Segurança em redes IEEE 802.11 utilizando cadeias SPKI

Marcello Milanez, Carlos Maziero, Edgard Jamhour

Programa de Pós Graduação em Informática Aplicada – PPGIA
Pontifícia Universidade Católica do Paraná – PUCPR
80.215-901 – Curitiba – PR, Brasil
{milanez,maziero,jamhour}@ppgia.pucpr.br

***Resumo**: A disseminação das redes sem fio tem impulsionado o desenvolvimento de novas tecnologias e padrões. Entretanto, com esta evolução surgiram problemas em áreas críticas, como segurança. Este trabalho propõe melhorias na segurança em redes sem fio seguindo o padrão IEEE 802.11. Nele é proposta uma forma de integração do modelo de autenticação/autorização não hierárquico SPKI ao ambiente de rede IEEE 802.11, fazendo também uso dos protocolos de segurança EAP e TLS.*

***Abstract**: The recent dissemination of wireless network has stimulated the development of new technologies and standards. However, as a consequence of this evolution, related problems in critical domains, like security, were also created. This work proposes improvements in security and flexibility of wireless networks which implement the IEEE 802.11 standards. It is proposed the integration of the non-hierarchical authentication/authorization SPKI model to the IEEE 802.11 environment, also combined with the integration of the EAP and TLS security protocols.*

1. Introdução

Com sua evolução, as redes sem fio têm possibilitado uma grande melhoria no acesso a informações, principalmente em relação à rapidez e à mobilidade. Entretanto, juntamente com os benefícios vieram alguns problemas, como a falta de segurança. Esse problema ocorre devido principalmente à utilização de ondas de rádio ao invés de cabos, o que facilita o acesso não autorizado às informações em trânsito.

No caso das redes sem fio no padrão IEEE 802.11, foram desenvolvidas alternativas para melhorar a segurança proporcionada pelo protocolo WEP (*Wired Equivalent Privacy*), que se mostrou pouco eficiente [Borisov et al. 2001; Arbaugh et al. 2001; Cam-Winget et al. 2003]. Uma das abordagens utilizadas foi trabalhar com a segurança das camadas superiores (a partir da camada de rede) a fim de complementar a segurança proporcionada pelo WEP. Passaram a ser utilizados protocolos como o EAP (*Extensible Authentication Protocol*) associado ao TLS (*Transport Layer Security*), permitindo a autenticação dos nós através de certificados X.509.

Baseado na infra-estrutura de segurança existente para redes IEEE 802.11 foi desenvolvido este trabalho, visando apresentar uma alternativa segura e que proporcione autenticação confiável utilizando EAP e uma maior flexibilidade no processo de autorização através do uso de certificados SPKI (*Simple Public Key Infrastructure*). Este artigo está assim estruturado: a seção 2 apresenta o padrão IEEE

802.11; a seção 3 apresenta os protocolos EAP e TLS e a segurança atual do padrão IEEE 802.11; a seção 4 apresenta a infra-estrutura SPKI; a seção 5 apresenta a proposta desenvolvida neste trabalho; a seção 6 detalha os principais roteiros de aplicação da proposta; a seção 7 apresenta detalhes da implementação e alguns resultados preliminares; finalmente, a seção 8 relaciona o presente trabalho às principais pesquisas sobre o tema.

2. Redes IEEE 802.11

O padrão de redes sem fio IEEE 802.11 foi definido em 1997, para velocidades entre 1 e 2 Mbps. Em 1999 foi definido o padrão 802.11b, para velocidades até 11 Mbps. Recentemente foram definidos os padrões 802.11a e 802.11g, ambos para velocidades até 54 Mbps. Em paralelo, os principais fabricantes da área formaram a WECA (*Wireless Ethernet Compatibility Alliance*), com o objetivo de certificar a interoperabilidade entre produtos, bem como garantir a compatibilidade dos mesmos com o padrão 802.11b.

O padrão IEEE 802.11 está focado nas duas primeiras camadas do modelo OSI (camadas física e de enlace). Dois tipos de equipamentos são definidos em uma WLAN: *estação* (ou *peer*), geralmente um computador equipado com uma placa de rede wireless, e *ponto de acesso* (*access point* ou *AP*), que atua como uma ponte entre as redes *wireless* e fixa. São definidas duas topologias de comunicação: *ad hoc mode*, onde as estações comunicam-se diretamente entre si, e *infraestructure mode*, onde existe a presença de um AP intermediando as comunicações e fornecendo acesso à rede fixa.

A segurança no padrão IEEE 802.11 está baseada em autenticação e privacidade, podendo operar em dois modos: *Open System* (somente autenticação) e *Shared Key* (autenticação e privacidade). Esta última utiliza o protocolo WEP (*Wired Equivalent Privacy*) como mecanismo de privacidade. A descrição do funcionamento da segurança no ambiente IEEE 802.11 será detalhada na seção 3.

3. Segurança em redes sem fio

Nesta seção serão apresentados alguns protocolos de segurança de redes sem fio. Na seqüência serão descritos aspectos de segurança do padrão IEEE 802.11.

3.1. EAP, TLS e EAP/TLS

O EAP (*Extensible Authentication Protocol*) é um protocolo de autenticação genérico. A definição do protocolo foi feita na RFC 2284 [Blunk e Vollbrecht 1998], com atualizações no *draft* 2284bis [Blunk et al. 2003]. O protocolo EAP foi projetado para conexões discadas PPP, sendo posteriormente adaptado para redes convencionais IEEE 802 e redes sem fio.

No EAP, são definidos dois elementos básicos: *autenticador* e *peer*. Em [Blunk et al. 2003], é definido um novo elemento denominado *servidor de autenticação*, que encapsula o serviço de autenticação. A descrição do *peer* em [Blunk et al. 2003], acrescenta que ele também pode ser um segmento de LAN ponto-a-ponto ou 802.11 a ser autenticado. O funcionamento básico do protocolo EAP é o seguinte:

1. o autenticador envia um pacote *Request* para autenticar o *peer*. O campo “tipo” do pacote indica o que está sendo solicitado (*Identity*, *MD5-challenge*, etc).

2. o *peer* envia um pacote *Response* em resposta a cada *Request*.
3. o autenticador finaliza a fase de autenticação com um pacote *Success* ou *Failure*, indicando sucesso ou falha no processo de autenticação.

O protocolo TLS [RFC 2246, Dierks e Allen 1999] fornece privacidade e integridade de comunicação. Ele é composto por duas camadas: *TLS Record Protocol* e *TLS Handshake Protocol*. O nível mais baixo é representado pelo *TLS Record Protocol*, responsável por cifrar e garantir a integridade das mensagens. O *TLS Handshake Protocol* permite a autenticação mútua e negociação do algoritmo de criptografia e chaves criptográficas antes do protocolo de aplicação transmitir ou receber dados.

O EAP fornece um mecanismo padronizado para suporte a métodos adicionais de autenticação, como *smart cards*, *Kerberos*, *public key*, *one-time passwords*. Entretanto, vários destes métodos permitem somente autenticação do cliente frente ao servidor. Em muitos casos é desejável o suporte à autenticação mútua, bem como a utilização de um mecanismo de estabelecimento de chaves de sessão. Essas necessidades levaram a unir o EAP ao TLS, que provê essas funcionalidades, constituindo assim o protocolo EAP-TLS [Aboba e Simon 1999].

3.2. Segurança no Padrão IEEE 802.11

A segurança definida no padrão IEEE 802.11 envolve autenticação e privacidade na camada de enlace. Há duas formas de autenticação: *Open Syst*em ou *Shared Key*, que podem ser utilizadas tanto para autenticação entre estação e AP quanto para autenticação entre estações em redes *ad-hoc*. A autenticação *Open System*, também chamada de autenticação nula, é a forma de autenticação mais simples, sendo o *default* em redes IEEE 802.11. As estações que solicitarem autenticação com esse mecanismo serão autenticadas, exceto caso uma estação se recuse a autenticar alguma estação em particular. O mecanismo envolve dois passos:

1. A estação solicitante declara sua identidade e solicita autenticação.
2. A estação solicitada informa o resultado da autenticação. Se o resultado da autenticação for "*successful*", as estações estarão mutuamente autenticadas.

A autenticação *Shared Key* envolve estações que compartilhem uma chave secreta. Não há necessidade de transmitir a chave secreta de forma aberta, mas o mecanismo de privacidade WEP é necessário (*Wired Equivalent Privacy*). A chave secreta deve ser entregue às estações participantes através de um canal seguro independente do IEEE 802.11. No modo *Shared Key*, a estação que inicia a autenticação é referenciada como *requester* e a outra é chamada *responder* [ANSI/IEEE 1999]. Esta forma de autenticação envolve quatro passos:

1. *requester* envia uma mensagem {*authentication request*} ao *responder* (AP) solicitando autenticação por *shared key*.
2. *responder* responde com uma mensagem {*authentication response*} contendo um desafio (*challenge*).
3. *requester* cifra o desafio com sua chave WEP e o devolve em uma nova mensagem {*authentication request*}.

4. Se o *responder* decifrar o *authentication request* e obter o desafio original, ele responde com um *authentication response* concedendo acesso ao *requester*.

O protocolo WEP (*Wired Equivalent Privacy*) visa fornecer às redes sem fio privacidade equivalente à das redes com fios [ANSI/IEEE 1999]. A privacidade oferecida pelo WEP se baseia em chaves criptográficas simétricas de 40 bits e um vetor de inicialização público de 24 bits (IV – *Initialization Vector*). Várias falhas de segurança foram recentemente observadas nesse protocolo [Borisov et al. 2001; Arbaugh et al. 2001; Cam-Winget et al. 2003], como por exemplo, repetição do mesmo IV, reuso de chaves e fragilidades no mecanismo de integridade. Para melhorar sua segurança, a IEEE criou uma solução temporária chamada TKIP (*Temporal Key Integrity Protocol*), que acrescenta os seguintes elementos [Cam-Winget et al. 2003]:

- MIC (*Message Integrity Code*) para evitar fraudes.
- Seqüenciamento de pacotes, para evitar ataques de *replay*.
- Construção de chaves criptográficas por pacote, para evitar ataques FMS (*Fluhrer-Mantin-Shamir*) [Fluhrer et al. 2001].

Como solução de longo prazo, a IEEE está desenvolvendo o CCMP (*Counter-Mode-CBC-MAC Protocol*) [Cam-Winget et al 2003], também com o objetivo de resolver as deficiências do WEP. Este novo protocolo utiliza o algoritmo de criptografia AES (*Advanced Encryption System*), incompatível com o WEP.

4. SPKI/SDSI

A proposta SPKI/SDSI surgiu em 1997 como alternativa à complexidade e a falta de flexibilidade da infra-estrutura de chave pública do padrão X.509. O modelo SDSI (*Simple Distributed Security Infrastructure*) [Lampson and Rivest 1996] define um espaço de nomes descentralizado, onde o proprietário de cada chave pública pode criar um espaço de nomes local relativo à sua chave. Esses espaços podem ser conectados de forma flexível a fim de possibilitar a criação de cadeias de autorização. O modelo SPKI (*Simple Public Key Infrastructure*) [Ellison 1999; Ellison et al. 1999] cria uma infra-estrutura de chave pública simples e flexível para especificar autorizações. Seu funcionamento consiste em identificar usuários através de suas chaves públicas e então vincular autorizações às mesmas.

Existem dois tipos de certificados em SPKI/SDSI: *certificados de nomes*, que vinculam chaves a um nome local, e *certificados de autorização*, que conferem autorização a um nome ou uma chave. Um certificado de nome é utilizado para vincular uma chave pública a um nome local pertencente ao espaço de nome local do emissor. Um certificado de nome é uma estrutura representada por (I, N, S, V):

- *Issuer* (I): chave pública do emissor que assina o certificado.
- *Name* (N): identificador que define o nome local.
- *Subject* (S): sujeito (chave ou um nome) que foi transformado no nome local pelo emissor.
- *Validity dates* (V): especificação de validade do certificado na forma [t1, t2], indicando que o certificado é válido em $t1 \leq t \leq t2$.

A função de um certificado de autorização é garantir ou delegar uma autorização específica do emissor para o sujeito. Esse certificado tem a forma (I, S, D, A, V), onde:

- *Issuer* (I): chave pública do emissor que concede a autorização.
- *Subject* (S): nome ou chave pública que representa o beneficiário da autorização.
- *Delegation* (D): bit de delegação. Quando verdadeiro permite ao sujeito delegar a autorização que está recebendo.
- *Authorization* (A): especificação de autorização que define as permissões que estão sendo concedidas no certificado.
- *Validity dates* (V): especificação de validade do certificado na forma [t1, t2].

5. Integração de SPKI ao protocolo EAP-TLS

Uma das abordagens mais difundidas para segurança em IEEE 802.11 é a que utiliza os protocolos EAP e TLS com autenticação por certificados X.509 [Arbaugh et al 2001; Borisov et al 2001]. Entretanto, existem limitações nessa abordagem: falta de flexibilidade pela dependência de autoridades de certificação (*CAs*), impossibilidade de delegação de autoridade e complexidade da administração do controle de acesso.

A proposta deste trabalho visa suprir as limitações acima identificadas. Ela se baseia na utilização de certificados SPKI em substituição aos tradicionais certificados X.509 para autenticação e autorização de usuários. Isto possibilita a descentralização de autoridade através de delegações e proporciona maior facilidade no controle de acesso. As listas de autorizações armazenadas no servidor de autenticação podem ser simplificadas, pois os próprios certificados contêm as informações de autorização concedidas para cada chave. Para a autenticação do servidor, continuam sendo utilizados certificados X.509.

Os elementos abordados na proposta deste trabalho são os seguintes:

- *Authentication Server* (AS): estação conectada à rede fixa, responsável pela autenticação e autorização do *peer* e negociação de parâmetros de segurança.
- *Authenticating peer*, *peer* ou *mobile STA*: estação móvel sem fio utilizada pelo usuário para se comunicar com o *Authenticator* (AP) e através dele acessar o AS para autenticação e autorização.
- *Authenticator* (AP): estação com uma interface IEEE 802.11 para se associar e comunicar com os *peers* e uma interface com a rede fixa para comunicar-se com o AS. O *Authenticator* é um AP que funciona como um elemento intermediário que encaminha as mensagens, possibilitando o acesso do *peer* ao AS e à rede.

O escopo deste trabalho corresponde ao período compreendido desde a associação estabelecida entre as estações IEEE 802.11, conforme descrito em [ANSI/IEEE 1999] até as confirmações de autenticação entre *peer* e AS.

6. Roteiros de aplicação

Para melhor descrever a proposta, foram definidos três roteiros que representam as situações básicas de aplicação da mesma, detalhados na seqüência:

1. O *peer* possui chave autorizada: neste caso, o usuário não possui certificados SPKI que comprovem a sua autorização, mas possui uma das chaves autorizadas pelo próprio AS.
2. O *peer* é capaz de gerar uma cadeia de certificados SPKI: neste caso, o usuário possui os certificados SPKI que comprovam sua autorização e então sua estação monta a cadeia de certificados SPKI e a envia ao AS.
3. O *peer* não possui chave autorizada nem é capaz de gerar uma cadeia SPKI: neste caso, o usuário inicialmente não possui certificados SPKI que comprovem sua autorização nem uma chave autorizada pelo próprio AS.

Roteiro 1: o peer possui uma chave autorizada

Neste roteiro o peer se identifica e solicita ao AS acesso a uma determinada sub-rede. O AS então informa ao *peer* que ele deve demonstrar a posse de uma chave autorizada através de uma cadeia de certificados SPKI, originada a partir de uma das chaves autorizadas pelo servidor e contidas na lista por ele enviada. O *peer* envia uma cadeia SPKI vazia ao servidor, pois possui uma das chaves autorizadas. Para facilitar a compreensão, o roteiro foi dividido em quatro passos:

1. Identificação do usuário e início do TLS.
2. Solicitação de acesso do usuário e envio da lista de chaves e certificado pelo AS.
3. Autenticação do servidor e autenticação/autorização do usuário.
4. Confirmação e finalização da autenticação e autorização.

O detalhamento deste roteiro está descrito a seguir e representado na figura 1.

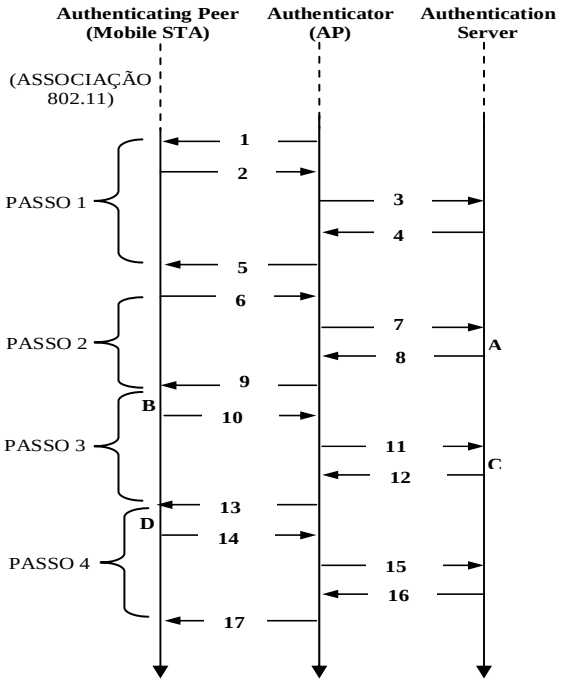

Figura 1: diagrama de tempo do roteiro 1

As mensagens trocadas no roteiro estão detalhadas na figura 2.

1) EAP-Request/Identity
2) EAP-Response/Identity *{chave pública}*
3) EAP-Response/Identity *{chave pública}*
4) EAP-Request/EAP-TLS (TLS Start)
5) EAP-Request/EAP-TLS (TLS Start)
6) EAP-Response/EAP-TLS (TLS client_hello) *{adição da requisição de acesso}*
7) EAP-Response/EAP-TLS (TLS client_hello) *{adição da requisição de acesso}*
8) EAP-Request/EAP-TLS (TLS server_hello, TLS certificate, [TLS server_key_exchange,] [TLS certificate_request,] TLS server_hello_done) *{lista de chaves autorizadas incluída na certificate request + requisição}*
9) EAP-Request/EAP-TLS (TLS server_hello, TLS certificate, [TLS server_key_exchange,] [TLS certificate_request,] TLS server_hello_done) *{lista de chaves autorizadas incluída na certificate request + requisição}*
10) EAP-Response/EAP-TLS (TLS certificate, TLS client_key_exchange, [TLS certificate_verify,] TLS change_cipher_spec, TLS finished) *{substituição do certificado pela SPKI certificate chain - vazia + (requisição, timestamp)Ku}*
11) EAP-Response/EAP-TLS (TLS certificate, TLS client_key_exchange, [TLS certificate_verify,] TLS change_cipher_spec, TLS finished) *{substituição do certificado pela SPKI certificate chain - vazia + (requisição, timestamp)Ku}*
12) EAP-Request/EAP-TLS (TLS change_cipher_spec, TLS finished)
13) EAP-Request/EAP-TLS (TLS change_cipher_spec, TLS finished)
14) EAP-Response/EAP-TLS
15) EAP-Response/EAP-TLS
16) EAP-Success
17) EAP-Success

Figura 2: mensagens do roteiro 1

Após a associação 802.11 entre o *peer* e o AP tem início o EAP-TLS:

Passo 1: Os pacotes de 1 a 3 representam a requisição, envio para o AP e encaminhamento da chave pública do *peer* para o AS. Os pacotes 4 e 5 indicam o envio da mensagem *TLS Start* para o AP e posteriormente para o *peer*.

Passo 2: Após o *peer* receber o pacote *TLS Start* (5):

6. O pacote enviado contém informações para a definição do TLS e a requisição de acesso SPKI, através da qual o *peer* solicita ao AS acesso à sub-rede.
7. As informações contidas em (6) são encaminhadas pelo AP ao AS.

A. Após receber a requisição de acesso, o AS a analisa e, após identificar as chaves públicas autorizadas para o acesso solicitado, gera a lista de chaves.

8. O pacote enviado contém informações para a definição do TLS, a cadeia de certificados X.509 para autenticação do servidor, informações criptográficas opcionais e a solicitação de autenticação do usuário através de certificados. A mensagem original *TLS certificate_request* foi modificada, substituindo a lista de CAs aceitáveis por uma lista de chaves públicas autorizadas. O campo *Requisição* é uma cópia da requisição de acesso enviada pelo *peer* em (6).
9. As informações contidas em (8) são encaminhadas pelo AP para o *peer*.

Passo 3: Após o *peer* receber as informações enviadas pelo AS em (9):

B. O *peer* analisa o certificado TLS enviado pelo AS para autenticar o servidor. Ele analisa a lista de chaves autorizadas recebidas, percebe que possui uma das

chaves autorizadas, gera uma cadeia SPKI vazia e um *timestamp*. Então, o *peer* cria uma seqüência SPKI contendo a requisição e o *timestamp*, assinando-a com sua chave privada e incluindo uma cópia de sua chave pública na assinatura SPKI.

10. O pacote enviado contém o certificado que, neste caso, não é um certificado padrão X.509, mas sim composto pela cadeia SPKI vazia e pela seqüência SPKI (item B), na forma *{[requisição, timestamp] Ku, cadeia SPKI}*, onde *Ku* é a chave do usuário. Acompanhando o certificado estão informações criptográficas para o AS e a resposta de autenticação do *peer* para o servidor.
11. As informações contidas em (10) são encaminhas pelo AP para o AS.

C. O AS verifica o certificado recebido em (11): analisa a validade do *timestamp*, verifica se a requisição recebida é a mesma enviada inicialmente pelo cliente, extrai a chave pública da assinatura, verifica a assinatura na seqüência *[requisição-timestamp]* utilizando a chave pública obtida. Como a cadeia SPKI recebida está vazia, o AS compara a chave pública do usuário com a lista de chaves públicas por ele autorizadas. Em seguida, o AS valida os parâmetros TLS enviados pelo *peer*.

12. Depois de confirmada a autenticação, a autorização da chave do usuário e validados os parâmetros TLS, o AS envia as informações criptográficas e a resposta de autenticação para o *peer*.
13. As informações contidas em (12) são encaminhas pelo AP para o *peer*.

Passo 4: Após o *peer* receber as informações do servidor de autenticação em (13):

D. O *peer* analisa as informações TLS recebidas e finaliza com sucesso a autenticação do servidor.

14. Para confirmar a autenticação do AS e o recebimento de (13), o *peer* envia um pacote *{EAP-Response/EAP-TLS (vazio)}*.
15. As informações contidas em (14) são encaminhas pelo AP para o AS.
16. O AS envia um pacote *{EAP-Success}*, indicando o sucesso da autenticação/autorização. Neste caso, o AS envia para o AP a chave de sessão negociada com o *peer* para que eles possam comunicar-se diretamente.
17. O pacote *{EAP-Success}* é encaminhado pelo AP para o *peer* e deste ponto em diante os dois poderão comunicar-se sem a presença do AS.

Roteiro 2: o *peer* possui uma cadeia de certificados SPKI válida

Neste roteiro, o *peer* comprova sua autorização através da composição e envio de uma cadeia de certificados (não nula) para ser validada pelo AS. Os passos 1, 2 e 4 são idênticos aos do roteiro 1. A diferença ocorre no passo 3, onde a autorização é comprovada através de uma cadeia de certificados SPKI (não nula) enviada ao servidor de autenticação, ao invés de uma chave autorizada. Neste caso, a cadeia SPKI é composta pelo *peer* utilizando os certificados do usuário, de forma que o emissor do primeiro certificado seja uma chave presente na lista fornecida pelo servidor e o beneficiário do último certificado seja a chave pública do usuário do *peer*. Esta cadeia é enviada ao AS que a valida e confirma a autorização.

Roteiro 3: o *peer* não possui chave autorizada nem cadeia SPKI válida

Este roteiro inicia da mesma forma que o primeiro (passos 1 e 2). Entretanto, não há chave autorizada presente nos certificados possuídos pelo usuário. Por isso, o *peer* envia uma cadeia SPKI vazia ao servidor de autenticação, não conseguindo assim comprovar que o seu usuário está autorizado. Isso gera o envio de uma mensagem de alerta pelo servidor de autenticação e o bloqueio de acesso à sub-rede para aquele usuário. Para que o usuário possa ser autorizado pelo servidor, ele deverá obter um certificado de autorização junto a outro *peer* e em seguida re-iniciar o processo de autenticação e autorização. Esse comportamento é detalhado nos sub-roteiros a seguir.

Sub-Roteiro 3.1: Falha na autorização do usuário

Neste sub-roteiro, são abordadas as mensagens e processamentos compreendidos até a confirmação de falha na autorização do usuário (passo 4), indicada pelo servidor de autenticação. Os passos 1 e 2 são idênticos aos descritos no roteiro 1. A diferenciação do passo 3 está na falha de autorização do usuário, pois ele não possui chave nem certificados SPKI que comprovem a sua autorização. O passo 4 se diferencia dos roteiros anteriores nos seguintes pacotes:

16. Para avisar o *peer* sobre o erro, o AS envia um pacote contendo o alerta.
17. As informações contidas em (16) são encaminhadas pelo AP para o *peer*.
18. Como não possui certificados que comprovem sua autorização, o *peer* envia um pacote vazio apenas para confirmar o recebimento da mensagem de alerta.
19. As informações contidas em (18) são encaminhadas pelo AP para o AS.
20. O AS envia um pacote indicando que a conversação foi encerrada e o processo de autenticação/autorização do usuário não foi concluído com sucesso.
21. As informações contidas em (20) são encaminhadas pelo AP para o *peer*.

Neste caso, para que o *peer* possa ser autorizado pelo servidor, ele deverá obter um certificado válido (ou cadeia de certificados), utilizando-se dos roteiros descritos em 3.2, para então reiniciar a troca de informações a partir do passo 1, como descrito anteriormente.

Sub-Roteiro 3.2: Obtenção de certificado de autorização SPKI

Duas alternativas são propostas para a obtenção de certificados de autorização SPKI, após a falha de autorização do usuário (sub-roteiro 3.1):

a) *busca de estações que possuam chave autorizada e poder de delegação*: o *peer* solicitante (estação que solicita o certificado) e o *peer* fornecedor (estação que poderá fornecer o certificado de autorização SPKI). Neste caso, as duas estações (e seus respectivos usuários) são desconhecidas (não haviam estabelecido contato prévio). O solicitante faz uma consulta ao fornecedor sobre a possibilidade de geração de um certificado de autorização SPKI, que contenha uma das chaves autorizadas pelo servidor e a identificação da sub-rede desejada. O solicitante envia as informações *{identificação do usuário, requisição [lista de chaves autorizadas pelo servidor] + identificação da sub-rede}* que deverão

estar contidas no certificado. A partir destas, o fornecedor pode decidir e emitir o certificado de autorização.

b) *comunicação com estação conhecida (interação pessoal prévia)*: Este roteiro usa o mesmo cenário que o anterior. Neste caso, considera-se que houve uma comunicação prévia (fora da rede) entre os usuários das duas estações, o que acaba tornando este roteiro um pouco mais simples que o anterior. A solicitação de informações é realizada pelo *peer* fornecedor (identificado pela comunicação prévia como capaz de gerar o certificado de autorização SPKI) que solicita informações *{identificação do usuário, requisição [lista de chaves autorizadas pelo servidor] + identificação da sub-rede}* para a emissão do certificado. Após recebê-las, o fornecedor emite o certificado e o envia ao solicitante.

7. Implementação

Para validar, mesmo que informalmente, a proposta apresentada neste trabalho, foi construída uma simulação do comportamento de cada um dos elementos de uma rede *wireless* IEEE 802.11 (composta por quatro *peers*, dois APs e um AS), de maneira similar ao que ocorre em uma rede real. A simulação foi construída em ambiente Java e permitiu observar o comportamento dinâmico dos diversos componentes nos roteiros descritos.

A opção pela simulação ao invés de uma validação em ambiente real foi feita por duas razões principais. A primeira foi a não-disponibilidade de uma infra-estrutura *wireless* (computadores portáteis, *access points* e interfaces de rede IEEE 802.11) para realizar o experimento. A segunda razão foi o fato de que, caso a infra-estrutura estivesse disponível, seria necessário realizar alterações de código nos protocolos de comunicação (TLS, por exemplo) para suportar a utilização de certificados SPKI, o que exigiria um esforço significativo. Além disso, geralmente o código-fonte dos diversos componentes relacionados não está disponível para efetuar as alterações necessárias.

Através da definição dos roteiros e da realização das simulações foi possível constatar que a proposta de utilização dos protocolos EAP e TLS juntamente com certificados SPKI é viável e pode ser utilizada em redes IEEE 802.11. Todavia, características temporais ainda devem ser incorporadas ao modelo de simulação para permitir avaliar o desempenho da proposta.

As simulações utilizando o programa desenvolvido em Java apresentaram resultados coerentes com a proposta inicial, e através delas foi possível retratar cada um dos roteiros propostos, inclusive as trocas de mensagens, e comprovar a flexibilidade no processo de autorização, especialmente pela utilização de certificados SPKI delegáveis.

8. Trabalhos Correlatos

Nesta seção são analisados os principais trabalhos publicados na área coberta por este artigo, ressaltando as diferenças entre os trabalhos anteriores e o aqui apresentado. Em [Madhusudhana e Ramachandran 2001] foi proposta uma integração entre certificados SPKI e TLS, sugerindo alterações nas mensagens TLS (*Certificate Request* e *Client Certificate*) e a extensão do campo *ClientCertificateType* para possibilitar a autorização do cliente via certificados SPKI, com a autenticação do servidor continuando a ser realizada através de certificados X.509. O foco desse *draft* é

a adaptação de mensagens TLS para suportar certificados SPKI em substituição a certificados X.509, do lado cliente. Essa proposta difere do proposto neste trabalho pelo fato de ser apenas conceitual (não especifica detalhes para sua implementação em nenhum tipo de ambiente), não abordar roteiros detalhados de funcionamento que retratem a sua aplicabilidade, além de tratar somente de TLS e SPKI, sem abordar a utilização do EAP.

Em [Burnside et al. 2002] foram apresentados os protocolos *device-to-proxy* para dispositivos *wireless* e *proxy-to-proxy* baseado em SPKI/SDSI, criados com o objetivo de permitir acesso de rede seguro a dispositivos móveis, incluindo os que possuem pouca capacidade de processamento (dispositivos leves). Neste artigo não foi considerada a autenticação do servidor e nem a confidencialidade (cifragem) dos dados trocados entre cliente e servidor.

Em [Clarke 2001] foi apresentado o lado servidor de um projeto denominado *Geronimo*, que explora a viabilidade do uso de SPKI/SDSI para fornecer controle de acesso pela Web, além de descrever um algoritmo para a montagem de cadeias de certificados em SPKI/SDSI. Também é sugerida uma possível utilização de SSL/TLS como forma de incrementar a segurança proposta, mas não são fornecidas indicações de como integrar esses diversos elementos.

9. Conclusão

Este artigo apresentou uma proposta para a melhoria da infra-estrutura de segurança em redes IEEE 802.11. Essa proposta está baseada nos protocolos no uso conjunto dos EAP e TLS e na estrutura de certificação não-hierárquica SPKI.

Através da proposta conceitual e sua posterior simulação foi possível constatar que o uso conjunto dos protocolos EAP e TLS é capaz de fornecer um nível de segurança satisfatório para as trocas de mensagens entre *peers*, APs e AS (devido à cifragem), e que a utilização de certificados de autorização SPKI proporciona uma maior descentralização (eliminando a dependência de CAs para a emissão de certificados) e agilidade (possibilitando delegações) no processo de autorização. Todavia, uma simulação mais detalhada deve ser construída, envolvendo a modelagem das características temporais dos protocolos, para permitir o estudo do desempenho da solução proposta.

Como trabalho futuro se considera ainda uma implementação da proposta em ambiente real, utilizando máquinas com interfaces de rede no padrão IEEE 802.11 a fim de analisar detalhadamente o comportamento de cada um dos elementos em relação a processamento e desempenho, além do impacto total das modificações necessárias para que sejam suportados certificados SPKI. Além disso, deverá ser realizada uma análise da viabilidade de implementar a infra-estrutura de segurança proposta em cenários reais, como, por exemplo, aeroportos, empresas, universidades ou outros locais em que haja demanda por redes móveis temporárias que forneçam um bom nível de segurança e um processo flexível de autenticação/autorização.

Referências

Aboba, B. and Simon, D. (1999) *PPP EAP TLS Authentication Protocol*. RFC 2716.

ANSI/IEEE (1999). *LAN MAN Standards Committee of the IEEE Computer Society*. ANSI/IEEE Std 802.11.

Arbaugh, W. A.; Shankar, N. and Wang, J. (2001) *Your 802.11 Network has no Clothes*. 1st IEEE Intl Conference on Wireless LANs and Home Networks.

Blunk, L. and Vollbrecht, J. (1998) *PPP Extensible Authentication Protocol (EAP)*. RFC 2284.

Blunk, L et al. (2003) *Extensible Authentication Protocol (EAP)*. Internet Draft (draft-ietf-eap-rfc2284bis-00).

Borisov, N.; Goldberg, I.; Wagner, D. (2001) *Intercepting Mobile Communications: The Insecurity of 802.11*. 7th Annual International Conference on Mobile Computing and Networking.

Burnside, M.; Clarke, D.; Mills, T.; Maywah, A.; Devadas, S.; Rivest, R. (2002) *Proxy-Based Security Protocols in Networked Mobile Devices*. 17th ACM Symposium on Applied Computing (Security Track), pages 265-272.

Cam-Winget N., Housley R., Wagner D., Walker J. (2003) *Wireless networking security: Security flaws in 802.11 data link protocols*. Communications of the ACM, Volume 46 Issue 5.

Clarke, D. (2001) *SPKI / SDSI HTTP Server / Certificate Chain Discovery in SPKI / SDSI*. Master´s thesis. Massachusetts Institute of Technology.

Dierks, T. and Allen, C. (1999) *The TLS Protocol Version 1.0*. RFC 2246.

Ellison, C. (1999) *SPKI Requirements*. RFC 2692.

Ellison, C. et al. (1999) *SPKI Certificate Theory*. RFC 2693.

Fluhrer, S.; Mantin, I.; Shamir, A. (2001) *Weaknesses in the key schedule algorithm of RC4*. 4th Annual Workshop on Selected Areas of Cryptography.

Lampson, B. and Rivest, R. (1996) *A simple Distributed Security Infrastructure*. http://citeseer.nj.nec.com/rivest96sdsi.html. Presented at CRYPTO'96 Rumpsession.

Madhusudhana, H.; Ramachandran, V. (2001) *SPKI Certificate Integration with Transport Layer Security (TLS) for Client Authentication and Authorization*. Internet Draft (draft-madhu-tls-spki-00).