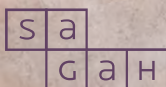


# APLICAÇÕES DE INTERNET DAS COISAS

Fabricao Machado da Silva



SOLUÇÕES  
EDUCACIONAIS  
INTEGRADAS



# Acesso a aplicações Web e Web das Coisas (WoT)

## Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Distinguir Internet das Coisas (IoT) de Web das Coisas (WoT).
- Definir Representational State Transfer (REST) e Resource Oriented Architecture (ROA).
- Analisar os modelos de acesso Context Based Access Control (CBAC) e Attribute Based Access Control (ABAC).

## Introdução

O princípio da Web das Coisas é a conexão de diversos dispositivos, objetos e serviços por meio da Web, utilizando seus padrões e suas tecnologias variadas. Tal modelo está se tornando cada vez mais comum, fazendo com que as aplicações sejam deixadas em segundo plano, percam relevância ou mesmo se tornem desnecessárias, o que simplifica problemas comuns.

Neste capítulo, vamos comparar e contrastar a Internet das Coisas com a Web das Coisas, vamos examinar os conceitos de REST e ROA e vamos analisar os modelos de acesso CBAC e ABAC.

## Distinção entre Internet das coisas e Web das Coisas

A diferença entre a Internet das Coisas (IoT, Internet of Things) e a Web das Coisas (WoT, Web of Things) pode parecer uma distinção direta e fácil de estabelecer. No entanto, os contrastes entre os dois conceitos são muito mais sutis do que se possa imaginar; cada detalhe da IoT e da WoT cria dois assuntos muito diferentes, cada um com seu próprio conjunto de ideias, estruturas e usos. Apesar de suas diferenças, a IoT e a WoT trabalham em uníssono e dependem uma da outra para funcionar sem problemas e com mais eficiência. Segundo Magrani (2018, p. 44):

A expressão IoT é utilizada para designar a conectividade e interação entre vários tipos de objetos do dia a dia, sensíveis à internet. Fazem parte desse conceito os dispositivos de nosso cotidiano que são equipados com ‘sensores capazes de captar aspectos do mundo real, como, por exemplo, temperatura, umidade e presença, e enviá-los a centrais que recebem estas informações e as utilizam de forma inteligente’.

Tudo na IoT está conectado, transformando objetos do cotidiano em dispositivos inteligentes, que são reconhecidos, localizados e até mesmo controlados pela Internet. Os objetos conectados à IoT podem ser desde muito simples a muito complexos. Dentre os mais simples e mais comuns, estão as encomendas enviadas com um número de rastreamento em uma etiqueta (*tag*) de identificação automática, seja na forma de código de barras, código QR, NFC ou RFID. Essas etiquetas permitem que seu pacote seja rastreado desde o fornecedor, passando pelo centro de expedição, até a entrega da encomenda. Um exemplo mais complexo é a tecnologia usada na manutenção e conserto de grandes equipamentos e máquinas. Os sensores podem ser instalados dentro de máquinas e enviar alertas e relatórios aos fabricantes quando algo não estiver funcionando corretamente. Isso pode evitar ao mau funcionamento dos principais equipamentos, levando a uma redução de custos e de ferimentos de funcionários.

Uma das limitações da IoT é que, quando muitos objetos tentam se comunicar de maneiras diferentes, é um desafio criar perfeitamente uma plataforma única de comunicação na qual todas as coisas conversem entre si de maneira eficaz. Uma iniciativa foi criar uma plataforma de comunicação completamente nova para que todos os dispositivos possam se comunicar de maneira eficaz, também conhecida como protocolo de camada de aplicativo (ou linguagem, mais simplesmente). No entanto, essa alternativa consome tempo e pode ser quase impossível de organizar. Por isso, muitos especialistas da área acreditam que devemos usar uma plataforma que já existe, como a World Wide Web (MAGRANI, 2018).

É aqui que o WoT entra em cena. A Web é um sistema já estabelecido que permite que todas as “coisas” se comuniquem. A Web pode ser usada como um tipo de aplicativo através do qual todos os objetos podem se comunicar da maneira mais eficiente. A WoT é muito semelhante à IoT em alguns aspectos, mas em outros é drasticamente diferente. O WoT foi inspirada na IoT, pois, em geral, os dispositivos cotidianos estão conectados à Web e podem se comunicar através de vários sistemas.

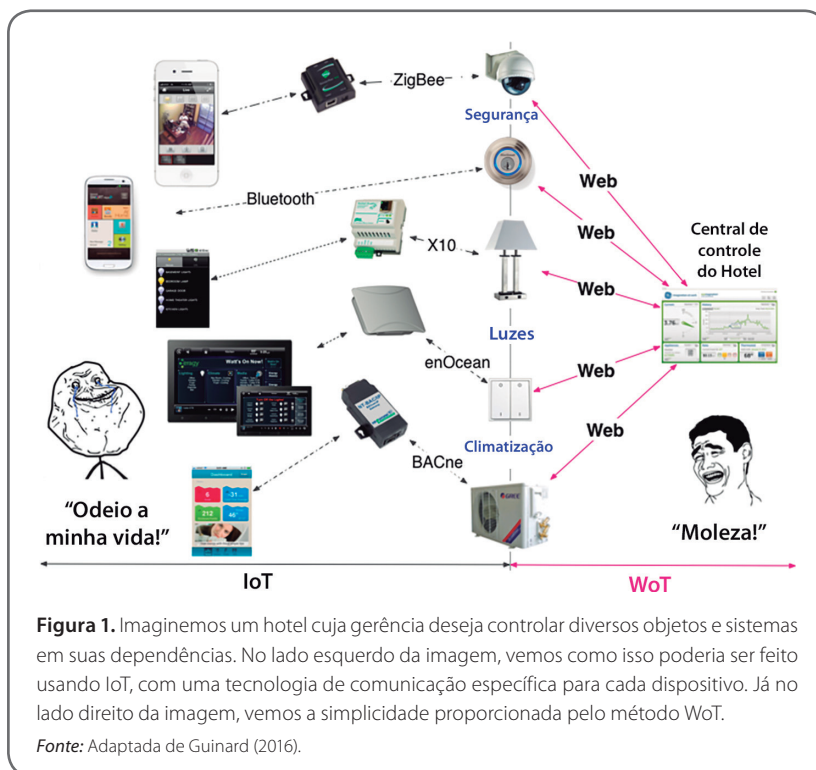
No entanto, um dos seus contrastes primordiais é que a WoT se concentra em reutilizar o sistema da Web já estabelecido para ajudar esses dispositivos a se manterem conectados permanentemente a um único aplicativo. Quando todos os dispositivos conectados são capazes de usar uma única base, a Web, para armazenar informações e se comunicar, a complexidade e a variedade dos protocolos de nível inferior por trás do modelo simples da Web oferece muitas vantagens. Assim como a Web se tornou a plataforma de integração global para aplicativos distribuídos pela Internet, a WoT facilita a integração de todos os tipos de dispositivos e aplicativos que interagem com eles. Em outras palavras, ocultando a complexidade e as diferenças entre os vários protocolos de transporte usados na IoT, a WoT permite que os desenvolvedores se concentrem na lógica de seus aplicativos, sem terem que se preocupar em como esse ou aquele protocolo ou dispositivo realmente funciona.



### Saiba mais

A Interface de Programação de Aplicações (API, Application Programming Interface) de serviços Web (Web services) é como uma estrada que liga duas cidades distantes que até então encontravam-se isoladas uma da outra sem este caminho. Uma API pode ser utilizada por diferentes aplicações de negócio, não necessitando que cada aplicação conheça os detalhes da sua implementação. As APIs são interfaces de integração que permitem a troca de informações entre aplicações, possibilitando inclusive que uma mesma aplicação apresente comportamentos específicos para uma versão móvel, outra para *desktop* e outra para Web.

Na Figura 1, é exibido um exemplo de IoT e WoT. A WoT é a capacidade de usar padrões modernos da Web diretamente em dispositivos incorporados. Ao alavancar todos esses padrões para cenários de IoT, permitimos a criação de novos tipos de aplicativos interativos e também garantimos que os dispositivos possam ser integrados a aplicativos e serviços da Web modernos com o mínimo de esforço.



## Conceitos de REST e ROA

O modelo de Transferência Representacional de Estado (REST, Representational State Transfer) representa uma abstração da arquitetura da Web. Resumidamente, o REST consiste em princípios/regras/restrições que, quando observados, permitem a criação de um projeto com interfaces bem definidas. Esse modelo de arquitetura foi descrito na tese de doutorado de Roy Fielding, que, além de ser coautor do HTTP, também foi cofundador do projeto do servidor HTTP Apache, um estilo arquitetônico que explora a tecnologia e os protocolos existentes da Web, incluindo HTTP e XML (W3C, 2004).

O REST acabou adotado como o modelo a ser utilizado na evolução da arquitetura do protocolo HTTP. O uso de serviços REST tem crescido amplamente, e grandes empresas como Google, Facebook, Yahoo!, Amazon, eBay e Microsoft vêm aderindo e oferecendo suporte a essa tecnologia. No cenário atual, em que o conceito de “serviço” ganha força a cada dia, REST

é atualmente uma opção importantíssima a se considerar na hora de definir como será disponibilizado um serviço.

Esse modelo permitiu o uso do HTTP de uma forma muito mais próxima da nossa realidade, dando sentido às requisições HTTP. Para melhor compreensão, vejamos exemplos de requisições em formatos fictícios:

- GET — `http://www.meusite.com/usuarios`
- DELETE — `http://www.meusite.com/usuarios/jackson`
- POST — `http://www.meusite.com/usuarios -data {nome: joaquim}`

Pela simples leitura, é possível inferir que no primeiro caso estamos pegando (GET) todos os usuários do *site*, ou seja, teremos uma lista de todos os usuários que estão cadastrados no sistema. Já no segundo caso, estamos apagando (DELETE) o usuário Jackson. No último exemplo, estamos usando o método POST, em que percebemos o envio de dados extras para cadastrar um novo usuário. Veja como ficou simples expressar o que queremos realizar ao acessar um determinado endereço usando verbos específicos para URLs específicas e usando dados padronizados, quando necessário. Pois estes princípios apresentados fazem parte do REST. Em outras palavras, nesses exemplos temos uma representação padronizada, verbos e métodos usados e ainda URLs.



### Saiba mais

Serviços Web do tipo REST possibilitam uma série de benefícios, mas é importante lembrar que, como não se trata de um padrão, você não necessariamente é obrigado a segui-lo para criar serviços ou aplicações Web. Podemos classificá-lo como uma espécie de “guia” com recomendações.

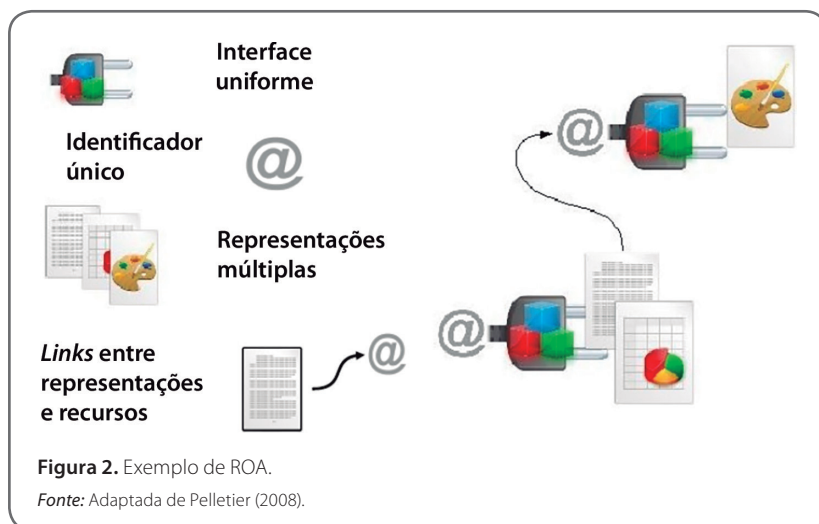
Por sua vez, a Arquitetura Orientada a Recursos (ROA, Resource Oriented Architecture) é o *design* estrutural que suporta a *internetworking* de recursos. Um **recurso**, nesse contexto, é qualquer entidade que pode ser identificada e receber atribuição de um Uniform Resource Identifier (URI). Em tecnologia da informação (TI), **arquitetura** refere-se à estrutura geral de um sistema de informação e às inter-relações de entidades que compõem esse sistema. O ROA é considerada uma arquitetura RESTful (ao estilo REST).

Dentro do conceito de ROA, os recursos incluem não apenas elementos de infraestrutura de TI, como servidores, computadores e outros dispositivos, mas também páginas da Web, *scripts*, páginas JSP/ASP e outras entidades, como semáforos.

O ROA apresenta quatro recursos básicos:

1. Endereçabilidade: um aplicativo é endereçável quando expõe um URI para todas as informações que possa servir.
2. Apátrida: toda solicitação HTTP ocorre em isolamento completo; o servidor nunca conta com informações de solicitações anteriores. Se essas informações fossem importantes, o cliente as teria enviado novamente nesta solicitação. O estado do aplicativo deve residir no cliente e o estado do recurso deve residir no servidor e ser o mesmo para todos os clientes.
3. Conexão: um serviço da Web está conectado na medida em que você pode colocá-lo em diferentes estados apenas seguindo os *links* e preenchendo os formulários.
4. Interface uniforme.

A Figura 2 apresenta um exemplo de ROA, em que são exibidas várias conexões de *links* entre os dispositivos e recursos em questão.



## Modelos de acesso CBAC e ABAC

O controle de acesso baseado em contexto (CBAC, *contexto-based access control*) é um recurso do *software* de *firewall* que filtra de forma inteligente os pacotes Transmission Control Protocol (TCP) e User Datagram Protocol (UDP) com base nas informações da sessão do protocolo da camada de aplicação. Pode ser usado para Intranets, Extranets e Internet.

O CBAC pode ser configurado para permitir o tráfego TCP e UDP especificado por meio de um *firewall* somente quando a conexão é iniciada na rede que precisa de proteção. Em outras palavras, o CBAC pode inspecionar o tráfego para as sessões originárias da rede externa. Além disso, o CBAC pode inspecionar o tráfego para as sessões originárias dos dois lados do *firewall*. Essa é a função básica de um *firewall* de inspeção com estado (HU *et al.*, 2018).

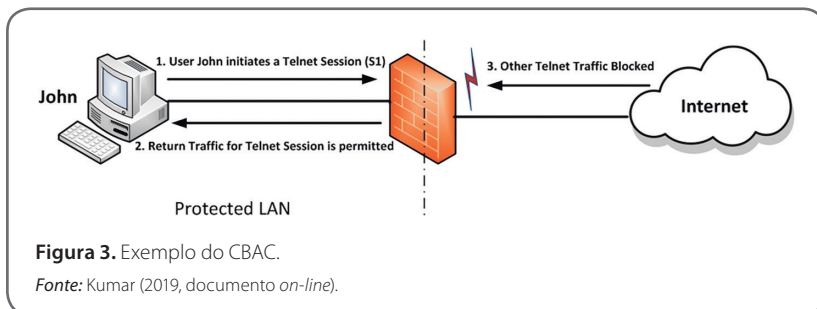
Sem o CBAC, a filtragem de tráfego fica limitada às implementações da lista de acesso que examinam pacotes na camada de rede ou, no máximo, na camada de transporte. No entanto, o CBAC examina não apenas as informações da camada de rede e da camada de transporte, mas também as informações do protocolo da camada de aplicativos — como informações de conexão File Transfer Protocol (FTP) — para conhecer o estado da sessão TCP ou UDP.

Isso permite o suporte de protocolos que envolvem vários canais criados como resultado de negociações no canal de controle FTP. A maioria dos protocolos multimídia, bem como alguns outros protocolos (como FTP, RPC e SQL\*Net), envolve vários canais de controle.

O CBAC inspeciona o tráfego que atravessa o *firewall* para descobrir e gerenciar informações de estado para sessões TCP e UDP. Essas informações de estado são usadas para criar aberturas temporárias nas listas de acesso do *firewall* a fim de permitir tráfego de retorno e conexões de dados adicionais para sessões permitidas (sessões originadas dentro da rede interna protegida).

Como o CBAC faz a inspeção profunda de pacotes, é chamado de um *firewall* IOS. O CBAC também oferece benefícios como prevenção e detecção de negação de serviço, alertas em tempo real e trilhas de auditoria. A Figura 3 ilustra a estrutura de funcionamento do CBAC.



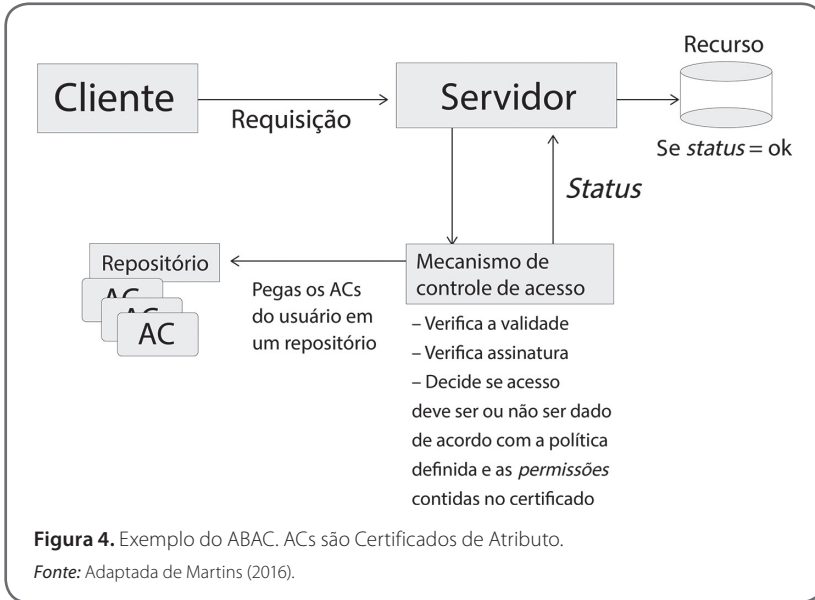


Por sua vez, o controle de acesso baseado em atributo (ABAC, *attribute-based access control*) pode fornecer controle de acesso contextual e refinado, o que permite um número maior de entradas discretas em uma decisão de controle de acesso e um conjunto maior de combinações possíveis dessas variáveis, para refletir possíveis regras, políticas ou restrições de acesso mais diversas e definitivas.

O ABAC permite que os administradores apliquem a política de controle de acesso sem o conhecimento prévio completo do assunto específico, usando outros pontos de dados que podem ser uma forte identidade de indicador. Quando combinados com outros atributos, os indicadores podem estabelecer uma base da confiança suficiente da identidade e propriedade do dispositivo para autorizar o acesso a serviços e transações. As políticas de controle de acesso que podem ser implementadas no ABAC são limitadas apenas pela linguagem computacional e pela riqueza dos atributos disponíveis.

O ABAC também pode fornecer recursos de controle de acesso mais dinâmicos e diminuir a necessidade de manutenção a longo prazo das proteções a objetos, já que as decisões de acesso podem se alterar entre as solicitações quando os valores dos atributos são modificados.

A Figura 4 traz uma demonstração do ABAC. Ela mostra que o serviço de autorização faz as mesmas validações que um serviço de autenticação, usando validação de data de expiração contra um atributo do próprio certificado e, em seguida, validando a revogação contra a Lista de Certificados Revogados (LCR) ou o Online Certificate Status Protocol (OCSP).



## Link

O link a seguir leva à página com a documentação WoT.

<https://qrgo.page.link/zjhW7>



## Referências

GUINARD, D. Web of things vs internet of things: 1/2. *Web of Things*, 2016. Disponível em: <https://webofthings.org/2016/01/23/wot-vs-iot-12/>. Acesso em: 22 nov. 2019.

HU, V. C. *et al.* *Attribute-based access control*. Boston: Artech house, 2018.

KUMAR, S. Context-based access control. *TechTutsOnline*, 2019. Disponível em: <https://www.techtutsonline.com/context-based-access-control/>. Acesso em: 22 nov. 2019.

MAGRANI, E. *A internet das coisas*. Rio de Janeiro: FGV Editora, 2018.

MARTINS, R. Autorização segura de dispositivos com certificado de atributo na internet das coisas. *Blog Atitude Reflexiva*, 2016. Disponível em: <https://atitudereflexiva.wordpress.com/2016/01/05/autorizacao-segura-de-dispositivos-com-certificado-de-atributo-na-internet-das-coisas/>. Acesso em: 22 nov. 2019.

PELLETIER, A. *Resource oriented architecture (ROA): 4<sup>th</sup> generation*. 2008. Apresentação. Disponível em: <https://pt.slideshare.net/toutantic/resource-oriented-architecture>. Acesso em: 22 nov. 2019.

W3C. *Web service architecture*. 2004. Disponível em: <https://www.w3.org/TR/ws-arch/#relwwwrest>. Acesso em: 22 nov. 2019.

### Leitura recomendada

INTERNET e web das coisas. *Ceweb.br*, [20--?]. Disponível em: <https://ceweb.br/wot/>. Acesso em: 22 nov. 2019.



### Fique atento

Os *links* para *sites* da Web fornecidos neste capítulo foram todos testados, e seu funcionamento foi comprovado no momento da publicação do material. No entanto, a rede é extremamente dinâmica; suas páginas estão constantemente mudando de local e conteúdo. Assim, os editores declaram não ter qualquer responsabilidade sobre qualidade, precisão ou integralidade das informações referidas em tais *links*.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:



SOLUÇÕES  
EDUCACIONAIS  
INTEGRADAS