UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

BENEDITO MENDES DUTRA NETO

SENSIBILIDADE AO CONTEXTO EM REDES COMUTADAS: a implantação do Spanning Tree Protocol na rede acadêmica da UFMA

São Luís
2015

BENEDITO MENDES DUTRA NETO

SENSIBILIDADE AO CONTEXTO EM REDES COMUTADAS: a implantação do Spanning Tree Protocol na rede acadêmica da UFMA

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Samyr Beliche Vale.

São Luís

2015

Dutra Neto, Benedito Mendes

Sensibilidade ao contexto em redes comutadas: a implantação do spanning tree protocol na rede acadêmica da UFMA / Benedito Mendes Dutra Neto. – São Luís, 2015.

62f.

Monografia (Graduação) – Curso de Ciência da Computação, Universidade Federal do Maranhão, 2015.

Orientador: Prof. Dr. Samyr Beliche Vale

1. Computação ubíqua. 2. Rede de computadores. I. Título.

CDU 004

BENEDITO MENDES DUTRA NETO

SENSIBILIDADE AO CONTEXTO EM REDES COMUTADAS: a implantação do Spanning Tree Protocol na rede acadêmica da UFMA

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Samyr Beliche Vale.

Aprovada em 05/08/2015

BANCA EXAMINADORA

Prof. Dr. Samyr Beliche Vale (Orientador)
Universidade Federal do Maranhão

Prof. Dr. Bruno Feres de Sousa
Universidade Federal do Maranhão

Prof. Dr. Mário Meireles Teixeira
Universidade Federal do Maranhão

A meus pais, Maria da Conceição e Milton Domingos, pela grande arte de viver.
A Karina Cristina, meu eterno amor, pelo meu plural em ti.

AGRADECIMENTOS

A Deus, pela saúde e força durante minha jornada.

Aos meus pais, Maria da Conceição e Milton Domingos, que nunca deixaram de me incentivar e torcer pelo meu sucesso. Um obrigado especial a minha mãe, a mais ansiosa e certamente a mais feliz com a conclusão desta etapa em minha vida.

Ao meu irmão, Sandro Karlo, que sempre se pôs a disposição para sanar minhas dúvidas relativas à programação. A sua esposa Rosyane e a mais nova estrela da família, Lara Vitória, minha afilhada amada, que ilumina nossas vidas e nos enche de alegrias.

A Maria do Socorro (*in memorian*), minha saudosa madrinha, por tudo que sempre significou na minha vida. Madrinha, obrigado pelos seus preciosos ensinamentos éticos e valorosos.

Aos meus avós, Iete (*in memorian*), Raimundo, Benedito (*in memorian*) e Maria José (*in memorian*) a quem sou grato pelos ensinamentos de vida.

A todos os meus tios, tias e primos, em especial àqueles com que tenho maior contato e grande admiração: Felipe, Kássya e Flávio (Dudu).

A Karina Cristina, minha eterna namorada e companheira, por ter me aguentado tão pacientemente durante esta etapa e pelas preciosas correções ortográficas.

Aos meus amigos, que são poucos, mas bons, com quem sei que posso contar a toda hora.

Ao meu orientador, Samyr Beliche, pelos ensinamentos e condução na construção deste trabalho, que acreditou e confiou nesta proposta.

Ao Núcleo de Tecnologia da Informação da UFMA, onde aproveitei e sou extremamente grato à disponibilidade e o interesse dos funcionários em transmitir suas experiências e conhecimentos, em especial ao mestre Leônidas, por quem sou grato pelos ensinamentos pessoais e profissionais.

E a todos os meus colegas, de curso, de trabalho e dos mais que estiveram presentes nesta caminhada e fizeram parte desta conquista.

“As tecnologias mais profundas são aquelas que desaparecem. Elas dissipam-se nas coisas do dia-a-dia até tornarem-se indistinguíveis.”

Mark Weiser

RESUMO

Manter a disponibilidade de uma rede de computadores através de um ambiente redundante, tolerante a falhas, que atua de forma transparente, possuindo métodos intuitivos para que o usuário usufrua das aplicações sem ter conhecimento específico através de ambientes sensíveis, adaptáveis e respondíveis às necessidades, são características da computação ubíqua. Este trabalho analisa o cenário da rede de dados da Universidade Federal do Maranhão (UFMA) e propõe uma melhoria através do uso do *Spanning Tree Protocol* visando otimizar a disponibilidade e evitar *loops* na camada de acesso da rede. Culmina com uma proposta de implementação de um aplicativo ubíquo para notificar os administradores da rede sobre um eventual *loop*.

Palavras-chave: Computação Ubíqua, Sensibilidade ao Contexto, Redes de Computadores, *Spanning Tree Protocol*.

ABSTRACT

Maintain the availability of a computer network via a redundant environment, fault-tolerant, which operates transparently, possessing intuitive method for the user to make use of applications without having specific knowledge through sensitive environments, adaptable and answerable are characteristics of ubiquitous computing. This paper analyzes the scenario of network data of the Federal University of Maranhão (UFMA) and proposes an improvement by using the Spanning Tree Protocol to optimize the availability and prevent loops in the network access layer. Culminates with a proposal to implement a ubiquitous application to notify the network administrators on a possible loop.

Keywords: Ubiquitous Computing, Context Awareness, Computer Networking, Spanning Tree Protocol.

LISTA DE FIGURAS

Figura 1. (a) LAN e (b) VLAN 27
Figura 2. Quadro 802.1Q 29
Figura 3. Broadcast storm 32
Figura 4. BPDU de configuração 34
Figura 5. Cabeçalho ISL 37
Figura 6. Rede hierárquica 40
Figura 7. Redundância topológica 42
Figura 8.1. *Loop* no mesmo switch 43
Figura 8.2. *Loop* entre switches 43
Figura 8.3. *Loop* em expansão irregular da rede 43
Figura 9. Estrutura da PRH 45
Figura 10.1. CLI do switch PRH-1 46
Figura 10.2. CLI do switch PRH-2 46
Figura 11. Monitoramento do switch 47
Figura 12. Monitoramento do STP 47
Figura 13. Arquitetura do STP Helper 49
Figura 14. Tela de cadastro 50
Figura 15. E-mail de alerta de loop gerado pelo STP Helper 51
Figura 16. Diagrama de atividades 52
Figura 17. Diagrama de sequência 53

LISTA DE SIGLAS

BPDU	Bridge Protocol Data Unit
DEC	Digital Equipamente Corporation
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Eletrical and Eletronics Engineers
ISO	International Organization for Standardization
LAN	Local Area Network
LLC	Logical Link Control
MAC	Media Access Control
MSTP	Multiple Spanning Tree Protocol
OSPF	Open Shortest Path First
PC	Personal Computer
PRH	Pró-Reitoria de Recursos Humanos
PVST	Per-VLAN Spanning Tree
RFC	Request for Comments
RSTP	Rapid Spanning Tree Protocol
SFP	Small Form-factor Pluggable Transceiver
STP	Spanning Tree Protocol
TCP	Transmission Control Protocol
VLAN	Virtual LAN
WAN	Wide Area Network

SUMÁRIO

1 INTRODUÇÃO 14
2 COMPUTAÇÃO UBÍQUA E SENSIBILIDADE AO CONTEXTO 16
2.1 Definição de computação ubíqua 17
2.2 Contexto e sensibilidade ao contexto 18
2.2.1 Contexto 19
2.2.2 Sensibilidade ao contexto 20
2.3 Características de aplicações sensíveis ao contexto 21
2.4 Técnicas e arquiteturas para o desenvolvimento de aplicações ubíquas sensíveis ao contexto 22
3 REDES DE COMPUTADORES 25
3.1 LANs Virtuais 25
3.1.1 Segmentação de redes 25
3.1.2 Definição de VLAN 26
3.1.3 Benefícios 28
3.2 Padrão IEEE 802.1Q 28
4 Spanning Tree Protocol 31
4.1 Loops 32
4.2 Bridge Protocol Data Unit - BPDU 33
4.2.1 BPDUS de configuração 34
4.2.2 BPDUS TCN (Notificação de Alteração da Topologia) 35
4.3 Estado das portas 35
4.4 Versões do STP 36
4.4.1 Per-VLAN Spanning Tree (PVST e PSVT+) 36
4.4.2 Rapid Spanning Tree (RSTP) 37
4.4.3 Multiple Spanning Tree (MSTP) 38
4.5 Spanning Tree e Sensibilidade ao Contexto 39
5 APLICAÇÃO PRÁTICA: implantação do STP na Rede Acadêmica da Universidade Federal do Maranhão 40
5.1 Modelos de rede e estrutura básica da Rede Acadêmica da Universidade Federal do Maranhão 40
5.2 Problemas encontrados 42
5.3 Solução proposta 44

5.3.1 Aplicação prática 44
5.4 Análise de resultados 47
5.5 Aplicativo ubíquo – STP HELPER 48
5.5.1 Diagrama de atividades 51
5.5.2 Diagrama de sequências 52
6 CONCLUSÃO 54
6.1 Trabalhos futuros 54
REFERÊNCIAS 56
APÊNDICE A 58
APÊNDICE B 60
APÊNDICE C 62

1 INTRODUÇÃO

A computação ubíqua é um termo dado à terceira era da computação. Com a evolução dos Sistemas de Informação Distribuídos, a era do PC, do inglês *Personal Computer*, caracterizada por dispositivos computacionais utilizados por uma só pessoa e dedicado a ela, se popularizou. A era da ubiquidade é caracterizada pela utilização de dispositivos móveis (smartphones, laptops, lousas digitais, etc.) com acesso a uma infraestrutura de rede (*e.g.* Internet) que compartilham recursos e oferecem serviços computacionais de maneira transparente ao usuário.

A convergência das tecnologias, coordenando-se entre si e fornecendo acesso a novos sistemas (de qualquer lugar e a todo o momento), diminuem a necessidade de interação do usuário proporcionando o desenvolvimento de aplicações mais adaptadas, gerando melhores resultados.

Para que isso funcione de forma eficiente, é necessária uma rede de computadores que garanta que todos os recursos de informação possam ser compartilhados rapidamente, de forma segura e confiável.

Como solução para tal, surge o *Spanning Tree Protocol*, que elimina *loops* e mantém a disponibilidade da rede através da utilização de apenas um caminho lógico.

O *Spanning Tree Protocol* é um protocolo desenvolvido pela DEC (*Digital Equipament Corporation*) e padronizado pelo IEEE 802.1d para resolver problemas circulares na redundância dos *links* de rede.

É importante que o administrador da rede esteja ciente das decisões que o STP está tomando para poder agir de forma pró ativa e eliminar os problemas da camada de enlace de rede. Portanto, este trabalho propõe a implantação do STP num setor universitário e o desenvolvimento de um aplicativo ubíquo sensível ao contexto que notifique o administrador sobre as decisões tomadas pelo protocolo para que elas não se perpetuem na rede.

Inicialmente, é apresentado um entendimento de contexto e sensibilidade ao contexto através de definições e propriedades, que norteiam e servem de base para entender por que o STP é um algoritmo sensível ao contexto ao ser aplicado em redes comutadas por pacotes.

O segundo capítulo deste trabalho trata especificamente de redes de computadores, suas classificações e modelos de referência, bem como a segmentação e o uso de LANs virtuais.

O terceiro capítulo contribui para esclarecer o funcionamento do STP, detalhando a operação do protocolo e expondo os problemas de uma rede sem STP, assim como seus objetivos e parâmetros necessários para ter-se uma topologia livre de *loops*.

O quarto capítulo trata da implantação do STP na rede acadêmica da Universidade Federal do Maranhão, especificamente na camada de acesso da rede, bem como a análise de resultados obtidos e também detalha a implementação de um aplicativo ubíquo de notificação de ciclos na camada de acesso ao usuário.

O último capítulo é a conclusão do trabalho, bem como os trabalhos futuros que podem ser realizados a partir desta pesquisa.

2 COMPUTAÇÃO UBÍQUA E SENSIBILIDADE AO CONTEXTO

Este capítulo apresenta um entendimento acerca dos conceitos necessários para o desenvolvimento deste trabalho. Inicia-se por um breve histórico da computação que conhecemos e que influenciaram o surgimento do termo ubiquidade, desde os grandes mainframes até a computação ubíqua, que conhecemos atualmente. Também serão apresentadas as definições e propriedades relacionadas à computação ubíqua, bem como os principais desafios existentes para o desenvolvimento de aplicações e algoritmos do gênero.

No intuito de esclarecer a proposta do tema, a segunda parte desse capítulo trás as definições e propriedades de Contexto e Sensibilidade ao Contexto de aplicações e algoritmos ubíquos sensíveis ao contexto. Inicia-se pela definição de contexto e suas categorias e, em seguida define-se o que é Sensibilidade ao Contexto e as principais técnicas e arquiteturas utilizadas.

Atualmente, o campo da computação ubíqua abrange uma ampla área de pesquisas e desenvolvimentos. Trabalhos de pesquisa referenciam comumente a Mark Weiser, que notoriamente cunhou o termo em 1991 enquanto trabalhava na empresa Xerox PARC.

Com a popularização dos computadores em decorrência do avanço tecnológico dos Sistemas de Informação Distribuídos e da diminuição do custo de produção de *hardware*, surgiram os populares PC's, do inglês *Personal Computer*. Um computador utilizado por uma só pessoa e dedicado a ela.

A transição entre a computação pessoal e a computação ubíqua foi marcada pelo advento da Internet. Ela pôs em contato elementos da computação centralizada com elementos da era PC, surgindo o paradigma cliente-servidor. De um lado, como clientes Web, estão os PC's e de outro lado, como servidores, estão os *mainframes.*

A consequência dessa interconexão de informações através da rede de computadores faz surgir um novo tipo de relacionamento, com vários computadores compartilhados por várias pessoas. Alguns desses computadores serão centenas daqueles que poderão ser acessados em consultas rápidas na Internet, outros em paredes, carros, roupas, etc., caracterizando a terceira era da computação: a computação ubíqua, cujo ponto de cruzamento ocorrerá entre 2005-2020 (WEISER, 1994).

A primeira vez que o termo computação ubíqua fora utilizado foi em 1988, por Mark Weiser, enquanto era diretor do Laboratório de Ciência da Computação da empresa Xerox PARC. Weiser vislumbrou um futuro onde as tecnologias da informação incorporaram-se em objetos do cotidiano para serem utilizados no apoio de atividades diárias: “As tecnologias mais profundas e duradouras são aquelas que desaparecem. Elas dissipam-se nas coisas do dia a dia até tornarem-se indistinguíveis.” (WEISER, 1991, p. 94).

Em um trecho do livro intitulado A Física do Futuro de Michio Kaku (2011), o autor apresenta como a ubiquidade moldará o mundo nos próximos cem anos:

> Computadores silenciosamente lendo nossos pensamentos serão capazes de realizar nossos desejos. Nós poderemos mover objetos apenas com a força da mente. Com o poder da nanotecnologia, poderemos pegar um objeto e transformá-lo em alguma coisa diferente. Embora coisas assim possam parecer inimaginavelmente avançadas, as sementes dessas tecnologias estão sendo plantadas nesse momento. É a ciência moderna e a tecnologia, e não mágicas e encantos que darão esse tipo de poder. (KAKU, 2011, p. 35).

Muito mais do que a computação móvel, essa tecnologia mudará fundamentalmente a natureza da computação, permitindo que objetos que encontramos na vida diária possam interagir com os usuários em ambos os mundos, físico e virtual. Para um bom entendimento da ubiquidade, faz-se necessário uma definição clara sobre o termo, bem como suas propriedades relevantes.

2.1 Definição de computação ubíqua

A ideia básica da computação ubíqua é de que ela move-se para fora das estações de trabalho e dos computadores pessoais e torna-se parte da nossa vida cotidiana, pervasiva[1]. O conceito foi introduzido ao vislumbrar novos sistemas e ambientes acrescidos de recursos computacionais capazes de prover serviços e informações quando e onde sejam desejados pelo usuário (*everywhere, everytime computing*, computação em toda parte, a todo tempo).

[1] A palavra “pervasiva” não consta nos dicionários de português por não possuir uma tradução literal, nem mesmo em dicionários especializados. Porém, na internet existem mais de 1.800 registros de buscas em páginas brasileiras. Pervasivo significa, então, aquilo que se infiltra, que penetra; espalhado, difuso.

Entretanto, a integração da tecnologia da informação em nossas vidas ainda está aquém da visão que Mark Weiser concluiu:

> Há mais informações disponíveis ao nosso alcance durante um passeio na floresta do que em qualquer sistema de computador, mas as pessoas acham uma caminhada entre as árvores relaxante e computadores frustrantes. Máquinas que se encaixam no ambiente humano vão fazer que o uso do computador seja mais refrescante do que um passeio na floresta. (WEISER, 1991, p. 104).

O que Weiser estava descrevendo seria nada mais nada menos do que a computação sem computadores. Neste contexto, temos a computação ubíqua, que não significa apenas “em todo lugar”, mas “em todas as coisas”. Objetos normais, como xícaras de café e pinturas nas paredes, podem ser considerados como locais para detecção e tratamento de informação, dotados de novas propriedades surpreendentes.

O melhor de tudo, as pessoas interagem com esses sistemas fluentemente e naturalmente, mal percebendo a computação a que estão envolvidos. A ubiquidade diz respeito a deixar nossas vidas mais simples através de ambientes digitais sensíveis, adaptáveis e respondíveis às necessidades humanas.

Esse paradigma leva em consideração que o ambiente computacional não deve impor restrições ao usuário para utilizá-lo (Weiser, 1991). Ao considerar o ambiente, assume-se que é necessária a existência do mesmo, de forma transparente, possuindo métodos intuitivos para o usuário interagir com a aplicação sem usar conhecimentos específicos.

2.2 Contexto e sensibilidade ao contexto

Esta seção apresenta as definições e propriedades de Contexto e Sensibilidade ao Contexto de aplicações ubíquas sensíveis ao contexto. Inicia-se pela definição de contexto e suas categorias e, em seguida define-se o que é Sensibilidade ao Contexto e as principais técnicas e arquiteturas utilizadas para o desenvolvimento de aplicações e implementação de algoritmos.

2.2.1 Contexto

A utilização de contexto é importante para aplicações interativas, onde o contexto do usuário se altera constantemente. A fim de facilitar o entendimento de como usar contexto e de como facilitar o desenvolvimento de aplicações sensíveis ao contexto, faz-se necessário entender o que é contexto.

O termo contexto é amplo em seu significado, então se torna necessário defini-lo previamente. Definições anteriores eram feitas através da enumeração de exemplos ou através da escolha de sinônimos.

No trabalho que introduziu o termo Sensibilidade ao Contexto, (Schilit *et al.* 1994) refere-se ao contexto como sendo a localização, as identidades das pessoas próximas e os objetos, bem como suas alterações. Semelhantemente, (Brown, 1997) definiu que contexto diz respeito à localização, as identidades das pessoas próximas e os objetos, assim como a hora do dia, a estação, a temperatura, etc.

Para (Ryan *et al.,* 1997, p. 269) "[...] contexto é a localização do usuário, o meio ambiente, a identidade e o tempo". Já (Dey, 1998) enumera contexto como sendo algumas características pessoais do usuário: estado emocional e foco de atenção, além das classificações tradicionais, como localização, data e hora, objetos e pessoas no ambiente.

Outras definições fornecem apenas sinônimos de contexto; por exemplo, ao se referir ao contexto como o ambiente ou a situação. Alguns autores o consideram como o ambiente do usuário e outros como o ambiente do aplicativo.

(Schilit *et al*., 1994, p. 85) afirma que "os aspectos mais relevantes para se definir o contexto são: onde você está, com quem está e quais os recursos estão disponíveis nas proximidades". Assim sendo, o contexto torna-se o ambiente de execução em constante mudança.

Porém, a definição que mais se aplica a este trabalho é a de Anind Dey:

> Contexto é toda a informação que pode ser utilizada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, lugar ou objeto que é considerado relevante para a interação entre um usuário e um aplicativo, incluindo o usuário e os próprios aplicativos. (DEY, 1998, p. 3-4).

Essa definição facilita a enumeração do contexto para um determinado cenário de aplicação. Se um trecho da informação pode ser usado para caracterizar a situação de um participante em uma interação, então essa informação é contexto.

Através dessa definição, o contexto passa ser indicado implicitamente ou explicitamente pelo usuário. Por exemplo, se a identidade do usuário é detectada implicitamente ou explicitamente por um login, a identidade do usuário ainda é contexto. (DEY & ABOWD, 1999).

A importância da categorização dos tipos de contexto auxilia os desenvolvedores a descobrir as partes mais funcionais que serão utéis para suas aplicações. As aplicações que são Sensíveis ao Contexto devem responder as perguntas: Quem? Onde? Quando e O quê? (o quê o usuário está fazendo) das entidades e usar essas informações a fim de determinar a situação corrente. (DEY & ABOWD, 1999, p. 4).

(Ryan *et al*. 1997) sugerem os seguintes tipos de contexto: localização, ambiente, identidade e tempo. Já (Schilit *et al*. 1994) lista que os aspectos importantes de contexto são a localização (onde você está), com quem você está e quais recursos existem nas proximidades.

Existem na prática alguns tipos de contexto que são mais importantes que os outros. São eles: localização, identidade, atividade e tempo. (DEY, 1999)

Essa categorização auxilia os desenvolvedores a escolher o contexto que será utilizado no desenvolvimento de suas aplicações, assim como os ajuda a estruturar o contexto já utilizado e até a procurar outro contexto relevante. Os quatro tipos de contexto primários definidos por Dey indicam os tipos de informações necessárias para a caracterização de uma situação e sua utilização como índices, proporcionando um caminho para a melhor utilização e organização do contexto.

Tendo definido contexto e suas categorias, é necessário pensar em como usar esse contexto. Na próxima seção será definido Sensibilidade ao Contexto, e será fornecido uma categorização dos recursos de aplicações Sensíveis ao Contexto.

2.2.2 Sensibilidade ao contexto

O primeiro trabalho que trouxe o termo Sensibilidade ao Contexto à discussão foi o de (Schilit & Teimer, 1994), como sendo um software que se adapta de acordo com o local de utilização, com o conjunto de pessoas próximas e objetos, bem como a alteração ao longo do tempo que os objetos podem sofrer.

No decorrer das pesquisas, diversos autores utilizaram o termo sensibilidade ao contexto como sinônimo de outros termos: adaptativa, reativa, ágil, situada, etc. (DEY & ABOWD, 1999). Essas definições dividiram a sensibilidade ao contexto em duas categorias: utilização e adaptação de contexto.

Ao definir aplicações Sensíveis ao Contexto, (Brown, 1999, p. 18) diz que são “aplicativos que fornecem informações automaticamente ou tomam medidas de acordo com o contexto do usuário”. Aplicações Sensíveis ao Contexto fornecem serviços computacionais de maneira transparente ao usuário, ou seja, sem a sua intervenção, para auxiliá-lo em atividades da vida cotidiana.

Já (Dey *et al.*, 1999) afirma que um sistema é Sensível ao Contexto se ele usa o contexto para fornecer informações relevantes e/ou serviços para o usuário, onde a relevância depende da tarefa do mesmo. Essa é uma definição mais geral, pois não exige que os sistemas detectem, interpretem e respondam ao contexto. É necessário apenas que responda, permitindo assim, que outras entidades computacionais possam realizar a detecção e interpretação do contexto.

Apenas a definição de Sensibilidade ao Contexto não é suficiente para gerar um entendimento de como as aplicações desse gêreno operam. Também é necessário categorizar os recursos para que estes sejam adaptáveis à aplicação e o próximo tópico abordará este assunto.

2.3 Características de aplicações sensíveis ao contexto

(Pascoe, 1998, p. 92) introduz um conjunto de quatro recursos que as aplicações Sensíveis ao Contexto suportam. Esta categorização ajuda a definir a estrutura das aplicações sensíveis ao contexto:

1. Sensoriamento de contexto: a aplicação detecta o contexto e simplesmente apresenta-o ao usuário, aumentando o sensoriamento do mesmo.
2. Adaptação de contexto: a aplicação utiliza o contexto para adaptar o seu comportamento em vez de fornecer uma interface uniforme em todas as situações.

3. Descoberta de recursos: a aplicação pode localizar e usar os recursos que compartilhem partes ou a totalidade do contexto a qual está inserida.
4. “Aumento” contextual: a aplicação amplia o ambiente através de informações adicionais, associando dados digitais com o contexto atual. As notas que um usuário faz em informações explícitas são aumentadas com a informações contextuais (localização, tempo, etc.).

Já (Dey et. al, 1999, p. 8) propõe uma lista de recursos sensíveis ao contexto que uma aplicação pode suportar, divididas em três categorias principais:

1. Apresentação de informações e serviços a um usuário;
2. Execução automática de um serviço;
3. Marcação do contexto para posterior recuperação de informações.

O contexto do usuário é muito dinânico. Ao utilizar aplicações com essas características, o usuário tem muito a ganhar através do uso eficaz de detectção implícita do contexto. Isso permite que o comportamento de um aplicativo possa ser personalizado de acordo com a situação atual do usuário.

Tais definições e categorizações auxiliam os pesquisadores e desenvolvedores a entender os limites da computação “ciente de contexto” e a selecionar o contexto de uso, como por exemplo, a estrutura das aplicações, definindo as funcionalidades necessárias para a implementação do software.

2.4 Técnicas e arquiteturas para o desenvolvimento de aplicações ubíquas sensíveis ao contexto

Um grande problema que dificulta as aplicações de fazerem um melhor uso do contexto tem sido a falta de suporte uniforme para o desenvolvimento e execução destes tipos de aplicações. Em sua maioria, essas aplicações foram desenvolvidas em *ad hoc* fortemente influenciado pela tecnologia subjacente usada para adquirir o contexto. Logo, resulta na falta de generalidade, fazendo com que cada aplicação nova seja construída do zero.

(Dey *et al*. 1999) enumeraram um processo de criação (*design process*) para o desenvolvimento de aplicações sensíveis ao contexto: “[...] acreditamos que a

dificuldade de construção de aplicações sensíveis ao contexto tem sido a falta de apoio a nível de infra-estrutura para este processo de criação.” (DEY, 1999, p. 8)

O processo de criação (adaptado por Dey, 1998) é como se segue:

1. Especificação: é a parte que especifica o problema a ser abordado através de uma solução de alto nível. Determina qual contexto é necessário para a aplicação.
2. Aquisição: determina os sensores ou dispositivos que fornecerão contexto e interpreta o contexto, se for o caso.
3. Entrega (*delivery*): fornece os métodos necessários para a entrega e contexto para uma ou mais aplicações remotas.
4. Recepção: recebe ou solicita o contexto e analisa as informações para determinar sua utilidade na aplicação.
5. Ação: analisa o contexto tratando-o como uma variável independente com combinando-o com outras informações recolhidas e escolhe o comportamento da aplicação, que se torna sensível ao contexto.

Para essas quatro primeiras etapas, existe um nível de infraestrutura de apoio, até mesmo para as aplicações sensíveis ao contexto mais triviais. Segundo (Dey, 1999), estes são os passos iniciais que tornam as aplicações deste gênero mais complexas e demoradas. Por isso, o conhecimento das atividades essenciais é fundamental para a construção de uma parte do software e incluem a compreensão do problema e a modelagem de uma solução para esse tal.

É uma árdua tarefa desenvolver aplicações que trabalham com uma grande variedade de informações que necessitam ser capturadas. Traduzir o contexto, manipular, comparar e apresentar ao usuário em um formato significativo demanda tempo e processamento. Por isso, os desenvolvedores devem escolher as técnicas mais fáceis de implementação, geralmente ditadas pelos sensores que estão sendo utilizados.

(Dey, 1999, p. 21) diz que isso vem em detrimento da generalização e implementação *ad hoc*[2]. Essa tendência de aplicativos que usam sensores emergiu contra o progresso da computação sensível ao contexto e possui quatro problemas:

1. Existência de poucas aplicações sensíveis ao contexto;
2. Pouca variedade de sensores;

[2] *Ad hoc* é uma rede de área local (LAN) que dispensa o uso de pontos de acesso, não possuindo nós. Nela, os dispositivos comunicam-se diretamente entre si, funcionando como roteadores.

3. Pouca variedade de tipos de contexto;
4. Incapacidade da evolução das aplicações.

Devido à maneira de desenvolvimento das aplicações SC, os desenvolvedores colocam pouco esforço para fazer que seus sensores sejam reutilizados por outras pessoas, o que resulta na falta de aplicativos básicos na área. Como muitos sensores não foram projetados para serem reutilizados torna-se difícil de integrá-los com aplicações existentes que não usam contexto, além de dificultar a adição dos sensores em aplicações já existentes.

3 REDES DE COMPUTADORES

As redes atuais são caracterizadas pelo seu objetivo e variedade de alternativas tecnológicas que as compõe quanto pelos sistemas de comunicação necessários para sua confiabilidade e capacidade de transmissão de dados. Implantar certa topologia de rede para prover suporte a um dado conjunto de aplicações é complexo, já que cada caso possui suas especificidades.

Uma rede de computadores visa garantir que todos os recursos de informação sejam compartilhados rapidamente, de forma confiável e segura. Para isso, a rede deve garantir que os meios de transmissão sejam eficientes, que os mecanismos garantam o transporte de informações entre os seus elementos, além de prover confiabilidade entre os vários sistemas de informação.

3.1 LANs Virtuais

Sempre que uma rede de comunicação de dados é estabelecida, é necessário que ela possua segurança para informações dos usuários e dos sistemas da rede. À medida que a rede cresce, essa tarefa torna-se mais complexa e é dever do gerente mantê-la sempre operante.

Neste tópico é apresentado a LAN Virtual – ou simplesmente VLAN – que melhora as questões relativas à segmentação da rede em níveis de segurança, escalabilidade e organização lógica, facilitando sua administração. Estarão sendo estudados a importância da segmentação da rede e o padrão IEEE 802.1q, que define o estabelecimento de VLANs.

3.1.1 Segmentação de redes

A segmentação de uma rede objetiva aumentar a largura de banda para os usuários sem tornar necessário a substituição de equipamentos. Através dela, pretende-se quebrar uma rede em redes menores com equipamentos apropriados. A utilização de hubs, switches, pontos lógicos de acesso, etc., precisa ser analisada

com cuidado pelo administrador da rede, pois podem apresentar um alto custo ou prejudicar o desempenho da rede.

Alguns motivos levam uma rede a ser segmentada:

- Distância de conexões: quando há uma grande distância geográfica, deseja-se assegurar que tráfego desnecessário não consuma a largura de banda;
- Segurança: quando há a necessidade de limitar acessos externos à rede e vice-versa;
- Limitações da largura de banda: é comum que algumas estações de trabalho ou servidores (como uma rede SAN[3], por exemplo) consumam grande parte da largura de banda;
- Limitações topológicas: numa rede em expansão, é comumente necessário adicionar mais nós à rede, mas há limitações de distância ou o endereçamento reservado no segmento já foi preenchido.

3.1.2 Definição de VLAN

Trata-se de um agrupamento lógico de estações, serviços e dispositivos de rede não restritos a um segmento físico de rede local. A Figura 1 ilustra esse aspecto e a compara com a LAN.

[3] *Storage Area Network* – SAN é uma rede ou sub-rede de alta velocidade para interconectar, de forma compartilhada, as *storages* com múltiplos servidores.

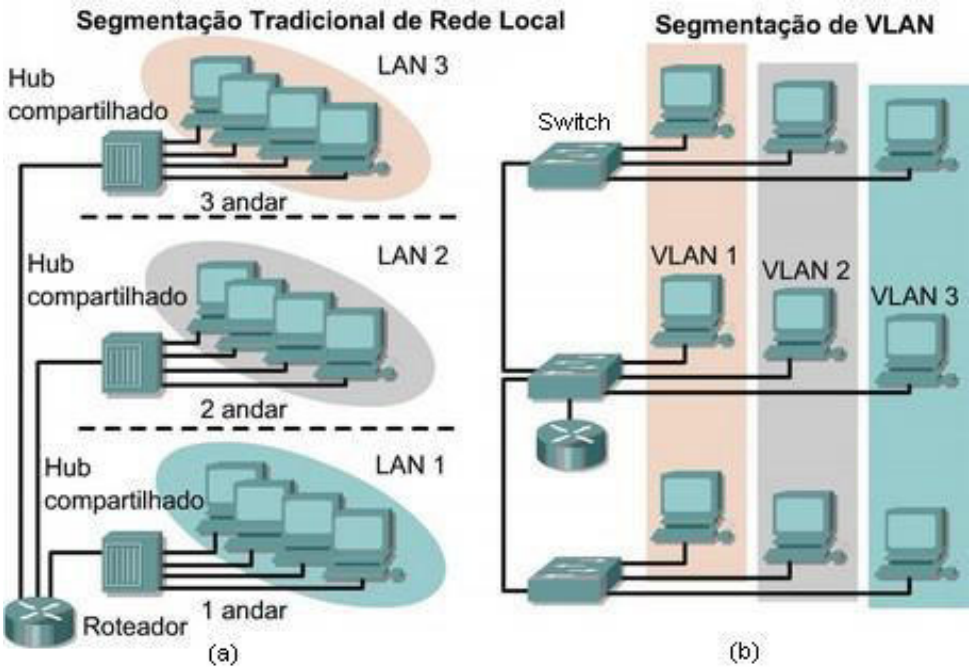


Figura 1. (a) LAN e (b) VLAN. (PRIOR, 2012, p. 2)

Em uma LAN, os terminais são conectados a um concentrador e fazem parte de uma mesma LAN, havendo uma limitação física. Numa VLAN, o concentrador pode atender diversas VLANs dependendo da identificação do segmento empregado.

VLANs são administradas separadamente dentro de uma mesma rede física. Elas são domínios que permitem o controle de *broadcast*[4], *multicast*[5] e *unicast*[6] dentro de um dispositivo de camada 2 (enlace).

Como exemplo, podemos tomar que vários computadores são usados em uma sala de conferências X e alguns outros em uma conferência Y. Os sistemas da conferência X podem comunicar-se uns com os outros, mas não estabelecem comunicação com os da conferência Y.

A criação de uma VLAN permite que os sistemas das conferências X e Y comuniquem-se entre si, mesmo estando em sub-redes físicas separadas. Eles podem ser geridos por um único dispositivo, mas ignoram os sistemas que não compartilham o mesmo identificador da VLAN.

[4] Encaminhamento um para todos.
[5] Encaminhamento um para vários ou vários para vários sem duplicação de mensagem. A mensagem duplica-se somente se o link para o destinatário divide-se em duas direções.
[6] Encaminhamento um para um (ponto a ponto).

3.1.3 Benefícios

Uma rede pode apresentar atraso no tempo de resposta de pacotes, devido a falhas nas conexões dos cabos, defeito em placas de rede ou até problemas nas aplicações. Através da segmentação da rede, a difusão de pacotes nos subdomínios reduz o tráfego de pacotes, proporcionando controle do tráfego de broadcast.

O gerenciamento é, sem dúvidas, um dos principais benefícios proporcionados pelo uso de VLAN's. Uma simples troca de VLAN no concentrador pode definir se a interface pertence a um segmento ou a um novo segmento, sem necessitar de intervenção presencial, como uma troca de cabeamento por exemplo.

O desempenho da rede é otimizado, visto que a segmentação reduz o tamanho do domínio de colisão e divide a largura de banda total da rede.

Hosts em VLAN's diferentes se comunicam somente se houver um roteador configurado para encaminhar pacotes entre as redes. Nesse ponto, podem-se configurar também algumas regras (políticas de acesso, segurança e prioridade (HAFFERMANN, 2009)) que possibilitam restrições de acesso entre os segmentos.

3.2 Padrão IEEE 802.1Q

O padrão IEEE 802.1, da hierarquia do IEEE 802, define os padrões de gerenciamento e interconectividade da rede e a interação deles com o modelo de referência OSI. O agrupamento de RFCs que tratam as VLAN's é o 802.1Q.

Essa especificação dispõe um método para inserir um pacote a uma VLAN. É adicionado um pacote de 4 bytes, onde 16 primeiros bits tratam do tipo de transmissão (ETYPE) utilizado (*Ethernet*, *Fast Ethernet* e *Gigabit Ethernet*). Os próximos 3 bits são a prioridade do quadro, 1 bit é nulo e os últimos 12 bits determinam o ID da VLAN do qual o quadro pertence. A Figura 2 representa um quadro no padrão 802.1Q.

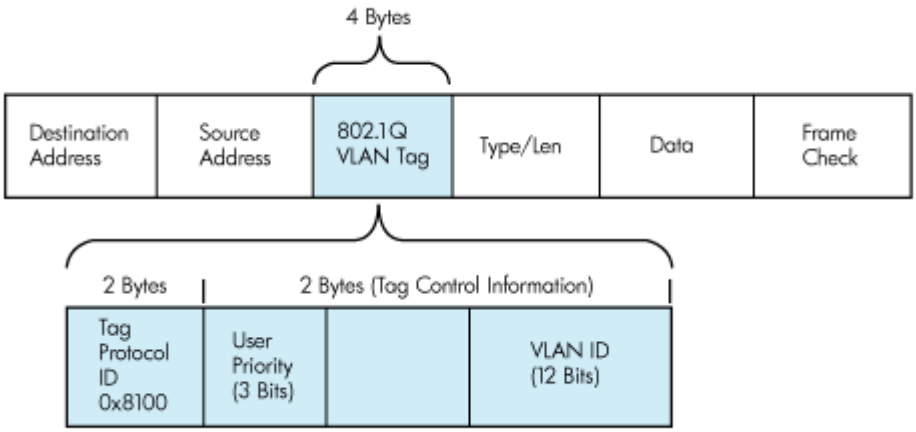


Figura 2. Quadro 802.1Q. (IEEE 802.1Q)

Os quadros 802.1Q podem ser classificados de três formas:

- *Untagged*, onde os pacotes não possuem marcação alguma (sem VLAN *tag*), portanto são tratados como pacotes padrões de Ethernet;
- *Priority Tagged Frame*, onde os três bits de prioridade são utilizados para tratar a prioridade dos pacotes. O padrão 802.1p define 8 níveis a serem obedecidos em relação a precedência dos pacotes (HAFFERMANN, 2009);
- *Tagged*, onde o campo VLAN ID determina a VLAN a qual o pacote pertence.

Existem três tipos de conexões para dispositivos que suportam ou não o padrão 802.1Q.

No tipo de conexão *Trunk* (tronco), as duas extremidades necessitam ter suporte ao padrão e os dispositivos devem reconhecer o campo VLAN ID dos pacotes. Isso por que as interfaces configuradas como *trunk* são geralmente utilizadas para fazer o tronco de uma conexão, como o *backbone* da rede, portanto, os pacotes que passarem por essas interfaces necessitam obrigatoriamente da marcação da VLAN (*tagged*).

O modo de conexão *access* (acesso) é o mais comum numa rede. Os quadros chegam nesse tipo de interface sem marcação VLAN (*untagged*), portanto uma estação que está conectada a uma interface *access* já recebe pacotes de broadcast da rede.

O terceiro modo de conexão é o *Hybrid* (híbrido), que trata da combinação dos dois anteriores. Várias VLANs podem passar no *tag*, mas no enlace de dados, apenas uma VLAN é definida.

Apesar de VLAN não ser o assunto priori deste trabalho, torna-se necessário uma descrição da mesma para que seja possível, no próximo capítulo, relacioná-la com o *Spanning Tree Protocol*.

4 SPANNING TREE PROTOCOL

A maioria das LANs atuais possuem diversos switches com o proposito de segmentar a rede e minimizar a quantidade de dados que trafegam internamente nela, tornando-a mais restritiva. Com o intuito de aumentar o nível de disponibilidade da rede, surgem os links redundantes.

Porém, a redundância da rede permite que quadros possam circular indevidamente, ocasionando perda de desempenho e *loops*. Como solução para tal, surge o 802.1D *Spanning Tree Protocol* (STP) entre switches da rede, operando na camada de enlace, objetivando criar um único caminho lógico, sem ciclos.

O protocolo foi criado pela (Radia Perlman, 1985), na época, engenheira da Sun Microsystems. Ela desenvolveu um método pelo qual as *bridges* na camada de enlace matem-se operando redundantemente e sem *loop*. Pense no *Spanning Tree* como uma árvore onde a *bridge* mantem um *path* na memória para transmissão otimizada de dados e tolerante a falhas.

A primeira versão original do STP foi criada pela DEC (*Digital Equipment Corporation*) e, depois de algum tempo, o IEEE (*Institute of Eletrical and Eletronics Engineers*) criou sua própria versão, o 802.1D STP.

O STP foi desenvolvido numa época em que a recuperação da conectividade após uma interrupção dentro de mais ou menos um minuto era considerado um desempenho adequado. Atualmente, as interrupções são resolvidas por soluções roteadas em que protocolos como o OSPF (*Open Shortest Path First*) fornecem um caminho alternativo em menos tempo.

> [...] o algoritmo STP cria uma árvore de espalhamento de interfaces que estão ou no estado *fowarding* (transmitindo pacotes) ou no estado *blocking* [...] se uma interface não apresentar motivos para estar nesse estado, ela é colocada em outro estado [...] (ODOM, 2008, p. 169-170).

A ideia por trás de uma topologia de *Spanning Tree* é que as bridges podem descobrir um subconjunto dentro da topologia corrente livre de *loop*, através da eleição de um switch principal (*root bridge*) e da seleção de uma única porta nos outros switches que enviarão os quadros de volta para o switch principal. O protocolo também garante que há conectividade suficiente para alcançar qualquer parte da rede, abrangendo toda a LAN.

Para melhor entendimento do protocolo, é necessário entender os problemas que levaram ao surgimento do mesmo.

4.1 Loops

Manter a redundância na rede assegura maior disponibilidade de acesso. Porém, se mal planejada, resulta em sérios problemas para o desempenho visto que os quadros podem ser enviados para todos os links ao mesmo tempo. O *loop* causa *broadcast storms* e corrupção na tabela do switch.

> Quando um switch recebe um *broadcast*, ele o repete em cada porta (exceto naquela em que foi recebido). Em um ambiente com *loop*, os *broadcasts* podem ser repetidos infinitamente. (DONAHUE, 2007, p. 67-68).

Em caso de redundância numa topologia que não utilize o STP, o processo de encaminhamento infinito de *broadcast* só será interrompido quando o *loop* for desfeito, através da remoção dos cabos ou desligamento do equipamento. Esse problema ocasiona na indisponibilidade dos links da rede. O cenário abaixo, ilustrado pela Figura 3, exemplifica esse problema.

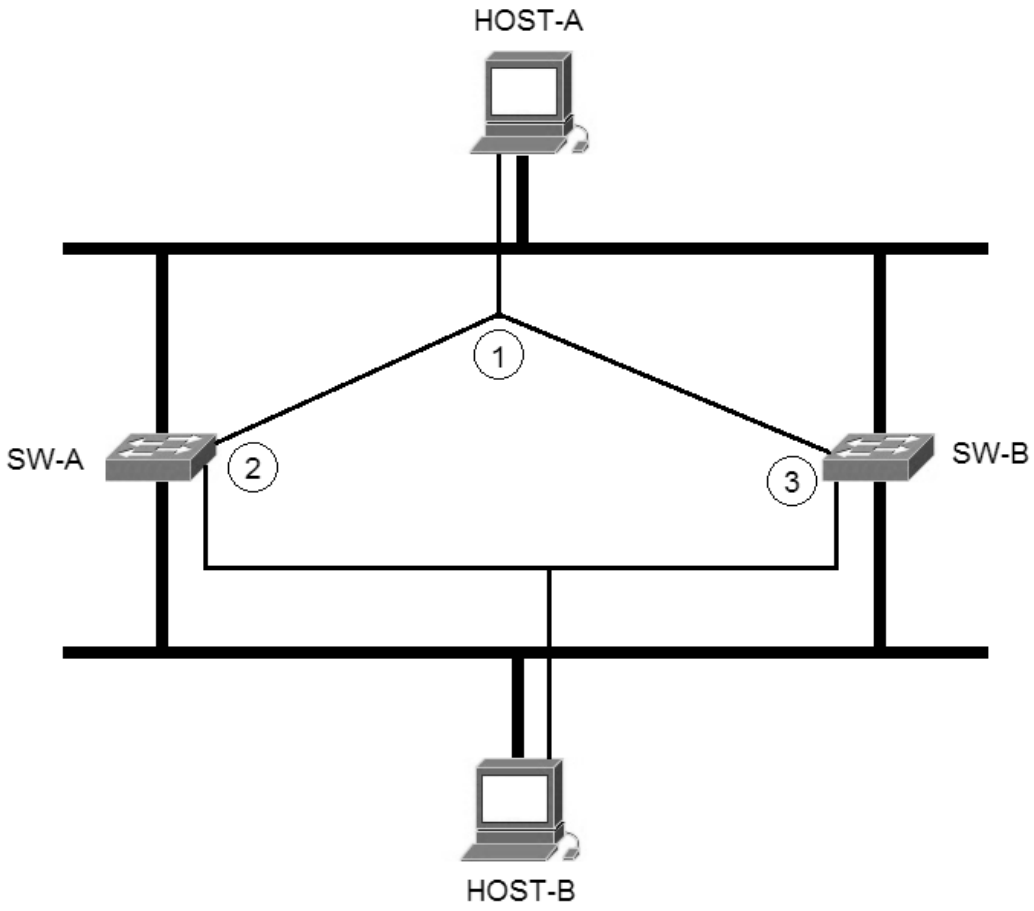


Figura 3. Broadcast storm. Fonte: autor.

Na situação 1, o HOST-A envia um quadro para o endereço de *broadcast*, ambos os switches, SW-A e SW-B, o recebem. Na situação 2, o switch SW-A recebe o quadro em uma interface de entrada, realiza a cópia e o envia numa interface de saída, ligada ao SW-B. Na situação 3, o switch SW-B recebe o quadro e o reencaminha de volta para o SW-A.

Essa operação continua repetindo-se no sentido anti-horário e horário, pois não foi descrito nenhum processo de encaminhamento do primeiro quadro recebido pelo SW-A. O *loop* ocorre em ambas às direções e, com o passar do tempo, a quantidade de cópias do quadro aumenta, resultando na transmissão de múltiplos quadros.

O *loop* também ocasiona a corrupção na tabela do switch, pois há a dupla adição de endereços MACs na tabela devido aos links redundantes e múltiplos loops, um dentro do outro, em ambas as direções. O equipamento acaba confundindo a localização dos dispositivos, pois recebe os quadros em mais de uma interface.

Segundo (Odom, 2008), os *loops* são evitados através do uso do protocolo *Spanning Tree*, que proporciona a criação de uma rede em que haja apenas um único caminho lógico ativo entre os switches.

Para isso, os equipamentos dentro de um domínio STP precisam ter conhecimento da topologia e receber informações sobre as alterações da mesma. Isso é possível graças ao BPDU, *Bridge Protocol Data Unit*.

4.2 Bridge Protocol Data Unit - BPDU

Quando um dispositivo é ligado pela primeira vez a uma porta do switch que está dentro de um domínio STP, ele não transmite os quadros de imediato. Ao invés disso, ele passa por uma série de estados enquanto os BPDUs processam e determinam a topologia da rede.

Existem dois tipos básicos de quadros BPDUs na especificação STP original: os de configuração e os de TCN (*Topology Change Notification*). O *Rapid Spanning Tree* (RSTP) utiliza um BPDU específico.

4.2.1 BPDUs de configuração

São quadros específicos que possuem informações relevantes ao algoritmo. Os equipamentos da rede que estão operando dentro do domínio STP criam esses BPDUs e os transmitem a fim de estabelecer a estrutura topológica livre de *loops* através da comparação dos parâmetros contidos nesses quadros.

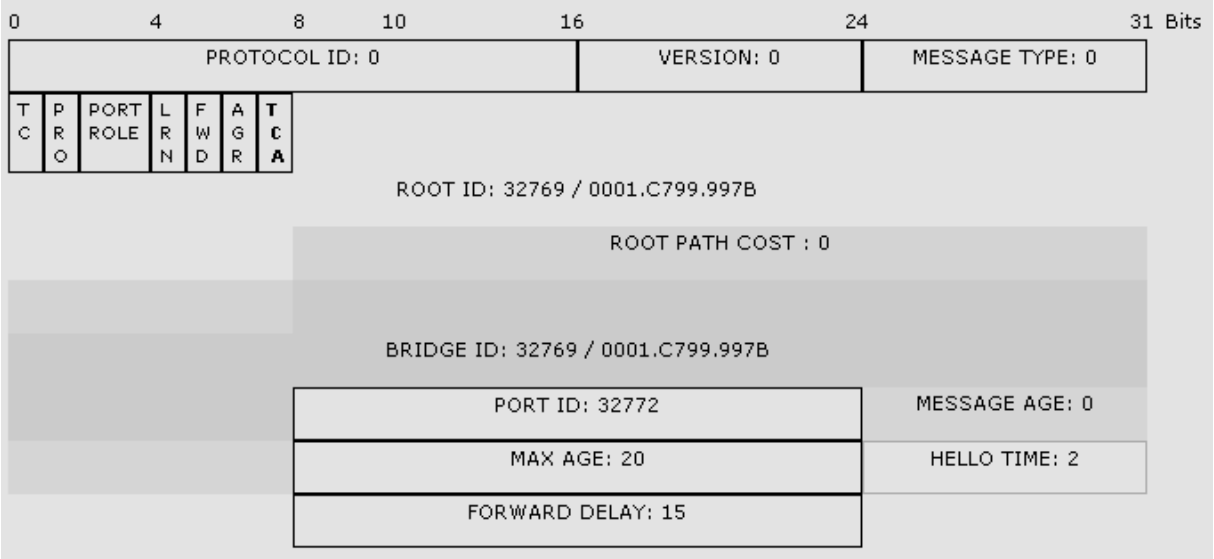


Figura 4. BPDU de configuração. (IEEE 802.1d)

Cada campo do quadro é descrito abaixo:

- *Protocol ID*: 2 bytes. É o algoritmo STP e o protocolo que está sendo usado;
- *Version*: 1 byte. Versão do protocolo;
- *Message Type*: 1 byte. Define o tipo de BPDU (de configuração ou TCN);
- *Flags*: São índices que indicam a mudança de topologia da rede:
    - *Topology Change Notification* (TCN): reconhece a mudança da topologia;
    - *Topology Change Notification Acknowledgment* (TCA): relata ao switch que os dados contidos no BPDU corrente foram lidos e salvos.
- *Root ID*: 8 bytes. Possui o ID do switch raiz. É através dele que os outros switchs identificam o switch raiz;
- *Root Path Cost*: 4 bytes. É o custo do caminho acumulado para o switch raiz;
- *Bridge ID*: 4 bytes. É a identificação do switch dentro do domínio STP;
- *Port ID*: 2 bytes. Identifica cada interface do switch através de um único valor.

- *Max Age*: 2 bytes. É a idade do quadro BPDU desde que ele foi gerado. Se houver perda de comunicação com o switch root, a idade é aumentada a fim identificar que são dados velhos, i.e. , é o tempo que o switch leva para concluir a modificação da topologia;
- *Forward Delay*: 2 bytes. A porta pode passar por alguns estados antes de estar encaminhando ou não os pacotes e este é o tempo mínimo em que ela permanece na identificação de cada estado;
- *Message Age*: 2 bytes. Tempo entre o anúncio de BPDUs.
- *Hello Time*: 2 segundos. É o tempo em que o equipamento lança o quadro BPDU para os outros switches.

4.2.2 BPDUs TCN (Notificação de Alteração da Topologia)

Simplesmente notifica a alteração da topologia. Esse BPDU informa os outros switches sobre as alterações ocorridas nas interfaces e são anunciados na rede através de um switch que não é a raiz.

Após a recepção de um BPDU TCN, o switch raiz irá definir um *flag* de *Topology Change* em suas BPDUs normais. Essa *flag* é propagada para todos os outros switches a fim de instruí-los para alterarem rapidamente suas entradas da tabela de encaminhamento.

Os BPDUs TCN são identificados pelo campo *Type* do BPDU de configuração, onde 0x00 identifica os BPDUs de configuração e 0x80 os de TCN.

4.3 Estado das portas

Para participar de um domínio STP, cada interface do switch passa por vários estados diferentes. A priori, ao conectar a porta na rede, ela está desabilitada e progride até chegar a um estado onde os dados possam trafegar. São eles:

- *Disabled*: Nesse estado, a porta é completamente não funcional e, portanto, não recebe ou transmite qualquer tipo de dado;

- *Blocking*: A porta nesse estado não é nem a porta raiz e nem uma porta de encaminhamento, mas é reconhecida como uma porta alternativa pela raiz. Ela não aprende endereços nem encaminha quadros ou BPDUs, mas pode ouvir BPDUs, visto que pode ser desbloqueada alguma hora;
- *Listening*: Nesse estado, a porta está sendo preparada para atividade, saindo de um estado de bloqueio existente. Ela ainda não aprende ou encaminha endereços, mas recebe e envia BPDUs.
- *Learning*: A porta nesse estado vai ser ativada para o encaminhamento de pacotes, mas precisa esperar que o *forwarding delay* expire (15 segundos, tipicamente). Isso permite que a porta adicione entradas em sua base de modo que não inunde outras portas enquanto não estiver no estado de *forwarding;*
- *Forwarding*: A porta nesse estado fica funcionando como qualquer outra porta do switch, filtrando e encaminhando quadros.

4.4 Versões do STP

O STP tradicional opera em uma instância única do protocolo, englobando todas as VLANs que trafegam na rede. Isso é denominado de CST (*Common Spanning Tree*) pelo IEEE 802.1Q. Porém existem outras versões do STP, que podem adaptar-se a cada LAN, podendo proporcionar um melhor balanceamento de carga da rede.

4.4.1 Per-VLAN Spanning Tree (PVST e PVST+)

O PVST é propriedade da Cisco Systems e possui maior flexibilidade que o CST. Cada VLAN que participa de um PVST precisa estar em um domínio STP (STPD) separado e o número identificador da VLAN (VLAN ID) necessita ser o mesmo do identificador STPD (STPD ID).

Essa versão permite que o tronco da VLAN seja encaminhado para algumas interfaces enquanto o bloqueia para outras. Desse modo, cada VLAN ou cada tronco é tratado como uma rede separada, efetuando o *load balance* (equilíbrio

na distribuição do tráfego) na camada de enlace, transmitindo algumas VLANs em um tronco e outras em outro, de acordo com a configuração realizada no STPD e livre de *loops*.

Pelo fato de ser proprietário, o PVST demanda o uso do ISL, um cabeçalho adicional no quadro Ethernet que identifica a VLAN que o quadro pertence, impedindo a comunicação entre o PVST e o CST (802.1Q). A Figura 5 ilustra um cabeçalho ISL.

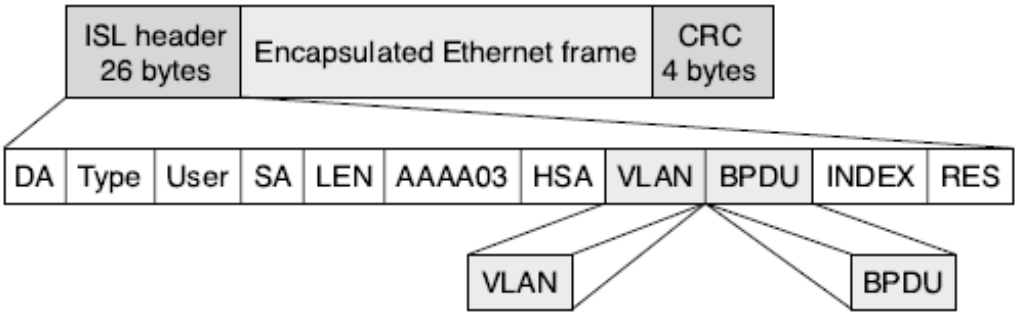


Figura 5. Cabeçalho ISL. (CISCO, 2011)

O PVST+ (*Per-VLAN Spanning Tree Plus*) foi desenvolvido para fornecer as mesmas funcionalidades do PVST e prover interoperabilidade entre o CST e o PVST, executando os dois protocolos, ISL e 802.1Q.

4.4.2 Rapid Spanning Tree (RSTP)

O IEEE introduziu o *Rapid Spanning Tree Protocol* em 2001 como 802.1w. O RSTP fornece mais rapidamente a convergência da árvore após uma mudança na topologia da rede, através da análise de novos comportamentos e regras.

Enquanto o CST pode levar de 30 a 50 segundos para responder a uma alteração de topologia, o RSTP é capaz de responder a 3 * *Hello times* (onde o padrão de uma mensagem *hello* é de 2 segundos) ou em poucos milissegundos em caso de falhas no link.

As portas dentro de um RSTP possuem funções diferenciadas, porém similares ao padrão, podendo ser: *root* (porta principal), *designated* (porta de encaminhamento para cada segmento), *alternate* (porta de caminho alternativo para

a *root*), *backup* (faz parte de um caminho redundante para um segmento onde outra porta *bridge* já está conectada), *disabled* (configurada manualmente pelo administrador de rede).

Nessa versão, o número de estados que uma porta pode passar é reduzido para três em vez de cinco do CST, são eles:

- *Discarding*: nenhum pacote é enviado por essa porta;
- *Learning*: a porta nesse estado ainda não encaminha pacotes, mas pode povoar a tabela de endereços MAC;
- *Forwarding*: a porta nesse estado está operacional, enviando e recebendo quadros.

O RSTP foi projetado para ser compatível com o 802.1q STP.

4.4.3 Multiple Spanning Tree (MSTP)

O *Multiple Spanning Tree Protocol*, definido pelo IEEE 802.1s, objetiva processar múltiplas instâncias RSTP, reduzindo ainda mais o tempo de convergência da árvore. O MSTP cria árvores separadas para cada grupo de VLAN.

O MSTP permite a formação de regiões MST que podem executar várias instâncias do MST (MSTI). Várias regiões e outras *bridges* STP são interligadas através de uma única árvore de expansão comum (CST).

Essa versão inclui todas as informações da *spanning tree* em um formato único de BPDU, reduzindo a quantidade de BPDUs necessárias em uma LAN e garantindo compatibilidade com o RSTP.

Cada instância MSTP – pertencentes a sua região – é conectada por uma *spanning tree* comum (CST), provendo comunicação para diferentes regiões. Regiões podem ser vistas como *bridges* que fazem parte de uma *spanning tree* (comum).

Cada versão depende do porte da rede. Em uma rede pequena, que não possui muitas *bridges*, o STP é uma solução viável. O RSTP é um protocolo que pode ser utilizado nas mesmas situações que o STP e em situações em que a rede seja de médio porte, tendo um menor tempo de convergência do que seu antecessor. Caso a rede seja demasiadamente grande, com várias VLANs, o MSTP é uma solução adequada.

4.5 Spanning Tree e sensibilidade ao contexto

Como visto no primeiro capítulo deste trabalho, para que uma aplicação seja considerada sensível ao contexto, é necessário que haja uma apresentação de informações ou serviços a um usuário; que esse serviço possua execução automática e que haja a marcação do contexto para posterior recuperação de informações (DEY, 1999).

Tomando por base o *Spanning Tree Protocol*, o contexto do domínio da rede ao qual está inserido e realizando uma análise dos requisitos que tornam uma aplicação ou sistema sensível ao contexto, é possível inferir que o protocolo acima citado é sensível ao contexto.

Trata-se se um protocolo que opera transparentemente para o usuário, diluído na topologia da rede, presente e invisível. É sensível ao contexto, pois através dos quadros BPDUs, consegue repassar informações referentes à mudança da topologia (contexto) da rede, a fim de estabelecer uma estrutura livre de *loops*.

Por se autoconfigurar, o STP age com base da necessidade implícita de haver disponibilidade do link, processando falhas sem necessidade de interação alguma.

O próximo capítulo apresenta a implantação do algoritmo *Spanning Tree Protocol* na rede acadêmica da Universidade Federal do Maranhão. O protocolo será aplicado na camada de acesso da rede, que age diretamente com o usuário final e é passiva de ocorrer *loops*.

5 APLICAÇÃO PRÁTICA: IMPLANTAÇÃO DO STP NA REDE ACADÊMICA DA UNIVERSIDADE FEDERAL DO MARANHÃO

Este capítulo apresenta o funcionamento estrutural básico físico e lógico da rede de dados da Universidade Federal do Maranhão. Primeiramente será apresentado o modelo de funcionamento da rede, detalhando cada camada. Logo, é apresentada a topologia e os objetivos da mesma.

Por fim, é apresentada a parte prática do trabalho: a implantação do *Spanning Tree Protocol* na camada de acesso da rede universitária.

5.1 Modelo de rede e estrutura básica da rede acadêmica da Universidade Federal do Maranhão

A rede acadêmica da UFMA segue o modelo de rede hierárquica, envolvendo a divisão da rede em camadas discretas, onde cada qual fornece funções específicas que definem sua função dentro da rede geral. Ao separarem-se as várias funções existentes dentro da rede, o desenho torna-se modular, facilitando a escalabilidade e o desempenho.

O modelo hierárquico típico é dividido em até três camadas: acesso, distribuição e núcleo (core). Na Figura 6 é exibido um exemplo de rede hierárquica com três camadas:

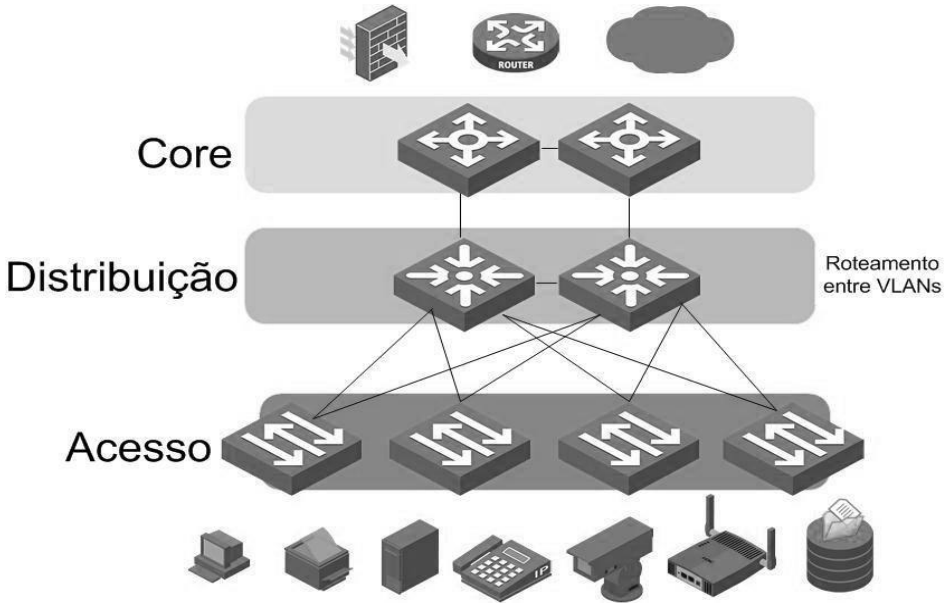


Figura 6. Rede hierárquica. Fonte: autor.

A camada de acesso é responsável pela interface dos dispositivos finais, como PC's, impressoras, telefones IP's, etc., que fornecem acesso ao restante da rede, é nela que os *loops* ocasionados pelos usuários acontecem.

Na camada de acesso podem estar roteadores, switches, hubs, bridges e pontos de acesso *wireless* – ou AP's (*access points*). Seu principal objetivo é fornecer um meio de conectar os dispositivos à rede e controlar quais tem permissão de se comunicar com a mesma.

A camada de distribuição agrega os dados recebidos dos switches da camada de acesso antes de serem transmitidos para a camada de núcleo, para que haja o roteamento até seu destino final.

Ela também é responsável pelo controle do fluxo do tráfego da rede usando políticas e determina domínios de broadcast, realizando funções de roteamento entre redes locais virtuais (VLANs, que serão estudadas mais adiante) definidas na camada de acesso.

A camada de núcleo – ou core – da rede hierárquica é o *backbone*[1] de alta velocidade das redes interconectadas. Como o core é essencial à interconectividade entre os dispositivos da camada de distribuição, é importante que seja altamente disponível e redundante.

A área do núcleo também pode se conectar a recursos de Internet. Como o núcleo agrega o tráfego de todos os dispositivos da camada de distribuição, ele deve ser capaz de encaminhar grandes quantidades de dados rapidamente. Esta camada pode ser dispensada para redes de pequeno porte, com até 200 elementos. Políticas de segurança geralmente não são aplicadas nesta camada, tendo em vista que o objetivo dela é realizar roteamento de grandes volumes de tráfego.

Uma rede extensa que não possui uma estrutura robusta é passível de falhas. O prévio conhecimento dessas falhas permite ao administrador ou gerente da rede prever políticas de redundância e de segurança.

A Figura 7 abaixo ilustra nos pontos pontilhados em destaque uma redundância estrutural entre switch cores, distribuições e acessos.

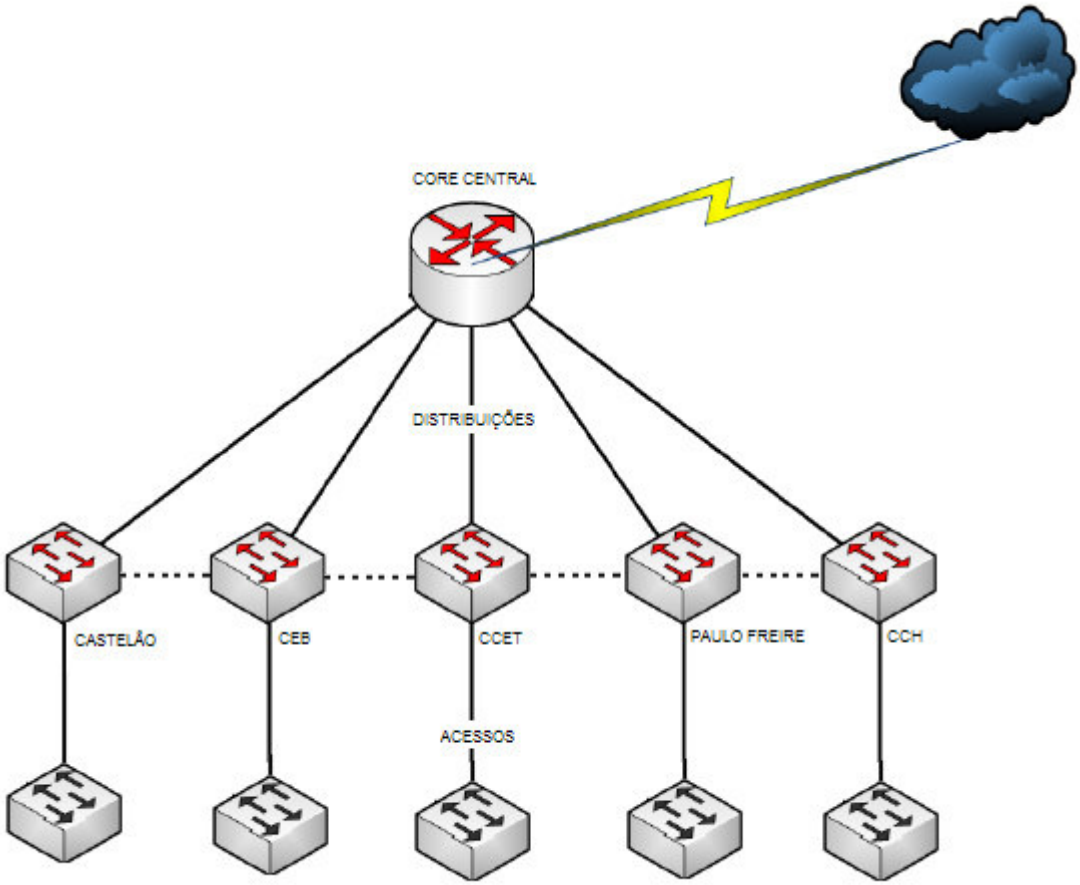


Figura 7. Redundância topológica. Fonte: autor.

É importante ter pelo menos um link redundante na interconexão entre os ativos da rede em sua estrutura topológica. Tal falha pode deixar a rede ou parte dela inoperante, deixando os usuários sem conectividade ou sem acesso aos serviços oferecidos caso o link apresente problemas.

5.2 Problemas encontrados

Os *loops* na camada de acesso são geralmente ocasionados devido à expansão irregular da rede. No intuito de aumentar a quantidade de pontos lógicos, o usuário insere um hub ou switch na rede e interliga vários pontos no mesmo acidentalmente, na ideia de aumentar a largura de banda para os computadores.

Loops geram indisponibilidade dos *links* da rede, devido aos *broadcasts storms* e corrupção nas tabelas dos switches. Na camada de acesso, eles podem ocorrer de três formas diferentes e são ilustrados pelas figuras abaixo:

1. No próprio switch: os *patch cords* podem ser ocasionalmente ligados de forma errônea no próprio equipamento, de uma porta a outra num segmento de acesso;

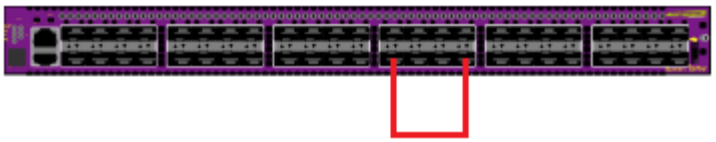


Figura 8.1. *Loop* no mesmo switch. Fonte: autor.

2. Entre switches: ocorre quando dois cabos são ligados errôneamentes entre dois switches que compartilham a mesma VLAN de acesso. Pode ocorrer também na ativação de *uplinks* entre os switches sem configuração prévia da porta;

Figura 8.2. *Loop* entre switches. Fonte: autor.

3. Fora do escopo da rede: o *loop* ocorre em um dispositivo que não faz parte da rede. Geralmente em expansões irregulares, como *hubs* ou switches que o usuário instala ou em má configuração do switch, em caso de *link aggregation*.

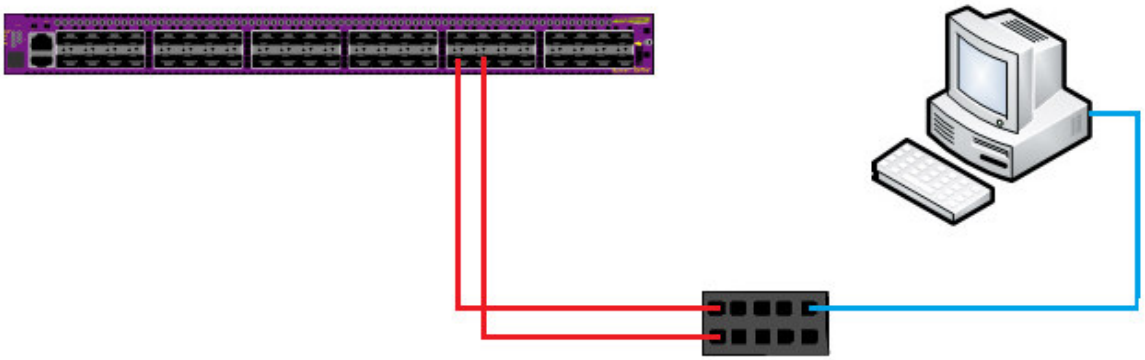


Figura 8.3. *Loop* em expansão irregular da rede. Fonte: autor.

Um *loop* nestes cenários apresentados é capaz de parar o funcionamento da rede como um todo. Aplicações que utilizam *broadcast*/*multicast*, como *streamings* ou em jogos de rede, são exemplos que podem ser citados como injetores de pacotes de difusão na rede, além dos próprios protocolos de rede, como ARP, DCHP, etc.

5.3 Solução proposta

A solução proposta para resolver os problemas de *loops* na camada de acesso é implantar o STP no modo de operação 802.1D em instâncias separadas para cada segmento da rede, ou seja, cada switch terá um domínio e instância STP.

Isso se deve ao grande número de VLANs existentes na rede de dados, onde tratar cada domínio STP separadamente favorece o gerenciamento dos domínios.

5.3.1 Aplicação prática

A Pró-Reitoria de Recursos Humanos – PRH da UFMA é o órgão executivo responsável pela gestão e pelo desenvolvimento dos Recursos Humanos, compete a ela orientar, promover, coordenar e supervisionar a execução das atividades relativas à administração e desenvolvimento dos Recursos Humanos na Universidade.

Devido ao grau de sua importância dentro da UFMA, o setor necessita de disponibilidade integral da rede, aliado ao fato de que a Pró-Reitoria já apresentou problemas recorrentes relacionados a *loops*, a aplicação prática deste trabalho será nos switches da camada de acesso da PRH. A Figura 9 abaixo ilustra a estrutura do setor em questão.

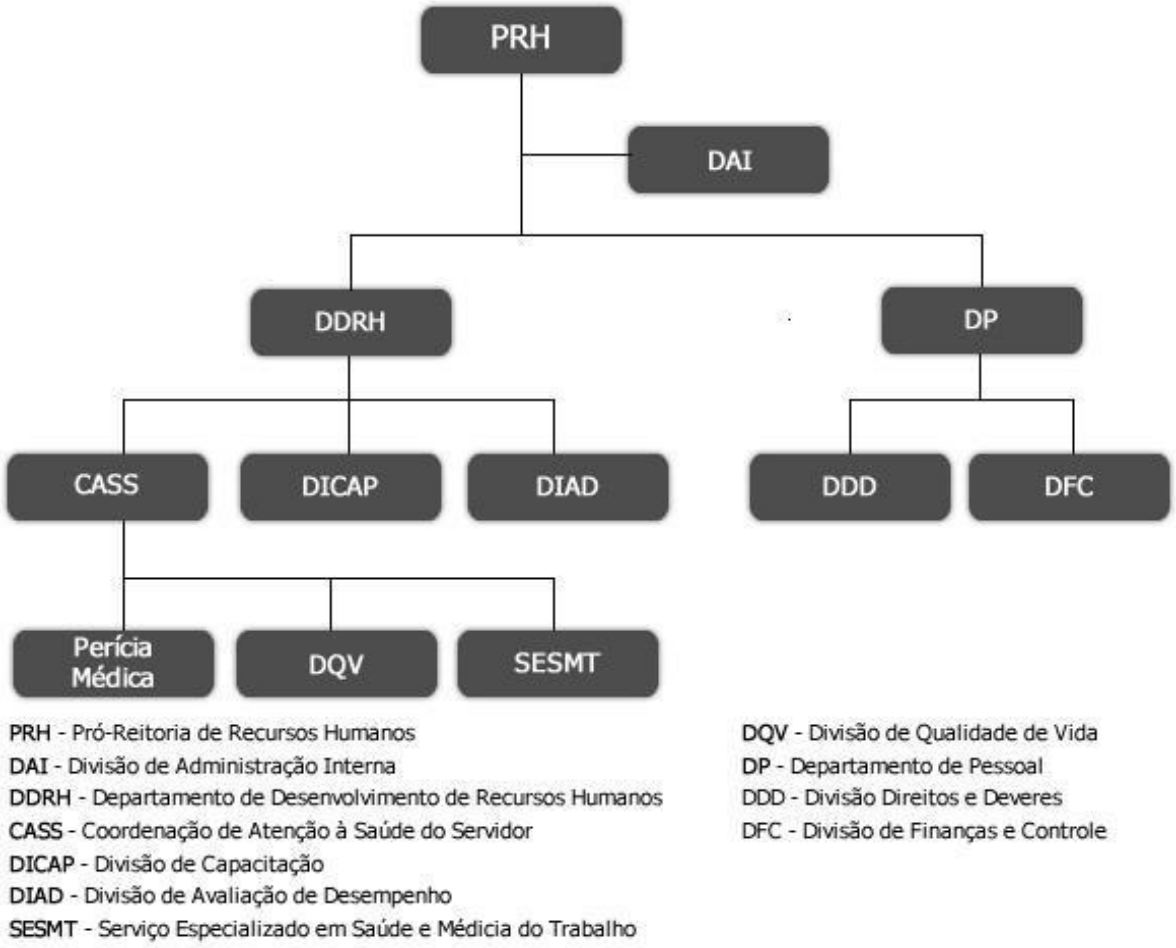


Figura 9. Estrutura da PRH. (Disponível em: www.prh.ufma.br)

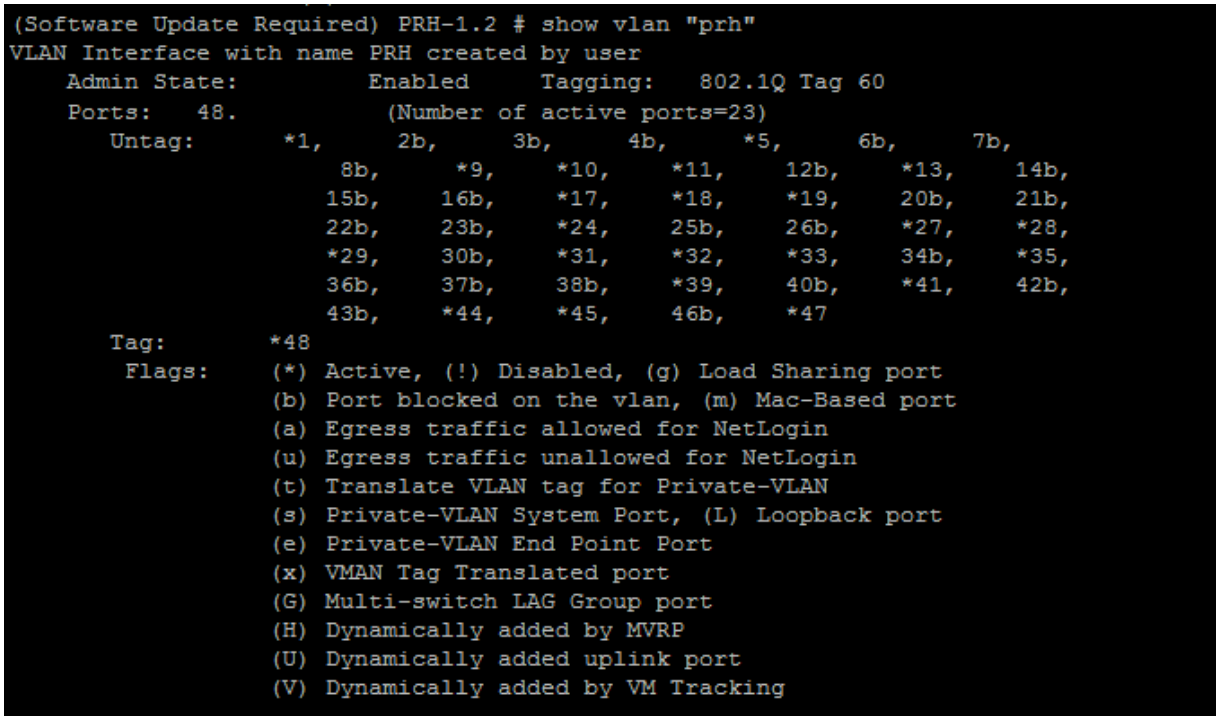
A PRH possui uma VLAN própria, cujo VID corresponde ao número 60. O segmento também possui 87 pontos de acesso, distribuídos entre oito salas, conforme mostrado na Figura 9.

Pela quantidade de pontos de acesso, foram necessários dois switches de 48 portas. O modelo utilizado foi o *Extreme Summit* x430-48t™, que possui 48 interfaces *Gigabit* e quatro portas SFP[7].

Os *uplinks* desses switches de acesso são configurados – por padrão – nas portas 48 e ambos vêm do switch de distribuição do Castelão-UFMA. Por motivos organizacionais, o primeiro switch foi nomeado de PRH-1 e o segundo de PRH-2.

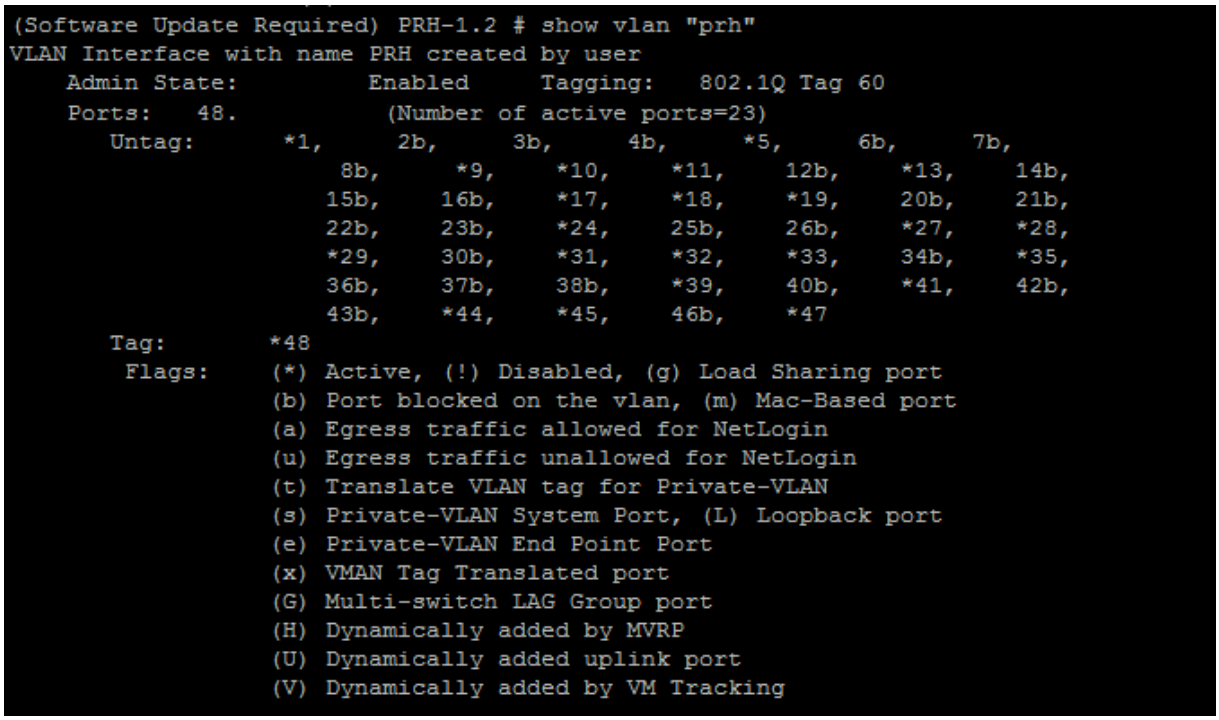
No PRH-1, as portas de acesso foram configuradas nas interfaces de 1 a 47, e no PRH-2, de um a 40, conforme mostram as Figuras 10.1 e 10.2.

[7] SFP é a sigla para *Small Form-Factor Pluggable*. É a especificação para um pequeno *transceiver* (dispositivo transmissor-receptor) que se conecta a porta também SFP de um switch. Os módulos SFP também são chamados de "mini-GBIC", devido ao tamanho. SFP converte os sinais elétricos em sinais ópticos e vice-versa.



```
(Software Update Required) PRH-1.2 # show vlan "prh"
VLAN Interface with name PRH created by user
    Admin State:         Enabled     Tagging:   802.1Q Tag 60
    Ports:   48.           (Number of active ports=23)
       Untag:      *1,      2b,     3b,     4b,     *5,     6b,     7b,
                       8b,     *9,    *10,    *11,    12b,    *13,    14b,
                      15b,    16b,    *17,    *18,    *19,    20b,    21b,
                      22b,    23b,    *24,    25b,    26b,    *27,    *28,
                      *29,    30b,    *31,    *32,    *33,    34b,    *35,
                      36b,    37b,    38b,    *39,    40b,    *41,    42b,
                      43b,    *44,    *45,    46b,    *47
       Tag:        *48
        Flags:     (*) Active, (!) Disabled, (g) Load Sharing port
                   (b) Port blocked on the vlan, (m) Mac-Based port
                   (a) Egress traffic allowed for NetLogin
                   (u) Egress traffic unallowed for NetLogin
                   (t) Translate VLAN tag for Private-VLAN
                   (s) Private-VLAN System Port, (L) Loopback port
                   (e) Private-VLAN End Point Port
                   (x) VMAN Tag Translated port
                   (G) Multi-switch LAG Group port
                   (H) Dynamically added by MVRP
                   (U) Dynamically added uplink port
                   (V) Dynamically added by VM Tracking
```

Figura 10.1. CLI do switch PRH-1. Fonte: dados de consulta.



```
(Software Update Required) PRH-1.2 # show vlan "prh"
VLAN Interface with name PRH created by user
    Admin State:         Enabled     Tagging:   802.1Q Tag 60
    Ports:   48.           (Number of active ports=23)
       Untag:      *1,      2b,     3b,     4b,     *5,     6b,     7b,
                       8b,     *9,    *10,    *11,    12b,    *13,    14b,
                      15b,    16b,    *17,    *18,    *19,    20b,    21b,
                      22b,    23b,    *24,    25b,    26b,    *27,    *28,
                      *29,    30b,    *31,    *32,    *33,    34b,    *35,
                      36b,    37b,    38b,    *39,    40b,    *41,    42b,
                      43b,    *44,    *45,    46b,    *47
       Tag:        *48
        Flags:     (*) Active, (!) Disabled, (g) Load Sharing port
                   (b) Port blocked on the vlan, (m) Mac-Based port
                   (a) Egress traffic allowed for NetLogin
                   (u) Egress traffic unallowed for NetLogin
                   (t) Translate VLAN tag for Private-VLAN
                   (s) Private-VLAN System Port, (L) Loopback port
                   (e) Private-VLAN End Point Port
                   (x) VMAN Tag Translated port
                   (G) Multi-switch LAG Group port
                   (H) Dynamically added by MVRP
                   (U) Dynamically added uplink port
                   (V) Dynamically added by VM Tracking
```

Figura 10.2. PRH-2. Fonte: dados de consulta.

A configuração do STP 802.1D não necessita que os equipamentos sejam desligados ou que a conectividade seja interrompida, podendo ser feita *in loco*.

5.4 Análise dos resultados

Para exemplificar a eficácia do protocolo na camada de acesso, alguns *loops* foram simulados no laboratório para prever o funcionamento do STP 802.1D.

Foi efetuado um *loop* no próprio switch entre as portas 9 e 10, gerando *broadcast storm* e corrupção da tabela do switch. Pode-se observar na Figura 11, através do monitoramento da CPU do switch, a sobrecarga que o equipamento sofre, chegando a 110.13% em 10 segundos de *loop*.

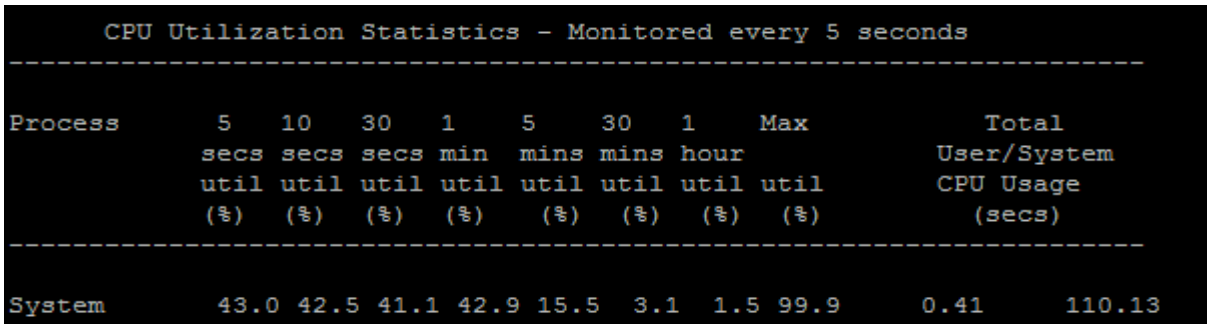


Figura 11. Monitoramento do switch. Fonte: dados de pesquisa.

Ao ativar o STP 802.1D no domínio e observar o monitoramento do protocolo, podemos perceber – na Figura 12 – que uma porta que estava causando ciclo foi bloqueada e a outra liberada. Esse é o princípio do STP, manter a disponibilidade da rede.

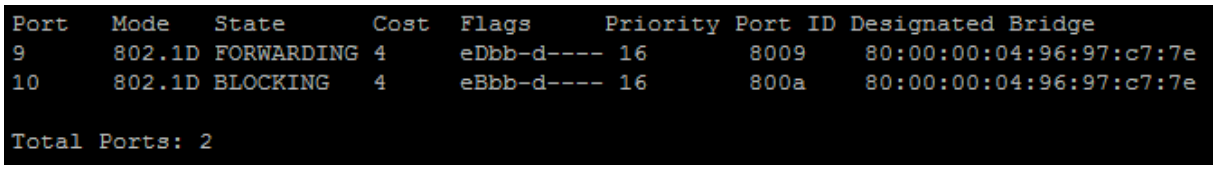


Figura 12. Monitoramento do STP. Fonte: dados de pesquisa.

Ao utilizar o STP para manter a disponibilidade da rede e sua redundância na topologia, o princípio observado é o mesmo. Como o STP formou-se nessas portas, os quadros são encaminhados somente pelas portas liberadas. Logo esse protocolo mantém uma estrutura redundante sem que possam ocorrer *loops*.

Outros protocolos não foram abordados porque seus princípios se assemelham ao do STP. Diferenciam apenas em detalhes sutis de velocidade de convergência e outros poucos relevantes, que no final melhoram a qualidade,

porém, o conceito principal mantém-se: colocar cada porta no estado ou de bloqueio ou de encaminhamento.

5.5 Aplicativo ubíquo - STP Helper

Dado a aplicação do STP na rede, não existe uma interação direta entre o administrador da rede e soluções implementadas pelo protocolo. O STP é capaz de tomar decisões que podem se perpetuar na rede caso o administrador não seja notificado das falhas e decisões do protocolo.

Então, este trabalho propõe o desenvolvimento de uma aplicação ubíqua sensível ao contexto, que fornece de maneira transparente ao administrador da rede as decisões tomadas pelo protocolo e implementadas no switch, relativas à detecção de *loops* e bloqueio ou encaminhamento de portas.

A aplicação notifica os administradores da rede – através do envio de um e-mail – dos *loops* ocorridos na camada de acesso à rede. Seu desenvolvimento foi realizado através de scripts de rotina em PHP e banco de dados MySQL.

A linha de switches Summit™ da empresa Extreme Networks armazena até 1.000 linhas de logs dentro de sua memória interna, portanto o STP Helper possui uma rotina que, de tempos em tempos, analisa o log gerado pelo equipamento.

As primeiras informações que o log dos switches da linha Summit possuem são a data em que ocorreu o evento, seguido da hora, do evento e da mensagem descritiva do evento, como se segue o exemplo:

Data (mm/dd/yyyy)	Horário	Evento	Descrição
07/02/2015	21:16:49.85	<Info:vlan.msgs.portLinkStateDown>	Port 23 link down
07/02/2015	21:16:49.16	<Info:vlan.msgs.portLinkStateUp>	Port 23 link UP at speed 10 Mbps and full-duplex
07/02/2015	21:14:36.51	<Warn:ipSecur.drpPkt>	DHCP violation occurred on port 22. Packet was dropped.

07/02/2015	21:14:36.51	<Warn:ipSecur.dhcpViol>	A Rogue DHCP server with IP 192.168.1.1 was detected on port 22

Tabela 2. Exemplo de log do switch Summit Extreme x430-48t.

Para que a aplicação interprete a ocorrência de *loops*, é necessário a captura dos *logs* dos equipamentos onde o STP está configurado. Então, um *parser*[8] analisa a sequência de entrada do *log*, buscando alterações no atributo que mudaram o estado das portas para bloqueadas em caso de *loop*.

Nos equipamentos da Extreme Networks, a notificação de *loop* é feita através de uma mensagem de *warning*, da seguinte forma:

```
<STP.DsblPortLoopDtect> Loop Detected on Port (%portId%) and port will be shutdown.
```

Onde `(%portId%)` é o número da porta que ocasionou loop na rede.

A Figura 13 ilustra a arquitetura do STP Helper.

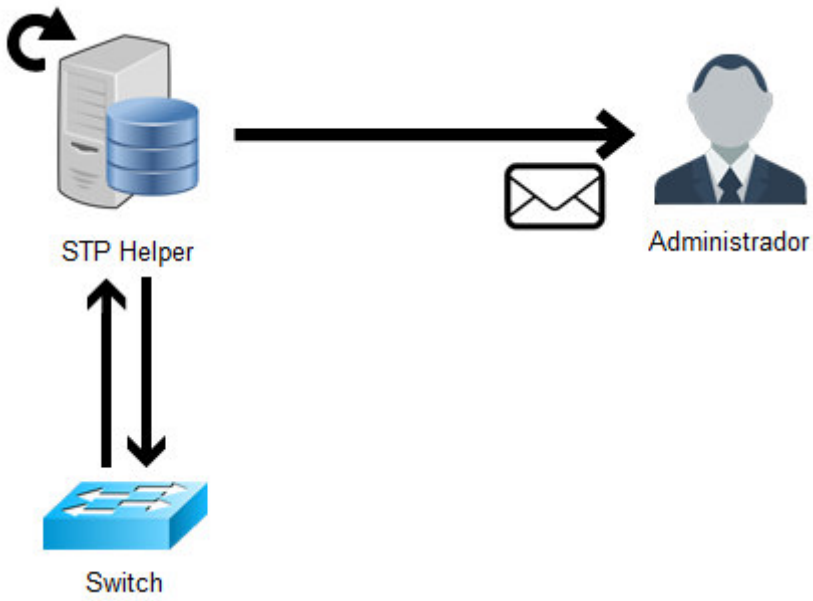


Figura 13. Arquitetura do STP Helper. Fonte: autor.

[8] *Parser* é um programa, ou parte de um programa que interpreta as entradas de um arquivo através do reconhecimento de palavras-chave ou da análise estrutural.

De acordo com a arquitetura, uma instância da aplicação funcionará seguindo os passos abaixo:

1. Abre o arquivo de log gerado pelo switch;
2. Se ocorrer o evento `<STP.DsblPortLoopDtect>` realiza a ação 2.1:
   2.1. Busca no log o nome do switch através da marcação `<SysName>`;
   2.2. Consulta os e-mails cadastrados no SGBD dos administradores de rede;
   2.3. Encaminha um e-mail aos administradores de rede informando que foi detectado um *loop* no equipamento, com o nome do equipamento, o número da porta e a porta que foi bloqueada pelo protocolo.
3. Se não encontrar nenhum evento `<STP.DsblPortLoopDtect>`, a aplicação ficará em *stand by*, aguardando a próxima rotina que a chamará novamente para analisar o log;
4. Volta ao passo 1 através da rotina que invocará o script novamente.

A Figura 14 abaixo mostra a interface de cadastro de usuário.

Figura 14. Tela de cadastro. Fonte: autor.

A Figura 15 ilustra a mensagem de e-mail que o STP Helper encaminha aos administradores de rede.

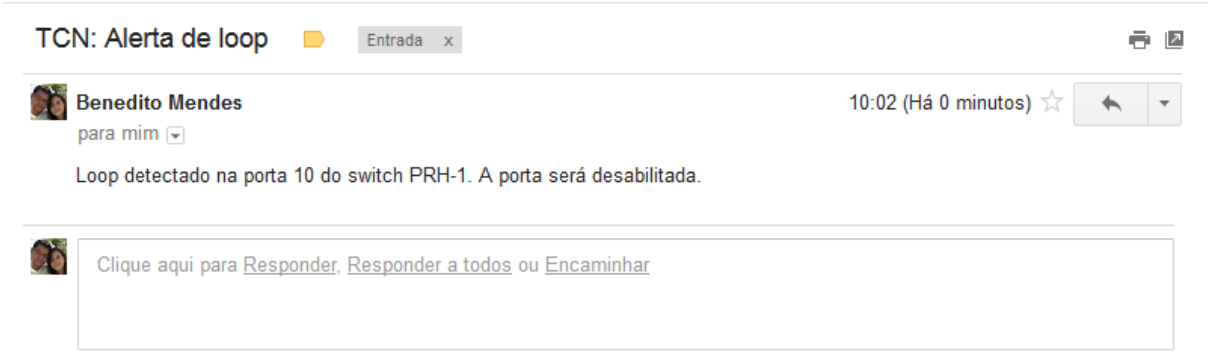


Figura 15. E-mail de alerta de loop gerado pelo STP Helper. Fonte: autor.

A mensagem enviada para os administradores da rede pelo aplicativo mostra com clareza a detecção do *loop*, especificando a porta, o nome do equipamento e a ação tomada pelo protocolo.

As próximas seções mostram o diagrama de atividades e o de sequência, ilustrando o fluxo de controle da aplicação.

5.5.1 Diagrama de atividades

O diagrama de atividades da Figura 16 descreve os fluxos de controle de uma atividade para outra em uma instância do aplicativo.

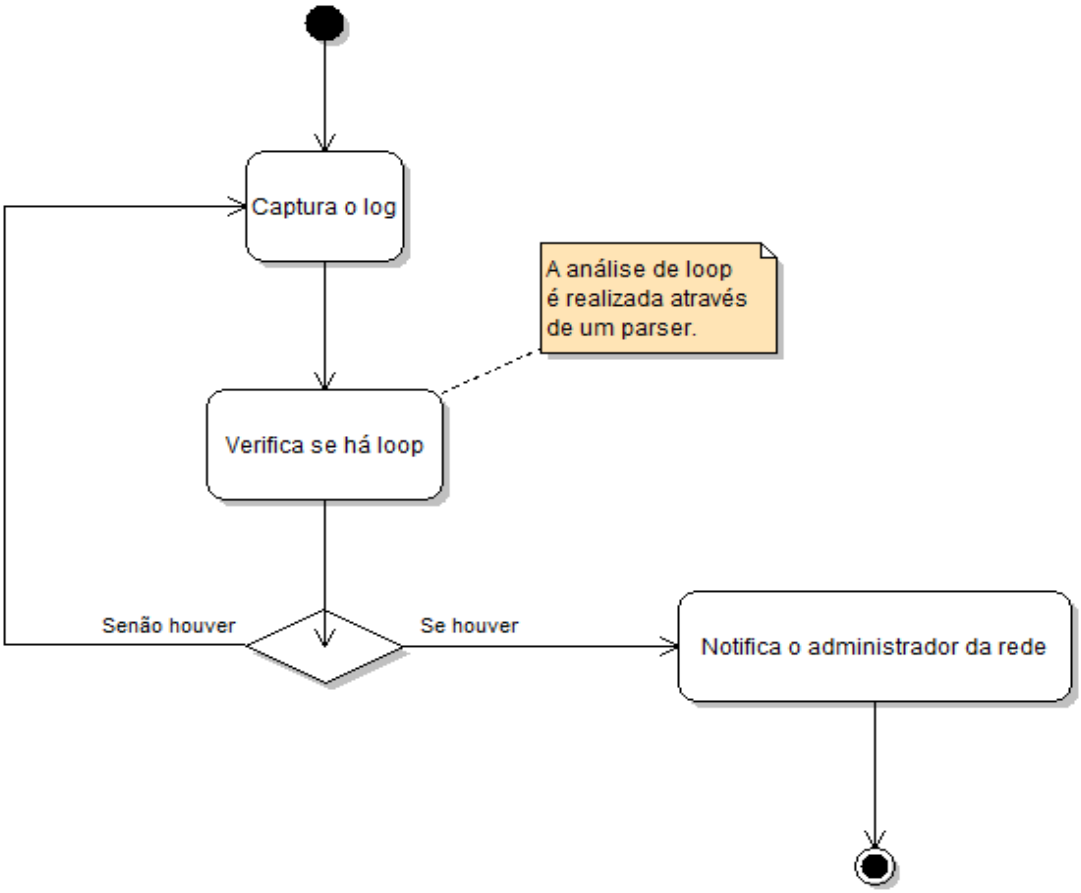


Figura 16. Diagrama de atividades. Fonte: autor.

5.5.2 Diagrama de sequência

O diagrama de sequências da Figura 17 mostra como as mensagens entre as entidades são trocadas no decorrer do tempo para a realização de uma instância da aplicação.

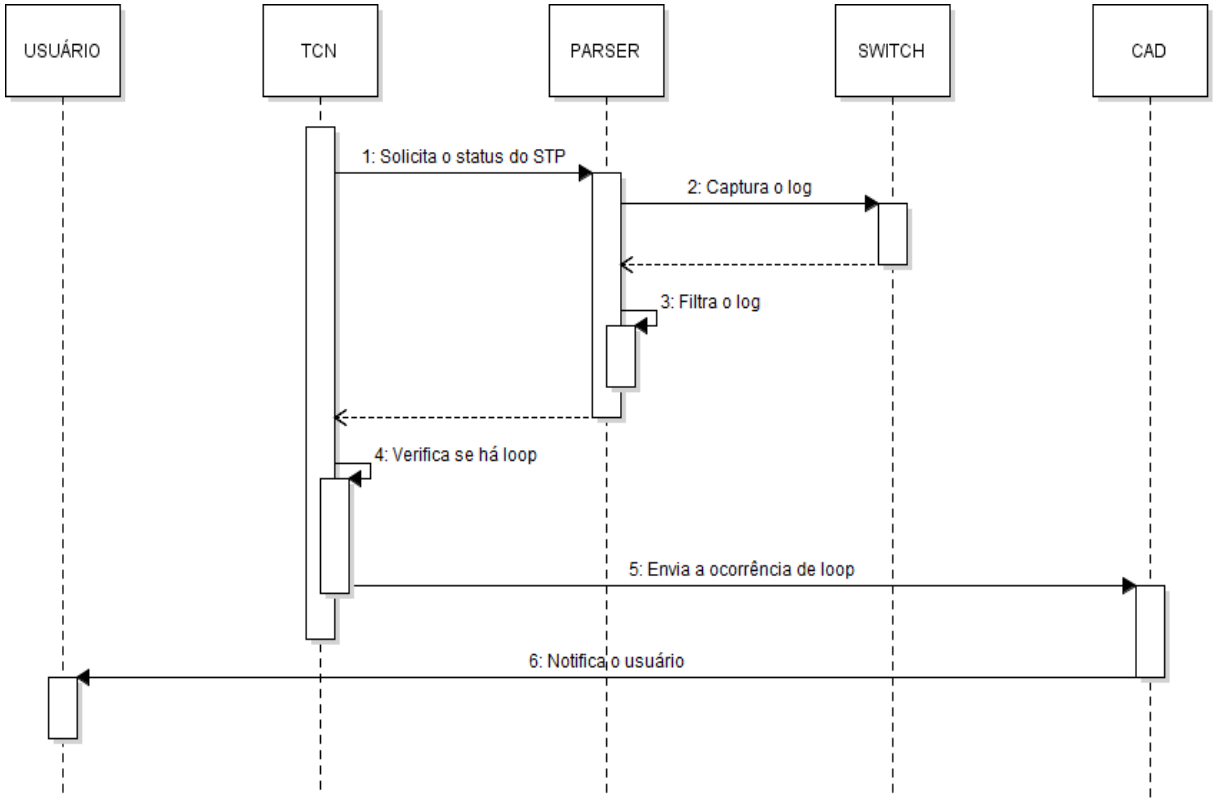


Figura 17. Diagrama de sequência. Fonte: autor.

As entidades mostradas nesse diagrama são respectivamente: o usuário; a aplicação para notificação de alteração na topologia (TCN); o script *parser*; o switch, responsável por gerar os *logs* e o CAD, que é o sistema responsável por consultar os administradores no SGBD e notificar o usuário através do e-mail.

O STP apenas faz o bloqueio lógico do *loop*, mas fisicamente ele ainda existe e precisa ser desfeito para que a porta possa ser reutilizada. Portanto, é de extrema importância que o administrador tenha ciência das alterações que o protocolo realizou para que elas não se perpetuem na rede, diminuindo os recursos da mesma e consumindo processamento do equipamento.

6 CONCLUSÃO

Com a implantação do *Spanning Tree Protocol* na PRH finalizada, o tráfego da rede continuou operante e, a partir dos testes simulados, livres de *loop*, o objetivo proposto foi alcançado. Qualquer problema relacionado será tratado pelo protocolo e bloqueado se houver necessidade.

Com a implementação da aplicação ubíqua, os administradores agora podem tomar ciência das decisões que o STP tomou na rede, realizando manutenção proativa e não deixando que o bloqueio gerado pelo protocolo de perpetue na rede, atingindo então, o objetivo proposto.

As mudanças físicas que a rede pode sofrer – como o aumento do número de pontos lógicos – não afetarão a integridade do protocolo nem a disponibilidade da rede, pois a segmentação garante essa flexibilidade em termos de configuração lógica.

Situações descritas na seção “problemas encontrados”, no capítulo 4, só ocorrerão caso haja a instalação de novos equipamentos e, se e somente se, esses equipamentos não receberem a configuração do STP.

Os argumentos apresentaram-se em uma linguagem acessível aos profissionais intermediários, atingindo também aqueles com conhecimentos avançados. Sendo assim, qualquer pessoa que vier a se interessar pelo estudo do tema terá uma ótima fonte de pesquisa. Por outras palavras, este trabalho abordou profundamente o *Spanning Tree* como um protocolo sensível ao contexto, uma vez que o estudo torna-se fácil, se há a teoria aplicada com a prática. E isso foi bem demonstrado durante a lapidação desta monografia no seu passo-a-passo.

6.1 Trabalhos futuros

No decorrer da implantação do trabalho, algumas ideias para futuras implementações na rede acadêmica foram surgindo. De início, fechar o *backbone* da rede redundantemente através de uma topologia híbrida.

Atualmente o cenário do *backbone* da rede universitária é caracterizado por uma estrela, onde o concentrador é o Núcleo de Tecnologias da Informação – NTI, ligando os switches de distribuição. A ideia é interligar os switches de

distribuição entre si, através de um anel e implantar o STP para evitar *loop* no *backbone* da rede.

Outro ponto que pode ser trabalhado é o gerenciamento dos ativos de rede. Os switches da universidade são gerenciáveis e suportam o protocolo SNMP (*Simple Network Management Protocol*), podendo enviar informações sobre tráfego nas interfaces, processadores, temperatura, entre várias outras informações importantes que o administrador pode utilizar para monitorar seu ambiente e tomar decisões.

Em termos de aplicação, a ideia é melhorar o STP Helper para que ele não apenas notifique o administrador das decisões tomadas pelo protocolo, mas que o administrador possa validar ou não as alterações no domínio.

REFERÊNCIAS

ABOWD, G. D. et al. **Cyberguide: a mobile context-aware tour guide.** Wirel. Netw., Kluwer Academic Publishers, Hingham, MA, USA. 421–433, out. 1997.

BROWN, P. J. **Triggering Information by Context**. Personal Technologies, 2(1) (1998) p. 1-9.

DEY, A.K.; ABOWD e G.D. **Towards a Better Understanding of Context and Context-awareness.** Submetido ao Handheld and Ubiquotous Computing - First International Simposium, 1999.

DEY, A.K.; SALBER, D.; FUTAKAWA, M. e ABOWD G. **An architecture to support context-aware applications**. Submetido ao 12º Simpósio ACM no User Interface Software and Technology, 1999.

DONAHUE, A. **Network Warrior.** O'Reilly, 2007, p. 59.

DYE, M. A.; MCDONALD, R. e RUFI, A. W. **Network Fundamentals CCNA Exploration Companion Guide**. [S.l.]: Cisco Press, 2008.

EXTREME NETWORKS. **Layer 2 Protocols**. Disponível em: < http://extrcdn.extremenetworks.com/wpcontent/uploads/2014/04/Layer_2_Protocols.pdf>. Acesso em: 18 maio 2015.

GROSS, J.; SACCOMAN, T. e SUFFEL, L. **Spanning Tree Results for Graphs and Multigraphs: A Matrix-Theoretic Approach**. World Scientific, 2015.

HAFFERMANN, L. **Segmentação de Redes com VLAN**, 2009. Disponivel em: <http://www.ppgia.pucpr.br/~jamhour/RSS/TCCRSS08A/Leonardo%20Haffermann%20-%20Artigo.pdf>. Acesso em: 18 abril 2015.

KAKU, M. **A Física do Futuro**. Rocco, 2011.

LYYTINEN, K. e YOO, Y. **Issues and Challenges in Ubiquitous Computing - Introduction**. Commun. ACM, 45(12):62–65, December 2002.

Modelos TCP/IP. Disponível em: <https://jbgsm.wordpress.com/2010/05/31/camadas-tcpip>. Acesso em: 19 abril 2015. Autor desconhecido.

ODOM, W.; HEALY, R. e MEHTA, N. **CCIE Routing and Switching Exam Certification Guide**. 3rd. ed. [S.l.]: Cisco Press, 2008.

PASCOE, J. **Adding generic contextual capabilities to wearable Computers.** Proceedings of 2nd International Symposium on Wearable Computers, October 1998, p. 92-99.

PRIOR, R. **Introdução às VLANs**. Disponível em: <http://www.dcc.fc.up.pt/~rprior/1112/LabRedes/VLAN.pdf>. Acesso em: 27 julho 2015.

PERLMAN, R. **An algorithm for distributed computation of a spanning tree in an extended LAN**. Proceedings of the ninth symposium on Data communications, p.44-53, 1985.

RYAN, N. S.; PASCOE, J. e MORSE, D. R. **Enhanced Reality Fieldwork: the Context-aware Archaeological Assistant**. In: GAFFNEY, V.; LEUSEN, M. van; EXXON, S. (Ed.). Computer Applications in Archaeology 1997. Oxford: Tempus Reparatum, p. 269-274. 1998.

SATYANARAYANAN, M. **Pervasive computing: Vision and challenges**. IEEE Personal Communications, 8:10–17, 2001.

SCHILIT, B.; ADAMS, N. e WANT, R. **Context-Aware Computing Applications**. 1st International Workshop on Mobile Computing Systems and Applications. 1994, p. 85-90.

SCHILIT, B. e THEIMER, M. **Disseminating Active Map Information to Mobile Hosts**. IEEE Network, 1994 p.22-32.

TANEBAUM, A. S. e WETHERALL, D. J. **Computer networks**. 5th. ed. Pearson, 2011.

WEISER, M. **The Computer for the 21s century**, Scientific American, p. 94-104 (1991).

WEISER, M. **The World is not a Desktop**. Interactions. Janeiro de 1994; p. 8.

802.1Q-2014. **IEE Standart for Local and Metropolitan Area Networks - Bridges and Bridged Networks**. 2014.

APÊNDICE A – Código-fonte do *parser*

```
<?php

      include 'conexao.php';
      require 'PHPMailer/PHPMailerAutoload.php';

      $sql = 'SELECT * FROM usuario';
      $result = mysqli_query($conexao, $sql);


      $txt_file    = file_get_contents('log.txt');
      $rows        = explode("\n", $txt_file);
      array_shift($rows);


      foreach($rows as $row => $data)
      {
          $row_data = explode(' ', $data);

          $info[$row]['data'] = $row_data[0];
          $info[$row]['horario'] = $row_data[1];
          $info[$row]['porta'] = $row_data[7];

            if ( $row_data[3] == '<STP.DsblPortLoopDtect>' )
            {

                  $mail = new PHPMailer;
                  $mail->isSMTP();
                  $mail->Host = 'smtp.gmail.com';
                  $mail->SMTPAuth = true;
                  $mail->Username = '**********';
                  $mail->Password = '**********';
                  $mail->SMTPSecure = 'tls';
                  $mail->Port = 587;
                  $mail->setFrom('noreply@localhost.com', 'Benedito
Mendes');

                  while( $sel = mysqli_fetch_array($result, MYSQLI_ASSOC)
):

                        $mail->addReplyTo( $sel['email'], $sel['nome'] );
                        $mail->addAddress( $sel['email'], $sel['nome'] );

                  endwhile;

                  $mail->WordWrap = 50;
                  $mail->isHTML(true);

                  $mail->Subject = 'TCN: Alerta de loop';
                  $mail->Body    = 'Loop detectado na porta ' .
$info[$row]['porta'] . ' do equipamento '.$info[$row]['nome'].' . A porta
```

```
será desabilitada. Horario: '. $info[$row]['data'] . $info[$row][
'horario']. '.  ';

                  if(!$mail->send()) {
                     echo 'Mensagem nao enviada. ';
                     echo 'Erro: ' . $mail->ErrorInfo;
                     exit;
                  }

            break;
          }
      }

?>
```

APÊNDICE B – Código-fonte da página de cadastro

```
<!DOCTYPE html>
<html lang="en">
  <head>

    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <title>TCN - Cadastro</title>

    <!-- Bootstrap core CSS -->
    <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet">

    <!-- Folha de estilos -->
    <link href="starter-template.css" rel="stylesheet">
    <script src="bootstrap/assets/js/ie-emulation-modes-
warning.js"></script>

  </head>

  <body>

    <nav class="navbar navbar-inverse navbar-fixed-top">
      <div class="container">
        <div class="navbar-header">
          <a class="navbar-brand" href="#">Notificação de Alteração de
Topologia</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
          <ul class="nav navbar-nav">
            <li class="active"><a href="#">Home</a></li>
          </ul>
        </div><!--/.nav-collapse -->
      </div>
    </nav>

    <div class="container">
      <br/><br/><br/>
      <div class="starter-template">
        <h3>Cadastro</h3>
        <br/>

            <!-- Formulário para cadastro de administrador -->
            <form class="form-horizontal" action="include.php"
method="post">

              <!-- NOME -->
              <div class="form-group">
                <label for="nome" class="col-sm-2 control-
label">Nome</label>
```

```
                <div class="col-sm-6">
                  <input type="text" class="form-control" id="nome"
name="nome">
                </div>
              </div>

              <!-- EMAIL -->
              <div class="form-group">
                <label for="endereco_empresa" class="col-sm-2 control-
label">E-mail</label>
                <div class="col-sm-6">
                  <input type="text" class="form-control" id="email"
name="email">
                </div>
              </div>


              <div class="form-group">
                <div class="col-sm-offset-2 col-sm-5">
                  <input type="submit" name="submit" value="Enviar">
                </div>
              </div>

            </form>

      </div>

    </div><!-- /.container -->

    <!-- Scripts Bootstrap -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></s
cript>
    <script src="bootstrap/js/bootstrap.min.js"></script>

  </body>
</html>
```

APÊNDICE C – Código-fonte do banco de dados e dos scripts em PHP

usuario.sql

```
CREATE TABLE IF NOT EXISTS `usuario` (
  `id_usuario` int(11) NOT NULL AUTO_INCREMENT,
  `nome` varchar(50) NOT NULL,
  `email` varchar(50) NOT NULL,
  PRIMARY KEY (`id_usuario`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=10;
```

conexao.php

```
<?php

      $conexao = mysqli_connect('localhost', '******', '******');
      mysqli_select_db($conexao,'tcn');

?>
```

include.php

```
<?php

      include 'conexao.php';

      $nome             = $_POST['nome'];
      $email            = $_POST['email'];

      $sql = "INSERT INTO usuario (nome, email) VALUES ('$nome',
'$email')";

      mysqli_query($conexao, $sql);
      header("Location: index.php");

?>
```