UNIVERSIDADE DA BEIRA INTERIOR

SISTEMAS DISTRIBUÍDOS E TOLERÂNCIA A FALHAS
2009/2010

ConfErr: A Tool for Assessing Resilience to Human Configuration Errors

(Uma ferramenta para avaliar a resiliência dos erros de configuração humanos.)

Artigo por:
Lorenzo Keller
Prasang Upadhyaya
George Candea

Da:
Ecole Polytechnique F´ed´erale de Lausanne (EPFL), Switzerland

Estrutura da Apresentação

- Introdução
- Modelos de Erros de Configuração
- Estrutura do ConfErr
- Plugins de Geração de Erros
- Casos de Estudo
- Conclusão

ConfErr

- Resiliência: combinação de factores que propiciam condições ao sistema para enfrentar e superar problemas e adversidades.

- Benchmark: acto de executar um programa de computador que avalia a performance de outro programa ou objecto, executando para isso uma série de testes e ensaios sobre o mesmo.

O que é o ConfErr?

- Ferramenta que testa e quantifica a resiliência dos sistemas de software face aos erros de configuração induzidos por humanos.

- Usa modelos de erro humanos baseados em psicologia e linguística para gerar erros de configuração realista. Depois injecta esses erros e mede os seus efeitos, produzindo um perfil de resiliência dos sistemas que estão a ser testados.

- O perfil de resiliência captura o quão sensível é o sistema às diferentes classes de erros de configuração.

1 - ConfErr : Introdução

- Os erros humanos são uma causa dominante de falhas em sistemas de computadores.
- Há mais de duas décadas, 42% dos incidentes em instalações High-End foram atribuídos a erros de operadores humanos.
- Um estudo mais recente afirma que 58% dos problemas encontrados em sistemas de bases de dados se devem a erros feitos por administradores desses sistemas.

1 - ConfErr : Introdução

- Erros de configuração são erros complicados, porque demoram um longo tempo a serem descobertos e arranjados, o que leva a longos tempos de reparação.

- Treinar melhores operadores pode ajudar, mas muitas vezes com benefícios limitados. Por exemplo, operadores de plantas nucleares sofrem um treino extensivo, mas mesmo assim são responsáveis por 44%-52% dos erros significantes em reactores.

1 - ConfErr : Introdução

- É então imperativo que o software de sistema crítico seja resiliente face a estes erros, visto que compensar o desenho pobre de um sistema é difícil. Foi mostrado que soluções como redundância e replicação não melhoram a disponibilidade do sistema face aos erros de operador.

- Para desenhar sistemas menos vulneráveis a erros os engenheiros de software precisam de ferramentas que quantifiquem os benefícios oferecidos pelas diferentes técnicas e implementações. É preciso pois uma ferramenta de benchmark objectiva.

1 - ConfErr : Introdução

- A chave para fazer essa ferramenta com resultados uniformes, económicos e compreensivos é testar os erros de configuração de modo automático. Este artigo apresenta o ConfErr, uma ferramenta que tenta alcançar esses objectivos.

- O ConfErr converte o conhecimento de erros humanos desenvolvido por psicólogos e linguistas numa ferramenta automática de medição da resiliência de um sistema sobre os erros de configuração.

1 - ConfErr : Introdução

- Segundo o artigo e os seus autores, é a primeira ferramenta para medir estes erros deste modo.

- O ConfErr gera e injecta automaticamente erros nos ficheiros de configuração dos sistemas, calcula a resiliência do sistema sobre esses erros e cria o perfil de resiliência do sistema.

- Este perfil pode ser feito de duas formas: criar um output de linha de comandos durante o desenvolvimento ou então um benchmark para comparar dois sistemas de funcionalidade equivalente

2 - Modelos de erros de configuração

- GEMS (Generic Error Modeling System): identifica vários níveis cognitivos nos quais os humanos resolvem problemas e dividem-nos em três níveis diferentes.

- O nível de habilidade mais baixo (skill-based) é usado para tarefas comuns e repetitivas. Erros simples e lapsos deste mesmo nível contam como 60% de todos os erros humanos. Erros de escrita e de detecção de dígitos são exemplos de erros deste nível.

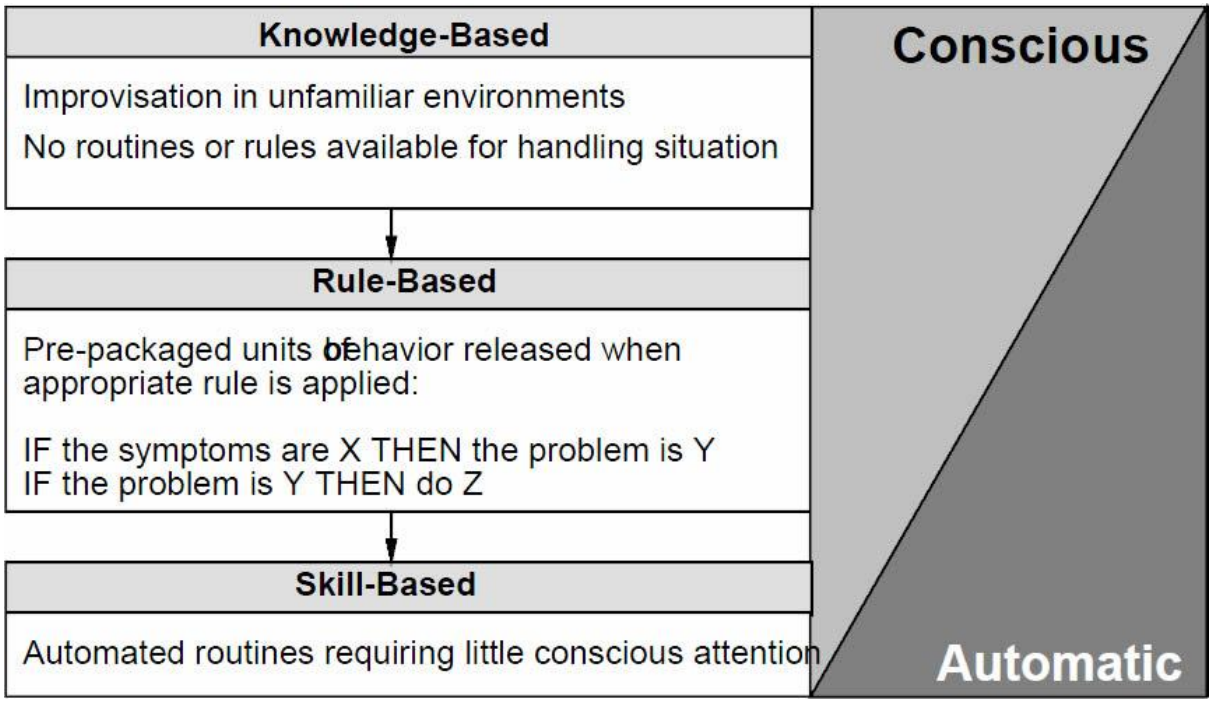
2 - Modelos de erros de configuração

- O nível seguinte (rule-based) é o nível em que a resolução dos problemas se obtém comparando a situação actual com instâncias prévias e usando as soluções descobertas anteriormente para resolver esses problemas. Erros deste nível contabilizam 30% de todos os erros humanos.

- O nível cognitivo mais alto (knowledge-based) é onde as tarefas são tratadas usando a razão sem o uso directo de regras ou habilidades prévias. Erros deste nível completam os restantes 10% dos erros humanos.

2 - Modelos de erros de configuração

- Níveis cognitivos do GEMS.



- Os geradores de erros do ConfErr englobam 3 modelos de erros para representar todos estes níveis cognitivos.

2.1 - Erros tipográficos

- Ocorrem durante o processo de dactilografia.
- Podem ser evitados com a revisão e correcção dos ficheiros de configuração recentemente editados e antes de estes serem aplicados ao sistema.
- Outra possibilidade é o próprio sistema identificar os problemas antes de aplicar os ficheiros. Mas à medida que os computadores ficam mais rápidos esta última abordagem torna-se mais usada, ao mesmo tempo que os programadores aumentaram a velocidade de programação e compilação sem reler o código.

2.1 - Erros tipográficos

- Os erros tipográficos podem ser divididos nas seguintes categorias:
  - Omissões: falta de uma letra numa palavra.
  - Inserções: uma letra extra introduzida numa palavra.
  - Substituições: troca de uma letra por uma outra incorrecta.
  - Alteração de letra maiúscula: letra maiúscula quando deveria ser minúscula e vice-versa.
  - Transposições: duas letras adjacentes numa palavra são trocadas.

2.2 - Erros estruturais

- Ficheiros de configuração geralmente têm uma estrutura bem definida. Há erros relacionados com esta estrutura pertencem aos três níveis cognitivos.

- No nível skill-based os erros que se tentaram representar são as repetições das directivas de configuração e má colocação das mesmas. Isto pode resultar do uso das operações de copy-paste.

2.2 - Erros estruturais

- Ao nível Rule-based foram modelados os erros de operador que resultam do uso de um formato de configuração similar ao correcto mas incorrecto.

- Erros do nível Knowledge-based tendem a resultar de uma incompatibilidade entre o modelo mental que o operador tem do sistema e o modelo actual do sistema.

2.3 - Erros de semântica

- Erros de semântica são introduzidos somente quando operando no nível cognitivo mais elevado.

- Neste modelo, são capturadas duas classes de erros semânticos:
  - O primeiro são as configurações inconsistentes, nas quais os parâmetros requeridos não são satisfeitos.

  - O segundo consiste nos erros que ocorrem quando o operador não sabe exactamente o significado de um parâmetro e usa-o para configurar um aspecto diferente do sistema.

3 - Estrutura do ConfErr

- O objectivo do ConfErr é transformar modelos de erro, como os que vimos anteriormente, em ferramentas práticas.

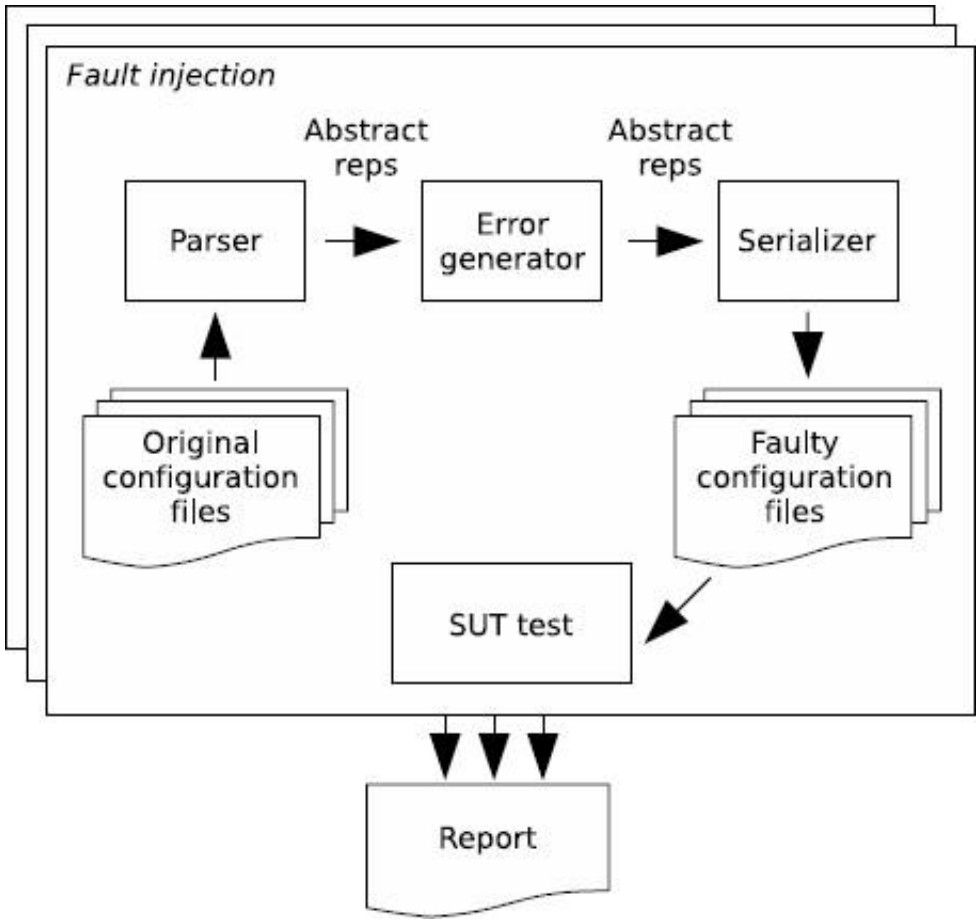
- O ConfErr gera os erros, injecta-os nos ficheiros de configuração, inicia o sistema sobre teste SUT (System-under-test) e mede o impacto de cada erro no sistema: tudo sem requerer intervenção humana.

3.1- Design

- O ConfErr pega nos ficheiros de configuração, altera-os adicionando-lhes erros, depois testa o sistema sobre teste (SUT) com as novas configurações e apresenta um relatório final.

3.1- Design

- Esta operação é ilustrada na imagem seguinte:



3.1- Design

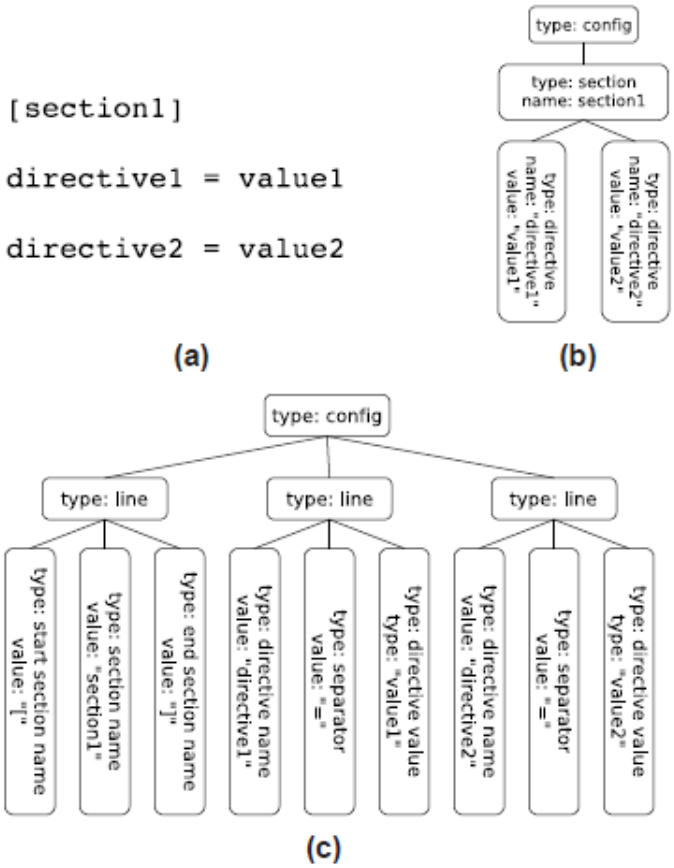
- Para gerar um perfil de resiliência para um dado sistema S, o ConfErr toma como input os ficheiros de configuração de S, um parsing especifico do sistema, um plugin de geração de erros e testes funcionais de domínio especifico.
- São então gerados ficheiros de configuração com erros para teste com o SUT, com três respostas possíveis:
  - *SUT failed to start.*
  - *SUT started but could not complete funcional tests.*
  - *All tests passed.*

3.1- Design

- O perfil de resiliência é então criado, indicando o resultado de cada teste de injecção, com o erro injectado e o comportamento resultante.

3.2 - Representação da configuração

- O modelo pode ser visto como uma árvore de informação de itens.



3.2 - Representação da configuração

- Para permitir cenários de injecção com flexibilidade, o ConfErr divide o processo de parsing em duas fases:
  - O ficheiro de configuração é colocado (*parsed*) numa árvore com uma representação XML.
  - Esta representação é mapeada para o formato requerido pelo plugin de erros, usando XSLT.
- **XSLT** é uma linguagem utilizada para transformar documentos XML num dado formato, para documentos XML de outros formatos.

3.2 - Representação da configuração

- Isto é necessário porque sistemas diferentes utilizam maneira diferentes de exprimir a mesma configuração.
- Não é possível utilizar uma única representação para representar todos os sistemas e tipos de falhas.

3.3 - Modelos de erro

- Modelos de erro são expressos com auxilio de um conjunto de modelos base.
- Estes modelos descrevem a transformação de uma árvore de configuração, tal como a duplicação/eliminação de um nodo.
- São parametrizados, permitindo que o utilizador especifique quais as transformações a aplicar em cada circunstancia.
- Dados um modelo e vários ficheiros de configuração, o ConfErr consegue gerar um conjunto de cenários de falha.

4 - Plugins de geração de erros

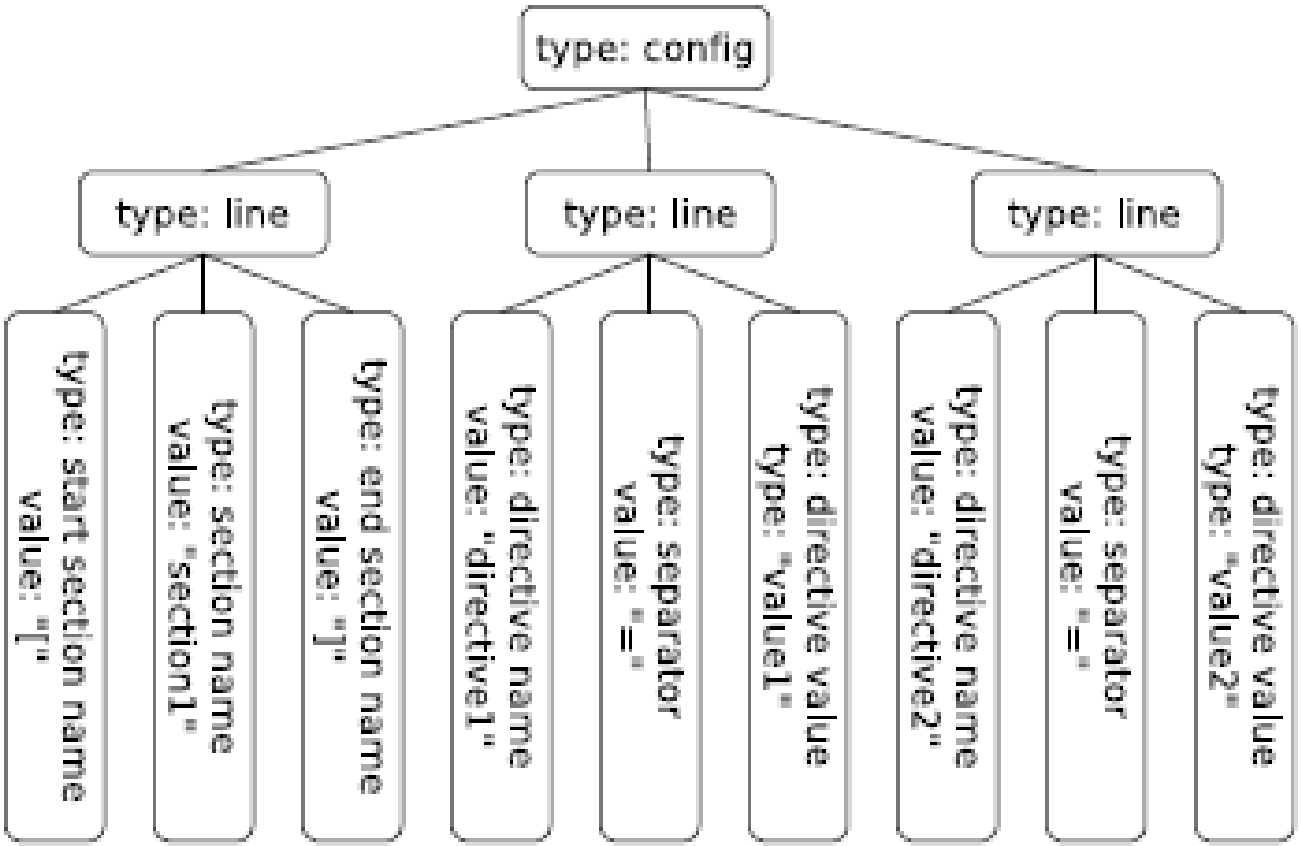
- Os geradores de erros do ConfErr são responsáveis por especificar a sequencia de mutações a fazer nas configurações, de modo a gerar perfis com resiliência significativa.

4.1 – Plugin de Erros Ortográficos

- Implementa uma colecção de sub-modelos, um para cada tipo de erro (erros ortográficos):
  - Omissões
  - Inserções
  - Substituições
  - Alteração de Letra Maiúscula
  - Transposições
- O plugin gera erros aos escolher subconjuntos aleatórios de erros ortográficos.

4.1 – Plugin de Erros Ortográficos

- Os ficheiros de configuração são representados como uma lista de tokens com tipos associados.

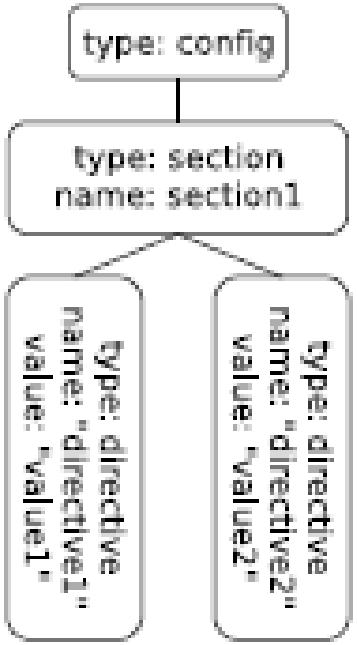


4.2 – Plugin de Erros Estruturais

- Implementa (erros estruturais):
  - Omissões
  - Inserções no sitio errado
  - Duplicação de directivas de configuração

4.2 – Plugin de Erros Estruturais

- Para a injecção de erros estruturais, os ficheiros de configuração são representados como uma árvore de directivas e secções.



4.3 – Plugin de Erros Semânticos

- Um plugin de erros semânticos, codifica erros que são específicos a uma classe de sistemas e que são normalmente baseados em documentos de referência.
- Por exemplo:
  - O RFC-1912 define uma lista de erros comuns para servidores DNS.

5 – Casos de estudo

- Nesta secção é ilustrado o uso do ConfErr para testar a resiliência aos erros de configuração em sistemas amplamente usados:
  - As bases de dados Postgres 8.2.5 e MySQL 5.1.22
  - O servidor web Apache 2.2.6
  - Os servidores de DNS BIND 9.4.2 e Djbddns 1.05.

5 – Casos de estudo

- O ConfErr necessita de 3 componentes especifico de sistema:
  - Ficheiros de configuração iniciais.
  - Parsers/serializers para configurações.
  - Scripts para organizar o ambiente, parar/arrancar o sistema e uma "diagnostic suite" para determinar os resultados da injecção dos erros.

- Combinando estes componentes com os geradores de erros e o ConfErr produz automaticamente um perfil de resiliência para o SUT.

5 – Casos de estudo

- Foram usados os ficheiros de configuração base que são lançados com os sistemas alvo.

- Para o MySQL, Postgres e Apache, os ficheiros de configuração consistem em secções, sendo estas constituidas por linhas, que podem estar vazias ou conter uma directiva.

- Uma directiva típica consiste num nome, um separador e um valor.

5.1 - Resiliência a Erros Ortográficos

- Resiliência a erros tipográficos:
  - Apesar de ser uma tarefa lenta e dispendiosa, encontrar erros tipográficos é bastante trivial, por isso é esperado que os SUT se portem bem face a esses erros.
  - Neste caso foi medida a resiliência para o MySQL, Postgres e Apache.
  - Foram injectados 3 tipos de erros:
    - Eliminação de directivas inteiras
    - Erros tipográficos no nome das directivas
    - Erros tipográficos nos valore das directivas

5.1 - Resiliência a Erros Ortográficos

- Resiliência a erros tipográficos:
  - Alguns erros foram detectados no arranque do sistema, outros por testes e outros nem sequer foram detectados.
  - A tabela seguinte sumariza os resultados.

	MySQL	Postgres	Apache
# of Injected Errors	327 (100%)	98 (100%)	120 (100%)
Detected by system at startup by functional tests	 270 (83%) 1	 76 (78%) 0	 46 (38%) 6 (5%)
Ignored	56 (17%)	22 (22%)	68 (57%)

5.1 - Resiliência a Erros Ortográficos

- Resiliência a erros ortográficos:
  - Testes funcionais não oferecem um poder de detecção significante comparado com os testes de arranque, à excepção de erros nas "listening ports" que são a razão para o valor de 5% nos erros detectados pelos testes funcionais do Apache.
  - Os perfis de resiliência revelam varias fraquezas inesperadas nos SUTs.
    - Por exemplo, o MySQL ignora valores fora dos limites.
    - O MySQL aceita sufixos errados, por exemplo, no caso de ter 1M3 (em vez de 13M) assimila como 1 milhão, parando logo quando encontra o M.

5.1 - Resiliência a Erros Ortográficos

- Os perfis de resiliência revelam varias fraquezas inesperadas nos SUTs.
  - Uma falha de desenho do MySQL permite que erros graves não sejam detectados. Esta falha resulta de um ficheiro de configuração que é usado para o servidor da DB e para outras ferramentas, tal como o "backup". Os erros na parte do ficheiro que configura o servidor DB é verificada no arranque do sistema enquanto que o resto do ficheiro não é verificado. O que pode levar a um erro grave aquando do uso da ferramenta de backup, que normalmente corre automaticamente e durante a noite, não dando feedback dos erros ao administrador.

5.1 - Resiliência a Erros Ortográficos

- Os perfis de resiliência revelam varias fraquezas inesperadas nos SUTs.

  - O parser do servidor Apache tambem revela fraquezas, por exemplo, no caso das directivas do tipo MIME só deveria tomar valores no formato “type/subtype”, como definido no RFC-2045. No entanto o Apache aceita qualquer tipo de strings nesse campo.

  - Outro erro no Apache é que no campo de endereço de email da directiva ServerAdmin só deveria aceitar um endereço de email (com @), mas tal como no caso do MIME, aceita qualquer tipo de strings.

5.1 - Resiliência a Erros Ortográficos

- Por vezes também são reveladas mais valias dos SUT nos perfis de resiliência.
  - A base de dados Postgres impõe limites nas directivas, por exemplo, um erro tipográfico introduzido na directiva max_fsm_pages (mudando o valor 15600 para 153600) resulta num aviso imediato do Postgres com uma mensagem de erro que indica que o máximo para este parâmetro é 16000.
  - Estes tipos de limites ajudam a encontrar e corrigir erros que poderiam resultar em implicações futuras de difícil diagnostico.

5.2 - Resiliência a Erros Estruturais

- Ficheiros de configuração de sistemas diferentes partilham frequentemente uma estrutura similar.
- Esta similaridade convida o utilizador a usar o modelo mental de um sistema para configurar outro.
- No entanto, algumas diferenças nos métodos de configuração dos sistemas podem levar à ocorrência de alguns erros.

5.2 - Resiliência a Erros Estruturais

- O ConfErr cria variações de ficheiros de configuração automaticamente.
- Estes podem ser usados para verificar o nível da implementação do operador.
- Podem também ser usados para descobrir que classes de variações são suportadas por um dado sistema.

5.2 - Resiliência a Erros Estruturais

- É esperado que as diferentes classes de variações sejam acmomodadas pelos SUT Apache, MySQL e Postgres.
  - Qualquer ordem das secções são permitidas
  - Qualquer ordem das directivas é permitido dentro de uma secção.
  - As directivas e os nomes de secção são “case sensitive”.
  - Espaços em branco entre nomes de directivas, separadores e valores são ignorados.
  - Os nomes das directivas podem ser truncados, se isto não implica uma colisão de nomes.

5.2 - Resiliência a Erros Estruturais

- Foram feitos uma série de experiencias nos 3 SUTs para descobrir quais dessas variações são suportadas e se a sua implementação está correcta.

- Para cada classe de variação foram testados 10 ficheiros de configuração diferentes.

5.2 - Resiliência a Erros Estruturais

- A tabela seguinte mostra os resultados obtidos.

	MySQL	Postgres	Apache
Order of sections	Yes	n/a	n/a
Order of directives	Yes	Yes	Yes
Spaces near separators	Yes	Yes	Yes
Mixed-case directive names	No	Yes	Yes
Truncatable directive names	Yes	No	No
% of assumptions satisfied	80%	75%	75%

5.2 - Resiliência a Erros Estruturais

- Notou-se que a maioria dos SUTs aceita quase todas as mutações, mas nenhum aceita todas.

- Os 3 sistemas deveriam ser flexíveis e aceitar todas as mutações.

5.3 - Resiliência a Erros semânticos

- O RFC-1912 define uma lista de erro de configuração comuns em DNS.
- Estes ocorrem a múltiplos níveis, desde a escolha de nomes até à relação entre registos de diferentes servidores.
- Usamos o ConfErr para testar o comportamento do BIND e do djbdns face a estes erros de configuração.

5.3 - Resiliência a Erros semânticos

- A tabela seguinte mostra o comportamento dos servidores DNS face a alguns dos erros descritos anteriormente.

Err#	Description of fault	BIND	djbdns
1.	Missing PTR	not found	N/A
2.	PTR pointing to CNAME	not found	N/A
3.	dupl name for NS and CNAME	found	not found
4.	MX pointing to CNAME	found	not found

5.3 - Resiliência a Erros semânticos

- O formato de configuração usado pelo djbdns permite que o administrador possa definir vários registos relacionados com apenas uma directiva.

- Nos ficheiros de configuração do djbdns não foi possível injectar os erros (1) e (2), porque não conseguiu voltar a transformar os ficheiros afectados novamente num ficheiro de configuração. Esta pode ser considerada uma mais valia do servidor djbdns .

5.3 - Resiliência a Erros semânticos

- Usar o ConfErr para injectar erros de configuração do mundo real que vão para alem de erros meramente sintácticos permite o estudo e teste do comportamento geral do sistema, não apenas do seu parser de configuração.

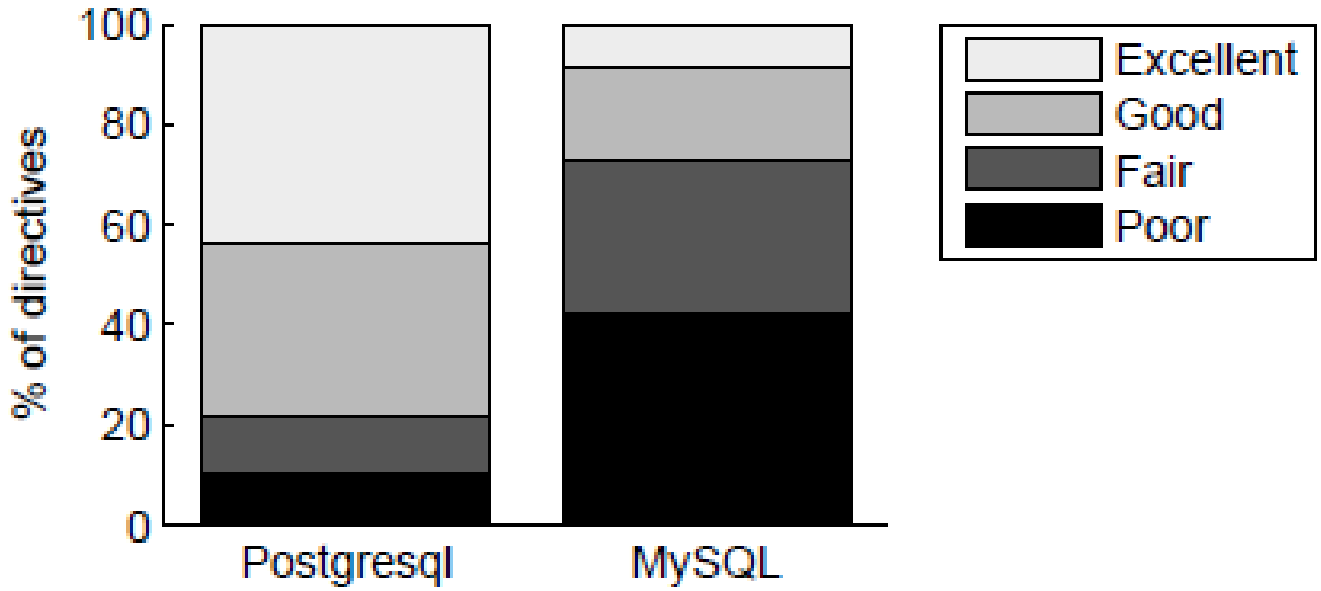
- Assim o administrador pode usar o ConfErr para identificar areas do sistema que precisem de melhorias

5.4 – Comparar Erros de Resiliência

- O ConfErr utiliza um script de benchmark que transforma automaticamente ficheiros de configuração iniciais em novos ficheiros válidos.
- Depois, cria ficheiros de configuração com falhas baseando-se nestes novos ficheiros e verifica o comportamento do sistema.
- Os erros são injectados próximo do local onde o ficheiro foi modificado (validamente), de modo a simular o modo como os erros normalmente surgem nas configurações.
- Isto simula o processo de configuração humano e constitui um benchmark humano primitivo.

5.4 – Comparar Erros de Resiliência

- Esta abordagem foi utilizada para comparar o Postgres com o MySQL.



Poor = 0-25% falhas detectadas

Fair = 25%-50% falhas detectadas

Good = 50%-75% falhas detectadas

Excellent = 75%-100% falhas detectadas

5.4 – Comparar Erros de Resiliência

- O Postgres conseguiu detectar mais de 75% dos erros ortográficos em quase 45% das suas directivas.
- O MySQL detectou menos de 25% dos erros ortográficos na mesma fracção das suas directivas.
- Estes resultados explicam-se pelo facto do Postgres possuir um forte mecanismo de verificação de constrangimentos para os seus parâmetros numéricos.

6 – Conclusão

- Erros de configuração são as causas dominantes para falhas de sistemas, mas são raramente tidas em conta ao desenhar, testar e avaliar sistemas.
- Teste directo deste tipo de erros tradicionalmente envolve pessoas reais, portanto pode tornar-se complexo, subjectivo e difícil de reproduzir.

6 – Conclusão

- O ConfErr é uma ferramenta que testa automaticamente o comportamento de um sistema quando encontra erros de configuração humanos.
- Deste modo, são utilizados modelos psicológicos e linguísticos que foram criados com base em estudos do comportamento humano.
- O ConfErr gera automaticamente erros de configuração realísticos, injecta-os em sistemas e determina o seu impacto nos mesmos.
- O ConfErr permite testar a resiliência de sistemas reais (com MySQL, Postgres, Apache, BIND, entre outros), de forma simples e sem esforço.

Obrigado!