1. Dados de identificação

Título: Utilização do driver TUN/TAP como mecanismo de depuração e acesso à pilha de tratamento de protocolos de rede disponível no kernel Linux.

Autor: Tiago Antônio Rizzetti

Universidade Federal de Santa Maria

RST 509, 1454. Santa Maria – RS

rizzetti@ctism.ufsm.br

2. Dados gerais

2.1. Objetivos da oficina prática

Durante o processo de desenvolvimento ou implementação de um protocolo de rede, são necessários testes exaustivos sobre o protocolo, a fim de garantir que este opere de acordo com as especificações de projeto. Em algumas ocasiões é interessante que esse teste seja feito de forma controlada e isolada, permitindo que o desenvolvedor seja capaz de identificar possíveis problemas do protocolo, normalmente através da realização de algum processamento extra durante a operação desse. Uma das formas de operacionalizar esse tipo de teste é através da utilização de interfaces virtuais, as quais são vistas pelo sistema como uma interface de rede física mas, que na verdade, são interfaces de rede lógicas, estabelecidas por software e que permitem um controle completo sobre os pacotes trafegados. Essas entidades são uma ferramenta de grande valia durante o teste e desenvolvimento de protocolos, sendo objetivo desta oficina apresentá-las e disponiblizar alguns exemplos de sua utilização no desenvolvimento de softwares de rede.

2.2. Revisão bibliográfica

Interface Virtual

Uma interface de rede virtual, do ponto de vista de sua funcionalidade, é bastante semelhante a uma interface de rede real, porém sem o hardware envolvido. O que permite aplicações interessantes em muitas situações, isolando os testes de software de rede, principalmente em se tratando de protocolos. Um driver virtual genérico, muito utilizado é o TUN/TAP,

disponibilizado no kernel do Linux desde sua versão 2.4.x. Esse driver cria um device na lista de dispositivos do linux que permite ler e escrever nele. Com isso tem-se duas funcionalidades inerentes, as quais promovem sua grande importância: captura de pacotes de rede e, injeção de pacotes de rede.

O Driver TUN/TAP

O driver TUN/TAP provê a recepção e transmissão de pacotes através de programas do espaço de usuário. O espaço de execução do programa é determinante para sua robustez e eficiência na execução. Tradicionalmente, programas em espaço de kernel tem seu debug mais complexo, no entanto são muito eficientes quando há necessidade de acesso intenso ao hardware. Já programas de espaço de usuário são acima de tudo mais seguros, visto que possuem as restrições necessárias para proteger o núcleo de execução do kernel do sistema. Com isso programas em espaço de usuário em geral, por estarem num maior nível de abstração, necessitam realizarem mais chamadas e processamento para realizar operações equivalentes no sistema.

Em função do driver TUN/TAP executar em espaço de usuário, ele possui uma caracteristica importante no desenvolvimento – facilidade de debug. Além disso, pode ser utilizado para emulação e acesso à pilha de protocolos disponibilizada pelo sistema operacional. Abaixo segue um caso de uso do driver TUN/TAP que exemplifica sua utilização na construção de software embarcado para equipamentos de telecom.

Utilizando o Driver TUN/TAP para inserir pacotes na pilha de protocolos do Kernel

Entre outras aplicações, ele pode ser utilizado para inserir pacotes de rede na pilha de tratamento de protocolos do kernel (normalmente a TCP/IP), sem a necessidade de uma interface física. Um exemplo de onde esse tipo de aplicação é importante são em equipamentos de rede. Nestes equipamentos normalmente há pelo menos dois processadores principais, dedicados a operacionalizar suas funcionalidades. Um deles é um processador dedicado, chamado Network Processor, que é altamente especializado, pois tem como objetivo fornecer um alto throughput para pacotes de dados da rede, mas que não é adequado para realizar o tratamento de protocolos de controle e gerenciamento. Os protocolos de controle em geral são tratados por um outro processador do sistema, que usualmentemente é um microcontrolador que executa o sistema host. No microcontrolador também são executados os aplicativos de espaço de usuário que permitem gerenciar o equipamento, alguns exemplos: telnet, ssh, ftp, tftp, sistema de gerencia remota do equipamento, etc. O driver TUN/TAP é especialmente útil para repassar pacotes à estes últimos.

Quando pacotes de controle/gerenciamento são recebidos, através da rede, pelo network processor, este identifica a natureza do pacote e quando o tratamento não é realizado por ele, o pacote em geral é encaminhado para o microcontrolador do host. Ao receber um pacote telnet por exemplo, este deve ser repassado ao daemon tratador deste tipo de pacote. Para isso utiliza-se o driver virtual, que insere esse pacote na pilha de tratamento de protocolos do kernel e, a partir de então, ele é tratado como um pacote normal recebido pela interface de rede. Com isso pode-se utilizar os recursos já disponíveis no sistema operacional de maneira eficiente.

Esse mesmo mecanismo pode ser empregado por exemplo para simular a entrada de pacotes na rede, utilizando a interface virtual, e verificando se estes passam pela pilha de protocolos e verificar se estes são tratados adequadamente.

Utilizando TUN/TAP no Linux

Esse driver é tratado no linux como um device usual no sistema, podendo ser acessado através da arvóres de dispositivos disponibilizados no /dev. O acesso ao TUN/TAP é feito através do arquivo que representa o device em /dev/net/tun.

Primeiramente, abre-se o dispositivo como um arquivo normal, para leitura e escrita, obtendo-se um descritor de arquivo. Em seguida, deve-se enviar uma *ioctl* para esse descritor, com a finalidade de configurar o dispositivo.

O device pode ser configurado tanto através de ioctls direcionadas ao descritor de arquivo, que o representa, quanto por utilização das ferramentas de configuração de devices de rede do linux, como o ifconfig.

Basicamente o driver TUN/TAP pode ser utilizado de duas formas distintas: como uma interface point-to-point (TUN) ou como um domínio de colisão ethernet, tal como um hub (TAP). O tipo do device desejado deve ser configurado através de chamadas ioctl ao sistema, no device representado pelo arquivo /dev/net/tunXX. Também é possível no linux realizar esse procedimento no formato dos comandos abaixo:

Interface do tipo TUN:

ifconfig <interface> <ip-local> pointopoint <ip-remoto>

Interface do tipo TAP:

ifconfig <interface> hw ether <MAC> <ip-local>

O programa irá comunicar-se com a interface de rede virtual por meio de seu descritor de arquivo, lendo e escrevendo no mesmo. Entretanto, há caracteristítas muito importantes que precisam serem observadas.

Para ler um pacote que tenha sido enviado para a interface de rede, será utilizada a chamada de sistema read(). Cada uma dessas chamadas lerá sempre um pacote diferente. Portanto é permitido ler somente um pacote inteiro de cada vez, se não for feito desta forma, o sistema operacional irá descartar o restante do pacote.

De maneira semelhante, cada chamada de sistema write() corresponderá a um pacote diferente que será recebido pela interface de rede virtual. Ou seja, não é possível dividir um mesmo pacote em varias chamadas write(), visto que cada escrita é interpretada como um pacote completo.

O formato de cada pacote de dados lido ou escrito no descritor de arquivo vai depender da configuração realizada no device.

2.3. Metodologia utilizada

- Abordar a importância do conhecimento proposto baseado em exemplos práticos que facilitem aos participantes visualizar sua aplicação em um ambiente real, justificando e motivando o aprendizado.
- Resgatar os conceitos necessários através da apresentação da revisão bibliográfica inerente ao tema.
- Mostrar exemplos práticos de sua utilização: Mostrando o processo durante a oficina, através da execução prática deste. Também será construído durante a oficina, o código, em linguagem C, necessário para captura/injeção de pacotes de um protocolo de rede qualquer sobre o driver virtual tun/tap no linux.

2.4. Estrutura física e softwares necessários

- Sistema operacional Linux, utilizando kernel >= 2.6.24, preferencialmente na versão Ubuntu com suporte ao driver TUN/TAP habilitado em kernel (usualmente isso é feito por padrão, sendo disponibilizado como um módulo de kernel);

- Ambiente de compilação GCC instalado, para a plataforma x86;

- Ferramenta de compilação de diretórios make;

- Software Wireshark;

- Software Packeth;

- Projetor (datashow);

- Alocação de um laboratório pode ser interessante, mas não é imprescindível.

2.5. Quantidade de vagas (máximo 24 participantes)

Para o bom andamento dos trabalhos, sugiro disponibilizar até 15 vagas.

3. Bibliografia principal e/ou básica utilizada na preparação da oficina prática

[1] Documentacao oficial do TUN/TAP para Linux.
/usr/src/linux/Documentation/networking/tuntap.txt

[2] Tanenbaum, Andrew. Redes de Computadores, 4ª Edição, 2003. ISBN: 8535.211853

[3] The Bug Magazine
http://www.thebugmagazine.org/magazine/bug02/0x04_tun-tap.txt

[4] TUN/TAP on wikepedia
http://en.wikipedia.org/wiki/TUN/TAP

[5] NP-3 Network Processor from EZChip
http://www.ezchip.com/Images/pdf/NP-3_Short_Brief_online.pdf

Curriculum Vitae

Tiago Antônio Rizzetti
Data de Nascimento: 09/09/1981.
Naturalidade: Santa Maria – RS
Endereço: RST 509, 1454. Bairro: São José. Santa Maria, RS
CEP: 97095-000
E-Mail: rizzetti@gmail.com

Formação

Mestrado em Computação, Universidade Federal de Santa Maria, UFSM, Brasil.
Área: Sistemas Paralelos e Distribuídos; Subárea : Computação Pervasiva.
2007/2009.

Graduação em Ciência da Computação.
Universidade Federal de Santa Maria, UFSM, Brasil.
2003/2006.

Atuação Profissional

Universidade Federal de Santa Maria
Analista de Tecnologia da Informação, atuando no gerenciamento da rede e no planejamento e implementação dos serviços de informática do Colégio Técnico Industrial de Santa Maria/UFSM.
Periodo: 06/2010 - Atual

Datacom Telemática.
Projetista de Desenvolvimento, atuando no desenvolvimento de software embarcado na plataforma powerpc, para equipamentos de telecom.
Periodo: 06/2008 - 06/2010

Bolsista ITI CNPQ.
Projeto PDSCE – Plataforma de Desenvolvimento de Sistemas Computacionais Embarcados. Área de Ferramentas de Apoio: Colaboração e Distribuição.
2005/2006.

Estagiário no Centro de Processamento de Dados – CPD/UFSM.
Função: Setor de Desenvolvimento do Sistema SIE/SIM (Sistema de Informações para o Ensino/ Sistema de Informações Municipais), módulo de Recursos Humanos.
2001/2005.

Apresentações

- RIZZETTI, T. A; AUGUSTIN, I. Um Ambinte de Contexto Personalizado e Orientado a Tarefas na Arquitetura ClinicSpace. Dissertação de Mestrado, para obtenção do Título de Mestre em Computação, 2009, UFSM, BR.
- AUGUSTIN, I. ; LIMA, J. C. D. ; SILVA, F. L. ; FERREIRA, . G. L. ; RIZZETTI, T. A. . Sistemas de Computação Móvel: da Mobilidade à Ubiquidade. Santa Maria: Escola Regional de Redes de Computadores (ERRC 2007), 2007 (Minicurso).
- RIZZETTI, T. A. ; LIMA, J. C. D. ; AUGUSTIN, I. . Construção de um Ambiente Distribuído e colaborativo para o projeto PDSCE. 2007. (Trabalho de Graduação);