Injeção de Falhas de Comunicação em Linux

Fábio Olivé Leite
(olive@conectiva.com.br)
Conectiva S/A
R. Tocantins, 89
Curitiba - PR

Taisy Silva Weber
(taisy@inf.ufrgs.br)
PPGC - UFRGS
Av. Bento Gonçalves, 9500
Porto Alegre - RS

Resumo

Este artigo apresenta a técnica de validação de protocolos de comunicação confiável através da injeção de falhas de comunicação, contextualizando a técnica em si e apresentando uma ferramenta de injeção de falhas de comunicação implementada no núcleo do Sistema Operacional Linux.

Palavras-chave: Injeção de Falhas; Sistemas Distribuídos; Comunicação Confiável.

Abstract

This article presents the reliable communication protocol validation technique through the use of communication fault injection, contextualizing the technique itself and presenting a tool for communication fault injection implemented in the kernel of the Linux Operating System.

Keywords: Fault Injection; Distributed Systems; Reliable Communication.

1 Introdução

Verifica-se hoje em dia um aumento explosivo da utilização de Sistemas Distribuídos, mesmo nos serviços mais vitais da sociedade. Como estes sistemas estão expostos à falhas tanto de *hardware* quanto de *software*, faz-se necessário garantir seu funcionamento correto, ou pelo menos a degradação não catastrófica de seus serviços, através de técnicas de Tolerância a Falhas.

A capacidade de comunicação de um nó de um Sistema Distribuído é tão importante quanto a sua capacidade de processamento, portanto existem vários protocolos de comunicação cujo propósito é atribuir à comunicação propriedades interessantes, tais como: confiabilidade, ordenação e atomicidade.

A comprovação da correção destes protocolos é feita através de provas matemáticas e lógicas de sua especificação, porém mesmo com uma especificação correta não se pode garantir uma *implementação* correta. A comprovação da correção de código fonte ou executável é extremamente complexa, ainda mais quando envolve a execução concorrente inerente aos Sistemas Distribuídos. A única forma viável, portanto, de se validar a implementação de protocolos de comunicação confiável é através de técnicas experimentais como a Injeção de Falhas [ARL 90].

2 Injeção de Falhas

A Injeção de Falhas pode ser vista como a maneira mais natural de se testar mecanismos de Tolerância a Falhas, visto que proporciona justamente a entrada que estes mecanismos processam, as falhas. De um modo geral, as técnicas de Injeção de Falhas podem ser classificadas em injeção por *hardware*, por *software* e por simulação [CAR 98].

A Injeção de Falhas por Hardware se dá através da utilização de equipamentos adicionais, que possam atuar nos níveis lógicos presentes nos vários níveis do sistema, como CPU, barramentos, etc. É a técnica mais efetiva por criar falhas (ou erros) reais, porém é cara (inclui criação de equipamentos específicos) e nem sempre permite total controle ou monitoração das falhas injetadas. Uma operação de injeção pode, por exemplo, resultar na queima de um componente.

No outro lado da moeda se encontra a Injeção de Falhas por Simulação, que consiste na simulação do funcionamento do sistema em teste em programas especializados. Esta técnica permite absoluto controle e monitoração do processo de validação, porém necessita que modelos de simulação completos do sistema em teste sejam gerados, o que pode não ser possível ou viável.

Como solução intermediária existe a Injeção de Falhas por Software, que permite a simulação em *software* de falhas (ou erros) de hardware. A Injeção de Falhas por Software pode se dar no meta-nível, no nível de aplicação, ou no nível de Sistema Operacional [BAR 99]. Cabe salientar que a presença do código injetor altera o ambiente de execução do sistema em teste, e o injetor deve ser projetado de forma a gerar a menor intrusão possível.

A Injeção de Falhas por Software no nível de Sistema Operacional é implementada de forma a simular a ocorrência de falhas nos serviços prestados aos processos. Como o Sistema Operacional provê uma máquina virtual aos processos, os quais não conhecem o "mundo exterior" senão pelas chamadas de sistema que o Sistema Operacional lhes proporciona, alterações no funcionamento interno desta máquina virtual são completamente transparentes aos processos. Falhas injetadas em serviços prestados pelo *kernel* são, portanto, indistinguíveis de falhas reais provenientes do universo físico.

3 Injeção de Falhas de Comunicação

A Injeção de Falhas de Comunicação se dá através da manipulação das mensagens que circulam em um Sistema Distribuído. Para caracterizar a manipulação de mensagens, faz-se necessário distinguir dois momentos distintos: a seleção de uma mensagem a ser manipulada, e a manipulação propriamente dita. Pode-se observar três tipos de seleção de mensagens. Seleção baseada em conteúdo da mensagem, baseada em fluxo de mensagens, e baseada em elementos externos. Da mesma forma, identifica-se manipulação do conteúdo da mensagem, da mensagem em si e de elementos externos [DAW 96].

A seleção baseada em conteúdo da mensagem permite que se selecione mensagens de acordo com as informações que carrega, como protocolo, endereços, números de série e *flags*. A seleção baseada em fluxo de mensagens não depende de conteúdo, mas sim de haver um fluxo de mensagens no sistema. Permite que se selecione uma a cada três mensagens, a quinta mensagem, ou mesmo uma seleção não determinística como 10% das mensagens. Já a seleção baseada em elementos externos leva em consideração temporizadores e sinalizadores, que podem ser controlados pela pessoa coordenando os testes.

De forma análoga, a manipulação do conteúdo da mensagem permite que se altere campos e dados internos da mensagem. A manipulação da mensagem em si trata de ações como o descarte da mensagem, o atraso da mensagem ou mesmo a duplicação da

ComFIRM_Test_Counter	0x20
ComFIRM_Action_Drop	0x60
ComFIRM_Action_Counter	0x78
ComFIRM_Test_Equal	0x00
ComFIRM_Action_Inc	0x01

Tabela 1: Exemplos de instruções da ferramenta ComFIRM

mensagem. Finalmente, a manipulação de elementos externos permite que a seleção de uma mensagem acione elementos como sinalizadores e temporizadores.

4 A Ferramenta ComFIRM

A ferramenta ComFIRM (*Communication Fault Injection through OS Resources Modification*) foi projetada para permitir a experimentação com vários elementos de seleção e manipulação de pacotes de rede e foi inserida no kernel do Linux. Ela permite todos os modos de seleção e manipulação de mensagens mencionados acima.

Como a ferramenta se situa dentro do núcleo do Sistema Operacional, seu controle se dá através de arquivos virtuais, situados no diretório `/proc/net`. Ao todo são quatro arquivos: `ComFIRM_Control`, que recebe comandos para ativação, desativação e controle do registro gerado pela ferramenta, `ComFIRM_Log`, do qual podem ser lidas as informações registradas pela ferramenta durante seu uso, e `ComFIRM_TX_Rules` e `ComFIRM_RX_Rules`, que contém as regras usadas para injeção de falhas de comunicação em mensagens transmitidas e recebidas, respectivamente.

Os experimentos são conduzidos através de conjuntos de regras, que são compostas de instruções primitivas de seleção e manipulação de mensagens. A ferramenta intercepta as rotinas `dev_queue_xmit()` e `netif_rx()`, pelas quais passam todas as mensagens transmitidas ou recebidas pelo núcleo do Linux. Isto permite ativar a ferramenta apenas quando uma mensagem é processada pelo Linux, minimizando a intrusão. A cada mensagem processada, as regras são percorridas e suas ações são efetuadas.

Como o processamento das regras deve ser bastante rápido, de forma a não adicionar um *overhead* excessivo ao tratamento de mensagens, elas são codificadas na forma de um *bytecode*, que se assemelha à uma linguagem de máquina de um processador, apenas sendo as instruções específicas para a injeção de falhas de comunicação. A tabela 1 mostra algumas instruções de seleção e manipulação de mensagens.

As instruções são concatenadas para formar uma regra. A avaliação começa com a primeira instrução e continua até que alguma instrução retorne resultado negativo. Por exemplo, um teste de contador que dê resultado falso termina a avaliação de uma regra. Ações sobre as mensagens sempre retornam resultado positivo. Sendo assim, para se criar uma regra que descarte a quinta mensagem de uma comunicação, utiliza-se uma ação de incremento de contador, um teste deste contador, e uma ação de descarte.

Codificando-se isto em *bytecode*, obtém-se `ComFIRM_Action_Counter` combinado com `ComFIRM_Action_Inc`, concatenado com `ComFIRM_Test_Counter` e finalmente a instrução `ComFIRM_Action_Drop`. Traduzindo para hexadecimal, isto resulta na seguinte seqüência: `0x79 0x00 0x20 0x00 0x05 0x00 0x00 0x00 0x60`. Obviamente a notação em *bytecode* não se adapta bem ao uso por pessoas, pois a memorização de todos os códigos se torna enfadonha. Pensando nisso, foi desenvolvida também uma interface gráfica, programada na linguagem Perl e usando o módulo PerlTk.

Esta interface (figura 1) facilita a execução de testes com a ferramenta ComFIRM,

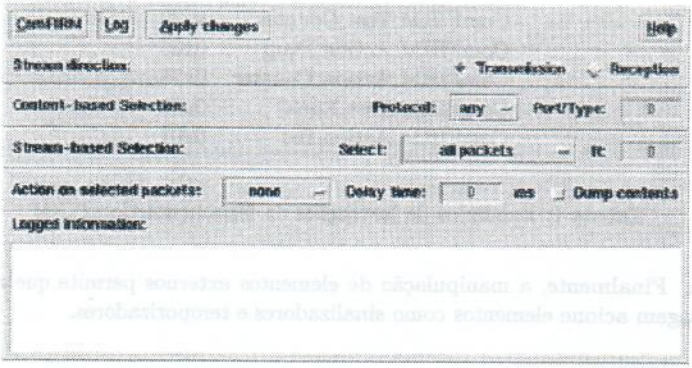

Figura 1: Interface gráfica experimental da ferramenta ComFIRM

porém não explora toda a sua potencialidade. Ainda assim, ela demonstra que é possível se criar interfaces amigáveis, que não exijam a memorização de inúmeros códigos. Esta ferramenta está concluída, e implementa todas as instruções de injeção de falhas de comunicação propostas. Nada impede, no entanto, que novas funcionalidades possam ser adicionadas com o tempo. Todo o código gerado é de livre distribuição e atualmente pode ser obtido por correio eletrônico com o autor.

5 Conclusão

Este artigo apresentou a técnica experimental de validação de protocolos de comunicação confiável através da Injeção de Falhas de Comunicação. Foram abordados técnicas de implementação de injetores de falhas, principalmente a de injeção de falhas por software, e especialmente a técnica de injeção de falhas de comunicação no nível de Sistema Operacional. Foi apresentada a ferramenta ComFIRM, que implementa a técnica demonstrada e se situa dentro do núcleo do Sistema Operacional Linux.

Referências

[CAR 98] CARREIRA, J; MADEIRA, H; SILVA, J. G. **Xception: A Technique for the Experimental Evaluation of Dependability in Modern Computers**. Transactions on Software Engineering, v. 24, n. 2, Fevereiro, 1998.

[ARL 90] ARLAT, J. et. al. **Fault-injection for dependability validation: a methodology and some applications**. IEEE Transactions on Software Engineering, Special Issue on Experimental Computer Science. New York, v. 16, n. 2, Fevereiro, 1990.

[BAR 99] BARCELOS, P. P; LEITE, F. O; WEBER, T. S. **Implementação de um Injetor de Falhas de Comunicação**. Anais do VIII Simpósio de Computação Tolerante a Falhas, 1999.

[DAW 96] DAWSON, Scott; JAHANIAN, Farnam; MITTON, Todd. **ORCHESTRA: A Fault Injection Environment for Distributed Systems**. University of Michigan, Department of Electrical Engineering and Computer Science, Techical Report n. 318, 1996.