterior transmissor, como forma de autenticação e segurança. Os proxies são autorizados a esconder alguma informação do pacote, se necessário.

Esta parte acerca da mobilidade IP é baseada em [8].

2.3.2 TACACS+

O protocolo TACACS + é a mais recente geração de TACACS. TACACS é um simples protocolo controlo de acesso baseado UDP, originalmente desenvolvido pela BBN para a MILNET. A Cisco tem melhorado TACACS várias vezes, e implementação da Cisco, baseada no original TACACS, está previsto como XTACACS.

TACACS + melhora os TACACS e XTACACS, separando as funções de Autenticação, Autorização e Contabilidade, e encriptando todo o tráfego entre o NAS e o daemon. Permite a arbitrariedade de comprimento e de troca de conteúdo de autenticação, o que possibilita o uso de qualquer mecanismo de autenticação.

A separação de autenticação, autorização e contabilidade é uma componente fundamental do projecto de TACACS+. Um importante benefício para separar autenticação da autorização é que a autorização pode ser um processo dinâmico.

TACACS + utiliza o TCP para ser transportado. O servidor deve ouvir na porta 49, que é a porta designada para o protocolo TACACS. Esta porta está reservada no Assigned Numbers RFC para UDP e TCP.

2.3.2.1 Cabeçalho do pacote TACACS+

Todos os pacotes TACACS + começam sempre com um cabeçalho de 12 octetos. O cabeçalho é sempre enviado em claro e descreve o resto do pacote. A figura 2.9 apresenta a estrutura do

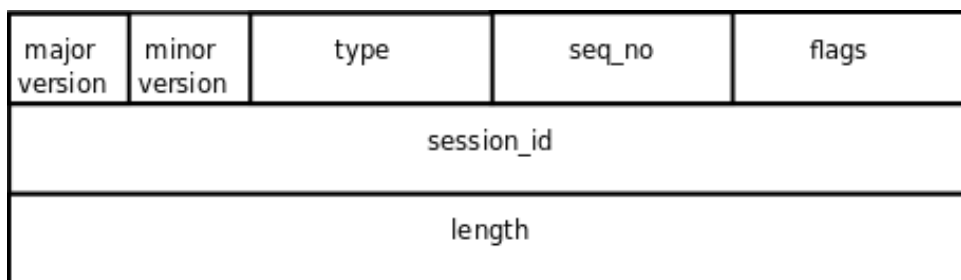


Figura 2.9: Formato do cabeçalho TACACS+

cabeçalho.

major_version

Este é o maior número de versão.

TAC_PLUS_MAJOR_VER: = 0xC

minor_version

O menor número de versão. Este destina-se a permitir revisões do protocolo, mantendo a compatibilidade com versões anteriores.

A menor versão 1 está actualmente definida para alguns comandos.

Quando um servidor recebe um pacote com um `minor_version` que não suporta, ele deve retornar um erro com o estado `minor_version` conjunto para o valor suportado mais próximo.

tipo

Este é o tipo do pacote. Valores válidos são:

TAC_PLUS_AUTHEN: = 0x01 (Autenticação)

TAC_PLUS_AUTHOR: = 0x02 (Autorização)

TAC_PLUS_ACCT: = 0x03 (Contabilidade)

seq_no

Este é o número sequencial do pacote actual para a sessão actual. O primeiro pacote TACACS+ numa sessão deve ter a sequência número 1 e cada um dos pacotes irá incrementar a sequência em uma unidade. Assim, os clientes só enviam pacotes contendo números de sequência ímpares, e os daemons só enviam pacotes contendo números de sequência pares.

O número sequencial nunca deve voltar a 1, ou seja, se o número de ordem 255 é alcançado, a sessão deve ser encerrada e ser reiniciada com número de sequência 1.

flags

Este campo contém várias flags.

O bit de flag “não encriptado” diz se o corpo do pacote TACACS+ está a ser encriptado.

TAC_PLUS_UNENCRYPTED_FLAG: = 0x01

Se esta flag estiver activa, o pacote não está encriptado. Se esta estiver inactiva, o pacote é encriptado. Pacotes não encriptados têm o propósito de realizar testes, e não são recomendados para a utilização normal.

A flag de ligação única:

TAC_PLUS_SINGLE_CONNECT_FLAG: = 0x04

Se um NAS define esta flag, isso indica que ele suporta multiplexagem de sessões durante uma única conexão TCP.

Se o servidor activa a flag na primeira resposta ao primeiro pacote de um NAS, isso indica a sua vontade de suportar ligação única sobre a conexão actual. O daemon pode activar essa flag, mesmo que o NAS não o tenha feito, mas o NAS não tem qualquer obrigação de continuar a fazê-lo.

session_id

O session_id é o identificador da sessão TACACS+. O valor deste campo deve ser aleatoriamente escolhido. Este campo não muda enquanto a sessão durar.

comprimento

Este campo indica o comprimento total do corpo do pacote, não incluindo o cabeçalho. Os pacotes são nunca são preenchidos para além deste comprimento.

2.3.2.2 Corpo do pacote TACACS+

O tipo de corpo do pacote TACACS+ é definido no cabeçalho. Em qualquer tipo de pacote são aplicadas as seguintes regras:

- Todo o corpo é protegido pelo mecanismo encriptação indicado no cabeçalho.

- Qualquer campo de dados de comprimento variável que não forem usadas devem ter um valor de comprimento igual a zero.
- Campos de comprimento fixo que não sejam usados devem ter valor zero.
- Todos os campos de dados e mensagens não podem terminar num valor nulo.
- Todos os valores de comprimento são têm sinal e estão na ordem de bytes da rede.
- Não deve haver padding em qualquer um dos campos ou no final de um pacote.

Encriptação do corpo do pacote

O corpo dos pacotes TACACS + pode ser encriptado, e apenas um mecanismo de encriptação deve ser utilizado dentro de uma única sessão.

Quando o mecanismo de encriptação se baseia numa chave secreta, esta refere-se a um segredo partilhado que é conhecido, tanto pelo cliente como pelo daemon. Existe a possibilidade de se usar apenas uma chave partilhada, ou usar uma chave para cada cliente e servidor. Por questões de segurança, a última opção deve estar disponível, mas a decisão sobre se a utilização de chaves distintas é adequada depende de onde se vai aplicar.

Após a desencriptação de um pacote, os comprimentos dos valores dos componente no pacote devem ser somados e comparado o resultado com o comprimento indicado no cabeçalho. Qualquer pacote que falhe nesta verificação deve ser descartado e sinalizado um erro. Normalmente, tais falhas podem ser esperadas quando não há correspondência entre as chaves do NAS e do servidor.

Se um erro tem de ser declarado, mas o tipo de pacote recebido não pode ser determinado, pode ser retornado um pacote com um cabeçalho idêntico com o número de sequência incrementado em uma unidade, e o comprimento definido a zero, para indicar um erro.

2.3.2.3 Autenticação

A autenticação TACACS + define três tipos de pacotes: START, CONTINUE e REPLY. START e CONTINUE são sempre enviados pelo cliente e REPLY é sempre enviado pelo daemon.

A autenticação começa com o cliente a enviar uma mensagem START para o daemon. A mensagem START descreve o tipo de autenticação a ser realizada, e pode conter o nome de utilizador e alguns dados de autenticação. O pacote START só é enviado na primeira mensagem numa sessão de autenticação, ou imediatamente após um pacote de reinício de sessão. O número de sequência de um pacote START é sempre igual a 1.

Em resposta a um START, o servidor envia um REPLY. A mensagem REPLY indica se a autenticação foi concluída, ou se que deve continuar. Se a resposta indica que a autenticação deve continuar, então também indica que novas informações são solicitadas. O cliente irá receber essas informações e devolvê-las numa mensagem CONTINUE.

O servidor deve sempre enviar um REPLY, tanto em resposta ao START como ao CONTINUE, sendo que a única excepção é a indicação, por parte do cliente, de “abort” na mensagem

CONTINUE, caso em que a sessão é imediatamente abortada.

O processo de autenticação

O mecanismo de autenticação é determinado pela acção e pelo campo `authen_type` no pacote START.

Em resposta a um pacote START, o daemon envia um pedido de mais informações (GETDATA, GETUSER ou GETPASS), ou uma terminação (PASS ou FAIL). As acções e significados, quando o servidor envia uma reinicialização, ERROR ou FOLLOW.

Quando o estado de uma mensagem REPLY é TAC_PLUS_AUTHEN_STATUS_GETDATA, TAC_PLUS_AUTHEN_STATUS_GETUSER ou TAC_PLUS_AUTHEN_STATUS_GETPASS, a autenticação continua, levando o cliente a solicitar mais informações ao usuário. O cliente deve então retornar um pacote CONTINUE contendo as informações solicitadas.

TAC_PLUS_AUTHEN_STATUS_GETDATA é o pedido genérico para mais informação. Os três fazem com que a mesma acção a seja executada, mas no caso de TAC_PLUS_AUTHEN_STATUS_GETUSER, o cliente pode saber que a informação que o utilizador responde com um nome de utilizador, e para TAC_PLUS_AUTHEN_STATUS_GETPASS, que uma resposta do utilizador representa uma palavra-passe.

2.3.2.4 Autorização

TACACS + autorização é uma forma de dar autorização remota de serviços. A sessão de autorização é definida como um único par de mensagens, um REQUEST seguido de um RESPONSE.

A mensagem REQUEST contém um conjunto fixo de campos que descrevem a autenticidade do utilizador ou processo, e um conjunto variável de argumentos que descreve os serviços e opções para as quais autorização é requerida.

O RESPONSE contém um conjunto variável de argumentos de resposta, que pode restringir ou modificar as acções dos clientes.

Os argumentos em ambos REQUEST e RESPONSE podem ser especificados como obrigatórios ou facultativos. Um argumento opcional é aquele que pode ou não ser usado, modificado ou mesmo compreendido por quem o recebe.

Um argumento obrigatório deve ser tanto compreendido como utilizado. Isto permite para alargar a lista de atributos ao mesmo tempo que proporciona compatibilidade com versões anteriores.

2.3.2.5 Contabilização

A contabilização em TACACS+ é muito semelhante à autorização. O formato do pacote é, também, semelhante. Existe uma parte fixa e uma outra extensível. A porção extensível usa todos os mesmos atributos que a autorização, e acrescenta outros.

Tudo o que é dito acerca do TACACS+ é baseado em [9].

2.3.3 Diameter

Diameter é um protocolo AAA (Autenticação, Autorização e Accounting) criado para ser o sucessor do RADIUS. Na altura em que este foi escolhido como sucessor do RADIUS, vários protocolos competiam para ocupar essa posição, sendo, todos eles, avaliados em vários aspectos como a escalabilidade, recuperação de falhas (failover), autenticação mútua entre cliente e servidor, segurança ao nível do transporte, confidencialidade dos dados, integridade dos dados, transporte de certificados, mecanismos de transporte confiáveis, possibilidade de correr sobre IPv4, possibilidade de correr sobre IPv6, suporte para proxies, possibilidade de transportar atributos de serviços específicos. Diameter foi o que cumpriu melhor todos os requisitos.

2.3.3.1 Especificações

A especificação base do Diameter estão definidas no RFC3588. Esta define os elementos base como as mensagens básicas do protocolo, atributos, e a estrutura dos atributos. A especificação base também explica de forma extensa as operações entre diferentes zonas, ou domínios, definindo claramente o papel dos vários agentes Diameter no processamento de rotas de mensagens Diameter, por exemplo, assim como explica os conceitos envolvidos no transporte de mensagens. Algo que é novo no Diameter, para este tipo de protocolos, é o conceito de aplicação. Uma aplicação pode ser um serviço, um protocolo, ou procedimento que utilize as estruturas do protocolo Diameter, assim como os seus servidores e proxies. Todas as aplicações do Diameter devem suportar as suas funcionalidades. Por mais estranho que pareça, no Diameter, a interacção com o NAS é, também, considerada uma aplicação. Apesar do volume e da percepção de ser bastante completa, a especificação base do Diameter não contém qualquer descrição acerca dos procedimentos de autenticação e autorização, nem da interacção com o NAS. As interacções com NAS são definidas num RFC diferente, assim como todas as outras aplicações. A especificação base do Diameter define apenas define que o cliente (NAS) é a entidade que permite a autenticação e/ou autorização de um utilizador ao servidor, mas não oferece qualquer detalhe acerca do procedimento das mensagens ou das próprias mensagens. No entanto, define mensagens que terminam sessões de utilizador. A especificação base do Diameter também especifica segurança mecanismos de segurança end-to-end e hop-by-hop. Isto é conseguido através da ocultação de atributos. O transporte do Diameter é feito sobre TCP ou sobre SCTP. Os agentes Diameter têm de suportar tanto ambos os protocolos de transporte, enquanto que o NAS (cliente) pode suportar apenas um deles, ou os dois.

Aplicações do Diameter

Aplicação	Identificador
Mensagens comuns	0
NASREQ	1
Mobile IP	2
Accounting	3
Relay	0xffffffff
Aplicação SIP	A ser determinado pelo IANA

Tabela 2.2: Identificadores de aplicação Diameter

A especificação base do Diameter apenas descreve o suporte de accounting, outros protocolos e serviços que usam o Diameter e os servidores do Diameter são considerados aplicações para o Diameter e são especificados em diferentes documentos. A cada uma das aplicações foi atribuído um identificador, como é mostrado na tabela 2.2. Assim, torna-se fácil para agentes Diameter comunicar entre si e dar a conhecer as aplicações que cada um suporta.

Em seguida faz-se uma lista de algumas das aplicações do Diameter:

Accounting: Esta é a única aplicação que é descrita da especificação base do Diameter.

Controlo de crédito: Dada a crescente popularidade de serviços pré-pagos, o suporte de um modelo pré-pago está a tornar-se cada vez mais importante para infra-estruturas AAA.

NAS: Esta aplicação descreve detalhes da interacção dos servidores Diameter com os NAS, para procedimentos de autenticação e outros. NASREQ, como é conhecida, suporta atributos RADIUS tanto quanto possível, para permitir compatibilidade com RADIUS.

Mobile IP: Enquanto que Mobile IP facilita o encaminhamento de pacotes e o transporte dados de uma estação móvel, o Diameter, entre outras coisas, facilita a verificação da identidade e mecanismos de autorização da estação móvel. A comunidade Mobile IP do IETF definiu mecanismos de gestão de chaves e segurança que são oferecidos por protocolos AAA, mas deixando à comunidade AAA a tarefa de definir os seus mecanismos AAA para os suportar. Esta aplicação permite a mobilidade através de vários domínios.

EAP: Esta é uma aplicação que define procedimentos para permitir a conversação EAP entre o NAS e o servidor, sobre Diameter.

Tipos de nós Diameter

Nó Diameter: um processo numa estação que implementa o protocolo Diameter.

Peer Diameter: um nó Diameter que tem uma ligação de transporte directa com outro nó.

Cliente Diameter: um dispositivo nos limites da rede que executa o controlo de acesso. Por exemplo, NAS ou o Foreign Agent do Mobile IP.

Servidor Diameter: um dispositivo servidor que trata pedidos AAA de uma zona, ou domínio. O servidor tem de implementar o protocolo Diameter, assim como as aplicações utilizadas nesse domínio.

Agente Diameter: um agente é um nó Diameter que oferece serviços de relay, proxy, redireccionamento ou tradução.

-Os agentes relay encaminham mensagens Diameter baseados nos atributos do Diameter relacionados com rotas e tabelas especiais de rotas. Os agentes relay não tomam qualquer decisão de políticas e, por isso, não examinam qualquer tipo de atributo que não os de rotas. Como consequência, estes agentes apenas precisam entender atributos de rotas. Um relay nunca origina uma mensagem, mas é capaz de tratar qualquer tipo de aplicação ou mensagem.

-Os agentes proxy podem ser vistos como agentes relay que também podem tomar decisões de políticas. proxies tipicamente não respondem ao cliente, mas podem gerar uma mensagem Reject no caso das políticas serem violadas.

-Agentes de redireccionamento não se encontram no caminho de encaminhamento e não alteram qualquer atributo. Então, eles não encaminham qualquer mensagem, mas indicam um servidor ao cliente através de mensagens de redireccionamento, de acordo com a configuração. Estes são capazes de lidar com qualquer tipo de mensagens, mas podem ser configurados para redireccionar apenas alguns tipos de mensagens.

-Agentes de tradução fazem a tradução de protocolos entre o Diameter e outros protocolos AAA, tal como o RADIUS. Estes são especialmente pensados para criar compatibilidade com RADIUS.

2.3.3.2 Mensagens Diameter

Diameter é um protocolo peer-to-peer, o que significa que tanto o cliente como o servidor podem originar tanto um pedido como uma resposta. A terminologia das mensagens do Diameter não inclui o termo “tipo de mensagem”, por isso pode-se dizer que existem apenas dois tipos de mensagens: pedidos e respostas. Em vez disso, Diameter usa o conceito de comandos. Os comandos distinguem-se entre si pelo código do comando que especifica o tipo de função que a mensagem Diameter pretende executar. A acção a ser executada, em conjunto com as mensagens

Version 8 bits	Message length (24 bits)
Flags 8 bits	Command Code (24 bits)
Application ID (32 bits)	
Hop-by-hop ID (32 bits)	
End-to-end ID (32 bits)	
AVPs...	

Figura 2.10: Formato das mensagens Diameter

recebidas, é definida pelo código de comando e pelos atributos incluídos nas mensagens.

Formato da mensagem

O formato de uma mensagem Diameter é representado pela figura 2.10.

“Version” indica a versão do protocolo Diameter, que por enquanto ainda é a número um.

“Command flags” especifica 4 flags:

- R, para assinalar um pedido (Request)
- P, para assinalar que a mensagem é “proxiable”
- E, para assinalar um erro na mensagem
- T, para assinalar que se trata de uma mensagem retransmitida.

“Command code” indica o comando associado à mensagem.

“Application ID” identifica a aplicação usada na mensagem.

“Hop-by-hop identifier” é um identificador usado para fazer a correspondência entre pedidos e resposta sobre um determinado salto na cadeia de nós Diameter.

“End-to-end identifier” é um identificador usado para detectar mensagens duplicadas. Tal como no campo anterior, este faz a correspondência entre pedidos e respostas.

Código de comando

Tanto os pedidos como as respostas de um determinado tipo de comando têm o mesmo código de comando, mas a mensagem do pedido activa a flag R. Tipicamente, a resposta contém um Result-Code AVP, indicando o resultado do processamento do pedido. A tabela 2.3 mostra alguns dos comandos da especificação base do Diameter.

Comando	Abreviação	Código	Descrição
Capabilities-Exchange Request/Response	CER/CEA	257	Permite a descoberta da identidade de um elemento Diameter e as suas capacidades
Disconnect-Peer-Request/Answer	DPR/DPA	282	Enviado a um elemento Diameter para informá-lo da intenção do remetente de fechar a ligação
Re-Auth-Request/Answer	RAR/RAA	258	Enviado por um servidor a um autenticador, para reautenticar ou reautorizar o utilizador
Session-Termination Request/Answer	STR/STA	275	Enviado pelo autenticador, em nome do utilizador, para terminar a sessão
Accounting Request/Answer	ACR/ACA	271	Enviado por um elemento Diameter para trocar informação de contabilização com outro elemento

Tabela 2.3: Exemplos de comandos Diameter

Tal como seria de esperar, mesmo que se apresentassem todos os comandos da especificação base do Diameter na tabela seria impossível esta conter todos os comandos existentes, dado que, por exemplo, comandos relativos à interacção com o NAS são descritos na especificação do NAS-REQ, acontecendo o mesmo para outras aplicações.

AVP

Muita da informação transportada pelo Diameter está contida nos AVPs (Attribute-Value Pair). A figura 2.11 apresenta a estrutura de um AVP.

O “AVP code” identifica o tipo de informação incluída no campo de dados do atributo. O Diameter reserva os códigos 0-255 para compatibilidade com os atributos definidos pelo RADIUS. O bit “M” das flags tem a finalidade de indicar se o nó Diameter necessita que o outro peer suporte o atributo para processar a mensagem. O bit “P” das flags indica a necessidade de encriptação, para segurança end-to-end. Os bits “R” estão reservados para uso futuro.

Em seguida, faz-se a apresentação de alguns dos atributos definidos na especificação base do Diameter:

-Origin-Host: Este atributo é adicionado pela estação que origina a mensagem Diameter e não pode ser modificado pelos agentes relay. Este tem de estar presente em todas as mensagens.

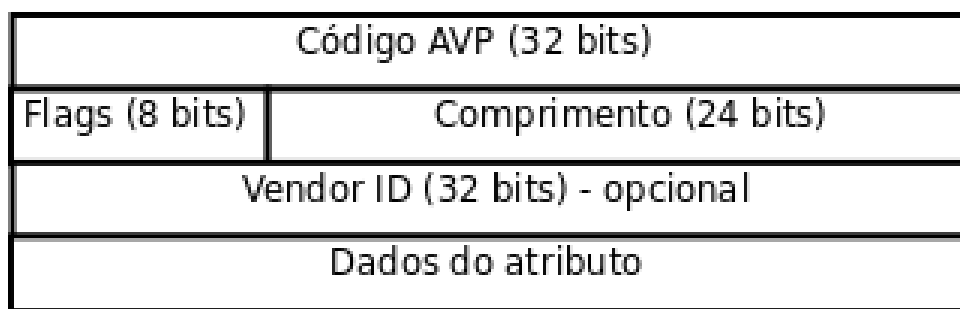


Figura 2.11: Formato dos atributos Diameter

-Origin-Realm: Este atributo contém o domínio da estação que gerou a mensagem Diameter e tem de estar presente em todas as mensagens. Tal como o Origin-Host, não pode ser alterado por agentes relay.

-Destination-Host: Este atributo, tipicamente, é usado para encaminhar uma mensagem para um destino fixo.

-Destination-Realm: contém o domínio de destino para onde a mensagem deve ser encaminhada.

-Routing: É usado para encaminhamento de mensagens. Agentes Diameter precisam de processar e alterar este atributo e, por essa razão, não pode ser protegido por segurança end-to-end.

-Result-Code: O seu conteúdo indica se um determinado pedido foi completado com sucesso ou não. Todas as mensagens de resposta têm de conter este atributo.

2.3.3.3 Conceitos de transporte e encaminhamento do Diameter

Conceitos de transporte

A robustez e o transporte são aspectos importantes na formação do protocolo Diameter. A especificação base diz que os clientes Diameter devem suportar TCP e/ou SCTP, enquanto que os agentes e servidores têm de suportar ambos TCP e SCTP. Na especificação são definidos dois conceitos importantes de transporte, a sessão e a conexão. A sessão é um conceito lógico ao nível da camada da aplicação e é estabelecida entre um dispositivo (normalmente, um cliente, um NAS) e um servidor. A conexão é, por outro lado, um conceito ao nível da camada de transporte e é estabelecida entre dois peers que enviam e recebem mensagens.

Conceitos de encaminhamento

De seguida, são explicados alguns dos principais conceitos de encaminhamento do protocolo Diameter.

-peer table: A peer table é usada no encaminhamento de mensagens e é referenciada pela realm-based routing table. Cada nó Diameter mantém uma destas tabelas que inclui entradas

acerca de cada um dos seus peers. Cada entrada desta tabela inclui informação sobre a identidade do peer, informação útil sobre o estado deste, se a entrada é estaticamente ou dinamicamente configurada.

-Realm-based routing table: Agentes Diameter consultam esta tabela para encaminhar a mensagem para o destino ou o próximo nó mais apropriado ao encaminhamento da mensagem.

Encaminhamento das mensagens Diameter

A não ser que o nó Diameter seja o destino final da mensagem e tenha que a processar localmente, o nó terá de encaminhar a mensagem através da consulta da peer table. Esta tabela contém todos os peers com quem pode comunicar directamente. De seguida, são apresentados alguns dos conceitos mais importantes.

-Os pedidos que não podem ser proxied e têm de ser processados localmente devem possuir uma indicação que mostre que são para consumo local. A indicação mais óbvia é a de que o atributo Destination-Host contém a identidade da estação local. Uma outra indicação, menos óbvia, é quando o atributo Destination-Host não está presente, mas o Destination-Realm está presente e inclui um domínio ao qual a estação local pertence e a aplicação em questão é suportada. Uma outra indicação é quando o pedido não contém Destination-Host ou Destination-Realm.

-Pedidos que precisam de ser enviados para um domínio específico, mas que podem ser processados por qualquer servidor dentro desse domínio, devem conter apenas o atributo Destination-Realm e não o Destination-Host .

-Pedidos que necessitam ser enviados para um servidor específico, devem conter ambos Destination-Realm e Destination-Host.

2.3.3.4 Negociação de capacidades

A especificação base do Diameter define o procedimento de negociação de capacidades, a qual permite aos peers terem a noção do suporte das capacidades (funcionalidades e aplicações) dos outros. Dois peers devem negociar capacidades antes de estabelecer qualquer ligação entre si. A negociação consiste em um peer enviar um capabilities exchange request (CER), ao qual o outro peer responde com um capabilities exchange answer (CEA). Ao enviar uma mensagem relativa a capacidades , cada peer indica as suas capacidades para cada aplicação. Isto optimiza interacções futuras dos peers, visto que cada peer evitará enviar comandos que relacionados com aplicações que outro não suporta. Como as mensagens Diameter poderão ter de ser encaminhadas por vários nós e agentes até chegarem ao destino, as mensagens terão de ser encaminhadas através de peers que suportem as capacidades necessárias para essas mensagens.

2.3.3.5 Segurança

Diameter obriga que tanto clientes como servidores suportem IPsec. Já no caso do TLS, é obrigatório para os servidores mas opcional para clientes. Uma das razões para facilitar a segu-

rança TLS para os clientes é o facto de estes não precisarem de suportar certificados. Assim, IPsec pode ser usado primeiramente para segurança dentro dum domínio, enquanto o uso adicional do TLS é aplicado em comunicações entre domínios.

Apesar do IPsec e TLS oferecerem protecção numa conexão, segurança end-to-end pode ser necessária, por exemplo quando um determinado atributo tem de ser protegido dos intermediários.

A especificação base do Diameter assume que as mensagens são protegidas pelo uso do IPsec ou TLS. Quando o IPsec não é utilizado, apenas a negociação de capacidades pode ser executada sem o auxílio do TLS. Os peers devem começar uma interacção TLS depois de uma negociação de capacidades. Se esta falhar, a conexão entre os peers deve cessar.

Todas as implementações Diameter devem suportar o modo de transporte do IPsec, com algoritmos de encriptação e autenticação para oferecer confidencialidade e autenticação a cada pacote transmitido. As implementações também devem suportar mecanismos anti-replay do IPsec. O Diameter obriga o uso do IKE (Internet Key Exchange) para autenticação de peer, gestão de chaves e negociação de associações de segurança do IPsec. No entanto, Diameter não obriga a implementação de todos os tipos de autenticação IKE e apenas obriga o uso de segredos partilhados, deixando o uso de certificados como opcional. Um problema que surge com uso de certificados com IPsec e IKE é que não é possível configurar Autoridades de Certificados para cada aplicação em separado. Isto implica uma falha de uso de certificados para o IPsec, em que a mesma política de segurança é aplicada a todas as aplicações.

Como parte integrante da segurança de cada conexão, não só os peers Diameter em cada lado da conexão precisam de se autenticar mutuamente, como também têm de autorizar tanto a conexão como a sessão. Por exemplo, o simples facto de um peer se autenticar com sucesso não significa que esteja automaticamente autorizado a agir como um servidor Diameter que suporta as aplicações que anuncia. Os seguintes procedimentos são necessários:

- Autorização de funcionalidade: Antes de iniciar uma conexão, um peer Diameter tem de se certificar que os seus peers estão autorizados a actuar nos seus papéis. Antes de se criar uma conexão, verificações de autorização são efectuadas em cada conexão ao longo de um caminho.

- O servidor, antes de autorizar uma sessão, tem de verificar se a rota percorrida pelo pedido é aceitável.

- Mensagens de accounting poderão igualmente estar sujeitas a uma verificação de segurança.

- Agentes Diameter, que fazem proxying de mensagens Diameter, também precisam de verificar a dados sobre as rotas das mensagens que vêm do servidor.

2.3.3.6 Detalhes de aplicações

Accounting

A figura 2.12 ilustra um exemplo de troca de mensagens da aplicação de accounting do Diameter. Os nomes das mensagens explicam-se por si próprios. A negociação de capacidades é

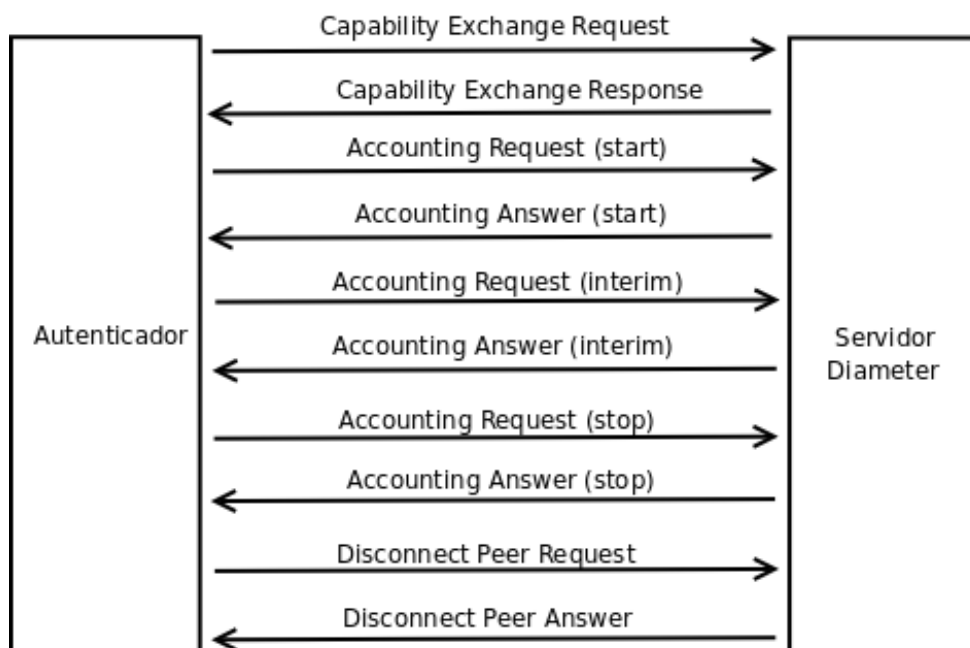


Figura 2.12: Exemplo de troca de mensagens da aplicação Accounting do Diameter

incluída para tornar a figura mais completa, garantindo que a aplicação é suportada pelos dois nós.

NASREQ

A especificação da aplicação NAS do Diameter não está ainda completamente definida. Como é sabido, a maior parte das implementações de autenticação baseiam-se no protocolo EAP, pelo que qualquer nó que suporte a aplicação EAP do Diameter terá igualmente de suportar o NASREQ.

O Diameter foi pensado para ser o sucessor do RADIUS, mesmo sabendo que o RADIUS é implementado de forma global e persistente. Assim, foi previsto um período de transição, no qual o Diameter co-existisse com o RADIUS. Isto fez a especificação da aplicação NAS dar especial atenção aos agentes de tradução.

Comandos introduzidos pelo NASREQ

-O comando AA-request (AAR) é enviado por um NAS para pedir autenticação e/ou autorização para um determinado utilizador.

-O comando AA-answer (AAA) é enviado em resposta a um AAR e pode incluir atributos relacionados com autorização.

Mobile IP

A especificação desta aplicação é bastante completa, não dá detalhes sobre autenticação, mas também autorização e accounting. Por exemplo, não só ajuda a especificar como o Foreign Agent

autentica a estação móvel que está a visitar uma outra rede, mas também como é que o Foreign Agent informa o Home Administrative Domain da estação móvel acerca dos recursos que a estação móvel consome na rede que visita. Contudo, deve-se salientar que esta especificação apenas se aplica ao IPv4 e não ao IPv6.

EAP

Como a aplicação EAP tem a ver com autenticação, esta está intimamente ligada à aplicação NASREQ. Isto é tão verdade que, para um agente suportar a aplicação EAP, terá também de suportar o NASREQ. Quando um servidor não suporta EAP, este deve permitir ao NAS negociar outros mecanismos de autenticação como o PAP ou o CHAP.

Apresenta-se de seguida, e de forma breve, o modelo de transporte do EAP sobre o Diameter:

- Mensagens EAP do NAS para o servidor são transportadas dentro de mensagens Diameter-EAP-Request. O EAP em si é transportado como atributo do Diameter.

- Mensagens EAP do servidor para o NAS são transportadas dentro de mensagens Diameter-EAP-Answer. Mais uma vez, o EAP em si é transportado como atributo do Diameter.

Embora, normalmente, o servidor EAP esteja co-alojado com o servidor Diameter, como o EAP e o Diameter continuam a ser mecanismos separados, a sua interacção requer algumas considerações:

- O EAP é falado entre o utilizador e o servidor. O NAS apenas entende Diameter, não podendo entender mensagens EAP, excepto o EAP success e EAP failure, agindo apenas como meio de comunicação entre o utilizador e o servidor para o resto das mensagens.

- O sucesso do processo de autenticação tipicamente é passado directamente ao utilizador através de um EAP success ou failure, enquanto que o resultado da autorização é passado ao NAS através do atributo Result-Code do Diameter.

- Muitas redes, hoje em dia, protegem a identidade do utilizador permitindo que este envie uma identidade falsa nos primeiros e desprotegidos pedidos EAP. Noutras alturas apenas as mensagens EAP, não as Diameter, contêm a identidade do utilizador. Um NAS que não entenda EAP poderá não ser informado acerca da verdadeira identidade do utilizador.

Toda a sub-secção sobre o Diameter, até este ponto, é baseada em [10].

2.3.3.7 Tradução Diameter-RADIUS

Durante a criação do projecto do Diameter, foi feito um grande esforço a pensar em estruturas que permitissem a co-existência do RADIUS com o Diameter. Um exemplo disso é o facto de se

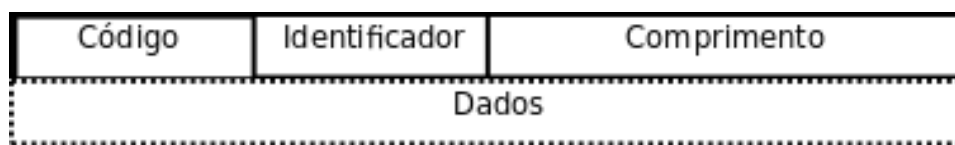


Figura 2.13: Formato dos pacotes EAP

garantir que a numeração dos atributos do RADIUS esteja incluída no espaço dos atributos do Diameter, para evitar a conversão de atributos. Dado o trabalho relativo à autenticação e autorização, o NASREQ é a especificação Diameter com grandes semelhanças com o RADIUS. Por essa razão, a aplicação NAS do Diameter é a primeira especificação que descreve interoperabilidade entre Diameter e RADIUS. Pretende-se conseguir esta interoperabilidade através de uma arquitectura que consiste em sistemas distintos de RADIUS e Diameter que se comunicam através de um agente de tradução nas suas fortalezas. O NASREQ descreve vários requisitos e procedimentos de conversão para os agentes de tradução RADIUS-Diameter. Esta parte é baseada em [11].

2.4 EAP

2.4.1 Estrutura do pacote EAP

2.4.1.1 Código

Este campo, com comprimento de um octeto, identifica o género de pacote EAP.

2.4.1.2 Identificador

Identificador é um campo com um comprimento de um octeto. Este campo torna possível associar pacotes EAP Response com EAP Request. Dado o valor do Identificador de um determinado pacote EAP Request, a resposta a esse pacote, o EAP Response, terá necessariamente de ter o mesmo valor do identificador, caso contrário será descartado. O mesmo é válido quando se trata de uma retransmissão. O pacote retransmitido terá o mesmo valor do Identificador do pacote perdido ou com erros.

2.4.1.3 Comprimento

Comprimento é constituído por dois octetos e indica o comprimento total, em octetos, do pacote EAP.

2.4.1.4 Dados

Os dados do pacote EAP dependem do género de pacote EAP utilizado e o Método, no caso de pacotes EAP Request e EAP Response.

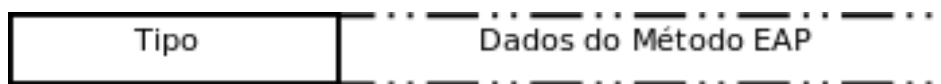


Figura 2.14: Formato dos dados EAP

2.4.2 Géneros de pacotes EAP

2.4.2.1 Request

Com o código “1”, Request é o pacote EAP que o Servidor usa para transmitir dados de Autenticação ao Suplicante. Nos casos em que existe autenticação mútua, o Suplicante também envia um Request a pedir que o Servidor se identifique.

2.4.2.2 Response

Este pacote, de código “2”, é enviado em resposta a um Request. Assim sendo, na maioria dos casos, Response é enviado estritamente pelo Suplicante, pelo que só nos casos de autenticação mútua é que seria enviado pelo Servidor.

2.4.2.3 Success

Este pacote, de código “3”, é enviado pelo Servidor ao Suplicante depois de todo o processo de autenticação ser feito com sucesso, dando permissão ao Suplicante de aceder a rede protegida.

2.4.2.4 Failure

Failure, de código “4”, é enviado pelo Servidor ao Suplicante depois de todo o processo de autenticação falhar, informando o Suplicante de que não pode aceder a rede protegida.

2.4.3 Tipos de EAP Request/Response

Os dados dos pacotes Request e Reponse são divididos na estrutura apresentada na figura 2.14.

2.4.3.1 Tipo

O campo Tipo, de comprimento um octeto, determina o tipo de EAP (Método EAP) usado para a autenticação.

2.4.3.2 Dados do Método EAP

Este campo contém os dados do Método EAP e tem tamanho variável.

2.4.4 Procedimentos

O processo de autenticação dá-se da seguinte maneira:

1. O Servidor envia um Request para autenticar a máquina ligada. Este pacote tem o um campo “tipo” para indicar o que é que está a ser pedido, ou seja, em muitos dos casos, o Método EAP a ser usado. Tipicamente, o Servidor envia inicialmente um Identity Request, mas este não é absolutamente necessário e pode ser omitido.

2. A máquina envia um pacote Response em resposta a um válido Request enviado pelo Servidor. Tal como com o pacote Request, o pacote Response tem o um campo “tipo” para indicar o que é que está a ser pedido.

3. Existe uma troca de pacotes Request e Response entre o Servidor e a máquina a autenticar, enquanto tal for necessário para o processo de autenticação. EAP é um protocolo “lock step”, pelo que, a não ser que seja enviado outro Request inicial, um novo Request não pode ser enviado antes de se receber um Response válido. O Servidor é responsável por retransmitir Requests. Depois de um certo número de retransmissões, o Servidor deve terminar a conversação EAP. O Servidor não deve enviar pacote Success, ou Failure, ao retransmitir, ou quando não consegue uma resposta da máquina a autenticar.

4. A conversação continua até que haja uma autenticação bem sucedida, ou até que o Servidor não consiga autenticar a máquina. No primeiro caso, o Servidor envia um EAP Success, enquanto que no segundo caso este transmite um EAP Failure.

O que é dito acerca do EAP é baseado em [12] e [13].

2.5 Métodos EAP

Existem 4 tipos de EAP obrigatórios em qualquer implementação do protocolo EAP, sendo esses o Identity (tipo “1”), Notification (tipo “2”), Nak (tipo “3”) e MD5-Challenge (tipo “4”). Para além destes, também deve ser suportado o tipo “254” (Expanded Types). Existem ainda outros que não são obrigatórios mas que são normalmente usados nas implementações, tais como One Time Password (tipo “5”), Generic Token Card (tipo “6”) e o tipo “255” para uso experimental. Finalmente, para além de tudo isto, existem muitos mais tipos de EAP definidos em vários RFCs e outros proprietários.

2.5.0.1 Identity

O tipo Identity é usado para inquirir a identidade do Suplicante. Geralmente, este é o Request inicial do processo. Um Response do tipo 1” deve ser enviado em resposta a um Request do mesmo tipo. É recomendado que o Identity Response seja usado para routing e para seleccionar o Método EAP a usar de seguida. Esse método deve incluir um mecanismo próprio para obter a identidade, para que não seja necessário responder em Indetity Response. Identity Request e

Response são enviados em texto, sem qualquer protecção ou encriptação, pelo que alguém poderá saber a identidade através dos pacotes ou até mesmo modificar ou falsificar identidade.

2.5.0.2 Notification

Este tipo é usado opcionalmente para transportar uma ser mostrada ao Suplicante. O suplicante deve responder ao Notification Request com um Notification Response, a não ser que o método utilizado expressamente proíba o uso de mensagens de notificação. Um Método EAP pode, nas suas especificações, que nenhuma notificação pode ser enviada durante o método. Neste caso o Suplicante descarta o Notification Request sem informar o Servidor. Em nenhuma altura um Nak Response deve ser enviado como resposta a um Notification Request.

2.5.0.3 Nak

Legacy Nak

O tipo Nak é válido apenas nos Response. É enviado em resposta a um Request em que o tipo de autenticação não pode ser aceite. O tipo “0” é usado para indicar que o Suplicante não tem alternativas viáveis e, portanto, o Servidor não deveria enviar outro Request depois de receber um Nak Response contendo um valor “0”. Como o Nak é válido apenas para Response e tem uma funcionalidade limitada, não deve ser usado como um indicador genérico de erros, tal como a comunicação de mensagens de erro, ou a negociação de parametros específicos para um determinado Método EAP.

Expanded Nak

Este tipo tem exactamente a mesma funcionalidade que o Nak anterior, apenas com a diferença de que só deve ser enviado em resposta a um Request do tipo “254” (Expanded Type). O Expanded Nak usa o próprio formato do Expanded Type, e o Response contém um ou mais tipos de autenticação desejados pelo Suplicante, todos no Expanded Type.

2.5.0.4 MD5-Challenge

O Request contém uma mensagem de “desafio” (message digest) ao Suplicante. O Suplicante responde enviando um MD5-Challenge Response com o resultado de uma transformação MD5 do desafio com a palavra-passe fornecida pelo utilizador.

2.5.0.5 One Time Password

O sistema One-Time Password está definido em “A One-Time Password System” (RFC2289) e em “OTP Extended Responses” (RFC2243). O Request contém um “desafio” OTP no formato

descrito no RFC2289. Um Response tem necessariamente de ser enviado em resposta ao Request, para além de ser do tipo “5” (OTP), “3” (Nak), ou “254” (Extended Nak). O Método EAP OTP é para ser usado apenas com um sistema One-Time Password, pelo que não deverá servir de suporte a palavras-passe em texto visível.

2.5.0.6 Generic Token Card

O tipo Generic Token Card é definido para ser usado com várias implementações Token Card, as quais requerem que o utilizador que introduza algo. O Request contém uma mensagem e o Response contém informação necessária para a autenticação. Tipicamente, esta seria informação lida do Token Card pelo utilizador e introduzida como texto. O Response deverá ser do tipo “5” (OTP), “3” (Nak), ou “254” (Extended Nak). O Método EAP Generic Token Card foi destinado a ser usado com Token Cards e não deve ser usado para servir de suporte a palavras-passe em texto visível.

2.5.0.7 Expanded Types (“254”)

Como muitas das implementações do EAP são específicas de produtores, o método Expanded Type está disponível para permitir aos produtores suportarem os seus próprios Expanded Types, incompatíveis com o uso geral. Expanded Type é usado também para expandir o espaço de tipos de métodos para além dos 255 originais.

2.5.1 Outros tipos de métodos EAP

2.5.1.1 EAP-TLS

EAP com Transport Layer Security (EAP-TLS), tipo “13”, fornece autenticação mútua, em que ambos Suplicante e Servidor de Autenticação provam a sua identidade um ao outro através de certificados. As comunicações entre eles é conseguida através de um tunel TLS encriptado, pelo que torna este método muito seguro.

2.5.1.2 EAP-TTLS

EAP-TTLS (Tunneled Transport Layer Security), do tipo “21”, foi desenvolvido como uma extensão do EAP-TLS. Tal como EAP-TLS, EAP-TTLS é baseado em certificados e autenticação mútua. No entanto, os certificados são apenas necessários do lado do servidor. Isto é feito ligando o Suplicante ao Servidor de Autenticação via TLS através de um tunel seguro. Os utilizadores podem autenticar-se eles próprios através de uma password, em vez de um certificado. Isto reduz bastante a complexidade do sistema de autenticação, pois não existe a necessidade de se instalar e gerir certificados nos dispositivos cliente.

2.5.1.3 PEAP

Do tipo “25”, PEAP (Protected Extensible Authentication Protocol), tal como EAP-TTLS, não precisa de certificados nos dispositivos cliente. PEAP oferece um transporte seguro dos dados de autenticação e, para tal, usa um túnel entre clientes PEAP e o Servidor de Autenticação.

2.5.1.4 LEAP

LEAP (Lightweight Extensible Authentication Protocol), do tipo “17”, foi desenvolvido para LANs sem fios. LEAP fornece encriptação através de chaves WEP (Wired Equivalent Privacy) dinamicamente geradas e faz uso de fortes palavras-passe para autenticar Suplicantes. É de notar que é possível facilmente penetrar numa rede com LEAP implementado através de uma ferramenta chamada ASLEAP, pelo que é fortemente recomendado o usado de um protocolo diferente em novas implementações.

2.5.1.5 EAP-FAST

EAP-FAST (Flexible Authentication via Secure Tunneling), do tipo “43”, foi desenvolvido como um substituto do LEAP. EAP-FAST fornece autenticação mútua através do uso de Protected Access Credential (PAC) em vez de certificados digitais. Um administrador pode distribuir PACs manualmente ou automaticamente através do Servidor de Autenticação. A vantagem do EAP-FAST é que não é necessária a instalação de certificados digitais.

2.5.1.6 EAP-SIM

EAP-SIM (Subscriber Identity Module) oferece autenticação mútua para cartões SIM instalados em telemóveis GSM. EAP-SIM permite que o cartão se autentique com um Servidor de Autenticação GSM e vice-versa.

Esta secção sobre os Métodos EAP é baseada em [14].

Capítulo 3

Análise Crítica e Comparativa

Este capítulo tem o objectivo de analisar de forma crítica cada um dos protocolos envolvidos em sistemas de autenticação, autorização e contabilização, de modo a perceber as vantagens e desvantagens de cada um. Com isto, pretende-se chegar a uma conclusão em relação à adequação de cada protocolo, nomeando um como parte integrante das soluções apresentadas.

3.1 Análise Crítica

3.1.1 RADIUS

3.1.1.1 Robustez do transporte de RADIUS

O transporte de RADIUS é feito sobre UDP. Inicialmente, na altura em que RADIUS foi criado, UDP parecia ser mais apropriado que TCP, dado que o TCP consome tempo no estabelecimento de uma ligação. Mas o facto de se utilizar UDP em vez do TCP faz com que seja possível a perda definitiva de pacotes, podendo criar vários problemas, principalmente em funções de contabilização. No próprio RFC do RADIUS foi feita uma nota por parte do Internet Engineering Steering Group (IESG) que diz: “This protocol is widely implemented and used. Experience has shown that it can suffer degraded performance and lost data when used in large scale systems, in part because it does not include provisions for congestion control. Readers of this document may find it beneficial to track the progress of the IETF’s AAA working group, which may develop a successor protocol that better addresses the scaling and congestion control issues.” Este parágrafo é baseado em [15] e [16].

3.1.1.2 Segredos partilhados estáticos e configurados manualmente

Não existe qualquer método para a determinação dinâmica e automática de segredos, no RADIUS. Os segredos são, por norma, configurados manualmente no Autenticador. Se o RADIUS for implementado numa rede de grandes dimensões, por uma questão de simplicidade, os gestores

muitas vezes chegam mesmo a usar as mesmas chaves ou segredos para vários, ou até mesmo todos, os Autenticadores. Para além disto, estas chaves já são, por si só, de longo prazo, ou seja, mudam muito raramente, sendo mais fácil descobri-las.

3.1.1.3 Procura do segredo partilhado

Para prevenir adulteração das mensagens, o servidor RADIUS usa o endereço IP de origem no pacote RADIUS UDP para procurar o segredo que partilha com o NAS. Isto pode criar vários problemas, principalmente quando o endereço IP do NAS muda. Por exemplo, um NAS pode ter a necessidade de obter o seu endereço IP dinamicamente através de DHCP, como pode ser o caso de muitas WLANs.

3.1.1.4 Encadeamento de proxies

Em implementações que utilizam proxies RADIUS entre o NAS e o servidor, o NAS apenas partilha um segredo com o primeiro proxy, com o qual comunica, e não com o Servidor de Autenticação propriamente dito. Isto significa que o NAS comunica com o servidor RADIUS baseado numa cadeia de confiança em vez de uma confiança directa com o servidor. O facto do RFC 2607 ter um carácter informativo faz com que a implementação não seja feita de forma global. O caminho para o próximo proxy em direcção ao servidor é determinado pelo Network Access Identifier (NAI), do NAS. No entanto, o RFC não especifica o procedimento de descoberta de caminho. Os proxies podem modificar o pedido ou resposta sem oferecer qualquer tipo de notificação ou assinatura. Isto não só cria incompatibilidade entre os pedidos do suplicante e os serviços prestados, como também abre a possibilidade de existirem ataques “Man-in-the-middle” (MITM) e corrupção das mensagens transmitidas, sem que seja possível a detecção. Mesmo quando existe autenticação dos elementos intermediários, esta é baseada em segredos partilhados com os proxies vizinhos. Por isto, a ocultação de atributos, desde o NAS ao servidor, não é possível.

3.1.1.5 Protecção do transporte

A ocultação de atributos oferece protecção selectiva na camada de aplicação. Esta não oferece qualquer protecção de segurança para as mensagens RADIUS ou camadas de protocolos (UDP, IP) em que estas mensagens assentam. Isto significa que o endereço IP pode facilmente ser adulterado ou outros atributos serem modificados.

O que é dito acerca das falhas de segurança do RADIUS é baseado em [17]

3.1.2 TACACS+

3.1.2.1 Chaves partilhadas estáticos e configurados manualmente

Tal como no RADIUS, as chaves de encriptação dos dados são configuradas manualmente em cada autenticador que seja instalado. Por este motivo, muitas vezes as chaves são copiadas de uns autenticadores para os outros, visto que se torna mais fácil gerí-los. Isto provoca uma falha de segurança.

3.1.2.2 Sem suporte à mobilidade

Este protocolo de autenticação, autorização e contabilização não suporta a mobilidade dos suplicantes. Não é, por isso, possível implementar uma cadeia de *proxies* de modo a autenticar um utilizador que esteja numa zona distante.

3.1.2.3 Protocolo servidor-cliente

O TACACS é um protocolo servidor-cliente. Isto significa que os pedidos são feitos pelo cliente e respondidos pelo servidor, em qualquer altura. Tal significa que não existe, nem em implementações muito específicas, a possibilidade de as mensagens serem iniciadas pelo servidor. Não é possível o servidor interromper uma ligação, ou sessão, quando detecta uma falha de segurança.

3.1.2.4 Grupo restrito de dispositivos compatíveis

Sendo o TACACS+ um protocolo proprietário da Cisco Systems, nem todos os NAS o suportam. Além dos dispositivos Cisco Systems, poucos suportam o TACACS+, sendo disso exemplo os sistemas Juniper Networks [18].

3.1.3 Diameter

3.1.3.1 Fragilidade das implementações

Quando se pretende implementar um servidor Diameter, um dos problemas que surgem de imediato é a complexidade de passos até que a instalação seja completa. São necessários vários tipos de pacotes de software e alguns necessitam de ser compilados e instalados manualmente. Ainda assim, muitas vezes existem problemas na execução das aplicações Diameter. Mesmo em versões pré-compiladas e empacotadas os problemas surgem, pois existe uma série de inconsistências na construção do software do Diameter, pelo que são necessários vários remendos para que as suas aplicações comecem finalmente a correr.

3.1.3.2 Inexistência de NAS que suportem Diameter

Tanto quanto foi pesquisado, não se encontraram NAS que suportassem Diameter. Uma das grandes empresas vocacionadas para redes de computadores, a Cisco Systems, numa página de

Internet sobre protocolos AAA [19], mencionou "DIAMETER is currently not supported in the Cisco IOS Software". Contudo, numa outra página, apresenta a configuração da aplicação de controlo de crédito, como se pode constatar em [20], mas não se encontra documentação em relação a outras aplicações bem mais importantes, como o NASREQ. Tudo leva a crer que o Diameter só é suportado por uma minoria muito pequena de dispositivos, que têm uma função muito específica.

3.1.3.3 Presença obrigatória de gateways

Dada a quase inexistência de NAS que suportem Diameter, a presença de gateways Diameter, ou seja agentes de tradução, são obrigatórios. Mesmo estes ainda foram pouco explorados pelo que existem muito poucas opções, sendo que as encontradas são soluções comerciais, como se verifica em [21]. A maior parte dos NAS suportam RADIUS.

3.1.3.4 Custos sem benefícios

Como foi explicado, não existem NAS que suportem Diameter. Também foi dito que isso obrigava a que se implementassem gateways Diameter, visto que os NAS comunicam em RADIUS. Assim, como resultado, ter-se-á um sistema com um servidor Diameter, fiável, seguro e com muitas funcionalidades, mas que no fim apenas executará funções RADIUS, pois o resto do sistema ainda não está preparado para suportar Diameter. Isto leva ao aumento de custos sem que haja grandes benefícios.

3.1.3.5 Complexidade

Depois de toda a explicação do protocolo Diameter, torna-se evidente que este é bastante complexo. Não só as suas especificações são bastantes extensas, como também existem dependências entre RFCs relativos ao protocolo.

3.1.4 EAP

O Extensible Authentication Protocol foi concebido para ser o suporte de muitos métodos de autenticação, sendo bastante implementado em muitos casos, principalmente na autenticação em redes sem fios. No entanto, este protocolo apresenta uma falha de segurança que pode facilitar o acesso a um utilizador intrusão. Os pacotes EAP não apresentam qualquer tipo de confidencialidade. Isto significa que qualquer utilizador que esteja a capturar pacotes de uma autenticação 802.1X pode acompanhar e compreender todos os passos da autenticação. Um utilizador mal intencionado consegue saber que método de autenticação está a ser utilizado, se o utilizador que se tenta autenticar o faz com sucesso e, inclusive, o nome de utilizador. Estas vulnerabilidades dão poder a um utilizador que tenta forçar a entrada à rede protegida, na medida em que lhe dá muita informação.

3.1.4.1 Métodos de autenticação EAP

Dado que os pacotes EAP não estão encriptados, os métodos de autenticação, mesmo que protegidos por algum tipo de codificação, tornam-se mais vulneráveis. O caso mais flagrante é o do MD5 Challenge, visto que o resultado de uma encriptação MD5 à palavra-passe pode ser capturada. Muitos métodos de autenticação introduzidos no EAP são vulneráveis a vários ataques. Mesmo métodos bastante seguros, como o EAP TTLSv0, Microsoft PEAPv0 e Cisco PEAPv1, estão sujeitos a ataques Man In The Middle, como é dito em [22].

3.2 Análise Comparativa

3.2.1 Vantagens do RADIUS

3.2.1.1 Globalmente implementado

O protocolo RADIUS é implementado a nível global, pelo que é fácil encontrar soluções adequadas a qualquer cenário em que seja necessário instalar um sistema de autenticação e controlo de acessos.

3.2.1.2 Compatibilidade dos NAS

Com a implementação do RADIUS em grande escala, praticamente todos os dispositivos autenticadores suportam este protocolo. Deste modo, pode-se pensar na implementação de um sistema de autenticação e controlo de acessos sem que haja a preocupação de que tipo de NAS utilizar.

3.2.1.3 Métodos de autenticação

Uma das consequências das muitas implementações do RADIUS é o facto deste estar preparado para, praticamente, qualquer tipo de autenticação, mesmo quando esta não envolve a plataforma de autenticação EAP.

3.2.1.4 Flexibilidade

A implementação do RADIUS a larga escala, a sua compatibilidade com praticamente todos os NAS e o suporte de qualquer método de autenticação faz com que o RADIUS se torne bastante flexível, podendo ser implementado em qualquer tipo de cenário.

3.2.1.5 Mobilidade IP

Apesar das soluções para a mobilidade IP serem proprietárias, e não existirem normas estabelecidas de como proceder, é possível implementar sistemas que suportem mobilidade. Esta é uma característica que faz com que o RADIUS se destaque do TACACS+, aproximando-se das características do Diameter.

3.2.1.6 Segurança no transporte

Embora pareça contraditório com o que é dito acerca deste assunto na Análise Crítica, esta acaba por ser uma vantagem do RADIUS em relação ao TACACS+. Tal como no caso da mobilidade IP, a implementação do IPsec é proprietária, não está normalizada e é opcional. No entanto, é possível implementá-lo.

3.2.2 Vantagens do TACACS+ em relação ao RADIUS

3.2.2.1 UDP e TCP

RADIUS usa UDP enquanto que TACACS+ usa TCP. TCP oferece um transporte orientado à conexão, enquanto que UDP oferece uma entrega de “melhor esforço”, não dando garantias de envio. RADIUS requer variáveis programáveis adicionais, como tentativas de retransmissão e “time-outs” para compensar o transporte de “melhor esforço”, mas não tem o nível de apoio que oferece um transporte TCP.

3.2.2.2 Encriptação de pacotes

RADIUS só encripta a palavra-passe nos pacotes Access-Request, do cliente para o servidor. O restante do pacote é não encriptado. Outras informações, como nome de utilizador, serviços autorizados, e contabilização, podem ser capturadas por um terceiro.

TACACS+ encripta todo o corpo do pacote, mas deixa um cabeçalho standard de TACACS+. Dentro do cabeçalho está um campo que indica se o corpo é codificado ou não. Para fins de depuração, é útil ter o corpo dos pacotes não encriptados. No entanto, durante o funcionamento normal, o corpo do pacote é totalmente encriptado para comunicações mais seguras.

3.2.2.3 Autenticação e Autorização

RADIUS combina autenticação e autorização. Os pacotes Access-Request, enviados pelo servidor RADIUS para o cliente, contêm informações de autorização. Isto torna difícil dissociar a autenticação e autorização.

TACACS+ utiliza a arquitectura AAA, que separa os três “As”. Isto permite separar soluções de autenticação que ainda podem usar TACACS+ para autorização e contabilização. Por exemplo, com TACACS+, é possível usar autenticação Kerberos e usar autorização e contabilização do TACACS+. Após um NAS se autenticar num servidor Kerberos, pede informação de autorização a um servidor TACACS+ sem ter que se reautenticar. O NAS informa o servidor TACACS+ que se autenticou com êxito num servidor Kerberos e, em seguida, o servidor fornece informações de autorização.

Durante uma sessão, se for necessária uma verificação adicional de autorização, o servidor de acesso confirma com o servidor TACACS+ para determinar se o utilizador tem permissão para utilizar um comando em particular. Isso proporciona maior controlo sobre os comandos que podem ser executadas sobre um servidor de acesso enquanto dissociada do mecanismo de autenticação.

3.2.2.4 Suporte para múltiplos protocolos

RADIUS não suporta os seguintes protocolos:

- AppleTalk Remote Access (ARA) protocol
- NetBIOS Frame Protocol Control protocol
- Novell Asynchronous Services Interface (NASI)
- X.25 PAD connection

TACACS+ oferece suporte para múltiplos protocolos.

3.2.2.5 Gestão de routers

RADIUS não permite aos utilizadores controlarem quais comandos podem ser executados num router e quais não podem. Portanto, RADIUS não é tão útil para a gestão de routers ou tão flexível para serviços de terminal.

TACACS+ fornece dois métodos para controlar a autorização de comandos do router, por utilizador ou por grupo. O primeiro método é o de atribuir níveis de privilégio a comandos e ter o router a confirmar com o servidor TACACS+ se o utilizador está autorizado a um específico nível de privilégio. O segundo método é o de indicar explicitamente no servidor TACACS +, por utilizador ou por grupo, os comandos que são permitidos.

3.2.2.6 Interoperabilidade

Devido a várias interpretações do RFC do RADIUS, o cumprimento do RFC RADIUS não garante interoperabilidade. Embora vários fornecedores implementem RADIUS clientes, isto não significa que sejam interoperáveis. A Cisco implementa mais atributos RADIUS e acrescenta mais de forma consistente. Se os clientes utilizam apenas atributos RADIUS standard nos seus servidores, podem interoperar entre vários fornecedores, desde que esses vendedores apliquem os mesmos atributos. No entanto, muitos vendedores implementam extensões que são atributos proprietários. Se um cliente usa um desses atributos estendidos que são específicas específicas de vendedores, a interoperabilidade não é possível.

3.2.2.7 Tráfego

Devido às diferenças referidas anteriormente entre TACACS+ e RADIUS, a quantidade de tráfego gerado entre o cliente e o servidor difere.

O que é dito acerca das vantagens do TACACS+ está escrito em [\[23\]](#).

3.2.3 Vantagens do Diameter em relação ao RADIUS

3.2.3.1 Fail-Over

Fail-Over é definido como sendo o processo de reencaminhar todos os pedidos pendentes para um agente diferente, quando existe uma falha com o primeiro. O RADIUS não tem um mecanismo de Fail-Over standard, pelo que as implementações podem diferir em termos de comportamento. No caso do Diameter é diferente, tem já definido nas suas especificações os métodos que tornam possível o Fail-Over.

3.2.3.2 Mensagens iniciadas pelo servidor

No RADIUS, a troca de mensagens iniciada pelo servidor é opcional, pelo que normalmente são iniciadas pelo cliente, tornando-se complicada a execução de determinadas operações por parte do servidor, como a reautenticação. Por outro lado, o Diameter é um protocolo "peer-to-peer" pelo que as mensagens podem ser iniciadas tanto pelo cliente como pelo servidor.

3.2.3.3 Transporte robusto

RADIUS usa UDP que, juntamente com a falta de especificações de retransmissão de pacotes, faz com que a robustez seja um problema, principalmente no que diz respeito à contabilização. Diameter, por outro lado, é transportado sobre TCP ou SCTP, sendo mais fiável e fácil de implementar.

3.2.3.4 Negociação de capacidades

Não só RADIUS não suporta mensagens de erro como também não tem maneira de descobrir se outros elementos que correm o mesmo protocolo suportam os vários atributos pede. Diameter consegue resolver os dois problemas, suportando mensagens de erro e também a funcionalidade de comunicar com outros elementos para descobrir as suas capacidades.

3.2.3.5 Segurança

As especificações base do RADIUS garantem apenas alguma autenticidade e integridade através do campo autenticador nos pacotes de resposta do servidor, e alguma confidencialidade com a ocultação de atributos. O Diameter, pelo contrário, prevê a implementação do IPsec e/ou TLS, para além da ocultação de atributos, no caso de dados sensíveis como palavras-passe.

3.2.3.6 Suporte para Agentes e Proxies

RADIUS não oferece suporte para agentes e proxies de forma clara. Como o comportamento esperado não está definido, este varia de implementação para implementação, podendo causar problemas de interoperabilidade. O conceito de encadeamento de proxies está definido, mas não está

feito de forma que impeça ataques e corrupção dos dados. O Diameter define o papel dos agentes e dos proxies o seu comportamento de forma explícita, dando suporte ao roaming entre domínios, definição de rotas das mensagens e segurança na camada de transmissão.

3.2.3.7 Descoberta e configuração de elementos Diameter

As implementações RADIUS normalmente requerem uma configuração manual do nome ou endereços dos servidores, juntamente com as chaves, ou segredos. Porque tal tarefa é complicada quando se trabalha numa rede de grandes dimensões, muitas vezes os operadores configuram os mesmos segredos, dando origem a falhas de segurança. Diameter procede à descoberta de elementos através do DNS, assim como também é activada a derivação dinamica de chaves de sessão através da segurança na camada de transmissão.

3.2.3.8 Compatibilidade com RADIUS

Apesar do Diameter não partilhar de um formato de mensagens comum com RADIUS, foi feito um grande esforço no sentido de permitir a compatibilidade entre os dois. Mesmo assim, é de esperar que sejam necessárias traduções de um protocolo para o outro através de gateways.

O que é explicado acerca das vantagens do Diameter é dito em [24].

3.3 Resumo e Conclusão

Na tabela 3.1 faz-se um resumo de tudo o que é dito nas análises crítica e comparativa, com o intuito de se chegar a uma conclusão.

3.3.1 Avaliação dos protocolos AAA

Depois desta breve comparação entre protocolos AAA, o Diameter parece ser o mais indicado em termos de segurança, fiabilidade e mobilidade.

Porém, apesar deste ser intitulado de “sucessor do RADIUS”, revela-se muito complexo. Cada uma das suas funcionalidades está descrita num RFC diferente, à excepção da contabilização que vem descrita no RFC do protocolo. Existem poucas implementações deste protocolo, sendo que a mais conhecida, o OpenDiameter, não se apresenta numa forma convidativa à sua instalação. Esta exige pacotes específicos e uma série de procedimentos, um tanto complicados, para que se consiga completá-la. Para além disso, é necessária a instalação de gateways Diameter para que funcione como tradutor do RADIUS, visto que ainda não existem NAS que suportem Diameter.

Tabela 3.1: Tabela de comparação entre protocolos AAA

	RADIUS	TACACS+	Diameter
Transporte	-	+	+
Integridade das mensagens	-	-	+
Autenticidade das mensagens	-	-	+
Privacidade das mensagens	-	+	+
Garantia de interoperabilidade entre implementações	-	+	+
Chaves dinâmicas entre elementos	0	0	+
Fail-Over	-	-	+
Roaming	-	0	+
Suporte por parte dos NAS	+	-	0
Difusão	+	-	-
Segurança do transporte	-	0	+
Simplicidade	+	+	-
Deteção de outros elementos	0	0	+
Separação dos 3 “As”	0	-	+
Troca de mensagens iniciadas pelo servidor	-	0	+
Negociação de Capacidades	-	-	+
Implementação prática	+	-	- / 0

Ou seja, actualmente, se se quiser implementar o Diameter é possível, mas apenas com as funcionalidades do RADIUS, fazendo com que os custos aumentem sem que haja benefícios.

No caso do TACACS+, consegue-se perceber que existe uma evolução em termos de privacidade dos dados trocados e de transporte, em relação ao RADIUS. No entanto, pergunta-se se tal será suficiente para suportar todo um processo de autenticação, autorização e contabilização. Não suporta mobilidade IP, nem atribuição de chaves dinâmicas de elementos da rede com o mesmo protocolo, detecção de outros elementos na rede e ainda a encriptação dos dados não é muito eficiente. Resumindo, este protocolo precisa de ser melhorado. Para além de tudo isto, o TACACS+ não é, de forma alguma, o protocolo mais implementado. Tudo o que foi afirmado acerca deste protocolo tem por base a comparação com RADIUS, feita pela própria Cisco Systems.

O RADIUS tem um número de implementações quase infindável a seu lado. Contudo, isto pode ser visto como algo negativo, pois não se garante a interoperabilidade entre diferentes implementações. Funcionalidades como Mobilidade IP e Fail-Over são possíveis com o RADIUS, no entanto, como nenhum destes assuntos é definido no seu RCF, a sua implementação é feita consoante o gosto de quem a desenvolve, ou nem sequer é feita, daí que nada é garantido. Para além disto, todas as suas características são fracas, quando comparadas com o Diameter.

3.3.2 Conclusão

Hoje em dia, praticamente todos os sistemas de autenticação e autorização baseiam-se no protocolo RADIUS. O protocolo Diameter permite a interoperabilidade entre os dois protocolos, apesar de serem precisos gateways na rede que façam a separação de zonas RADIUS de zonas Diameter, e façam a tradução entre elas. Porém, apesar do Diameter ser o protocolo AAA mais

seguro, ainda não está preparado para uma actual implementação, pois não existem NAS que comuniquem em Diameter e a sua implementação, nos dias de hoje, apenas traz custos sem qualquer benefício. Por tudo isto, o protocolo mais indicado para a segurança no acesso a uma rede é o RADIUS, devendo-se, contudo, ter sempre em vista uma evolução para Diameter.

Capítulo 4

Exposição de protocolos acessórios

Neste capítulo, é dada uma breve explicação sobre protocolos que têm valor para a solução apresentada.

4.1 HTTP

O HTTP (Hypertext Transfer Protocol) é um protocolo da camada de aplicação com a leveza e velocidade necessárias para sistemas de informação distribuídos e colaborativos. HTTP foi usado pela iniciativa World-Wide Web global information desde 1990.

Sistemas de informação práticos exigem mais funcionalidade do que simples recuperação, incluindo pesquisa, actualização front-end, e anotação. HTTP permite um conjunto de métodos que são utilizados para indicar o efeitos de um pedido. Baseia-se na disciplina de referência fornecida pelo Uniform Resource Identifier (URI), como uma localização (URL) ou nome (URN), para indicar o recurso em que um método está a ser aplicado. As mensagens são passadas num formato semelhante ao que é utilizado pelo Internet Mail e o Multipurpose Internet Mail Extensions (MIME).

HTTP também é usado como um protocolo genérico para comunicação entre user agents e proxies / gateways com outros protocolos Internet, tais como SMTP, NNTP, FTP, Gopher, e WAIS, permitindo acesso básico aos recursos disponíveis a partir de diversas aplicações e simplificando a execução de user agents.

4.1.1 Funcionamento global

O protocolo HTTP é baseado num paradigma pedido / resposta. Um cliente estabelece uma ligação com um servidor e envia um pedido ao servidor, sob a forma de um método de pedido, URL, e versão do protocolo, seguido por uma mensagem idêntica ao MIME contendo modificadores do pedido, informações do cliente, e possível conteúdo do corpo da mensagem. O servidor responde com uma linha do estado, incluindo a versão do protocolo da mensagem e um código

de sucesso ou de erro, seguido por uma mensagem idêntica ao MIME contendo informações do servidor, meta-informação da entidade e possível conteúdo do corpo da mensagem.

A maioria da comunicação HTTP é iniciada por um user agent, e consiste num pedido a ser aplicado a um recurso num determinado servidor de origem.

4.1.2 Tipos de mensagens

As mensagens HTTP consistem em pedidos do cliente para o servidor e respostas do servidor para o cliente. Uma mensagem HTTP pode ser um Simple-Request ou um Simple-Response, no caso da versão 0.9 do HTTP, ou então um Full-Request ou Full-Response para o HTTP/1.0.

Full-Request e Full-Response utilizam o formato de mensagem genérico do RFC 822 para a transferência de entidades. Ambas as mensagens podem incluir campos cabeçalho opcional e um corpo de entidade. O corpo de entidade é separado do cabeçalho por uma linha nula.

```
Full-Request = Request-Line;
(General-Header
|Request-Header
|Entity-Header)
CRLF
Entity-Body
```

```
Full-Response = Status-Line
(General-Header
|Response-Header
|Entity-Header)
CRLF
Entity-Body
```

Simple-Request e Simple-Response não permitem a utilização de qualquer informação de cabeçalho e são limitados a um único método pedido (GET).

```
Simple-Request = "GET"SP Request-URI CRLF
```

```
Simple-Response = [Entity-Body]
```

Utilização do formato Simple-Request é desencorajado, pois impede o servidor de identificar o tipo de media da entidade devolvidos.

4.1.2.1 Pedido

Uma mensagem de pedido de um cliente para um servidor inclui, na primeira linha da mensagem, o método a ser aplicado ao recurso, o identificador do recurso, bem como a versão do

protocolo em uso. Para compatibilidade com a versão HTTP/0.9 mais limitada, existem dois formatos válidos para um pedido HTTP: Simple-Request e Full-Request.

Simple-Request = "GET"SP Request-URI CRLF

Full-Request = Request-Line
(General-Header
| Request-Header
| Entity-Header)
CRLF
Entity-Body

Se um servidor recebe um HTTP/1.0 Simple-Request, ele deve responder com um Simple-Response HTTP/0.9. Um cliente HTTP/1.0 capaz de receber um Full-Response nunca deve gerar um Simple-Request.

Método do pedido

O Método na mensagem indica o método a ser realizado sobre o recurso identificado pelo Request-URI. O método é sensível a maiúsculas e minúsculas. Os métodos mais conhecidos são o GET, HEAD e POST.

A lista dos métodos aceitáveis por um determinado recurso pode mudar dinamicamente; o cliente é notificado através do código de retorno da resposta, se um método não é permitido num determinado recurso. Servidores devem retornar o código de estado 501 (Não implementado) se o método não é reconhecido ou não implementado.

4.1.2.2 Resposta

Depois de receber e interpretar uma mensagem de pedido, um servidor responde na forma de uma mensagem de resposta HTTP. A resposta pode ser um Simple-Response ou um Full-Response.

Simple-Response = [Entity-Body]

Full-Response = Status-Line;
(General-Header
| Response-Header
| Entity-Header)
CRLF
Entity-Body

Um Simple-Response, só deve ser enviado em resposta a um Simple-Request HTTP/0.9 ou se o servidor suporta apenas o protocolo mais limitado, HTTP/0.9. Se um cliente envia uma HTTP/1.0 Full-Request e recebe uma resposta que não comece com um Status-Line, que deve assumir que a resposta é um Simple-Response, e analisar em conformidade. Note que o Simple-Response consiste apenas no corpo de entidade e é terminado pelo servidor ao fechar a ligação.

4.1.3 Definição dos métodos

O conjunto de métodos comuns para HTTP/1.0 é definido abaixo.

4.1.3.1 GET

O método GET significa obter qualquer informação (sob a forma de uma entidade) e é identificado pelo URI do pedido. Se o URI do pedido se refere a um processo de produção de dados, são os dados produzidos que serão retornados como a entidade na resposta e não o texto de origem do processo, a menos que o texto seja o produto do processo.

A semântica do método GET muda para um "GET condicional", se o pedido incluir um campo de cabeçalho If-Modified-Since. Um método GET condicional requer que recursos identificados sejam transferidos apenas se tiver sido modificado desde a data indicada pelo cabeçalho If-Modified-Since. O método GET condicional tem o propósito de reduzir o uso de rede, permitindo que entidades de cache sejam actualizadas sem exigir múltiplos pedidos ou transferências de dados desnecessárias.

4.1.3.2 HEAD

HEAD é idêntico ao método GET exceptuando o facto de que o servidor não deve devolver qualquer Entity-Body na resposta. A meta-informação contida nos cabeçalhos HTTP em resposta a um pedido HEAD deve ser idêntica à informação enviada em resposta a um pedido GET. Este método pode ser utilizado para obter meta-informação sobre o recurso identificado pelo URI do pedido sem transferir o Entity-Body em si. Este método é muitas vezes utilizado para ensaios links de validade, a acessibilidade, e as recentes modificações.

Não existe um pedido "HEAD condicional" análogo ao GET condicional. Se um campo de cabeçalho If-Modified-Since é incluído num pedido HEAD, este deve ser ignorado.

4.1.3.3 POST

O método POST é utilizado para solicitar que o servidor de destino aceite a entidade contida no pedido como um novo subordinado do recurso identificado pelo Request-URI no Request-Line. POST é concebido para permitir um método uniforme para cobrir as seguintes funções:

- Anotação dos recursos existentes;

- Colocar uma mensagem num bulletin board, newsgroup, mailing list, ou grupo similar de artigos;
- Disponibilização de um bloco de dados, tais como o resultado ao submeter um formulário, para um processo de manipulação de dados;
- Extendendo uma base de dados através de uma operação de anexação.

A verdadeira função exercida pelo método POST é determinada pelo servidor, e é geralmente dependente do Request-URI. A entidade postada é subordinada ao URI da mesma forma que um ficheiro está subordinado a um directório que o contenha, uma notícia está subordinado a um grupo de notícias no qual está postada, ou um registo é subordinado a base de dados.

Um sucesso POST não exige que a entidade seja criada como um recurso no servidor de origem ou tornada acessível para referência futura. Ou seja, a acção realizada pelo método POST pode não resultar num recurso que pode ser identificado por um URI. Neste caso, quer 200 (OK) ou 204 (sem conteúdo) são a resposta adequada, dependendo se a resposta inclui uma entidade que descreve o resultado ou não.

Se um recurso foi criado no servidor de origem, a resposta deve ser 201 (criado) e conter uma entidade (de preferência do tipo "text / html"), que descreva o estado do pedido e se refira ao novo recurso.

Um Content-Length válido é necessário em todos os pedidos HTTP/1.0 POST. Um servidor HTTP/1.0 deve responder com um 400 (mau pedido) mensagem, se não puder determinar o comprimento do conteúdo da mensagem do pedido.

Aplicações não devem guardar informações sobre as respostas a um pedido POST porque a aplicação não tem forma de saber se o servidor retornaria uma resposta equivalente num pedido futuro.

4.1.3.4 HTTPS

O HTTP, foi originalmente usado em claro na Internet. No entanto, o aumento da utilização do HTTP para aplicações sensíveis exigiu medidas de segurança. SSL, e o seu sucessor TLS, foram concebidos para proporcionar segurança orientada a canais. Em termos conceptuais, HTTP / TLS é muito simples. Simplesmente usa-se o HTTP sobre TLS precisamente como se poderia usar HTTP sobre TCP.

4.1.3.5 Início de conexão

O agente que actua como o cliente HTTP deve, também, agir como o cliente TLS. Deve iniciar uma conexão com o servidor na porta apropriada (neste caso, na porta 443) e, em seguida, enviar o TLS ClientHello para iniciar um handshake TLS. Quando o handshake termina, o cliente pode então iniciar a primeira solicitação HTTP. Todos os dados HTTP devem ser enviados como "Application Data" do TLS. O comportamento normal HTTP, inclusive em ligações permanentes, deve ser seguido.

4.1.3.6 Encerramento de conexão

TLS oferece uma facilidade segura de encerramento de conexão. Quando um alerta de encerramento válido é recebido, é garantido que não serão recebidos mais dados nessa ligação. Implementações TLS devem iniciar uma troca de alertas de encerramento antes de fechar uma ligação. A implementação TLS pode, depois de enviar um alerta de encerramento, fechar a conexão sem esperar que o outro peer envie seu alerta de encerramento, gerando um "encerramento incompleto". Note que uma aplicação que o faz pode optar por reutilizar a sessão. Isso só deve acontecer quando a aplicação sabe (normalmente através da detecção de fronteiras de mensagens HTTP) que recebeu todos os dados importantes para o processo que decorre.

Qualquer aplicação que recebe um encerramento da ligação sem antes receber um alerta de encerramento válido (um "encerramento prematuro") não deve reutilizar essa sessão. Note que um encerramento prematuro não põe em causa a segurança dos dados já recebidos, mas apenas indica que os dados posteriores poderão ter sido truncados. Porque o TLS está abstraído das mensagens HTTP, é necessário examinar os dados HTTP em si (especificamente o cabeçalho Content-Length) para determinar se o corte ocorreu dentro de uma mensagem ou entre mensagens.

4.1.3.7 TLS

O TLS foi concebido para correr sobre um transporte fiável e, por isso, está bem adaptado para protocolos de aplicação que corram sobre TCP. O TLS é um protocolo cliente-servidor assimétrico, o que significa que normalmente o servidor se autentica perante o cliente usando o seu certificado, enquanto que o cliente nem sempre tem de apresentar um certificado. Mesmo assim, a autenticação mútua através da apresentação de certificados é suportada pelo TLS. Para além de oferecer segurança à camada de transporte, o TLS trata das suas negociações de chaves de forma protegida e sem necessitar de outro protocolo para o fazer.

Troca de mensagens TLS para geração de chaves

A troca de mensagens é mostrada na figura 4.1 e descrita da seguinte forma:

- O iniciador da sessão, o cliente, envia uma mensagem client hello ao servidor. Esta mensagem inclui uma lista dos tipos de encriptação que o cliente suporta. Um número aleatório é também incluído para evitar ataques de replay.
- Depois de verificar que tipos de encriptação o cliente suporta, o servidor envia um server hello, também contendo a lista dos tipos de encriptação que suporta e um número aleatório. Para além disso, esta mensagem contém um identificador da sessão a ser estabelecida.
- Para se autenticar perante o cliente, o servidor envia o seu certificado numa mensagem server certificate. O nome e a chave pública do servidor estão incluídos no certificado, que é assinado pela Autoridade de Certificados (Certification Authority – CA).
- O cliente usa as chaves públicas contidas no certificado para verificar a sua autenticidade. O cliente extrai a chave pública do servidor para encriptar futuras mensagens enviadas ao servidor.

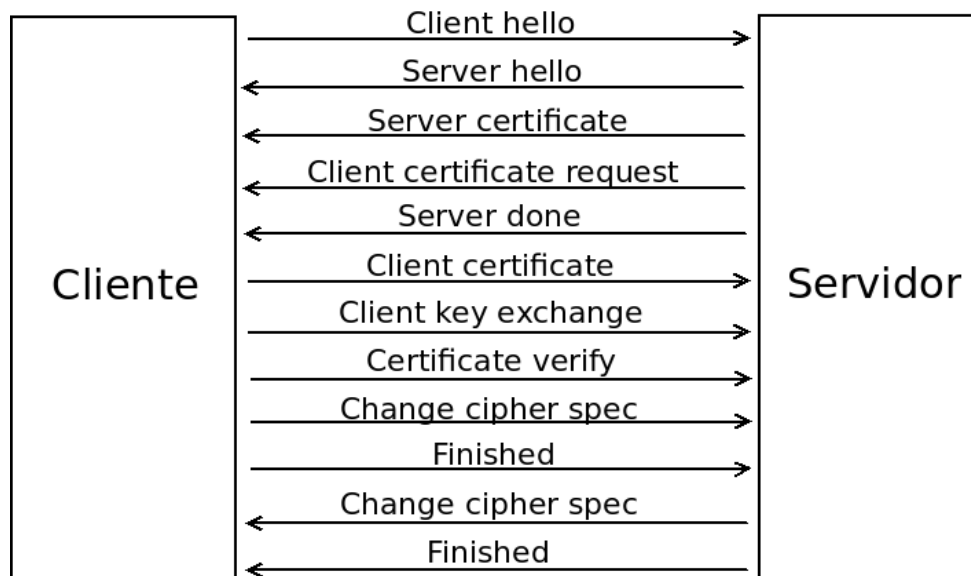


Figura 4.1: Negociação TLS

- Se o servidor necessitar da autenticação do cliente, este fará o pedido do seu certificado, terminado esta sequência de mensagens com uma server done.
- O cliente então gera um número aleatório, chamado pre-master key, e encripta esse número com a chave pública do servidor, enviando-o numa mensagem client key exchange para o servidor.
- Se requerido pelo servidor, o cliente envia o seu certificado usando a mensagem client certificate.
- O servidor descodifica a pre-master key usando a sua chave privada. Se também foi pedida a autenticação do cliente, o servidor verifica a autenticação fornecida pelo cliente.
- Agora que ambos cliente e servidor possuem a pre-master key, eles criam a master key fazendo uma hashing ao número aleatório fornecido pelo cliente, o número aleatório fornecido pelo servidor, e a pre-master key. Neste ponto, a negociação está completa e as chaves foram trocadas.
- O servidor envia uma mensagem change cipher spec para indicar que as seguintes mensagens serão protegidas utilizando as encriptações e chaves que foram negociados. O servidor envia também uma mensagem finished ao cliente.

A master key gerada na negociação TLS é usada para gerar futuras chaves simétricas, oferecendo encriptação e autenticação aos dados trocados a partir desse ponto.

A descrição feita ao HTTP e HTTPS é baseada em [25], [26], [27], [28] e [29].

4.2 SNMP

O SNMP (Simple Network Management Protocol) é um protocolo de gestão de dispositivos de rede. Ele permite a um gestor de uma rede recolher informações dos dispositivos que a compõem, dando a possibilidade de resolver problemas que apareçam na rede, assim como melhorá-la.

4.2.1 Arquitectura

No SNMP existem duas entidades que permitem a sua operação.

4.2.1.1 Estação de gestão

Uma estação de gestão corre uma aplicação de gestão que monitoriza e controla os elementos de rede. A aplicação de gestão faz pedidos para ler ou escrever informação de variáveis, contidas na estrutura de informação de gestão dos elementos de rede.

4.2.1.2 Elementos de rede

Os elementos de rede, ou na linguagem SNMP, os dispositivos geridos, correm agentes SNMP, responsáveis por executar as funções de gestão pedidas pelas estações de gestão.

4.2.2 Informação de Gestão

A informação de gestão contida nos dispositivos geridos está organizada numa estrutura de informação de gestão, chamada SMI (Structure of Management Information). A SMI usa a linguagem ASN.1, criando uma árvore de objectos que suportam a informação para o protocolo de gestão, como se mostra na figura 4.2.

4.2.2.1 Management Information Base

A Management Information Base (MIB) especifica que elementos de dados determinado sistema deve ter, assim como os seus significados. A MIB define, também, que tipo de operações se podem realizar sobre os elementos de dados. A figura 4.3 ilustra a MIB.

4.2.2.2 Object Identifier

O Object Identifier, tal como o nome indica, é o identificador de um objecto. O OID pode ser entendido como o caminho na árvore da SMI para encontrar o objecto pretendido. Um OID é representado por números separados por pontos, mas também pode ser aplicado de uma forma mais inteligível para os humanos, traduzindo os números por nomes.

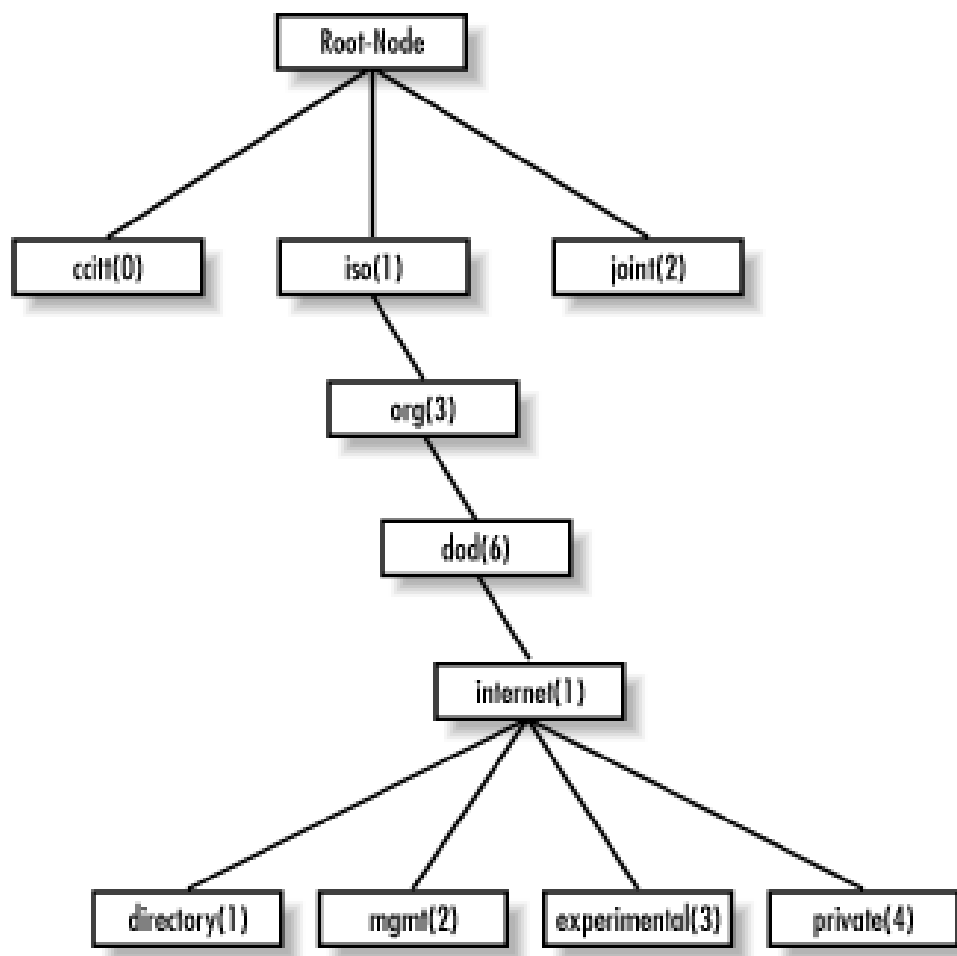


Figura 4.2: Estrutura de informação de gestão (SMI)

4.2.3 Protocolo

4.2.3.1 Mensagens

As mensagens SNMP têm a estrutura apresentada na figura 4.4.

Como se pode ver, uma mensagem é constituída por um campo que informa da versão do SNMP, um outro, comunidade, e uma unidade de dados (Protocol Data Unit - PDU). O campo da comunidade (community string) serve de autenticação, dando permissão de leitura, ou leitura e escrita, conforme o seu valor. A PDU define a operação executada pela mensagem SNMP.

Uma troca de mensagens convencional é mostrada na figura 4.5.

Como se pode ver, em resposta aos pedidos feitos pela estação de gestão, o agente SNMP envia um Get-response. É de notar que as mensagens Trap são enviadas pelo agente sem que qualquer mensagem tenha sido recebida, ao contrário do Get-response, é enviada para notificar de eventos importantes no agente.

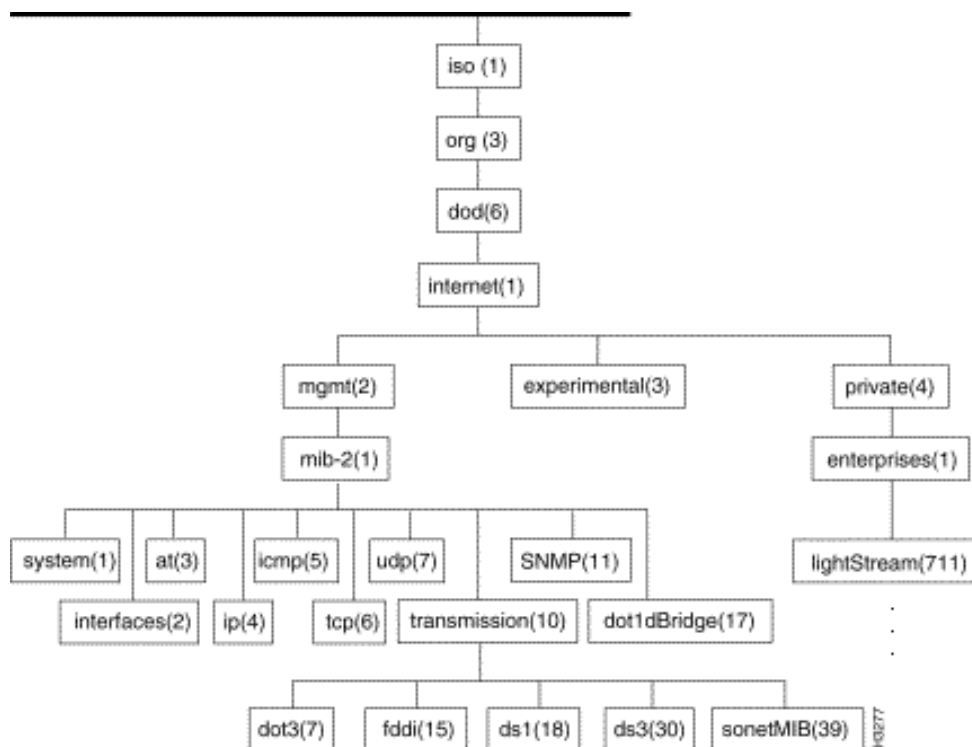


Figura 4.3: Management Information Base (MIB)

4.2.3.2 Operações SNMP

Todas as implementações têm, obrigatoriamente, de suportar as operações da primeira versão do SNMP. Essas operações são explicadas sucintamente.

GetRequest e GetNextRequest

Estas operações são usadas pela estação de gestão para requisitar informação aos elementos de rede.

SetRequest

Esta operação é usado pela estação de gestão para definir o valor de um OID num dispositivo gerido.

GetResponse

Esta operação é utilizada pelo agente SNMP para responder aos pedidos da estação de gestão, GetRequest e GetNextRequest, assim como efectuar a confirmação à operação SetRequest.



Figura 4.4: Estrutura das mensagens SNMP

Trap

Esta operação é enviada pelo dispositivo gerido à estação de gestão, para o informar de eventos importantes.

4.2.4 SNMPv3

A terceira versão do SNMP foi criada para resolver falhas de segurança existentes em versões anteriores.

4.2.4.1 Ameaças à segurança

O SNMPv3 tenta ultrapassar uma série de ameaças à segurança:

- Modificação de informação;
- Dissimulação de um utilizador com permissões;
- Falta de confidencialidade;
- Reordenação maliciosa das mensagens.

4.2.4.2 Medidas de segurança

Com as falhas identificadas, a versão 3 do SNMP tem como medidas de segurança:

- Verificação de integridade dos dados;
- Autenticação;
- Confidencialidade;
- Protecção anti-replay.

4.2.4.3 Modelo de segurança baseado no utilizador

Como forma de oferecer segurança ao SNMP, foi criado um modelo de segurança baseado no utilizador (User-based Security Model - USM). Através deste modelo são definidas regras que permitem que as trocas de mensagens SNMP sejam minimamente seguras, evitando as ameaças referidas em "Ameaças à segurança".

Com este modelo, um utilizador apenas conseguirá ter acesso e modificar objectos num agente SNMP se souber uma palavra-passe. Esta palavra-passe é também a chave que permite a encriptação das mensagens, utilizando o algoritmo MD5 ou SHA. Assim, a segurança é garantida, dado

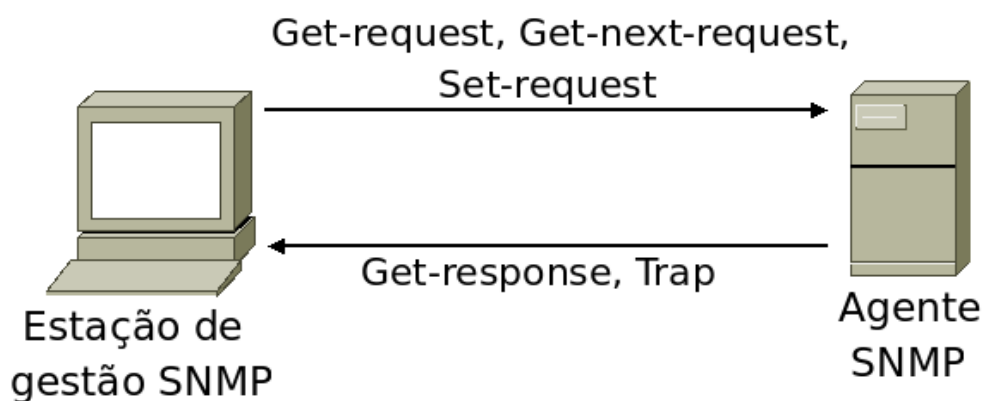


Figura 4.5: Troca de mensagens SNMP

que a informação só é cedida quando o utilizador se autentica com sucesso, para além de que as mensagens são encriptadas. Tal não acontecia em versões anteriores, em que as permissões eram dadas consoante a *community string*, que era passada em claro, com a agravante de não se realizar qualquer tipo de encriptação.

A descrição do SNMP é baseada em [30] e [31].

Capítulo 5

Problema e Solução

5.1 Problema

Analisada a teoria, é necessário perceber as falhas introduzidas pelos sistemas de controlo de acesso actuais. Existem dois tipos de falhas associados a estes sistemas: o acesso à rede, e a segurança da autenticação e do acesso. A primeira situação refere-se à possibilidade de utilizadores ou máquinas se ligarem a uma rede. Se o utilizador ou máquina não suportar os protocolos apresentados, ou se, por outro lado, a rede a que pretende aceder não dispõe dos elementos de rede necessários à implementação de um sistema de controlo de acessos, então existirão problemas de acesso. Não só pelos utilizadores não suportarem os protocolos exigidos, mas também pela rede não estar preparada. Quanto à segurança da autenticação e do acesso, existem também algumas dificuldades, no sentido em que hoje em dia é cada vez mais difícil proteger um sistema de autenticação, pois existe um crescente conhecimento por parte de quem tenta quebrar estes mecanismos, pelo que é imperativo dificultar acesso às credenciais de autenticação. Neste caso fala-se, precisamente, de ataques que têm como objectivo o acesso à rede que, em circunstâncias em que a autenticação é executada normalmente, não é autorizado. Os sistemas de controlo de acesso actuais são vulneráveis a ataques de dicionário, Denial Of Service, Man In The Middle, entre outros.

5.1.1 O problema do acesso

Quando um utilizador, ou máquina, tenta aceder a uma rede através do sistema convencional de controlo de acessos, necessita do software apropriado, neste caso o software do Suplicante 802.1X, para o fazer. Quando esse software não está instalado na máquina, ou não é suportado, a autenticação válida, e consequente autorização de acesso à rede, não é possível. Para além disso, a máquina pode ter algumas restrições nas funcionalidades apresentadas, isto é, a máquina pode estar restrita a determinado software ou ao nível de desempenho exigido por este. A rede a que uma máquina tenta aceder pode, também, não ser a mais apropriada para a implementação de um sistema convencional de controlo de acessos. Nem sempre uma rede tem os elementos de rede mais apropriados para implementar um sistema de controlo de acessos. A entidade responsável pela

manutenção da rede pode não ter poder económico suficiente para renovar os seus equipamentos, de modo a que se torne possível o controlo de acessos fiável.

5.1.1.1 Acesso de computadores pessoais

Quando se pensa no acesso de computadores pessoais, a primeira intuição pode ser a de que qualquer software pode ser instalado e que, por isso, o problema estaria automaticamente resolvido. Isto pode não ser possível, dado que nem todo o software é compatível, e nem todas as máquinas suportam qualquer software. Disso é exemplo o caso de estudo da Universidade da Florida, em que foi necessário uniformizar o software do Suplicante para que tudo funcionasse, como se pode constatar em [32]. Para além disso, para utilizadores menos experientes, pode ser complicada a instalação e configuração do software Suplicante, pelo que convém ser avaliado o acesso para esses utilizadores.

5.1.1.2 Acesso de outras máquinas

A maioria dos acessos de máquinas são, de certeza, de computadores pessoais. No entanto, não só os computadores pessoais necessitam que seja controlado o acesso. Se se pretender que outro tipo de máquina, por exemplo uma impressora, aceda a rede, é necessário avaliar até que ponto é possível instalar o software necessário para implementar a autenticação através do 802.1X. Dado que nem sempre isso é possível, é preciso encontrar alternativas para que o acesso destas máquinas à rede seja controlado. Para além disso, as funcionalidades suportadas pelo dispositivo podem não proporcionar a conveniente autenticação da máquina, pelo que também é necessário encontrar soluções para esses casos.

5.1.1.3 Funcionalidades da rede

Numa rede em que os seus elementos que a suportam são básicos, pode não ser possível implementar de forma imediata um sistema de controlo de acessos. Assim, mesmo que todas máquinas que a pretendem aceder suportem a autenticação através do 802.1X, pode não se conseguir controlar o seu acesso, visto que a rede não está preparada para o efeito. Isto deve-se, muitas vezes, ao facto de a entidade responsável pela manutenção da rede não estar disposta a investir em novo equipamento, para simplesmente suportar autenticação. O resultado disso é a existência de uma rede com várias funcionalidades, mas que não implementa qualquer tipo de controlo de acessos. Existe software que implementa autenticação 802.1X e que poderia ser instalado numa máquina, mas dado que foi concebido para redes sem fios, não consegue resolver tão bem o problema quando se passa para uma rede por cabo.

5.1.2 O problema da segurança

Um sistema de controlo de acessos não se pode limitar simplesmente a controlar os acessos de forma despreocupada, assumindo que todos os seus utilizadores cumprem os protocolos de autenticação e não falsificam a sua identidade, fazendo-se passar por outro utilizador. A forma como os sistemas de controlo de acessos são concebidos actualmente faz com que sejam vulneráveis a vários tipos de ataques, pelo que nem sempre um utilizador autenticado seja de facto quem diz ser. Para além disso, existem falhas relacionadas com o controlo do processo de autenticação. Se um utilizador passar a ter o comando do sistema de autenticação e de controlo de acessos, pode ter acesso a informação delicada, como as credenciais de autenticação, modificar as configurações, ou causar um Denial Of Service.

5.1.2.1 RADIUS

Como é sabido, o protocolo RADIUS apresenta uma série de falhas de segurança. Dessas falhas, podem-se destacar as chaves partilhadas estáticas e configuradas manualmente, a procura das chaves tendo como referência o endereço IP de origem, a cadeia de confiança dos proxies e a falta de protecção dos protocolos abaixo da camada de aplicação. Na verdade, o Diameter foi pensado para substituir o RADIUS por corrigir muitas das suas falhas de segurança, mas, dado que actualmente a sua implementação não traz vantagens em relação ao RADIUS, este não é usado. Assim, todos os defeitos do RADIUS continuam presentes na grande maioria dos sistemas de controlo de acesso.

5.1.2.2 EAP

O Extensible Authentication Protocol foi concebido para ser o suporte de muitos métodos de autenticação, sendo implementado em muitos casos, principalmente na autenticação em redes sem fios. No entanto, este protocolo apresenta uma falha de segurança que pode facilitar o acesso a um utilizador intrusão. Os pacotes EAP não apresentam qualquer tipo de confidencialidade, logo um utilizador que se queira fazer passar por outro pode simplesmente ouvir os pacotes de autenticação de outro utilizador até que este seja devidamente autenticado e autorizado a aceder a rede. Com isto, o utilizador que executa o ataque tem a oportunidade de recolher várias informações, incluindo, muitas vezes, o nome de utilizador da vítima. Para agravar esta falha, muitos métodos de autenticação introduzidos no EAP são vulneráveis a vários ataques. Mesmo métodos bastante seguros, como o EAP TTLSv0, Microsoft PEAPv0 e Cisco PEAPv1, estão sujeitos a ataques Man In The Middle [22].

5.1.3 O problema em estudo

Apesar dos problemas apresentados serem igualmente importantes, este estudo foca o problema do acesso, pois primeiramente é necessário pensar no controlo do acesso, antes de se pensar num acesso seguro. Não faz sentido pensar em segurança no acesso a uma rede se primeiramente

não forem controlados os acessos. Assim, este estudo pretende resolver o problema do acesso de computadores pessoais que não suportem 802.1X, dando-lhes a possibilidade de se autenticarem de forma simples, sem a instalação de novo software. Da mesma forma, tenta resolver o problema do acesso de máquinas, que não os computadores pessoais, de forma fácil e sem instalação de novo software. Também se pretende encontrar uma alternativa para máquinas em que o software que suportam não é suficiente para permitir o controlo de acessos. Finalmente, tem-se igualmente como objectivo o melhoramento do software de autenticação em computadores pessoais por forma a permitir o acesso fiável por cabo.

5.2 Solução

Dados os problemas apresentados é necessário encontrar soluções. Seguidamente, são analisadas e apresentadas soluções para as várias situações referidas na caracterização do problema.

5.2.1 Funcionalidades da rede

Numa rede em que os seus elementos que a suportam são básicos, pode não ser possível implementar de forma imediata um sistema de controlo de acessos. Assim, mesmo que todas máquinas que a pretendem aceder suportem a autenticação através do 802.1X, pode não se conseguir controlar o seu acesso, visto que a rede não está preparada para o efeito. Existe software de autenticador que funciona de forma fiável em redes sem fios, o HostAP. No entanto, este não é tão compatível com a rede por cabo como com a rede sem fios. Como o HostAP não suporta o Port Access Entity, a porta Ethernet fica permanentemente aberta. Assim, este software apenas autentica utilizadores, sem que tenha oportunidade de controlar a porta Ethernet, como é descrito em [33]. Dado que o objectivo é controlar o acesso, algo necessita de ser alterado. É imperativo que o HostAP possa, de facto, controlar a porta Ethernet de forma simples. Ao implementar o HostAP, é necessário criar uma Network Address Translation (NAT) para que este funcione adequadamente. Aproveitando este facto, o código fonte do HostAP pode ser modificado por forma a controlar a porta Ethernet, deixando que apenas o tráfego dos utilizadores autenticados atravessasse a NAT. Supõe-se que HostAP corre numa máquina, na qual foi criada uma NAT em que todo o tráfego interior é bloqueado, ou seja, não é encaminhado. Se, ao autenticar um utilizador, o HostAP adicionar uma regra nas IPtables para passar a encaminhar o tráfego com o endereço MAC de origem da máquina do utilizador que acabou de se autenticar, esta situação fica solucionada. Para esta situação o cenário é o apresentado pela figura 5.1.

Como se constata na figura 5.1, esta solução recorre aos mesmos princípios dos sistemas de controlo de acesso convencionais. O suplicante tenta a autenticação através do Autenticador, o qual, por sua vez, comunica com o Servidor de Autenticação. A única diferença reside no software do HostAP, que é adaptado para endereçar melhor o problema do acesso de uma rede por cabo. O Servidor de Autenticação é o RADIUS, pois, tal como é descrito na análise comparativa de vários protocolos AAA, o RADIUS é, actualmente, o mais adequado em sistemas de autenticação e controlo de acessos.

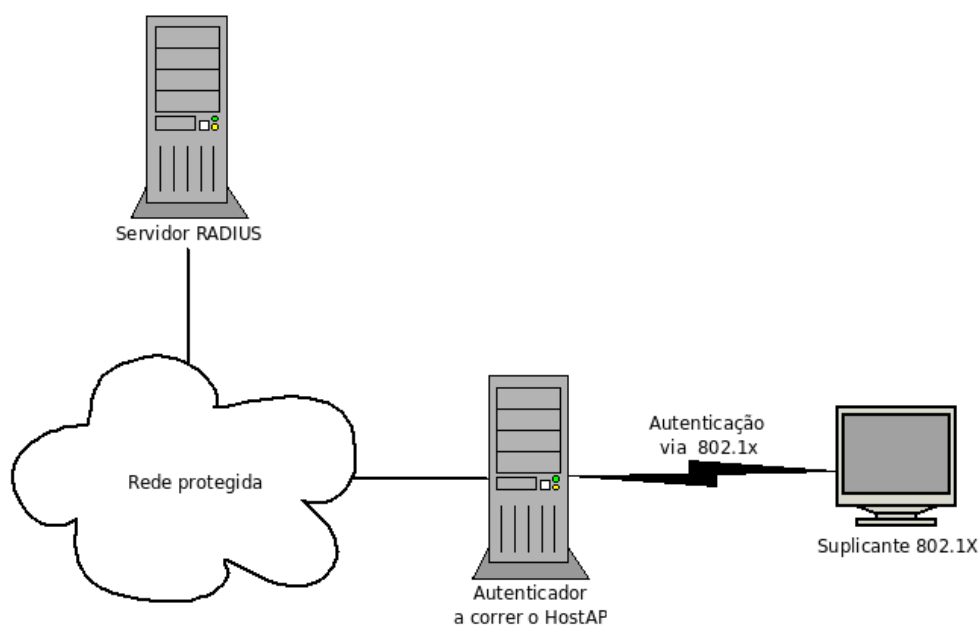


Figura 5.1: Autenticação 802.1X

5.2.2 Acesso de computadores pessoais

O software utilizado para suporte do Suplicante 802.1X nem sempre é compatível entre implementações [32]. Para além disso, utilizadores menos experientes podem ter mais dificuldade em configurar o acesso.

5.2.2.1 Solução aplicada aos computadores pessoais

Dado que o HTTP é um dos protocolos mais universais e versáteis, este é escolhido como ferramenta da solução apresentada. Qualquer computador pessoal, hoje em dia, tem instalado um navegador de Internet, utilizando o protocolo HTTP e suportando, inclusive, o HTTPS. Assim, se o HTTPS for utilizado como meio de autenticação, torna-se numa solução simples e minimamente fiável e segura. Na figura 5.2 é mostrado o diagrama de rede aplicado a esta solução.

Como é ilustrado na figura, os Suplicantes têm acesso a uma rede IP aberta, sem controlo. Nesta rede, também se encontra o servidor de acesso, ou autenticador, o qual tem acesso à rede protegida. É o autenticador que faz a ponte entre a rede aberta e a rede protegida. Este é, também, servidor HTTPS e cliente RADIUS.

5.2.2.2 Processo de autenticação

Quando um utilizador se tenta autenticar, é-lhe fornecido o nome ou endereço IP do autenticador. Ao introduzir o nome no navegador de Internet, o utilizador tem acesso à página de Internet fornecida pelo servidor HTTPS que corre no autenticador. Esta página terá, numa versão simplificada, um campo para nome de utilizador e um outro para a introdução da palavra-passe. Este caso

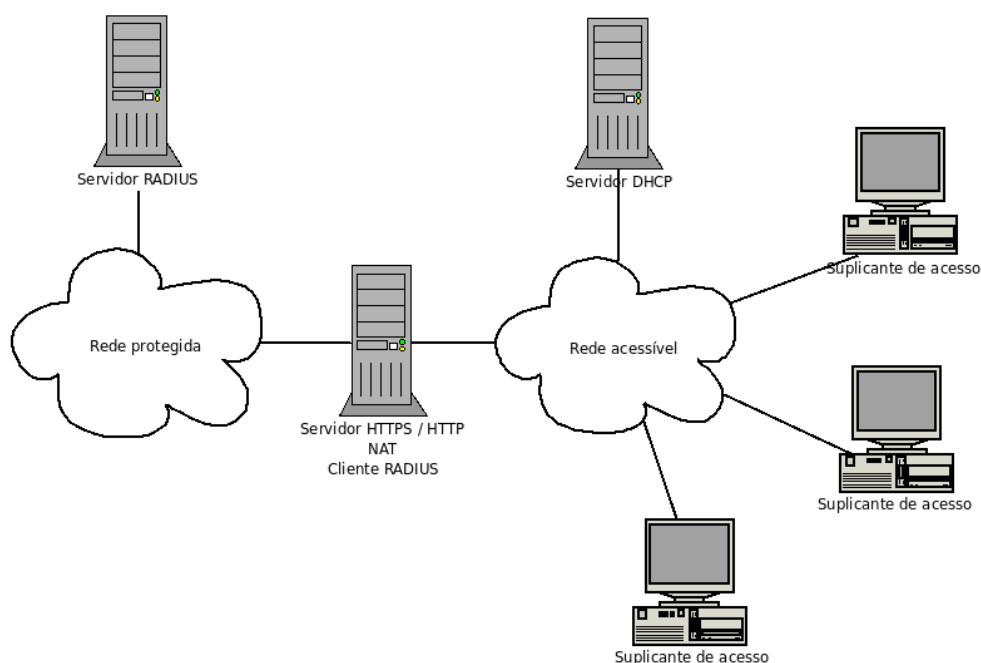


Figura 5.2: Controlo de acesso de PCs

seria aplicado a vários métodos de autenticação. Se o sistema de controlo de acessos implementar o método de autenticação EAP-TLS, pode-se incluir na autenticação o upload do certificado do Suplicante. Assim que o utilizador submete os seus dados, o autenticador inicia uma troca de mensagens com o servidor RADIUS no intuito de autenticar o suplicante. Quando o utilizador é autenticado com sucesso, o autenticador mostra uma página de sucesso ao suplicante e altera as IPTables de forma a permitir o tráfego com endereço MAC de origem do da máquina suplicante. Caso contrário, o autenticador mostra ao suplicante uma página dizendo que a autenticação não foi bem sucedida. Esta solução requer que no autenticador seja instalado um servidor HTTPS e um cliente RADIUS. O uso do protocolo HTTPS tem a finalidade de ser de fácil utilização e ao mesmo tempo seguro, protegendo os dados de autenticação do utilizador. O cliente RADIUS é a ponte entre o servidor HTTPS e o servidor RADIUS da rede protegida.

5.2.2.3 Trabalho anterior

Na verdade, este tipo de solução já existe enquanto processo de autenticação [34], mas usa a autenticação HTTP como método de autenticação. A escolha do HTTPS como suporte à autenticação tem por base a robustez do protocolo, fazendo com que a autenticação se torne mais segura, pois assenta numa protecção TLS. Para além disto, a forma como a solução é implementada permite a utilização de vários métodos de autenticação.

Na solução apresentada, no lado da ligação suplicante-autenticador, os dados estão protegidos por um túnel TLS e, do lado da ligação autenticador-servidor, os dados estão protegidos, não só

pela encriptação MD5 intrínseca à ocultação de atributos do RADIUS, mas também pela encriptação associada a diferentes métodos de autenticação, independentemente de quais sejam. Assim, o método de autenticação aplicado apenas depende do tipo de dados exigidos ao utilizador, na página de Internet que lhe é apresentada no navegador. Por outro lado, a autenticação HTTP que, do lado da ligação suplicante-autenticador, está protegida por uma encriptação MD5 e, do lado da ligação autenticador-servidor, os dados estão protegidos pela ocultação de atributos e pela encriptação do método de autenticação. No entanto, como na autenticação HTTP apenas é pedido ao utilizador o nome de utilizador e a palavra-passe, os métodos de autenticação que podem ser aplicados estão limitados a estas credenciais de autenticação, como é o caso do MD5-Challenge.

Como agravante da falta de flexibilidade e robustez da autenticação HTTP, houve também mais dificuldade na implementação desta do que do HTTPS.

Resumindo, uma autenticação baseada no HTTPS é mais robusta, segura, flexível e fácil de implementar que a autenticação HTTP, sendo estas as razões da escolha feita em termos de protocolo de suporte.

5.2.2.4 Demonstração prática

Para provar o conceito, foram feitos testes práticos. Foram instalados os pacotes `apache2`, `php5` e `openssl` no dispositivo autenticador. Este tem o endereço IP 172.16.2.41 associado à interface `eth0`, ligada à rede protegida e com ligação a Internet, e o endereço 172.16.0.41/24 associado à interface `eth1`, ligada à rede acessível. Foi activado o módulo *SSL* do *Apache*, assim como o sítio relacionado com este. As configurações *IPtables* do dispositivo, neste momento, são as apresentadas no seguinte texto.

```
*filter
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
COMMIT

*mangle
:PREROUTING ACCEPT [164:15203]
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
:POSTROUTING ACCEPT [147:63028]
COMMIT

*nat
:PREROUTING DROP [14:672]
```

```
:POSTROUTING ACCEPT [9:684]
:OUTPUT ACCEPT [9:684]
-A POSTROUTING -o eth0 -j MASQUERADE
-A PREROUTING -d 172.16.0.41 -p icmp -j ACCEPT
-A PREROUTING -d 172.16.0.41 -p TCP --dport 443 -j ACCEPT
-A PREROUTING -s 172.16.0.41 -p TCP --sport 443 -j ACCEPT
COMMIT
```

O dispositivo suplicante tem o endereço IP 172.16.0.43 associado à interface ligada à rede acessível, tendo como *gateway* o dispositivo autenticador. Neste momento, o dispositivo suplicante não está autenticado, pelo que não consegue aceder a Internet, podendo apenas comunicar com o dispositivo autenticador através de pacotes ICMP ou HTTPS (porta 443). Os testes que a seguir são apresentados demonstram o não acesso à Internet e a comunicação com o dispositivo autenticador.

```
PING 172.16.0.41 (172.16.0.41) 56(84) bytes of data.
64 bytes from 172.16.0.41: icmp_seq=1 ttl=64 time=0.205 ms
64 bytes from 172.16.0.41: icmp_seq=2 ttl=64 time=0.159 ms
64 bytes from 172.16.0.41: icmp_seq=3 ttl=64 time=0.219 ms

--- 172.16.0.41 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.159/0.194/0.219/0.028 ms
```

Ao executar o comando "ping www.sapo.pt", não se obtém qualquer resposta, não resolvendo sequer o nome "www.sapo.pt".

Quando um utilizador pretende aceder a rede protegida a partir o dispositivo suplicante, este digita o endereço IP do autenticador no navegador de Internet, não esquecendo que o autenticador apenas aceitará pedidos em HTTPS. Depois disto aparecerá ao utilizador uma página como a apresentada na figura 5.3.

O utilizador, em seguida, insere as suas credenciais de autenticação e o autenticador estabelece uma ligação com o servidor RADIUS, autenticando o utilizador.

Independentemente do método EAP utilizado, o servidor RADIUS terá no seu ficheiro de configuração *users* uma linha com o formato

```
utilizador    Cleartext-Password := "palavra-passe"
```

para definir um utilizador.

Autenticado o utilizador, o autenticador recebe uma autorização do servidor RADIUS, para permitir o acesso do dispositivo suplicante. O autenticador, recebendo a autorização, define uma

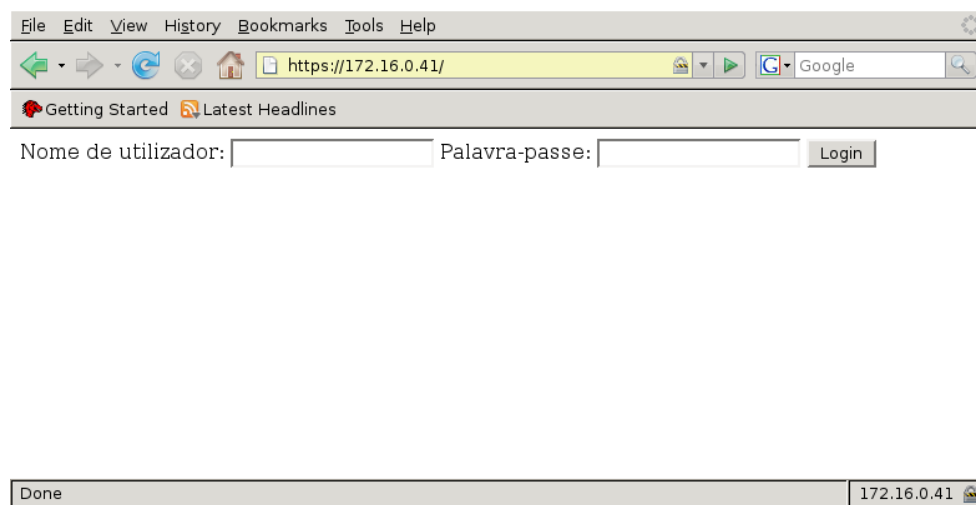


Figura 5.3: Página de autenticação HTTPS

regra na IPtables que permite o tráfego do dispositivo suplicante, como é mostrado no código seguinte.

```
*filter
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
COMMIT

*mangle
:PREROUTING ACCEPT [164:15203]
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
:POSTROUTING ACCEPT [147:63028]
COMMIT

*nat
:PREROUTING DROP [14:672]
:POSTROUTING ACCEPT [9:684]
:OUTPUT ACCEPT [9:684]
-A POSTROUTING -o eth0 -j MASQUERADE
-A PREROUTING -d 172.16.0.41 -p icmp -j ACCEPT
-A PREROUTING -d 172.16.0.41 -p TCP --dport 443 -j ACCEPT
-A PREROUTING -s 172.16.0.41 -p TCP --sport 443 -j ACCEPT
-A PREROUTING -m mac --mac 00:21:5A:C5:60:1E -j ACCEPT
```

COMMIT

Finalmente, o dispositivo suplicante passa a ter permissão de acesso à rede protegida, como se comprova com o teste ping apresentado.

```
PING www.sapo.pt (213.13.146.140) 56(84) bytes of data.  
64 bytes from sapo.pt (213.13.146.140): icmp_seq=1 ttl=115 time=10.9 ms  
64 bytes from sapo.pt (213.13.146.140): icmp_seq=2 ttl=115 time=6.74 ms  
64 bytes from sapo.pt (213.13.146.140): icmp_seq=3 ttl=115 time=6.79 ms  
  
--- www.sapo.pt ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 1999ms  
rtt min/avg/max/mdev = 6.742/8.163/10.954/1.975 ms
```

5.2.2.5 Vantagens e Desvantagens

A vantagem que se destaca é, de facto, a facilidade de utilização por parte do utilizador que pretende aceder a rede protegida. Este não necessita de ter grandes conhecimentos acerca de computadores pessoais, apenas precisa de saber utilizar o navegador de Internet instalado no seu computador, o que, regra geral, é bastante simples e intuitivo. Outra vantagem da solução é que a autenticação é feita de forma bastante segura, dando poucas hipóteses de que alguém descubra as credenciais de autenticação.

A principal desvantagem desta solução é o seu nível de segurança. Na verdade, o processo de autenticação é bastante seguro, visto que corre sobre uma encriptação SSL. No entanto, depois de um utilizador se autenticar, qualquer máquina pode forjar o seu endereço físico e fazer-se passar pela máquina do utilizador que se autenticou. Dado que o acesso, depois da autenticação e dada a autorização, se baseia num controlo de endereços MAC, este pode facilmente ser enganado. Uma outra máquina apenas tem de esperar que um utilizador se autentique para passar a ter acesso também.

5.2.3 Acesso de outras máquinas

A maioria das máquinas presentes numa rede são computadores pessoais. No entanto, nem todas as máquinas pertencem a este grupo. Dispositivos como impressoras, telefones VoIP, câmaras, entre outros, necessitam de ser ligados a uma rede, garantindo o seu acesso legítimo. Estes elementos de rede não suportam HTTP ou HTTPS como clientes, logo é necessário encontrar uma outra solução. Este tipo de dispositivos suporta, normalmente, um protocolo de gestão de redes, o SNMP. Este protocolo pode ser usado como ferramenta de autenticação num sistema de controlo de acessos.

O SNMP é um protocolo bastante comum entre elementos de rede que não os computadores pessoais, pelo que este é escolhido como ferramenta da solução apresentada. Na figura 5.4 é apresentado o diagrama de rede da solução para estes dispositivos.

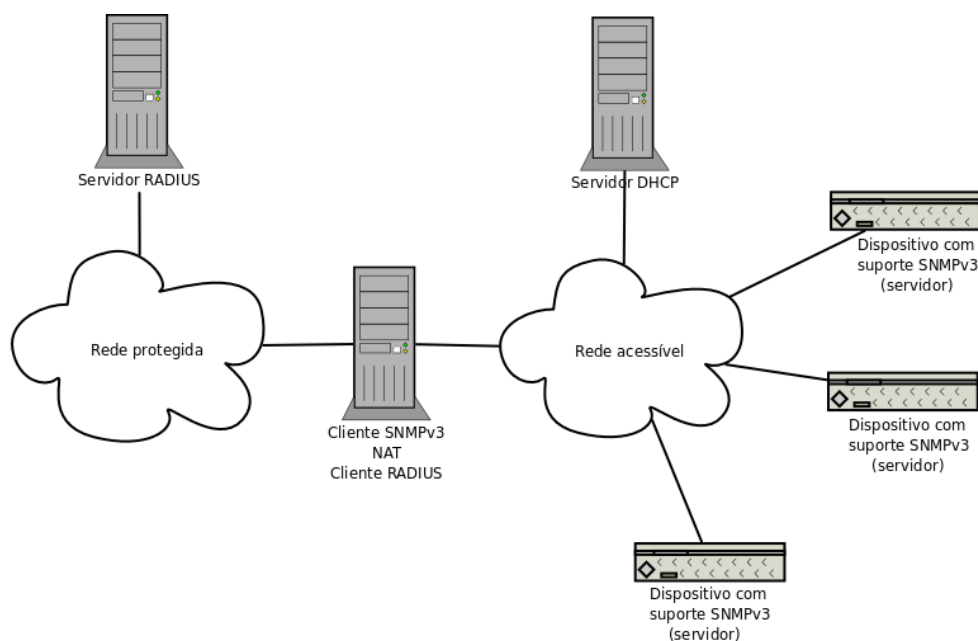


Figura 5.4: Controle de acesso de dispositivos com suporte SNMPv3

5.2.3.1 Escolha da versão do SNMP

O SNMP é um protocolo muito utilizado na gestão de redes. No entanto, como se pode constatar na descrição feita a este protocolo, apenas a versão 3 do SNMP garante a confidencialidade dos dados transferidos ao cliente SNMP. Por este motivo, apenas o SNMPv3 deve ser utilizado com este tipo de autenticação. Caso contrário, qualquer outra máquina que esteja a recolher o tráfego da rede pode replicar uma autenticação que esteja a ser realizada com SNMPv1, ou SNMPv2c. Mesmo quando não existem dados a serem transferidos na rede sobre estas versões do SNMP, basta que um utilizador saiba a community string de uma máquina que se sabe ser legítima de aceder a rede protegida, para automaticamente ser capaz de replicar uma autenticação através das versões 1 e 2c do SNMP. Isto é possível mesmo sem que qualquer autenticação tenha sido previamente executada.

5.2.3.2 Processo de autenticação

Um novo elemento de rede com suporte SNMPv3 tem de configurar um cliente SNMPv3, atribuindo-lhe um nome de utilizador e palavra-passe. Estes dados têm de ser conhecidos apenas pelo autenticador da rede protegida, de modo garantir que só os dispositivos de autenticação têm acesso a dados sensíveis do dispositivo suplicante. A máquina suplicante tem também que criar um OID para o alojamento do nome de utilizador e um outro para alojar a palavra-passe. Assim que a máquina suplicante acede a rede não protegida, o autenticador autentica-se no servidor SNMPv3 do suplicante. Se esta autenticação falhar, nada mais acontece e o suplicante não tem acesso à rede protegida. Se a máquina suplicante autenticar com sucesso o autenticador, o autenticador consulta

os OIDs relativos ao nome de utilizador e à palavra-passe. Desta vez, cabe ao autenticador a função de autenticar o suplicante. Se a autenticação falhar, o suplicante não tem acesso à rede protegida. Se a autenticação for bem sucedida, é dada autorização ao suplicante de aceder a rede protegida, através da definição de uma regra das IPtables que permita o encaminhamento do tráfego com o endereço MAC do dispositivo que acabou de se autenticar.

5.2.3.3 Demonstração prática

Para provar o conceito, foram feitos testes práticos. Foi instalado o pacote *snmp* no dispositivo suplicante, de modo a que se possa utilizar um cliente SNMP. Este tem o endereço IP 172.16.2.41 associado à interface *eth0*, ligada à rede protegida e com ligação a Internet, e o endereço 172.16.0.41 associado à interface *eth1*, ligada à rede acessível. As configurações *IPtables* do dispositivo, neste momento, são as apresentadas no seguinte texto.

```
*filter
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
COMMIT

*mangle
:PREROUTING ACCEPT [164:15203]
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
:POSTROUTING ACCEPT [147:63028]
COMMIT

*nat
:PREROUTING DROP [14:672]
:POSTROUTING ACCEPT [9:684]
:OUTPUT ACCEPT [9:684]
-A POSTROUTING -o eth0 -j MASQUERADE
-A PREROUTING -d 172.16.0.41 -p icmp -j ACCEPT
-A PREROUTING -d 172.16.0.41 -p UDP --sport 161 -j ACCEPT
-A PREROUTING -s 172.16.0.41 -p UDP --dport 161 -j ACCEPT
COMMIT
```

O dispositivo suplicante tem o endereço IP 172.16.0.43 associado à interface ligada à rede acessível, tendo como gateway o dispositivo autenticador. O suplicante implementa um servidor SNMP, definindo o utilizador do autenticador, assim como os OIDs das credenciais de

autenticação. No exemplo apresentado, o utilizador autenticador é "luis", com a palavra-passe "1234567890" e encriptação MD5. As credenciais de autenticação do dispositivo suplicante são "luis" para nome de utilizador e "123456" para a palavra-passe. Foi usado o OID do INESC (1.3.6.1.4.1.668), acrescentando-se números mais ou menos aleatórios na árvore do OID para realizar a demonstração.

```
group qwerty usm luis
access qwerty "" any noauth exact all all none
createUser luis MD5 1234567890

extend .1.3.6.1.4.1.668.987654.1 = "username = luis"
extend .1.3.6.1.4.1.668.987654.2 = "password = 123456"
```

Neste momento, o dispositivo suplicante não está autenticado, pelo que não consegue aceder a Internet, podendo apenas comunicar com o dispositivo autenticador através de pacotes ICMP ou SNMP (porta 161). Os testes que a seguir são apresentados demonstram o não acesso à Internet e a comunicação com o dispositivo autenticador.

```
PING 172.16.0.41 (172.16.0.41) 56(84) bytes of data.
64 bytes from 172.16.0.41: icmp_seq=1 ttl=64 time=0.205 ms
64 bytes from 172.16.0.41: icmp_seq=2 ttl=64 time=0.159 ms
64 bytes from 172.16.0.41: icmp_seq=3 ttl=64 time=0.219 ms

--- 172.16.0.41 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.159/0.194/0.219/0.028 ms
```

Ao executar o comando "ping www.sapo.pt", não se obtém qualquer resposta, não resolvendo sequer o nome "www.sapo.pt".

Quando se pretende que o dispositivo suplicante acesse a rede protegida, o autenticador faz dois pedidos SNMPv3, para os OIDs .1.3.6.1.4.1.668.987654.1 (nome de utilizador) e .1.3.6.1.4.1.668.987654.2 (palavra-passe), recebendo a seguinte resposta:

```
SNMPv2-SMI::enterprises.668.987654.1.2.1.2.1.61 = STRING: "username = luis"
SNMPv2-SMI::enterprises.668.987654.2.2.1.2.1.61 = STRING: "password = 123456"
```

Confirmando que as credenciais de autenticação estão correctas, o autenticador define uma nova regra na *IPtables*, permitindo tráfego do dispositivo suplicante. As *IPtables* terão um formato como o apresentado no texto seguinte.

```
*filter
```

```

:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
COMMIT

*mangle
:PREROUTING ACCEPT [164:15203]
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
:POSTROUTING ACCEPT [147:63028]
COMMIT

*nat
:PREROUTING DROP [14:672]
:POSTROUTING ACCEPT [9:684]
:OUTPUT ACCEPT [9:684]
-A POSTROUTING -o eth0 -j MASQUERADE
-A PREROUTING -d 172.16.0.41 -p icmp -j ACCEPT
-A PREROUTING -d 172.16.0.41 -p UDP --sport 161 -j ACCEPT
-A PREROUTING -s 172.16.0.41 -p UDP --dport 161 -j ACCEPT
-A PREROUTING -m mac --mac 00:21:5A:C5:60:1E -j ACCEPT
COMMIT

```

Finalmente, o dispositivo suplicante passa a ter permissão de acesso à rede protegida, como se comprova com o teste ping apresentado.

```

PING www.sapo.pt (213.13.146.140) 56(84) bytes of data.
64 bytes from sapo.pt (213.13.146.140): icmp_seq=1 ttl=115 time=10.9 ms
64 bytes from sapo.pt (213.13.146.140): icmp_seq=2 ttl=115 time=6.74 ms
64 bytes from sapo.pt (213.13.146.140): icmp_seq=3 ttl=115 time=6.79 ms

--- www.sapo.pt ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 6.742/8.163/10.954/1.975 ms

```

5.2.3.4 Vantagens e Desvantagens

Uma das vantagens desta solução é a fácil configuração do dispositivo que se pretende ligar à rede. Facilmente se configura um acesso e se criam os OIDs para servirem o propósito do controlo de acessos. Também o dispositivo que toma o papel de autenticador é facilmente configurável,

fazendo-o ler as variáveis de autenticação no dispositivo controlado. Assim como na solução aplicada aos computadores pessoais, o processo de autenticação é seguro, protegendo as credenciais de autenticação.

Esta solução, porém, apresenta duas desvantagens. A primeira, é que não são todos os dispositivos que suportam a terceira versão do SNMP. Mesmo os que suportam o SNMPv3, poderão já suportar o 802.1X. Deste modo, está-se a resolver o problema do acesso para uma minoria. A segunda desvantagem tem a ver com a segurança. Tal como acontece na solução aplicada aos computadores pessoais, o processo de autenticação é bastante seguro. Mas, dada a autorização de acesso, este pode facilmente ser forjado, visto que o controlo é feito exclusivamente através do endereço físico das máquinas que se autenticam.

5.2.4 Alternativa

Dado o problema do acesso e esgotadas as alternativas, em termos de protocolos capazes de auxiliar uma autenticação, é necessário encontrar uma solução para controlar o acesso de máquinas que não suportem os protocolos 802.1X, HTTPS ou SNMPv3. Um simples controlo estático de endereços físicos não é suficiente, pois facilmente se muda o endereço físico de uma máquina para que se passe a ter permissão de acesso. De mesma forma, limitar estaticamente um endereço IP associado a um endereço MAC não traz grandes melhorias, visto que o endereço IP é ainda mais passível de ser alterado. Desta forma, a única forma de se controlar o acesso de uma máquina é através do seu tráfego.

Normalmente, máquinas, que não computadores pessoais, sem suporte de protocolos complexos, têm uma função muito específica. Impressoras apenas recebem e geram tráfego associado a impressões, telefones VoIP só trocam tráfego relacionado com chamadas telefónicas, etc. Dada esta realidade, é possível controlar o acesso das máquinas à rede protegida, deixando-as gerar apenas o tipo de tráfego que é esperado destas. Se, para além disto, se controlar o período de tempo em que a máquina está autorizada a aceder a rede protegida, poder-se-á implementar um mecanismo de controlo de acessos com alguma fiabilidade. Assim, a solução passa por um controlo do endereço físico, do endereço IP, do tráfego gerado pela máquina ligada e do período de tempo em que há acesso. Ou seja, o dispositivo que exerce a função de autenticador apenas encaminha tráfego de determinado endereço IP, associado a um endereço físico, e apenas dos serviços esperados para a máquina com o papel de suplicante, num determinado intervalo de tempo.

A figura 5.5 apresenta o diagrama de rede aplicado a esta solução.

5.2.4.1 Demonstração prática

Como forma de demonstrar esta solução, foram definidas regras nas IPtables de modo a, primeiro, não permitir qualquer tráfego, e depois, permitir tráfego de determinado endereço físico associado a um endereço IP e a um serviço.

No início, a IPtables tinha as seguintes regras:

```
*filter
```

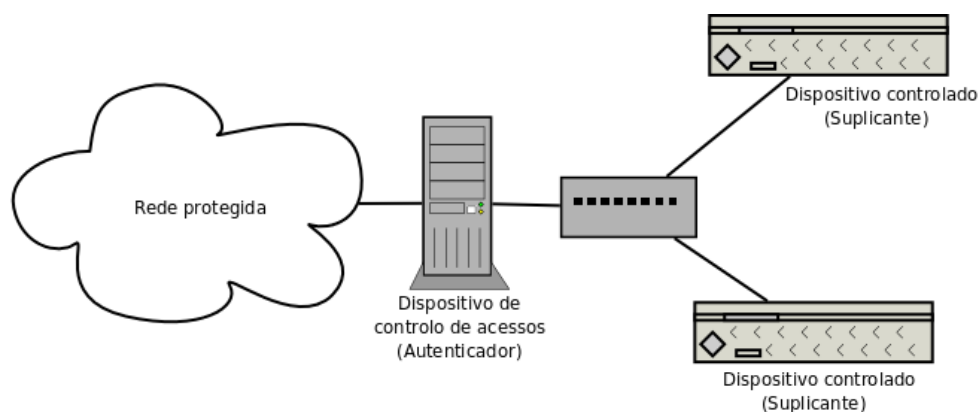


Figura 5.5: Diagrama de rede da solução alternativa

```
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
COMMIT

*mangle
:PREROUTING ACCEPT [164:15203]
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
:POSTROUTING ACCEPT [147:63028]
COMMIT

*nat
:PREROUTING DROP [14:672]
:POSTROUTING ACCEPT [9:684]
:OUTPUT ACCEPT [9:684]
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT
```

Desta forma, o dispositivo que se pretende ligar à rede protegida não consegue comunicar em qualquer tipo de serviço.

Depois foram definidas novas regras de forma a permitir apenas o tráfego do endereço físico do dispositivo suplicante (00:21:5A:C5:60:1E), com determinado endereço IP, neste caso 172.16.0.43 e para um serviço específico. O serviço testado é o protocolo HTTP, pois permitirá a confirmação de que apenas este protocolo é permitido e mais nenhum.

```
*filter
```

```

:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
COMMIT

*mangle
:PREROUTING ACCEPT [164:15203]
:INPUT ACCEPT [164:15203]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [147:63028]
:POSTROUTING ACCEPT [147:63028]
COMMIT

*nat
:PREROUTING DROP [14:672]
:POSTROUTING ACCEPT [9:684]
:OUTPUT ACCEPT [9:684]
-A POSTROUTING -o eth0 -j MASQUERADE
-A PREROUTING -s 172.16.0.43 -p TCP --dport 80 -m mac --mac
\\00:21:5A:C5:60:1E -j ACCEPT
-A PREROUTING -d 172.16.0.43 -p TCP --sport 80 -m mac --mac
\\00:21:5A:C5:60:1E -j ACCEPT
COMMIT

```

No navegador de Internet do dispositivo suplicante tentou-se ver a página *www.google.com*, mas não era possível fazê-lo, como mostra a figura 5.6.

Depois de uma longa espera, o navegador de Internet do suplicante apresenta a página que se mostra na figura 5.7.

Isto acontece porque, visto que o autenticador apenas deixa passar tráfego HTTP associado ao suplicante, não é possível haver troca de pacotes DNS, como se mostra na figura 5.8.

Sabendo que o endereço IP do nome *www.google.com* é 74.125.77.99, este foi inserido no navegador de Internet do suplicante, podendo-se assim ver a página, como pode ser visto na figura 5.9.

5.2.4.2 Vantagens e Desvantagens

A grande vantagem desta solução é o reduzido esforço de configuração do dispositivo que controla os acessos.

Apesar da facilidade de implementação, a solução tem inconvenientes. Um controlo baseado no endereço físico e no endereço IP é bastante vulnerável, pois é extremamente simples de ser ultrapassado. Deste modo, quase que se pode contar apenas com o controlo de tráfego e de tempo

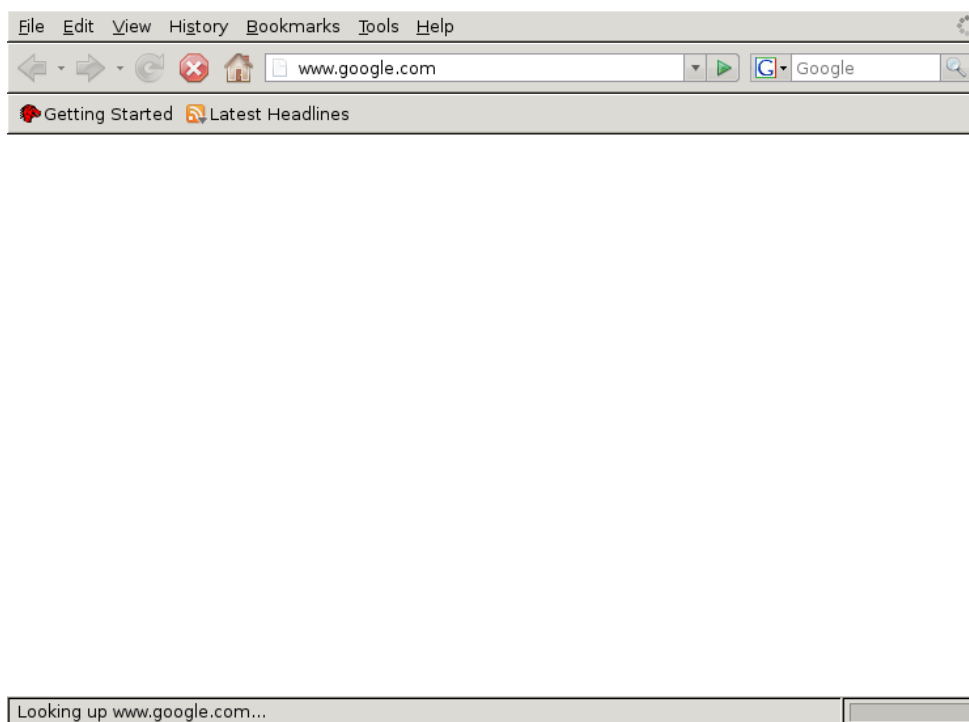


Figura 5.6: Tentativa de acesso à página *www.google.com*

que, por si só, não constitui um controlo de acessos propriamente dito. Mesmo com o controlo de tráfego, este pode ser perigoso, na medida em que alguém pode usar os serviços disponíveis de forma maliciosa. Por exemplo, se se replicar uma impressora, documentos, cujo conteúdo tenha elevado valor, que sejam impressos por um utilizador através dessa impressora impostora serão perdidos para quem os receber, sem que qualquer página seja impressa. Isto significa que é preciso analisar o contexto em que esta solução é aplicada. Apesar desta solução ser tão vulnerável quanto as soluções baseadas em HTTPS e SNMPv3 após a autenticação de uma máquina, esta é bem mais vulnerável pois nem sequer existe um verdadeiro processo de autenticação. As outras soluções dependem de credenciais de autenticação protegidas por encriptação, enquanto que nesta solução apenas é necessário conhecer a máquina autorizada a aceder a rede.

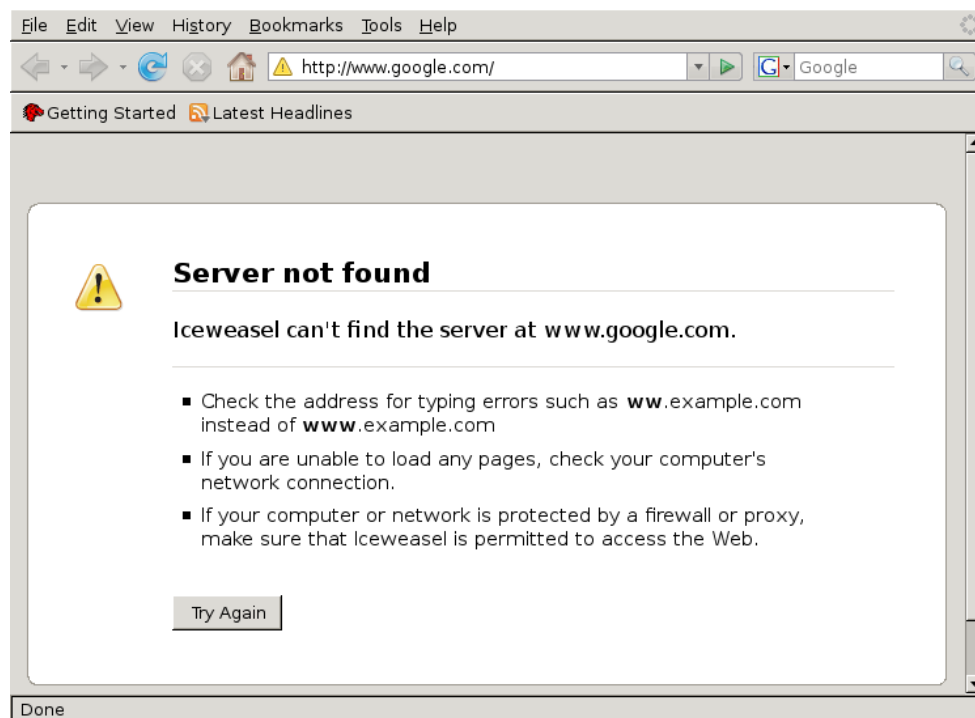
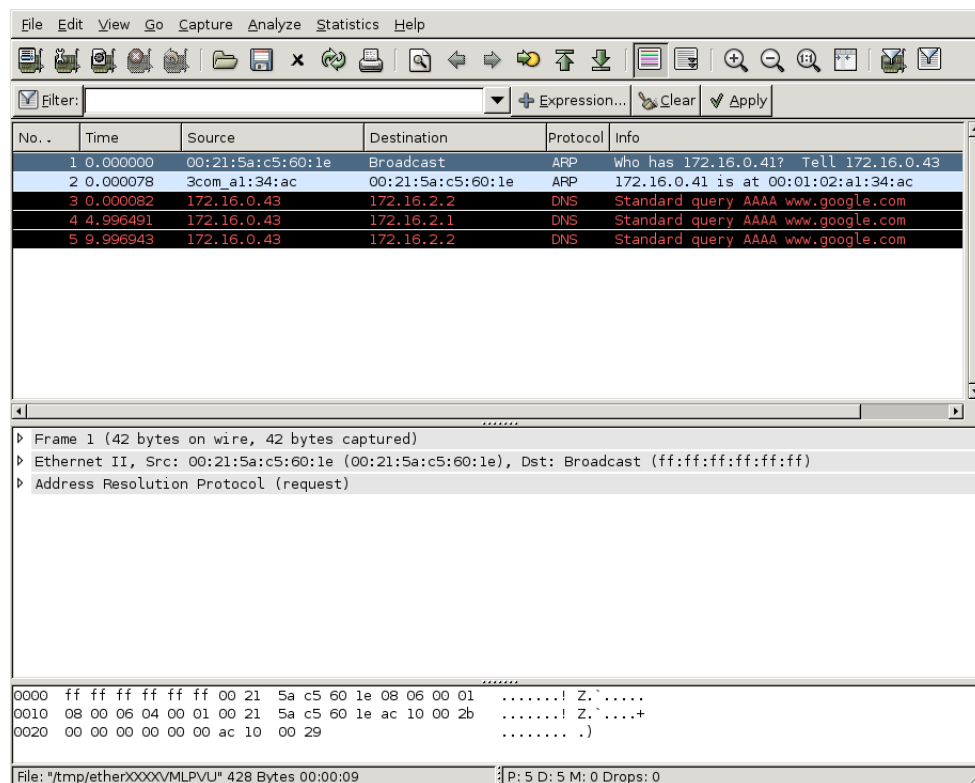
Figura 5.7: Resultado da consulta de *www.google.com*

Figura 5.8: Captura wireshark - sem resposta DNS

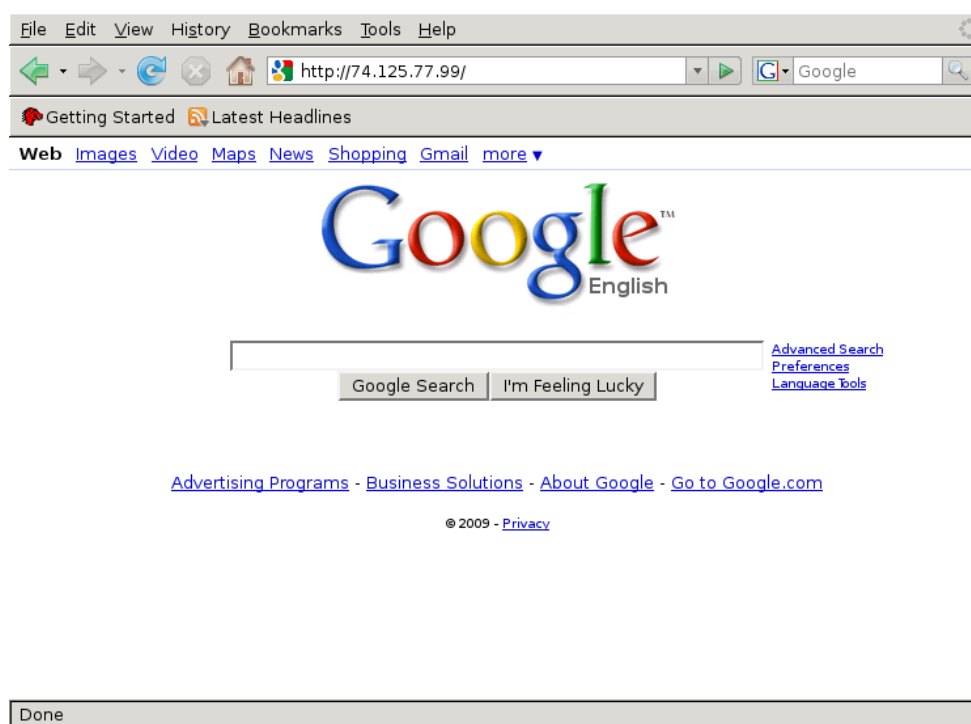


Figura 5.9: Acesso da página www.google.com através do endereço IP

Capítulo 6

Conclusão e Trabalho Futuro

6.1 Conclusão

Este estudo focou, primeiramente, a compreensão dos conceitos relacionados com os sistemas de autenticação e controlo de acessos, descrevendo os procedimentos e protocolos implicados. A partir deste estudo, analisaram-se os protocolos mais relevantes, nomeadamente protocolos AAA, a fim de perceber as suas falhas e ponto fortes, com o objectivo de se saber que protocolos usar. Posteriormente, foram estudados protocolos fora do âmbito convencional do controlo de acesso a redes, com o propósito de os apresentar, com vista à solução apresentada. Depois, foram expostos os problemas relacionados com o controlo de acessos, definindo qual seria analisado e porquê. Definido o objectivo do trabalho, foram apresentadas várias soluções que tentam dar solução a vários cenários, fazendo demonstrações práticas de como essas soluções funcionariam na prática. Finalmente, foram expostas as principais vantagens e desvantagens de cada solução, o que permitiu chegar a uma conclusão: os meios que permitem o controlo de acessos são pouco fiáveis, em termos de segurança, para máquinas que não suportem o protocolo 802.1X, visto que podem ser ultrapassados. No entanto, estes meios são necessários quando se pretende controlar o acesso de máquinas a uma rede.

6.2 Trabalho Futuro

Neste trabalho foram propostas soluções, demonstradas as suas utilidades, mas é necessário dar continuidade ao trabalho realizado. No seguimento deste estudo, é preciso trabalhar nas soluções apresentadas, assim como nos problemas que ainda carecem de solução. Desta forma são apresentadas várias matérias para estudo ou trabalho futuro:

- Modificação e compilação do código fonte do HostAP, fazendo com que este software altere as IPtables quando autentica um suplicante;

- Desenvolvimento de uma aplicação que controle os acessos a uma rede local, Ethernet, através do HTTPS. O autenticador conterá uma página de Internet que o suplicante abrirá, para que, submetendo os dados de autenticação, se autentique e tenha acesso à rede protegida;

- Desenvolvimento de uma aplicação que controle os acessos a uma rede local, Ethernet, através do SNMPv3. O autenticador deverá requisitar as credenciais de autenticação ao suplicante, através do SNMPv3 e, se os objectos requisitados forem válidos numa autenticação, permitirá o acesso do suplicante;

- Estudo de uma solução para as falhas de segurança do protocolo EAP, visto que este protocolo carece de confidencialidade quando aplicado ao 802.1X. Os pacotes de autenticação de um suplicante podem ser capturados por um utilizador mal intencionado, que os pode usar para ganhar acesso à rede;

- Estudo mais aprofundado da migração RADIUS-Diameter que ainda não apresenta vantagens relativamente ao uso exclusivo do RADIUS. Quando se utiliza o Diameter, implementando tradutores para comunicar com os NAS, os custos de implementações são maiores, sem que haja quaisquer benefícios. Deste modo, são necessárias algumas mudanças na implementação do Diameter, para que a transição se torne mais atractiva;

- Aperfeiçoamento das implementações Diameter, pois estas ainda são bastante frágeis e inconsistentes.

Referências

- [1] Jim Geier. *Implementing 802.1X Security Solutions for Wired and Wireless Networks*, chapter 3 - "EAPOL Protocol", pages 55 – 62. WILEY, 2008.
- [2] Jim Geier. *Implementing 802.1X Security Solutions for Wired and Wireless Networks*, chapter 4 - "RADIUS Protocol". WILEY, 2008.
- [3] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 6 - "Remote Access Dial-In User Service (RADIUS)", pages 131 – 134. WILEY, 2005.
- [4] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 6 - "Remote Access Dial-In User Service (RADIUS)", page 135. WILEY, 2005.
- [5] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 6 - "Remote Access Dial-In User Service (RADIUS)", page 139. WILEY, 2005.
- [6] A. Rubens W. Simpson C. Rigney, S. Willens. *Request for Comments: 2865 - Remote Authentication Dial In User Service (RADIUS)*. June 2000.
- [7] C. Rigney. *Request for Comments: 2866 - RADIUS Accounting*. June 2000.
- [8] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 6 - "Remote Access Dial-In User Service (RADIUS)", pages 141 – 143. WILEY, 2005.
- [9] D. Carrel. *The TACACS+ Protocol - Version 1.78*, January 1997. <http://tools.ietf.org/html/draft-grant-tacacs-02>.
- [10] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 7 - "Diameter: Twice the RADIUS?", pages 147 – 168. WILEY, 2005.
- [11] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 7 - "Diameter: Twice the RADIUS?", page 171. WILEY, 2005.
- [12] Jim Geier. *Implementing 802.1X Security Solutions for Wired and Wireless Networks*, chapter 3 - "EAPOL Protocol", pages 63 – 67. WILEY, 2008.
- [13] J. Vollbrecht J. Carlson H. Levkowitz Ed. B. Aboba, L. Blunk. *Request for Comments: 3748 - Extensible Authentication Protocol (EAP)*. June 2004.
- [14] Jim Geier. *Implementing 802.1X Security Solutions for Wired and Wireless Networks*, chapter 5 - "EAP_Methods Protocol". WILEY, 2008.
- [15] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 6 - "Remote Access Dial-In User Service (RADIUS)", page 130. WILEY, 2005.

- [16] A. Rubens W. Simpson C. Rigney, S. Willens. *Request for Comments: 2865 - Remote Authentication Dial In User Service (RADIUS)*, page 1. June 2000.
- [17] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 6 - "Remote Access Dial-In User Service (RADIUS)", page 134. WILEY, 2005.
- [18] *Configuring TACACS+ Authentication*, Maio 2009. <http://www.juniper.net/techpubs/software/junos/junos94/swconfig-system-basics/configuring-tacacs-authentication.html>.
- [19] *Authentication, Authorization, and Accounting Overview*, Março 2009. http://www.cisco.com/en/US/products/ps6638/products_data_sheet09186a00804fe332.html.
- [20] *Diameter Credit Control Application*, Março 2009. https://www.cisco.com/en/US/docs/ios/12_4t/12_4t11/ht_diam.html.
- [21] *Diameter Gateway*, Março 2009. <http://www.traffixsystems.com/site/content/t2.asp?Sid=60&Pid=319>.
- [22] Bernard Aboba. *Wireless LAN Security Site*, Abril 2009. <http://www.drizzle.com/~aboba/IEEE/>.
- [23] *TACACS+ and RADIUS Comparison - Cisco Systems*, Março 2009. http://www.cisco.com/en/US/tech/tk59/technologies_tech_note09186a0080094e99.shtml.
- [24] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 7 - "Diameter: Twice the RADIUS?", pages 168 – 170. WILEY, 2005.
- [25] H. Frystyk T. Berners-Lee, R. Fielding. *RFC 1945 - Hypertext Transfer Protocol – HTTP/1.0*, May 1996.
- [26] J. Mogul H. Frystyk T. Berners-Lee R. Fielding, J. Gettys. *RFC 2068 - Hypertext Transfer Protocol – HTTP/1.1*, January 1997.
- [27] J. Mogul H. Frystyk L. Masinter-P. Leach T. Berners-Lee R. Fielding, J. Gettys. *RFC 2616 - Hypertext Transfer Protocol – HTTP/1.1*, December 2002.
- [28] E. Rescorla. *RFC 2818 - HTTP Over TLS*, May 2000.
- [29] Madjid Nakhiri and Mahsa Nakhiri. *AAA and Network Security for Mobile Access*, chapter 4 - "Internet Security and Key Exchange Basics", pages 91 – 95. WILEY, 2005.
- [30] M. Schoffstall J. Davin J. Case, M. Fedo. *RFC 1157 - A Simple Network Management Protocol (SNMP)*, May 1990.
- [31] B. Wijnen U. Blumenthal. *RFC 3414 - User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*, December 2002.
- [32] Cloudpath Networks Inc. *XpressConnect Allows University of Florida Network Team To Move Forward*, Abril 2009. http://www.cloudpath.net/document/case_studies/univ_of_florida.pdf.
- [33] *Authentication failed, but I still can send packets*, Abril 2009. <http://lists.shmoo.com/pipermail/hostap/2008-November/018851.html>.

- [34] *The Authentication module for Apache*, Maio 2009. http://freeradius.org/mod_auth_radius/.