Extensão de Segurança para o Perfil DPWS

Igor Thiago Marques Mendonça[1], Joni da Silva Fraga[1], Roberto Alexandre Dias[2]

[1]Programa de Pós-Graduação em Engenharia de Automação e Sistemas – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brazil

[2]Curso de Mestrado em Mecatrônica – Instituto Federal de Santa Catarina (IF-SC)

{igor,fraga}@das.ufsc.br, roberto@ifsc.edu.br

Abstract. *Recently Web Services has been the target of standards and specifications to determine its application directly in embedded devices. The DPWS specification is an example of efforts in this direction. This paper presents a proposed extension of a Security Service to the model's functional DPWS. This extension treats the security aspects independently, following the definitions of WS-Security and WS-SecureConversation. This extended model is applied to a prototype system for managing the distribution of electricity in order to evince the efficiency of the propositions and choices.*

Resumo. Recentemente Serviços *Web* vêm sendo alvo de padrões e especificações visando a sua aplicação diretamente sobre dispositivos embarcados. A especificação DPWS é um exemplo de esforço neste sentido. Neste trabalho é apresentada uma proposta de extensão de um Serviço de Segurança ao modelo funcional do DPWS. Esta extensão trata de maneira independente os aspectos de segurança seguindo as definições de *WS-Security* e *WS-SecureConversation*. Este modelo estendido é aplicado em um protótipo de um sistema de gerenciamento da distribuição de energia elétrica no sentido de evidenciar a eficiência das proposições e escolhas.

1. Introdução

Atualmente Serviços *Web* estão sendo objeto de padrões e especificações que visam a implementação dos mesmos diretamente sobre dispositivos embarcados. Estas especificações fornecem as abstrações necessárias para tornar estes dispositivos facilmente integráveis em ambientes de larga escala e heterogêneos. A especificação *Devices Profile for Web Services* (DPWS) [DPWS 2009] é um destes esforços recentes cujo objetivo é descrever um conjunto mínimo de serviços e infraestrutura para levar Serviços *Web* a dispositivos embarcados. Estas especificações, ao incluir nos pequenos dispositivos a arquitetura orientada a serviços (SOA), tornam possível a integração destes diferentes tipos de aplicações sem a necessidade de recorrer, por exemplo, à integração de Serviços *Web* com tecnologias como o SNMP. Com esta nova realidade, vários projetos como SIRENA (www.sirena-itea.org), SODA (www.soda-itea.org) e SOCRADES (www.socrades.eu) têm sido desenvolvidos, levando essa tecnologia recente para segmentos antes refratários ao SOA, como a indústria automotiva, de automação residencial e ao chão de fábrica.

Este artigo descreve nossos esforços no uso do DPWS em dispositivos medidores na coleta do consumo de energia elétrica de residências. O sistema de aquisição resultante

obtém medidas de consumo fazendo uso das próprias linhas de distribuição de energia elétrica, de recursos de redes metropolitanas sem fio e da Internet. O uso destes meios abertos para interações constitui um fator de risco significativo para a segurança das informações e do próprio sistema de medição. Sem mecanismos adequados não é possível garantir as propriedades de segurança das mensagens de medição e de controle dos dispositivos medidores. A concessionária ficaria vulnerável a fraudes e a revelação de dados confidenciais de seus clientes. Poucos trabalhos que propõem o uso da Internet para a aquisição de medidas se preocupam com a segurança.

Diante disto, este trabalho propõe uma extensão à arquitetura do DPWS, na forma de um Serviço de Segurança, que torna disponível mecanismos para a garantia das propriedades de confidencialidade e integridade das informações transmitidas. Este modelo estendido foi usado no sistema de aquisição citado. Um protótipo deste sistema de supervisão e aquisição de consumo de energia elétrica em unidades residenciais foi desenvolvido para avaliar a funcionalidade desta extensão diante de questões como o desempenho e a segurança. Vários testes foram realizados no sentido de comprovar a eficiência das escolhas feitas neste trabalho.

O artigo está dividido em 6 seções. A seção 2 apresenta a especificação DPWS. As extensões propostas para a segurança que formam base das nossas contribuições estão na seção 3. Na seqüência, a seção 4 apresenta a aplicação alvo, detalhes da implementação do modelo de segurança e a avaliação das contribuições propostas. Na seção 5 são recuperados e discutidos os principais trabalhos relacionados. Por fim, na seção 6 são apresentadas as conclusões finais e perspectivas futuras.

2. *Devices Profile for Web Services* (DPWS)

A Arquitetura Orientada a Serviço (SOA) tem ganhado espaço significativo nos últimos anos na integração de aplicações. Muitas áreas têm mostrado aceitação ao uso do alto nível de infraestruturas de comunicação baseada em serviços. Recentemente o desenvolvimento de infraestruturas de serviços próprias para dispositivos embarcados tem sido objeto de grande interesse. Com o avanço da tecnologia e a grande disponibilidade de acesso através de redes (redes sem fio ou a cabo), estes dispositivos podem, por exemplo, através de conceitos SOA, ser facilmente integrados a sistemas de informação de corporações, ou se tornarem visíveis em níveis mais altos de aplicações distribuídas presentes na Internet. Porém, considerando a natureza destes sistemas de larga escala, fica bem claro que os benefícios desta evolução somente poderão ser usufruídos, se os aspectos de segurança forem propriamente endereçados nas infraestruturas de suporte.

Na seqüência, apresentamos as principais características da especificação ou Perfil de Dispositivos para Serviços *Web* (DPWS).

2.1. Modelo DPWS

O DPWS define um conjunto mínimo de funcionalidades que permite dispositivos com limitações de recursos suportarem Serviços *Web*. Este conjunto mínimo, formado basicamente pelo SOAP[1], serviços de descoberta e de notificação de eventos, descrição

[1] Definido pelo consórcio W3C, o SOAP é um protocolo baseado em XML para a troca de mensagens entre clientes e provedores de serviços no ambiente de Serviços *Web*. O SOAP é independente de linguagem, opera com diversos sistemas operacionais e sobre protocolos já consolidados, como o HTTP, o SMTP, o FTP, o RMI/IIOP, etc.

de serviços, deve permitir implementação direta de serviços de aplicação em dispositivos embarcados em geral, sem comprometer a conformidade com a padronização de Serviços *Web*.

O modelo computacional enfatizado na especificação DPWS preconiza que dispositivos podem assumir diferentes papeis: são ou consumidores de serviços (clientes), ou serviços ou, ainda, ambos. No caso de serviços, dois tipos são distinguidos: *serviços de hospedagem* e *serviços hospedados*. A Figura 1 ilustra como os dois tipos de serviços se enquadram no modelo.

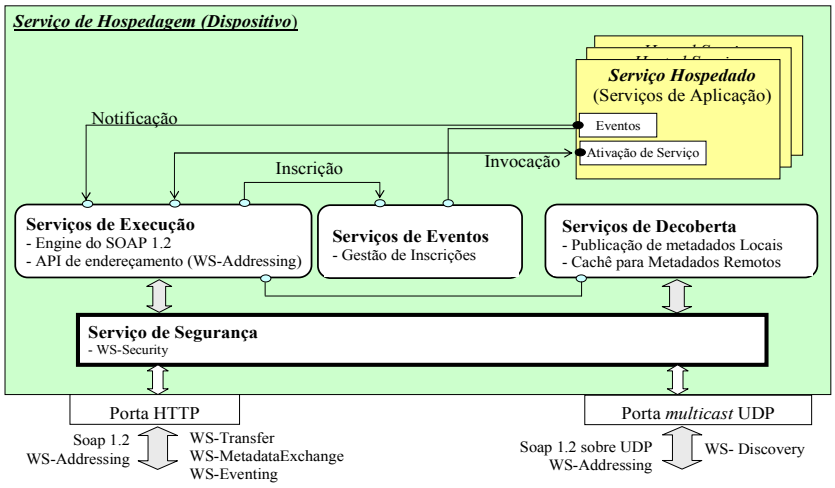

Figura 1. Modelo computacional do DPWS

Os chamados *serviços de hospedagem* (ou *dispositivo*) são parte importante do modelo DPWS. Vários aspectos não funcionais que atuam na evolução de serviços de aplicação estão concentrados neste componente *dispositivo*, na forma de serviços embutidos. Serviços que permitem a descoberta dinâmica e a troca de metadados (interfaces WSDL[2] e seus anexos como o WS-Policy[3], XML Schemas[4], etc.) são exemplos destes serviços. O próprio motor do protocolo SOAP 1.2 é também parte destes serviços do componente de *hospedagem*.

Os *serviços hospedados* são Serviços *Web* específicos das aplicações e fornecem o comportamento funcional das mesmas. Um dispositivo pode possuir diversos serviços hospedados, com cada um destes possuindo um endereço lógico próprio. Estes serviços hospedados dependem do dispositivo e fazem uso de seus serviços embutidos como, por exemplo, o serviço de descoberta [Jammes 2005].

Dos serviços embutidos o Serviço de Descoberta (*WS-Discovery* [OASIS 2009]) é usado por dispositivos para se anunciarem em uma rede e para serem descobertos por clientes. O *WS-Discovery* usa o SOAP sobre a pilha UDP/Multicast IP para difundir e

[2] A WSDL (http://www.w3.org/TR/wsdl), definida pela W3C, é uma gramática XML, extensível, para descrever interfaces de Serviços *Web*. Através da WSDL é possível separar a descrição das funcionalidades oferecidas por um serviço dos detalhes concretos da sua implementação.

[3] A especificação WS-Policy [W3C 2007] provê uma gramática extensível e flexível que possibilita expressar "asserções de política" associadas a um serviço *web*. Estas asserções de políticas estabelecem os tipos de credenciais requeridas pelo serviço, algoritmos de cifragem suportados, etc.

[4] XML *Schemas* são usados para a definição de formatos de dados usados na construção de mensagens enviadas e recebidas pelos serviços.

escutar mensagens de descoberta. Outro destes serviços embutidos, o *WS-Eventing*[5] apresenta operações de *Publicação* e *Inscrição* que permitem a recepção de eventos. A combinação de serviços de aplicação com o serviço embutido de eventos permite clientes se inscrevam para recepção de mensagens assíncronas (eventos) produzidas por serviços hospedados.

Os metadados de serviços hospedados ficam disponíveis para clientes através do uso das especificações do *WS-MetadataExchange* (http://www.w3.org/TR/ws-metadata-exchange) e *WS-Transfer* [W3C 2006a]. O *WS-MetadataExchange* define formatos para encapsular metadados de um Serviço *Web*. A especificação *WS-Transfer* descreve um mecanismo para aquisição de recursos (representações de metadados) usando a infraestrutura de comunicação de Serviços *Web*. Os recursos em aquisição através do *WS-Transfer* possuem suas representações dadas em XML e são endereçáveis através de *endpoints*. Estas trocas de mensagens ocorrem segundo as especificações SOAP 1.2 (http://www.w3.org/TR/soap). As informações de cabeçalho do SOAP seguem o *WS-Addressing* [W3C 2006b], permitindo roteamento em nível de aplicação, além da disponibilidade sobre qualquer protocolo de transporte (HTTP, SMTP, TCP, UDP, etc).

2.2. Segurança no DPWS

Os aspectos de segurança são tratados como opcional nas especificações do DPWS. São feitas recomendações básicas de mecanismos para a segurança nas comunicações envolvendo dispositivos no modelo de Serviços *Web*. Estas recomendações visam essencialmente a garantia de propriedades como a integridade, confidencialidade e autenticação nas comunicações. No perfil, estas recomendações são divididas em duas partes: (i) assinatura opcional em nível de mensagens para o tráfego UDP usado no *WS-Discovery* e (ii) cifragem criptográfica em nível de transporte.

A cifragem em nível de transporte se resume à indicação do uso do TLS/SSL [Dierks 1999]. A outra parte da recomendação tem como alvo o *WS-Discovery* que é peça fundamental para a integração dos componentes nos ambientes de rede. Os protocolos na descoberta dinâmica estão fundamentados na pilha UDP/IP *Multicast* para as comunicações *Multicast*. O uso do TLS/SSL fica impossibilitado na descoberta de serviços. Para contornar o problema, o DPWS em suas recomendações propõe o uso de mecanismos de assinaturas, utilizando certificados x.509.v3, em mensagens no nível de aplicação para garantir a integridade e autenticação das mensagens em *multicast*.

Um dos problemas das recomendações citadas é que as mesmas desconsideram roteamentos de mensagens SOAP, muito comum através de dispositivos, no seu encaminhamento para o receptor final desejado. Este roteamento de mensagens SOAP é concretizado em nível de protocolos de aplicação. Com isto o TLS e seu homônimo SSL não são suficientes para garantir a segurança fim a fim das comunicações. Embora não seja muito enfático nas especificações DPWS, o uso da especificação *WS-Security* [OASIS 2004] se faz necessário para que haja estas garantias fim a fim.

[5] http://www.w3.org/TR/ws-eventing

3. Modelo de Segurança Proposto: a Extensão do DPWS com o *WS-Security*

A *WS-Security*, principal especificação de segurança para Serviços *Web*, apóia-se nos padrões XML-Signature [Bartel et al. 2002] e XML-Encryption[6] [Imamura et al. 2002] para prover trocas de mensagens seguras. A especificação visa ser flexível, sendo possível utilizar uma grande variedade de mecanismos de segurança, provendo suporte para diferentes tipos de credenciais de segurança[7] (*security tokens*), múltiplos formatos para assinatura e várias tecnologias de cifragem de dados. A multiplicidade de opções é importante para alcançar a interoperabilidade entre diferentes tecnologias de segurança.

Na literatura, alguns trabalhos defendem de maneira mais enfática o uso do *WS-Security* no modelo do DPWS [Hernández et al. 2009]. Com isto, assumem então as propriedades de confiabilidade, autenticidade e integridade como garantidas em nível de protocolos de aplicação (portanto, garantias fim a fim). Este é o caminho tomado nas experiências com o DPWS em nosso trabalho.

Como conseqüência, o presente trabalho introduz no modelo do DPWS, o Serviço de Segurança, já ilustrado na Figura 1. Este serviço, por sua vez, trabalha de maneira independente interceptando mensagens de entrada e de saída do dispositivo, atuando no tratamento dos aspectos de segurança nas mensagens, sempre seguindo os padrões especificados pela *WS-Security*. A Figura 2 exibe a arquitetura desse serviço que é descrita na seqüência.

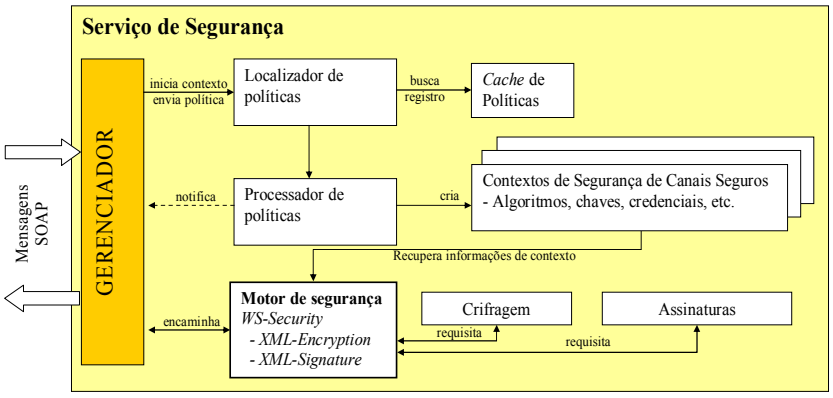

Figura 2. Funcionalidades do Serviço de Segurança

A Tabela 1 descreve os módulos do Serviço de Segurança apresentados na Figura 2 e suas funcionalidades. Dos elementos exibidos na figura, cifragem e assinaturas são implementados conforme a necessidade da aplicação desenvolvida com o dispositivo, os outros são elementos fixos da arquitetura. O motivo para a variação das APIs de cifragem e assinaturas está na diversidade de algoritmos de criptografia existentes. Cada dispositivo poderá ter sua implementação de acordo com sua capacidade computacional e requisitos de segurança.

[6] As recomendações XML-Signature e XMLEncryption permitem expressar assinaturas digitais e cifragem de dados em formato XML, sendo que os dados assinados e/ou cifrados podem ser ou não documentos XML.
[7] Entre os formatos de credenciais de segurança estão o UserNameToken, X.509, Kerberos e SAML.

Tabela 1. Módulos do Serviço de Segurança

Módulos	Função
Cifragem e Assinaturas (*APIs*)	Estes módulos estão disponíveis em forma de bibliotecas e são responsáveis pelas cifragens e decifragens de dados e a assinatura e a verificação das mesmas, respectivamente. Suas implementações variam de acordo com as necessidades e a capacidade do dispositivo.
Contextos de segurança	Para cada canal seguro estabelecido com o dispositivo é criado um contexto de segurança, armazenado na forma de política de segurança (*WS-Policy*).
***Cache* de políticas**	Módulo que armazena as políticas (metadados) de interfaces de serviços que o dispositivo tem conhecimento recente.
Localizador de políticas	Este componente é usado na recuperação e armazenamento de políticas. Identifica e armazena no *cache* local as políticas compartilhadas com seus pares conhecidos.
Processdor de políticas	Realiza comparação entre políticas. No processo de estabelecimento de canal seguro, será criado um contexto de segurança válido quando esta comparação for bem sucedida.
Motor de segurança	Cifra, decifra, assina e verifica assinaturas em mensagens SOAP seguindo o *WS-Security*.
Gerenciador	Elo entre os componentes, responsável por tomadas de decisões e respostas aos pares. É a partir deste módulo que ocorrem as interceptações necessárias para a implantação dos mecanismos e políticas que atuam sobre as mensagens.

Como mostrado na Figura 1, o Serviço de Segurança está posicionado entre a atual arquitetura de DPWS e sua interface de saída. As mensagens de aplicação que chegam ou saem do dispositivo são interceptadas e processadas por este, segundo as especificações *WS-Security*.

O estabelecimento de um contexto de segurança envolve trocas internas entre os módulos do Serviço de Segurança, além de trocas externas que serão explicitadas na seção seguinte.

3.2. Estabelecimento de Canais Seguros

Para se estabelecer canais seguros é necessário o acesso das informações dos pares sobre as alternativas de protocolos suportados para autenticação, confidencialidade e distribuição de chaves nos serviços. Estes informações são anunciadas na forma de asserções de políticas de Qualidade de Proteção (QoP), seguindo a especificação *WS-Policy*. Tais políticas devem estar anexadas às interfaces WSDL correspondentes. No estabelecimento de canais seguros, clientes e dispositivos fazem uso destas políticas para se adequarem mutuamente, permitindo a definição dos contextos de segurança destes canais (escolha de algoritmos, chaves e etc.). O conceito de contextos de segurança para Serviços *Web* é introduzido pela especificação *WS-SecureConversation* [OASIS 2007] que define mecanismos de segurança para uma série de mensagens, diferentemente da *WS-Security* que é focada na segurança de uma mensagem por vez.

No modelo de segurança proposto neste texto para o DPWS, seguimos a especificação *WS-SecureConversation* em alguns pontos no estabelecimento e alteração dos contextos de segurança e de como são geradas e transmitidas chaves derivadas[8] da chave de sessão. Esta especificação define três caminhos para se estabelecer um contexto de segurança. O adotado no nosso caso foi o da solução negociada em que parceiros trocam suas políticas e definem suas chaves e algoritmos também (trocas diretas entre as partes envolvidas). Para a definição da chave criptográfica compartilhada de um canal seguro foi assumido o protocolo *Diffie-Hellman* de distribuição de chaves [RFC 1999]. Esta escolha foi baseada nas características das redes formadas com pequenos

[8] Chaves derivadas: são chaves criptográficas com prazo de validade, derivadas de uma chave compartilhada para evitar a exposição desta última. Estas chaves derivadas, geradas através de um algoritmo, são usadas no lugar da compartilhada.

dispositivos que são de topologia normalmente variável e auto-organizável, muitas vezes sem hierarquias rígidas.

A Figura 3 ilustra o processo de descoberta, a troca de *metadados* e o estabelecimento de um canal seguro através da criação de um contexto entre um cliente e um dispositivo seguindo as recomendações das especificações DPWS [DPWS 2009] e *WS-SecureConversation*. Na figura citada, inicialmente, o cliente envia uma mensagem de descoberta do tipo "*Probe*" ou "*Resolve*" (passo 1). O dispositivo responde com mensagens "*Probe Match*" ou "*Resolve Match*"[9] (passo 2). Na sequência (passos 3 e 4) a requisição e resposta (*Get* e *GetResponse*), através de *WS-MetadataExchange* e *WS-Transfer,* permite a busca de *metadados* do dispositivo. Nos passos 5 e 6 são trocadas mensagens de *metadados* (*WSDL, WS-Policy etc*) dos serviços hospedados.

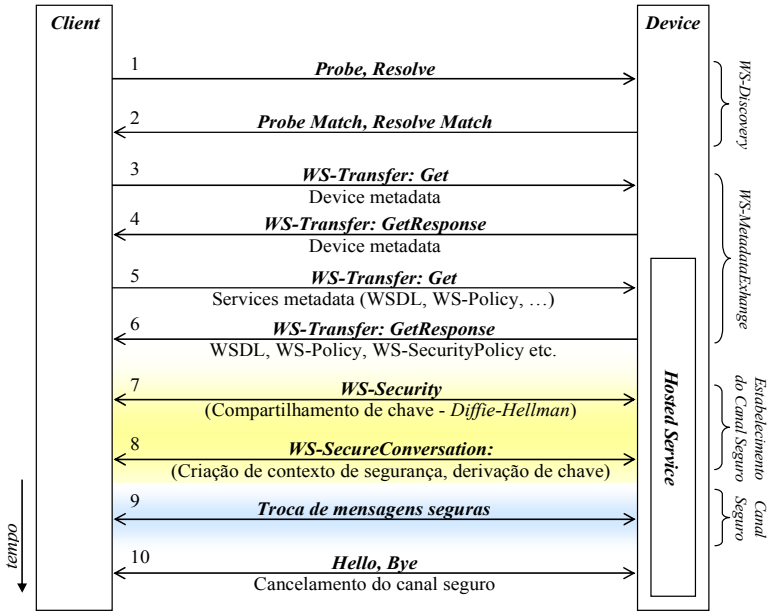

Figura 3. Estabelecimento de canais seguros

Até este ponto, as trocas de mensagens são as previstas pelas especificações DPWS. Para estabelecer os canais seguros da extensão de segurança proposta, é necessário que se definam mais trocas. De posse dos *metadados* de um serviço hospedado, o cliente pode verificar se possui os requisitos necessários (as credenciais) para o estabelecimento de uma conexão segura e/ou autenticada com o mesmo. No passo 7, de posse das credenciais necessárias, o cliente inicia o processo de estabelecimento de uma chave compartilhada com o Serviço Hospedado. Após validar as credenciais do cliente, o Dispositivo inicia as trocas do protocolo *Diffie-Hellman*, criando uma chave de sessão compartilhada com o cliente. O passo 8, sintetiza as trocas usando o *WS-SecureConversation* para a consolidação do contexto de segurança entre os pares, permitindo, entre outras coisas, a geração de chaves derivadas usadas nas cifragens/decifragens do canal. A consolidação de um contexto de segurança é representada pelo envio de um *token* que representa a concretização final do canal seguro

[9] A mensagem "*Resolve Match*" indica que o dispositivo que a emitiu corresponde ao tipo de serviço solicitado pelo cliente.

entre as partes. A partir deste ponto a troca de mensagens pode ocorrer de maneira segura.

Caso um dispositivo venha deixar a rede ou uma sessão de comunicação, este deve enviar uma mensagem do tipo "*Bye*", que irá remover o canal. Uma mensagem do tipo "*Hello*" quando enviada por um dispositivo ou cliente que já tenham um canal estabelecido também remove um canal seguro antes estabelecido. Isto ocorre caso o dispositivo tenha se desconectado sem enviar a mensagem "*Bye*". O retorno com a mensagem "*Hello*" deve iniciar um novo processo de estabelecimento de canal seguro.

Nestas trocas para o estabelecimento de canais seguro, é importante salientar que um cliente ou dispositivo poderá suportar diversos protocolos e mecanismos alternativos, sendo assim, sua política deverá listar em ordem de preferências dos mesmos, suas escolhas para a autenticação, cifragem e estabelecimento de chaves.

4. Uso do Modelo Estendido do DPWS

Nesta seção descrevemos a adaptação do modelo para a aplicação alvo definida para a experimentação de nossas contribuições.

4.1. Aplicação alvo

A aplicação alvo deste trabalho é um sistema de controle e medição de energia elétrica para consumidores residenciais. Neste cenário, como exibido na Figura 4, distinguimos três tipos de componentes no modelo computacional da aplicação: os dispositivos (*D*) que fazem a medição de energia elétrica nas residências, os concentradores de informações (*C*), posicionados na rede de distribuição de energia elétrica, com a função de segmentar os dispositivos em grupos de aproximadamente 30 unidades de medição, e um servidor na empresa (*S*) responsável pelo gerenciamento e armazenamento das informações coletadas. Uma característica deste cenário é que os componentes na rede de distribuição de energia elétrica (os dispositivos e os concentradores) estão interligados pelo próprio cabeamento de energia elétrica, por isso, fazem uso da tecnologia PLC (*Power Line Communication*) na comunicação entre si. As comunicações entre concentradores e o servidor da empresa ocorre através da Internet.

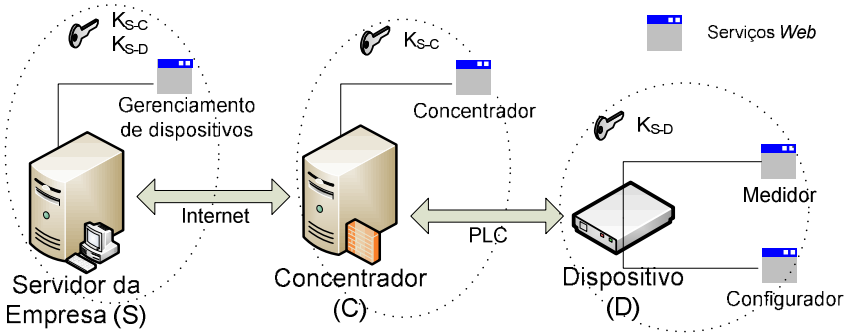

Figura 4 - Componentes e Serviços *Web* da aplicação alvo

A adaptação do nosso modelo estendido do DPWS à aplicação alvo é facilitada pelas características desta última. O gerenciamento de dispositivos sendo feito a partir do servidor da empresa (S), torna possível a centralização da distribuição de chaves criptográficas neste mesmo servidor. Definimos o compartilhamento estático de cada dispositivo medidor com o servidor de chaves criptográficas. Cada dispositivo já sai da

empresa com chave própria compartilhada com servidor. Com isto, o passo de estabelecimento de chave compartilhada, Figura 3 (passo 7) não se faz necessário, da forma como foi definido no modelo geral. Nesta aplicação o servidor faz a distribuição de chaves de sessão para comunicações usando as chaves estáticas de cada participante.

No modelo, o dispositivo se faz conhecer por um concentrador através do serviço de descoberta. Este repassa as informações do dispositivo para o servidor da empresa. A partir destas informações o servidor faz uso das chaves estáticas do concentrador e do dispositivo para distribuir as chaves de sessão para as comunicações entre concentrador e dispositivo.

4.2. Protótipo

Para compor a camada necessária para o desenvolvimento de aplicações baseadas nas especificações de DPWS, escolheu-se o *framework* SOA4D (https://forge.soa4d.org), que por sua vez, é baseado em outra ferramenta para Serviços *Web* chamada gSOAP (http://www.cs.fsu.edu/~engelen/soap.html). A escolha se deu por dois principais motivos: (i) são ferramentas de código aberto e (ii) são baseadas em linguagem C, facilitando o porte dos desenvolvimentos para dispositivos embarcados. Para compor a camada de qualidade de proteção e prover os aspectos de segurança definidos no modelo, adotou-se como base as bibliotecas fornecidas pelo *xmlsec1* (http://www.aleksey.com/xmlsec) em conjunto a *openssl* (http://www.openssl.org).

Ainda no escopo da camada de qualidade de proteção, o conjunto de bibliotecas *xmlsec1* com *openssl* fornecem os mecanismos de implementação para *XML-Encryption* e *XML-Signature*, mas não tratam os aspectos da especificação *WS-Security*, que nesta abordagem, são implementados diretamente pelo Serviço de Segurança. O gSOAP, que fornece os *engines* necessários para trocas e interpretações de mensagens SOAP, permite a criação de complementos, com trechos de código, que agregam as novas funcionalidades ao suporte e aplicações. Fazendo uso deste recurso, o Serviço de Segurança foi desenvolvido e pode ser incluído em aplicações, independente das trocas, por *handlers* interceptando as mensagens a partir do suporte do gSOAP.

Na configuração da Figura 5, que ilustra o cenário utilizado nos testes, é assumido um computador caracterizado como Concentrador (*C*) de informações. Entre o Concentrador *C* e os dois dispositivos de medição (D_1 e D_2) é criada uma rede PLC com *modems* Yitran (IT800D) sobre a rede elétrica em laboratório do grupo. Para os dispositivos D_1 e D_2, foram usados kits de desenvolvimento microcontrolados modelo TS-7200 fabricados pela empresa *Technologic Systems Inc*. Estes equipamentos são dotados de processadores ARM9 de 200MHz, com memória *flash* interna do tipo STRATA de 8MB, dentre outros periféricos, possui duas portas seriais e uma porta Ethernet 10/100 Mbits. Acompanham o equipamento um sistema operacional Linux (Debian), cabos de conexão com PC e ferramentas de software (*Cross Toolchain ARM compiler GCC*) para compilação de programas em Linux.

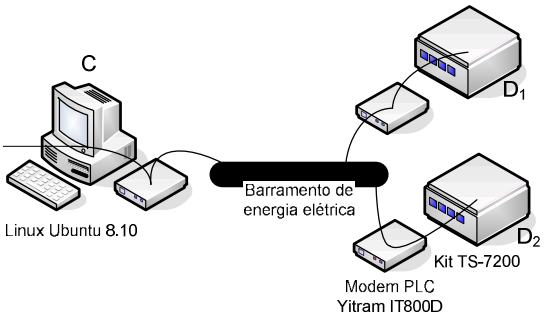

Figura 5. Configuração da rede para os ensaios.

Os dispositivos D_1 e D_2 e o Concentrador C foram programados usando o *toolkit SOA4D*, para que incluíssem o suporte completo do DPWS já estendido com o Serviço de Segurança proposto neste trabalho. No Concentrador são implementados um serviço hospedado de eventos, cuja função é concentrar os dados de medição de energia elétrica enviados pelos dispositivos D_1 e D_2 e um cliente de serviço que configura os dispositivos. Em D_1 e D_2, que são dispositivos DPWS, são implementados dois serviços hospedados: *Configurador* e *Medidor*. O primeiro, com dois métodos, simulam a liberação e o corte de consumo de energia elétrica no local onde o dispositivo está instalado. O segundo, fonte de dados de consumo, notifica via serviço de eventos o concentrador C sobre a atual medição de consumo.

4.3. Avaliação da Implementação e das Escolhas do Modelo Estendido

Nos testes realizados foram verificados principalmente os custos da criptografia e o próprio desempenho da rede PLC, que é de banda estreita. Os testes realizados indicaram que o modem usado tem banda efetiva mínima de 2,2 Kbps, mesmo com degradação da qualidade da rede elétrica e atenuação para grandes lances de cabo. O tamanho médio das mensagens SOAP de notificação com dados de medição, enviadas dos dispositivos D_1 e D_2 para o concentrador, é de aproximadamente 7,2 Kbits. Com o uso do Serviço de Segurança, a inclusão dos cabeçalhos *WS-Security* aumenta o tamanho das mensagens de medição para aproximadamente 24 Kbits.

A ocupação de rede por cada dispositivo pode ser dividido em duas etapas, na primeira, durante sua entrada na rede, transfere em média 314,4 Kbits de dados. A Tabela 2 ilustra o consumo de rede desta etapa, segmentado por protocolos e tipos de mensagens que ocorrem. Na segunda etapa, os dispositivos enviam mensagens periódicas (15 minutos de acordo com a norma ABNT 14522) notificando o consumo de energia elétrica, cada uma com aproximadamente 24 Kbits. Está incluído nestes valores a sobrecarga de dados dos protocolos TCP/IP.

Os valores da Tabela 2 foram obtidos através de um software capturador de pacotes de rede, chamado *Wireshark* (http://www.wireshark.org), e desconsideram retransmissões. O estabelecimento de chaves e contexto de sessão tiveram os cálculos baseados nos exemplos disponíveis na especificação *WS-SecureConversation*.

Considerando as medidas dos tamanhos de mensagens citadas e a taxa de transmissão obtida para o pior caso na rede PLC, a entrada de cada dispositivo na rede, com segurança, levaria 2min20seg. O envio das mensagens periódicas de notificação de consumo (~24 Kbits) leva em média 11 segundos para serem entregues. Mesmo num

cenário onde existam 30 dispositivos em uma rede PLC com todos enviando notificações de medidas simultaneamente a cada 15 minutos, teríamos um total de 720 Kbits para transmitir todas estas informações. Isto ocuparia o canal de comunicação por aproximadamente 5min30seg. Estes números dão a garantia que o uso do DPWS estendido sobre uma rede PLC de banda estreita é compatível com a norma NBR14522.

Tabela 2. Custo médio de utilização de rede de um dispositivo

Protocolo	***Tipo mensagem***	***Tamanho***
WS-Discovery	*Hello (~23,2 Kbits), Probe (~20 Kbits), ProbeMatch (~24,8 Kbits), Bye (~20 Kbits)*	**~88 Kbits**
WS-MetadataExchange	*Metadados do dispositivo (~28 Kbits), WSDL (~56 Kbits), Políticas (~32 Kbits)*	**~116 Kbits**
Serviço de Segurança	*Estabelecimento de chave de sessão – WS-SecureConversation*	**~56 Kbits**
WS-Aplicação	*Libera_Consumo*	**~27,2 Kbits**
WS-Eventing	*Inscrição*	**~27,2 Kbits**
	Total	**~314,4 Kbits**

O pior caso seriam todos os 30 dispositivos entrando no sistema ao mesmo tempo. Se assim acontecer, a quantidade de dados a transmitir seria de aproximadamente 9,6 Mbits, que na taxa de 2,2 Kbps levaria até 1h13min para que todos os dispositivos estejam notificando medições. Existe a possibilidade de suprimir o TCP/IP e o HTTP, utilizando o SOAP diretamente sobre PLC, com isso, a redução no tamanho das mensagens que utilizam UDP seria de aproximadamente 1,8 Kbits e as que utilizam TCP+HTTP seria de 8 Kbits. Ao invés dos 314,4 Kbits da primeira etapa teríamos em torno de 224 Kbits, o que reduzia em 38 segundos a etapa de entrada do dispositivo no sistema. Nas mensagens de notificação de consumo, para os 30 dispositivos, a redução seria de 1min54seg.

As proposições do modelo estendido do DPWS procuram prover meios para suprir as necessidades de segurança nas trocas de dispositivos em geral que ficam disponíveis via SOA a aplicações na Internet ou em sistemas abertos como a rede PLC. As trocas de mensagens feitas pelo Serviço de Segurança foram implantadas seguindo os padrões de segurança *WS-Security* e *WS-SecureConversation*, propostos para a tecnologia Serviços *Web*. Estes padrões aportaram funcionalidades que garantem a segurança fim a fim em ambiente onde existe o roteamento em nível de aplicação.

Os componentes da arquitetura, sempre que entram na rede, estão de posse de uma chave simétrica (estática) compartilhada com o serviço de gerenciamento de dispositivos, como foi exibido na Figura 4. Em seus processos de identificação na rede, estes componentes usam a chave compartilhada e o algoritmo de criptografia AES[10] para receber as chaves complementares (as chaves de sessão, também simétricas) para as comunicações com seus pares. Através do uso das operações de cifragem do *XML-Encryption* e das chaves de citadas, a confidencialidade das informações emitidas é garantida fim a fim. O concentrador, por exemplo, lida com mensagens de vários dispositivos sem que haja possibilidade da quebra da confidencialidade destas no mesmo.

Os aspectos de integridade e de validade são também garantidos com o uso do *XML-Encryption*, *hashes* e *nonces*. Os *hashes* usados na verdade são MACs, calculados

[10] http://www.csrc.nist.gov/publications/fips/fips197/fips-197.pdf

com a função HMAC-SHA1 que é disponível em biblioteca do *xmlsec1*. A validade é garantida por *nonces* e a posse das chaves simétricas somente pelos pares comunicantes. Ataques de *replay*, por exemplo, são facilmente detectados com estes mecanismos citados.

Nas proposições deste trabalho, outro aspecto a ser considerado é o uso de chaves estáticas para a comunicação com o serviço de gerenciamento de dispositivos no servidor da concessionária. Estas chaves até poderiam ser trocadas periodicamente ou ainda, usado algum esquema de chaves derivadas para diminuir a exposição das mesmas. Porém considerando a finalidade da aplicação (sistema de medição), acreditamos que os custos relacionados para diminuir a exposição destas chaves não compensam pela baixa probabilidade de revelação das mesmas. Por fim, a escolha da criptografia simétrica neste trabalho é justificável principalmente pelas restrições computacionais dos dispositivos embarcados.

5. Trabalhos Relacionados

Diversos trabalhos na literatura abordam a utilização de Serviços *Web* em dispositivos embarcados, poucos detalham ou falam dos aspectos de segurança. Algumas iniciativas, como em [Castro et al. 2001], utilizam uma infraestrutura *ad-oc* para realizar a monitoração remota de dispositivos, sem uso de padrões como, por exemplo, o DPWS.

Alguns trabalhos fazem uso da grande difusão do SNMP (*Simple Network Management Protocol*). É o caso do trabalho proposto em [Almqvist 1994]. Os autores, neste caso, investigam e testam este protocolo no gerenciamento de dispositivos remotos. Procuram mostrar a eficiência do SNMP no gerenciamento de equipamentos na área de energia elétrica. A característica principal deste protocolo é a sua simplicidade, mas no próprio artigo são feitas ressalvas sobre a falta de mecanismos de segurança adequados, a entrega de notificações e de alarmes não confiável (devido ao uso do UDP) e a característica estática da estrutura de armazenamento de variáveis remotas (MIB - *Management Information Base*).

Aiko Pras *et al* [Pras et al. 2004] fazem uma comparação entre o SNMP e Serviços *Web* nos aspectos de transmissão de dados, consumo de memória, tempo de processamento e *round trip delay*. Segundo os autores, o SNMP é mais eficiente quando a requisição é feita para poucos objetos. Contudo, se a requisição deve tratar com muitas fontes de dados, Serviços *Web* se tornam mais efetivos devido às possibilidades de agregação e de roteamento em nível de aplicação deste último.

Desde o surgimento do primeiro *Draft* da especificação DPWS, este vem sendo estudado por diversos grupos de pesquisa, objetivando principalmente aplicações práticas do mesmo. Um dos primeiros trabalhos que abordam estas especificações em dispositivos embarcados veio de uma iniciativa européia, através do projeto SIRENA, que teve por objetivo desenvolver uma infraestrutura de serviços para aplicações em redes de dispositivos embarcados. Neste projeto a arquitetura SOA e as especificações DPWS são usadas para interligar dispositivos embarcados entre quatro áreas distintas de aplicações: a industrial, telecomunicações, automotiva e de automação residencial.

Outros dois projetos tomaram como base os estudos e experiências do projeto SIRENA. São neste caso o projeto SODA (*Service Oriented Device & Delivery Architecture*) que tinha como objetivo a criação de uma plataforma baseada em SOA, abrangente, escalável e fácil de implantar para dispositivos de baixo custo; e o

SOCRADES que investiga a aplicabilidade do DPWS para um grande número de dispositivos objetivando a integração de chão de fábrica com uma infraestrutura orientada a serviços ligada às atividades da camada de negócios de uma organização. Em [Karnouskos 2009], mostra com os resultados de simulações do projeto SOCRADES a escalabilidade do DPWS. Com auxílio de uma plataforma de simulação de agentes, são criados milhares de dispositivos DPWS e testados os serviços de descoberta e as trocas de mensagens entre estes. Outro ponto explicitado nesta experiência é que a adoção de SOA nos dispositivos e equipamentos que estão nos níveis de chão de fábrica permite a integração direta dos mesmos aos níveis mais altos e gerenciais de uma aplicação.

Em [Martínez et al. 2008] é apresentado uma proposta de extensão para a especificação de segurança do DPWS. Desenvolvida no âmbito do projeto SODA, essas extensões seriam disponibilizadas através de bibliotecas para os desenvolvedores de serviço e clientes no ambiente DPWS. O modelo apresentado é mais ambicioso que o apresentado no nosso trabalho, pois apresenta módulos de "Autenticação Local" e "Gerenciadores de Autorização", etc. A nossa proposição se abstém destes serviços e mecanismos por entender que controles são próprios para níveis mais altos de uma aplicação. Os controles de segurança para níveis mais baixos devem ser mais simples e eficientes. No seu serviço de segurança, os autores citados também estão prevendo o uso do *WS-Security* para a segurança considerando as possibilidades de roteamento em nível de aplicação e a necessidade de garantias fim a fim, mas em trabalho subsequente ([Hernández 2009]), onde um modelo é apresentado, afirmam não obedecer totalmente a especificação *WS-Security*. O serviço de segurança criado também como o nosso é incluído na forma de um serviço embutido. Porém, na base necessária para estabelecer contexto de segurança e estabelecimento de chaves, não usam protocolos padronizados como o *WS-SecureConversation*, como é feito na abordagem apresentada no nosso texto.

6. Conclusões

Este artigo apresentou uma experiência com DPWS e uma proposta de extensão ao mesmo com *WS-Security* e *WS-SecureConversation* para prover segurança fim a fim. Nas especificações DPWS a segurança é tratada como opcional. Não são dadas alternativas para segurança fim a fim para aplicações que são em muitos casos envolvidas com roteamento. Esta proposta de extensão, como contribuição, parece, portanto, bem pertinente. As extensões são agrupadas em um Serviço de Segurança que é incluído na plataforma usada na forma de complemento e tem suas diferentes funções ativadas por *handlers* na execução dos diferentes protocolos definidos. O protótipo montado mostrou a efetividade das soluções adotadas no modelo.

O DPWS e as extensões desenvolvidas estão sendo aplicados em sistemas de controle e aquisição de dados, prevendo atender um setor ainda carente de automação que é o setor elétrico. As experiências na bibliografia com este tipo de aplicação, envolvendo o acesso a dispositivos via Internet, normalmente envolvem a integração de diversas ferramentas com Serviços *Web*. A solução proposta neste texto não envolve integração de ferramentas atuando em diferentes níveis da aplicação. Dispositivos são integrados via Internet em aplicações orientadas a serviço. Mas a contribuição definitiva da proposta é o aporte de mecanismos de segurança cujas proposições da literatura pouco fazem menção nestes sistemas que envolvem ambientes abertos como a Internet.

References

Almqvist, L e Wikstrom, R. (1994) “Standardizing energy management by using simple network managementprotocol”, In: Telecommunications Energy Conference, 1994. INTELEC '94., 16th International.

Bartel, M., Boyer, J., e Fox, B. (2002). XML-Signature Syntax and Processing. W3C.

Castro, A L S et al. (2001) “Power Quality Monitoring Instrument for Energy Distribution Feeder”, 11th IMEKO TC-4 Symposium Trends in Electrical Measurement and Instrumentation and 6th EuroWorkshop on ADC modelling and testing, Lisbon - PORTUGAL, September 13-14.

Dierks, T. et al. (1999) “The TLS Protocol”, Version 1.0, http://www.ietf.org/rfc/rfc2246.txt, IETF RFC 2246, January 1999.

DPWS (2009). Devices Profile for Web Services. OASIS.

Hernández, V., López, L., Prieto, O., Martínez, J. F., García, A. B. e Da-Silva., A. (2009) “Security Framework for DPWS Compliant Devices”. In: Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference.

Imamura, T., Dillaway, B., e Simon, E. (2002). XML Encryp. Syntax and Proc. W3C.

Jammes, F., Mensch, A. e Smit, H. (2005) “Service-Oriented Device Communications Using the Devices Profile for Web Services”. In: Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing, páginas: 1-8, Grenoble, France.

Karnouskos, S. e Tariq, M. M. J. (2009) “Using Multi-Agent Systems to Simulate Dynamic Infrastructures Populated with Large Numbers of Web Service Enabled Devices”. In: The 9th International Symposium on Autonomous Decentralized Systems – ISADS, Athens, Greece.

Martínez, J. F., López, M., Hernández, V., Jean-Marie, K., García, A. B., López, L., Herrera, C. e Sánchez-Alarcos, C. J. (2008) “A security architectural approach for DPWS-based devices”. In: CollECTeR Ibéroamérica 2008 conference. Madrid, Spain.

OASIS (2004). WS Security: SOAP Message Security 1.0. OASIS.

OASIS (2007). WS-SecureConversation. OASIS.

OASIS (2009). Web Services Dynamic Discovery (WS-Discovery). OASIS.

Pras, A., Drevers, T., van de Meent, R. e Quartel. D. (2004) “Comparing the Performance of SNMP and Web Services-Based Management”, In: ETRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, FALL 2004. IEEE 2004.

RFC 2631 (1999). “Diffie-Hellman Key Agreement Method”. E. Rescorla Junho 1999.

W3C (2006a). Web Services Transfer (WS-Transfer). W3C. Disponível em: http://www.w3.org/Submission/WS-Transfer/

W3C (2006b). Web Services Addressing 1.0 - Core. W3C. Disponível em: http://www.w3.org/TR/ws-addr-core/

W3C (2007). Web Services Policy 1.5 – Framework. W3C. Disponível em: http://www.w3.org/TR/ws-policy/