MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
ENGENHARIA DE COMPUTAÇÃO - SE/8

Luan Ferreira Cardoso, Ricardo Sollon Zalla, Venicius Gonçalves da Rocha Junior

DevOps: aproximando a área de desenvolvimento da operacional

Rio de Janeiro
4 de maio de 2016

Luan Ferreira Cardoso, Ricardo Sollon Zalla, Venicius Gonçalves da Rocha Junior

DevOps: aproximando a área de desenvolvimento da operacional

Trabalho apresentado ao Curso de Engenharia de Computação do Instituto Militar de Engenharia como Verificação Especial do Projeto de Fim de Curso.

Instituto Militar de Engenharia

Orientador: Scooper

Coorientador: Coorientador ?????

Rio de Janeiro

4 de maio de 2016

Instituto Militar de Engenharia
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP: 22290-270

Cardoso, Luan; Zalla, Ricardo e Gonçalves, Venicius
S586d DevOps: aproximando a área de desenvolvimento da operacional / Luan Ferreira Cardoso, Ricardo Sollon Zalla, Venicius Gonçalves da Rocha Junior. - Rio de Janeiro: Instituto Militar de Engenharia, 2016.

17f. : il., graf., tab. : -cm.

Projeto de Fim de Curso - Instituto Militar de Engenharia
Orientador: Scooper.

1 - DevOps 2 - Desenvolvimento e Operação

CDU ???.???.??

Luan Ferreira Cardoso, Ricardo Sollon Zalla, Venicius Gonçalves da Rocha Junior

DevOps: aproximando a área de desenvolvimento da operacional

Trabalho apresentado ao Curso de Engenharia de Computação do Instituto Militar de Engenharia como Verificação Especial do Projeto de Fim de Curso.

Trabalho aprovado. Rio de Janeiro, 4 de maio de 2016:

Prof. Scooper
Orientador, D. Sc., do IME

Prof. Humberto
Convidado, M. c., do IME

Prof. Chorem
Convidado, D. c., do IME

Rio de Janeiro

4 de maio de 2016

Sumário

Resumo

Resumo em pt

Palavras-chave: DevOps, desenvolvimento, operação, ambientes.

Abstract

Abstract in English

Keywords: DevOps, development, operation, environment.

1 Introdução

1.1 Motivação

1.2 Objetivo

1.3 Justificativa

1.4 Metodologia

Para alcancar os objetivos desejados, o trabalho divide-se em três etapas:

1. Obtenção dos criptogramas
2. Projeto da rede neural
3. Aprendizado e análise da rede

A primeira etapa consiste em obter os textos em claro, obter as chaves, e obter os criptogramas. A parte de obtenção de textos serve para evitar *overfitting*, que consiste na perda de generalidade que ocorre em procedimentos de aprendizado de máquina quando um pequeno erro dentro da amostra deixa de indicar um erro pequeno fora da amostra,podendo levar ao efeito contrário [1].

A intenção é obter textos que não tenham correlação entre si. Foi considerado obter os textos da *DBPedia*.

A *DBPedia* é uma base de dados criada com base na *Wikipedia*, que relaciona as informações contidas em seus artigos, com dados de locais, pessoas, conceitos; uma base ontológica. O acesso à *DBPedia* e feito através de requisições *SPARQL* ao servidor da *DBPedia*. Foram obtidos os 400 textos mais extensos da base em português, com base nos resumos dos artigos, e depois concatenados dois a dois, seguindo o modelo do n-ésimo maior com o n-ésimo menor. Os tamanhos dos textos variam entre 35KB e 11KB.

Após a obtenção dos textos, eles devem ser preprocessados a fim de que possam ser criptografados.

Após isso, os textos ficam prontos para serem criptografados. Para a criptografia, foi utilizada a biblioteca *PyCrypto*[1], que foi utilizada tambem para a geração das chaves. Foram geradas mil (1000) chaves para cada algoritmo. As chaves foram geradas segundo o tamanho máximo da chave aceito pelos algoritmos. Com isso, tem-se um milhão de criptogramas para a rede neural.

A segunda etapa consiste

1.5 Estrutura

[1] Documentação disponível em: <https://www.dlitz.net/software/pycrypto/api/current/>

2 Conceitos iniciais

2.1 Algoritmos criptográficos

2.1.1 Advanced Encryption Standard - AES

O AES é uma primitiva de criptografia que se destina a composição sistemas simétricos. É uma cifra de bloco, ou seja, divide a mensagem em blocos de tamanho fixo (128 bits, ou 16 bytes, no caso do AES). Como toda cifra de bloco, pode ser transformada numa cifra de fluxo (opera em dados de tamanho arbitrário) através de um modo de operação. Pode ser utilizada com chaves de 128, 192 ou 256 bits (o algoritmo Rijndael, que originou o AES, permite tamanhos diversos de chaves).

Em outras palavras, é um algoritmo que recebe como entrada para cifrar um bloco de 128 bits e uma chave do tamanho escolhido, e devolve uma saída também de 128 bits (criptograma). A decifragem recebe como entrada o criptograma (bloco de 128 bits) e devolve como saída um bloco de 128 bits. Se a chave for a correta, a saída será idêntica à mensagem original.

Para dimininuir a possibilidade de quebra da cifra, busca-se minimizar qualquer correlação visível entre a entrada e a saída, de modo que a mesma] possa ser deduzida simplesmente observando-se um número muito grande de ccriptogramas (ou de pares mensagem/criptogramas). Então, usa-se uma série de rodadas em que os bytes sofrem transformações não lineares, porém reversíveis.

2.1.1.1 Corpo de Galois

Todas as operações no AES tratam os bytes de entrada como um corpo finito (ou corpo de Galois) em 2^8 (ou GF(2^8)). Define-se o conjunto e suas operações a seguir:

- O conjunto $[0, 255]$ são todos os valores possíveis para um byte e um desses elementos é chamado “zero” (no caso, o próprio 0);
- A adição, que se aplica a quaisquer dois elementos nesse conjunto e cujo resultado também é um elemento desse conjunto, consiste na operação XOR (OU exclusivo).

- A multiplicação define-se escrevendo o byte na forma polinomial (cada bit é representado pelo coeficiente do polinômio do sétimo grau). Opera-se a multiplicação de maneira convencional, atentando somente a execução da soma dos monômios conforme definido no item acima (somar x com x, por exemplo, teria como resultado $0x$, o que corresponde ao bit 0). Após a obtenção do polinômio produto, faz-se a redução modular via XOR com os coeficientes do polinômio redutor $x^8 + x^4 + x^3 + x + 1$ (alinhando-se os dígitos mais significativos), até a obtenção de um polinômio de grau menor que o redutor.

Em cada rodada são executados os procedimentos na seguinte ordem (exceto a última, que não possui o passo de trocar colunas):

1. S-BOX de Rijndael (ByteSub());
2. Deslocamento de linhas (ShiftRow())
3. Permutação de colunas (MixColumns())
4. Adição da chave de rodada (AddRoundKey())

A seguir é descrito de forma mais minuciosa cada método executado.

2.1.1.2 S-BOX de Rijndael

Com base na aritmética em GF(2^8), foi concebida uma tabela de consulta S-Box, destinada a transformar um byte em outro diferente, de uma forma não linear. Duas tabelas são usadas: uma para a função direta e outra pra inversa.

1. Em primeiro lugar, calcula-se o inverso multiplicativo do byte (byte que multiplicado pelo substituído gera o byte "01". O zero não tem inverso, então ele é mapeado para ele mesmo;
2. Em seguida, os oito bits do resultado são submetidos a uma transformação afim, para tornar o método mais resistente contra ataques algébricos, conforme $b'_i = b_i + b_{(i+4)mod8} + b_{(i+5)mod8} + b_{(i+6)mod8} + b_{(i+7)mod8} + c_i$, onde b'_i é o novo bit, b_i se refere ao bit de posição i no byte substituído e c_i se refere ao bit de posição i no byte $63h$.

O resultado é um novo bloco em que os bytes aparecem substituídos pelos obtidos na S-BOX. Essa tabela foi projetada com o objetivo de ser resistente à criptanálise linear ou

diferencial, e permitindo a substituição da transformação afim por outra caso se descubra algum backdoor no futuro. Em resumo, é uma função invertível cuja saída não corresponde ao texto em claro.

2.1.1.3 Deslocamento de linhas

As linhas da matriz do bloco são ciclicamente deslocadas, onde a linha 0 permanece e demais são deslocada por um dado número de bytes. Os valores do deslocamento estão em função do tamanho do bloco e são descritos na tabela abaixo:

$n_{colunas} \times n_{colunas\ de\ estado}$	**1**	**2**	**3**
4	1	2	3
6	1	2	3
8	1	3	4

2.1.1.4 Permutação de Colunas

Cada coluna de um estado é multiplicada e reduzida módulo $m(x) = x^4 + 1$ por um polinômio constante $p(x) = 03hx^3 + 01hx^2 + 01hx + 02h$ na forma matricial, na chamada matriz circulante, onde os coeficientes do polinômio compõe a primeira coluna e as demais obtidas por permutação circular dos elementos da coluna. No caso de um bloco composto por 4 linhas, a matriz fica representada por:

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

2.1.1.5 Adição da Chave de Rodada

Uma chave é adicionada ao estado utilizando um XOR. A chave é derivada da chave K original através do procedimento Key Schedule() e possui tamanho igual ao bloco do estado. O procedimento Key Schedule() é feito da seguinte forma:

1. Total de chaves de rodada = $TamanhodoBloco \times (N\acute{u}meroderodadas + 1)$;
2. A chave K inicial é expandida usando o procedimento KeyExpansion(), que depende do algoritmo AES utilizado (AES-128,AES-192 ou AES-256);

3. As chaves são extraidas da chave expandida de p em p words, onde p é o número de colunas do bloco cifrado, denominado estado. Em resumo, as k_i chaves recebem as$i-$ésimas words da chave expandida gerada pelo método KeyExpansion().

Após o término das (10,12 ou 14) rodadas escolhidas, obtem-se o criptograma do texto de entrada.

Para decriptografar, basta executar as funções inversas baseadas em $GF(2^8)$, seguindo a sequência reversa de transformações.

2.1.2 Alleged Rivest's Cipher version 4 - ARC4

asf

2.1.3 Blowfish

2.1.4 Data Encryption Standard - DES

O algoritmo de criptografia *Data Encryption Standard* foi desenvolvido pela IBM na década de 70, e foi largamente utilizado pela indústria, sendo escolhido pela NSA como um padrão para criptografia. [2]

Esse é um algoritmo de bloco, ou seja, ele divide a mensagem em blocos (no caso do DES, de 8 bytes), e cada bloco é criptografado/decriptografado separadamente.

A criptografia consiste em 16 etapas, iterativamente, onde em cada etapa consiste em uma permutacao das metades do bloco, e da aplicação de uma funcao especifica sobre uma metade do bloco.

Essa operação específica consiste em permutações com expansão e aplicação da operação XOR, e a aplicação de S-boxes (caixas de substituição).

2.1.5 Triple Data Encryption Standard - DES3

O algoritmo Triplo DES é um reforço ao DES para aumentar a sua segurança. Ele é a aplicação do DES três vezes seguidas, utilizando chaves de 128 ou 192 bits. Quando se utiliza uma chave de 192 bits, esta é subdividida em 3 chaves de 64 bits ($K1$, $K2$ e $K3$). Caso a chave seja de 128 bits, está é dividida em 2 chaves ($K1$ e $K2$) e a terceira chave é adotada igual à segunda ($K2 = K3$).

A aplicação se da pelo seguinte método (para criptografia)[2]:

1. Primeiro encripta-se usando a primeira chave ($K1$)
2. Depois, decripta-se o resultado do primeiro passo usando a segunda chave ($K2$)
3. E, por ultimo, encripta-se novamente usando a terceira chave sobre o resultado do segundo passo ($K3$)

A decriptografia se da fazendo o reverso da criptografia.

2.2 Redes Neurais

As redes neurais são ferramentas de aprendizado de máquina largamente utilizadas em problemas de classificação. Haykin [3] define:

> Uma rede neural é um processador maciçamente paralelamente distribuído constituído de unidade de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos:
>
> 1. O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem.
> 2. Forças de conexão entre neurônios, conhecidos como pesos sinápticos, são utilizados para armazenar o conhecimento adquirido.

2.2.1 Benefícios das redes neurais

As redes neurais são um recurso computacional com poder de generalização e de estrutura maciçamente paralela. Por esse motivo, elas podem ser implementadas por meio de núcleos integrados que realizam operações simples.

As redes neurais apresentamas seguintes propriedades[3]:

1. *Não linearidade.* Um neurônio artificial pode ser linear ou não-linear

symbol	value	unit
zNa	11	-
zF	9	-
$EmaxNa$	0.545	$[MeV]$

3 Metodologia para criptoanálise

3.1 AAA

4 Conclusões

Texto Conclusão

Referências

1 ABU-MOSTAFA, Y. S.; MAGDON-ISMAIL, M.; LIN, H.-T. *Learning from data: a short course*. [S.l.]: AMLbook.com, 2012.

2 STALLINGS, W. *Cryptography and Network Security: Principles and Practice*. Prentice Hall, 2011. ISBN 9780136097044. Disponível em: <https://books.google.com.br/books?id=wwfTvrWEKVwC>.

3 HAYKIN, S. *Neural Networks and Learning Machines*. Prentice Hall, 2009. (Neural networks and learning machines, v. 10). ISBN 9780131471399. Disponível em: <https://books.google.com.br/books?id=K7P36lKzI_QC>.