

Redes Neurais e análise de potência

Neural Networks and power analysis

Jorge Luis Varella

Universidade Católica de Petrópolis
jorge.varella@icloud.com

Giovane Quadrelli

Universidade Católica de Petrópolis
giovane.quadrelli@ucp.br

Resumo

Este artigo apresenta um breve resumo sobre o algoritmo de aprendizado não supervisionado K-Means e suas características, no que tange a sua implementação básica e aplicabilidade. O artigo descreve a sua utilização em áreas onde a clusterização é fator primordial. Uma análise da utilização do algoritmo é feita sob a ótica de análise de potência em um ambiente de criptoanálise em criptografia de curva elíptica e criptografia RSA.

Palavras-chave: K-Means. Criptoanálise. Clusterização

Abstract

This article presents a brief summary about the K-Means unsupervised learning algorithm and its characteristics, with respect to its basic implementation and applicability. The article describes its use in areas where clustering is a primary factor. An analysis of the use of the algorithm is made from the perspective of power analysis in a cryptanalysis environment in elliptic curve cryptography and RSA encryption.

Keywords: K-Means. Cryptanalysis. Clustering

1. INTRODUÇÃO

Para prover segurança em um ambiente computacional pode-se aplicar diversas técnicas, sendo uma delas a criptografia. Uma criptografia forte, (que pode ser medida pelo tempo e recursos que se exige para recuperar o dado), precisa resistir as diversas formas de ataque que buscam obter do sistema informações sigilosas. Estes ataques podem ocorrer no

** Received 28 June 2017; received in revised form in 14 September 2017; accepted in 6 December 2017; published online 22 December 2017.

próprio algoritmo de criptografia, assim como em problemas computacionais subjacentes decorrentes de sua implementação (LADEIRA, 2016).

Uma das classes deste tipo de ataque é o de canal lateral, que faz uso de informações colidas durante a execução de uma primitiva criptográfica. Este tipo de ataque utiliza variações no tempo de execução, no consumo de energia e emanações eletromagnéticas dentre outras características do sistema a ser atacado.

Contra medidas podem ser executadas nestes ataques como por exemplo modificações de software e hardware.

Este artigo irá abordar o algoritmo *K-Means* empregado para *clustering* nos ataques de consumo de energia.

2 ALGORITMO *K-MEANS*

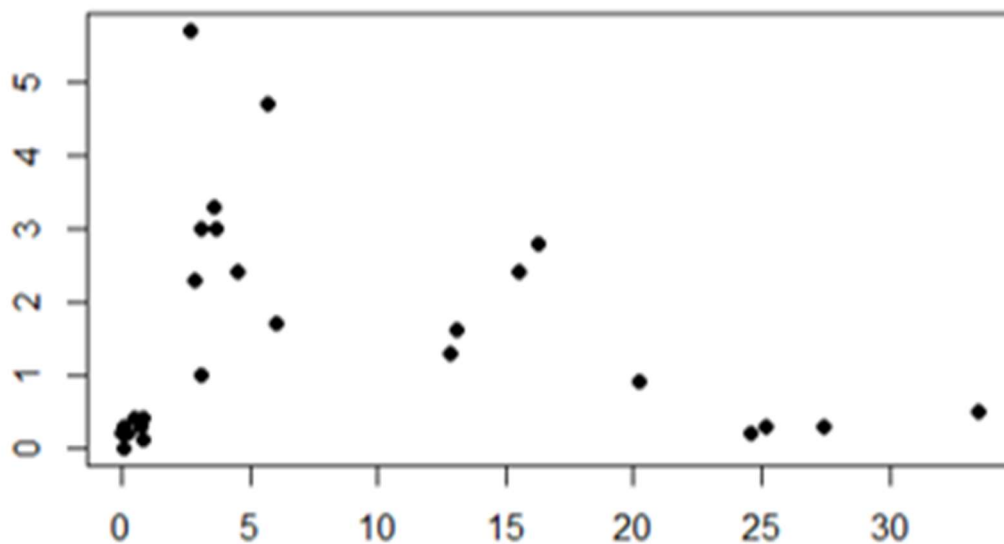
O *K-Means* é um algoritmo de aprendizado não supervisionado, comumente utilizado em mineração de dados e na resolução de problemas de agrupamentos.

Diferente do aprendizado supervisionado, onde se informa ao sistema o que ele deve procurar ou aprender, no aprendizado não supervisionado não se sabe exatamente o que está se tentando ensinar ao sistema, isso faz com que seja preciso recorrer a agrupadores lógicos de segmentação para encontrar similaridade entre os dados da amostra, e como isso seja possível definir um padrão (NOGARE, 2016).

Definido o padrão, se assume que é isto que está se tentando ensinar ao sistema, e que por sua vez irá aprender a encontrar este padrão sempre que for acionado. A partir deste padrão, todos os novos itens que tenham a mesma similaridade com aquele segmento serão agrupados.

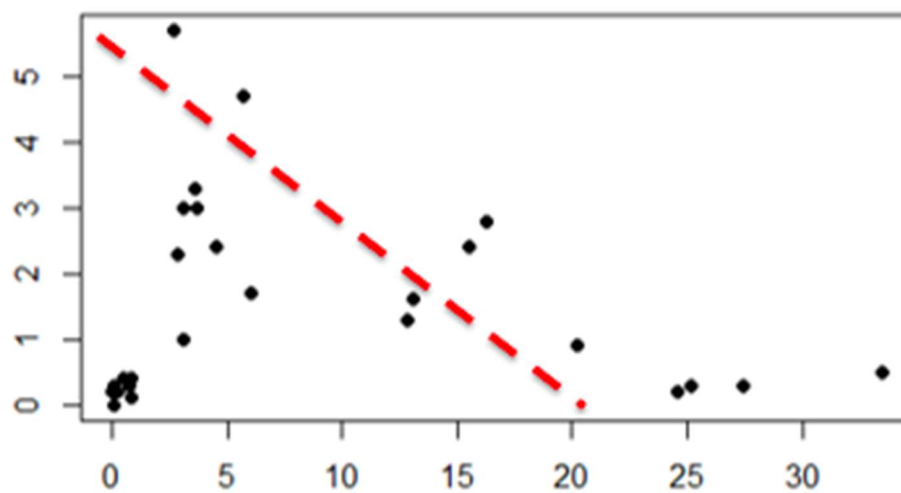
Na figura 1 é possível observar o modelo original e nas figuras 2 e 3 a separação de grupos, que pode ser feito de diversas formas.

Figura 1 – Modelo original



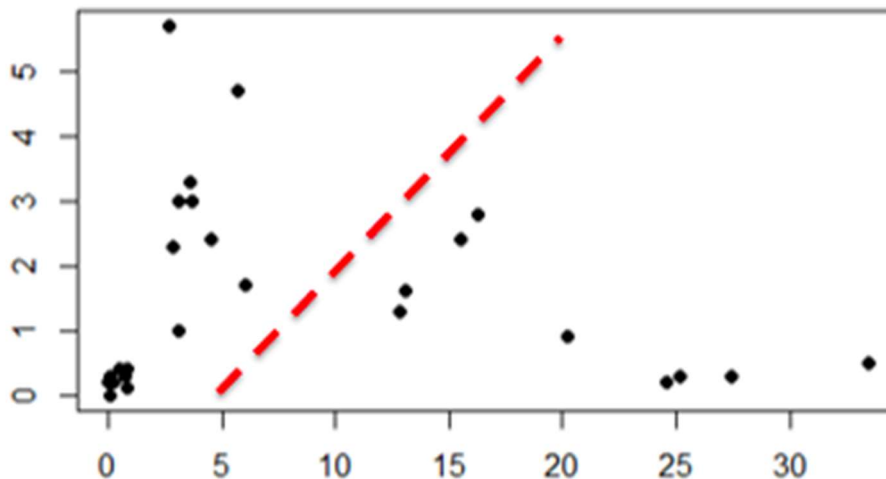
Fonte: MOORE, 2016

Figura 2 – Separação de Grupo A



Fonte: MOORE, 2016.

Figura 3 – Separação de Grupo B



Fonte: MOORE, 2016.

3. ALGORITMO K-MEANS

O algoritmo *K-Means* usa uma forma simples e fácil de classificar um conjunto de dados por meio de um número de *cluster*, que deve ser fixado antes da execução do algoritmo.

Segundo Matteucci (2016) o conceito principal é definir k centroides, um para cada cluster pré-definido, esses centroides devem ser definidos por conta da separação entre os resultados. Desta forma a escolha deve ser feita para colocá-los distantes um dos outros. A seguir deve-se fazer com que cada ponto pertença a um conjunto de dados e vinculá-lo ao centroide mais próximo. Quando da conclusão desta etapa todos os pontos devem pertencer a um grupo.

O próximo passo é recalculer k novos centroides como pontos centrais dos *clusters* do passo anterior. Depois de ter esses novos k centroides calculados, uma nova ligação deve ser feita entres os mesmos pontos do conjunto de dados e o novo centroide mais próximo. Um *loop* deve ser executado com esta função e como resultado, é possível observar que os k centroides mudam de localização até que não aconteçam alterações.

O algoritmo *k-Means* tem como objetivo minimizar uma função objetivo, neste caso específico uma função quadrática de erro que pode ser vista conforme mostra Matteucci, (2016) na equação 1.

Equação 1 - Função Objetivo

$$J = \sum_{j=1}^k \sum_{i=1}^x \left\| x^{(j)}_i - c_j \right\|^2$$

Percebe-se que onde $x_i^{(j)}$ e c_j representam a distância entre o ponto e o centro do *cluster*.

O algoritmo pode ser definido pelas seguintes etapas:

K representa os objetos que se pretende agrupar e são a representação dos centroides iniciais.

Associa cada objeto ao grupo centroide mais próximo.

Quando todos os objetos tiverem sido associados, recalcule as posições dos centroides **K**.

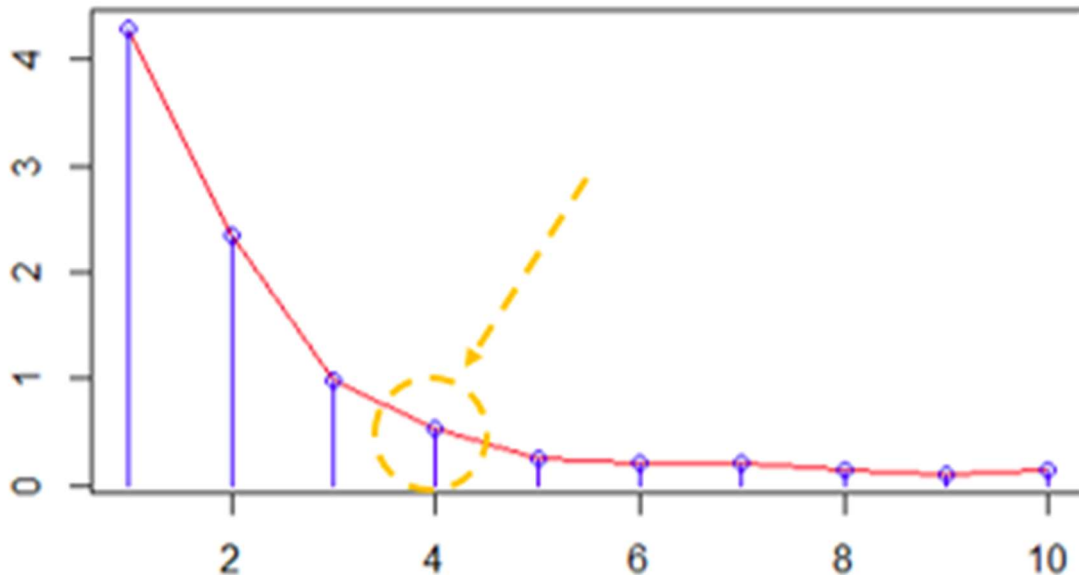
Repita os passos até que os centroides não mudem de posição.

Este procedimento termina sempre com um resultado de separação, embora não necessariamente ótimo. O algoritmo *K-Means* é sensível aos centros de agrupamento selecionados aleatoriamente, por conta disso é necessária sua definição assertiva e que deve ser tomada como uma premissa.

Como mencionado anteriormente o algoritmo deve ser executado várias vezes até que os centroides não apresentem mudança e com isso reduzir seus efeitos.

A escolha de **k** pode ser feita através do método *Elbow*, o método tem esse nome por ter seu resultado parecido com um braço e o que se procura é um cotovelo para definir um número de **k** a serem criados, com base no número de amostra. Este método faz crescer a quantidades de clusters a partir de 1 (um) e analisa o melhor resultado após cada incremento, quando o benefício não for mais relevante encontrasse o melhor resultado de **k** (LADEIRA, 2016), como pode ser visto na figura 5.

Figura 5 – Método Elbow



Fonte: NOGARE, 2016.

O algoritmo *K-Means* é um dentre vários de propõem a resolver problemas de *clustering*, e se destaca pela sua simplificação e fácil aplicação como veremos ao longo deste artigo.

O exemplo de código de um algoritmo *K-Means* faz parte deste artigo, e foi baseado em um exemplo do site do MatLab. Isso torna possível materializar explicação dada até este momento.

Código *K-Means*. MatLab (2016):

```

clc
clear all
close all

%% Gerando os pontos

Sigma = [0.5 0.05; 0.05 0.5];
f1 = mvnrnd([0.5 0] ,Sigma,100);
f2 = mvnrnd([0.5 0.5],Sigma,100);
f3 = mvnrnd([0.5 1] ,Sigma,100);
f4 = mvnrnd([0.5 1.5],Sigma,100);
F = [f1;f2;f3;f4];

```

```

%% K-means
K = 4; % Números de Clusters
KMI = 400; % K-Means Interações
CENTS = F( ceil(rand(K,1)*size(F,1)) ,:); % Cluster Centro
DAL = zeros(size(F,1),K+2); % Distâncias
CV = '+r+b+c+m+k+yorobocomokoyrsbscsmsksy'; % Cores dos vetores

for n = 1:KMI

    for i = 1:size(F,1)
        for j = 1:K
            DAL(i,j) = norm(F(i,:) - CENTS(j,:));
        end
        [Distance CN] = min(DAL(i,1:K)); % 1:K Distância do cluster ao centro 1:K
        DAL(i,K+1) = CN; % K+1 Identificação do cluster
        DAL(i,K+2) = Distance; % K+2 Distância mínima
    end
    for i = 1:K
        A = (DAL(:,K+1) == i); % Cluster K Pontos
        CENTS(i,:) = mean(F(A,:)); % Novo centro do cluster
        if sum(isnan(CENTS(:))) ~= 0 % Ponto randômico
            NC = find(isnan(CENTS(:,1)) == 1); %
            for Ind = 1:size(NC,1)
                CENTS(NC(Ind),:) = F(randi(size(F,1)),:);
            end
        end
    end
end

%% Plotando
clf
figure(1)
hold on

for i = 1:K
    PT = F(DAL(:,K+1) == i,:); % Ponto de cada cluster
    plot(PT(:,1),PT(:,2),CV(2*i-1:2*i),'LineWidth',2); % Plotar o ponto
    plot(CENTS(:,1),CENTS(:,2),'*k','LineWidth',7); % Plotar cluster
end

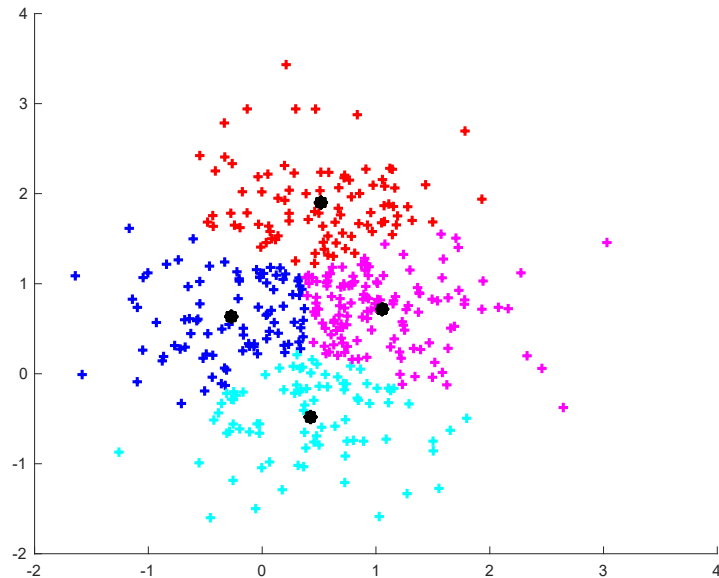
hold off
grid on
pause(0.1)

end

```

A figura 6 representa o resultado da operação do algoritmo, onde é possível identificar o agrupamento em torno dos centroides.

Figura 6 - Agrupamento



Fonte: MatLab, 2016.

Como pode ser visto a existem 4 áreas bem distintas onde houve a aglutinação dos pontos, após 400 interações pré-definidas no algoritmo.

4. APLICAÇÃO EM CRIPTOANÁLISE

A aplicação do algoritmo *K-Means* no ambiente de criptoanálise está ligado a um tipo de ataque chamado “Ataque Horizontal”, que é uma metodologia para ataques de canal lateral, cujo alvo são protocolos criptográficos assimétricos baseados em RSA e ECC e que fazem uso da exponenciação modular no caso do algoritmo RSA e multiplicação escalar no caso de ECC. Em teoria estes ataques permitem recuperar os bits do expoente/escalar secreto (LADEIRA, 2016).

Especificamente em um ataque lateral a criptografia de curva elíptica é possível o que o ataque seja executado em uma das camadas de sua implementação, ou seja:

Aplicação.

Protocolo.

Multiplicação escalar.

Adição e duplicação de ponto.

Aritmética de ponto finito ou aritmética de inteiros grandes.

Protocolos de criptográficos em criptografia de curva elíptica, são implementados a partir dessas primitivas e que foram listadas anteriormente é importante ressaltar a interdependência entre as camadas e suas operações, e que isto impacta diretamente em seu desempenho e segurança.

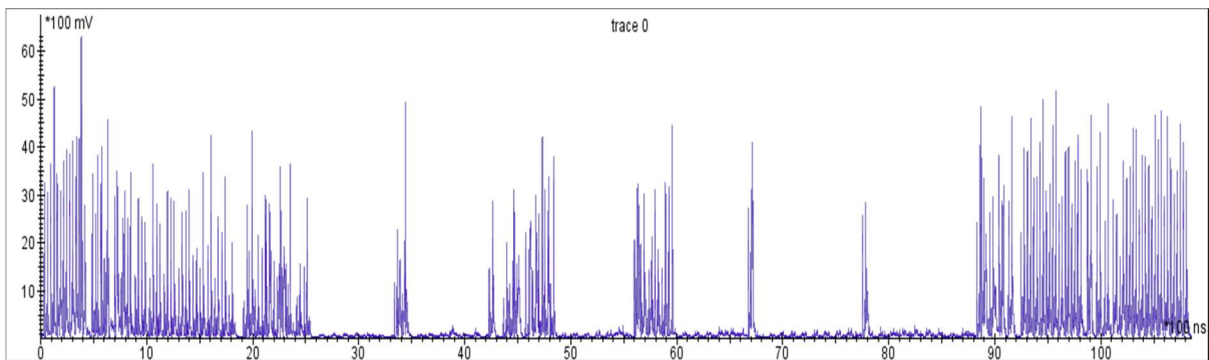
Há várias formas de ataques a criptografia de curva elíptica como ataque de tempo e contramedidas como randomização escalar, mas não são o foco deste artigo. A análise será feita tomando como base o algoritmo *K-Means* e a análise de potência especificamente.

O método de *correlation analysis* segue o princípio de análise potência, e neste contexto o *trace* é dividido em vários seguimentos (*Clustering*), como pode ser visto na figura 7, e para cada um deles é atribuído um valor hipotético, e a relação entre a amostra e o valor hipotético é calculada.

Os ataques horizontais requerem pré-processamento avançados de *trace*, caracterização e avaliação de vazamento, o principal problema desta abordagem está em extrair o vazamento a partir de um único trace.

Métodos de aprendizagem não supervisionada com o *K-Means*, tema deste artigo é a forma recentemente aplicada para resolver esta limitação. Este método pode ser aplicado em um único trace e assim permitindo a identificação de classes, (agrupamento) específico na operação como pode ser visto na figura 7.

Figura 7 – Trace a ser analisado



Fonte: LADEIRA, 2016.

Ainda sobre o *K-Means* é possível afirmar que é um algoritmo rígido, isto é cada instância é atribuída a um único *cluster*, característica importante na sua utilização em criptoanálise.

A partir deste ponto é possível fazer a verificação de uma chave conhecida, que consiste em determinar os pontos de interesse, isto é, os índices das amostras onde o vazamento é forte. Esta análise é realizada na fase de testes do ataque para determinar o número traces necessários para avaliação. Um análise, mas profunda deste tipo de ataque poderá ser vista em futuros artigos.

5 ANÁLISE E RESULTADO

Capturados os *traces* de potência com objetivo de recuperação de chaves, os mesmos foram analisados da seguinte forma:

Agrupamento: Onde em cada vetor é aplicado o algoritmo *K-Means*. Tal resultado por ser visto como um escalar aproximado.

Estimativa do Escalar: Escalares aproximados são combinados em um escalar único.

Grau de Confiança: Calcula-se o grau de confiança de cada bit recuperado. Este grau de confiança é um número real entre 0 e 1 e depende do tipo de ataque que está sendo realizado.

Taxa de sucesso: Taxa de sucesso é a média de bits do escalar recuperados corretamente.

Obtido estes dados é possível se ter indicadores para o ataque horizontal, cuja taxa de sucesso e grau de confiança devem ser altos, deve ser ressaltado que a qualidade das medidas dos traces está diretamente ligada ao sucesso da coleta dos indicadores, ou seja, o equipamento usado nestas medidas deve ter qualidade e ser adequado a este fim.

Caso a quantidades de bits sem erro seja grande, o ataque de força bruta é viável se e somente se os bits contendo erros não sejam adjacentes.

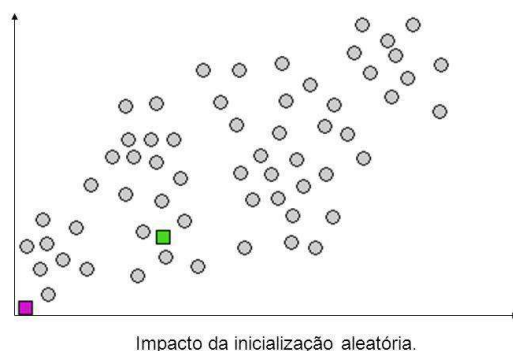
A combinação de força bruta e outros algoritmos, como por exemplo Montgomery Ladder tem-se mostrado mais eficientes no ataque horizontal, e cujo detalhamento não será parte deste artigo.

6.0 Considerações Finais

O algoritmo *K-Means* tem aplicações diversas quando a premissa de clusterização se faz necessária, mas é valido destacar também algumas fraquezas, tais como a maneira popular de sua inicialização que é a escolha aleatória de k nas amostras, os resultados iniciais dependem dos valores iniciais para as medias, dentre outros, a figura 8 mostra o resultado da escolha aleatória, onde a o resultado da separação não é eficiente.

Figura 8 – Escolha aleatória

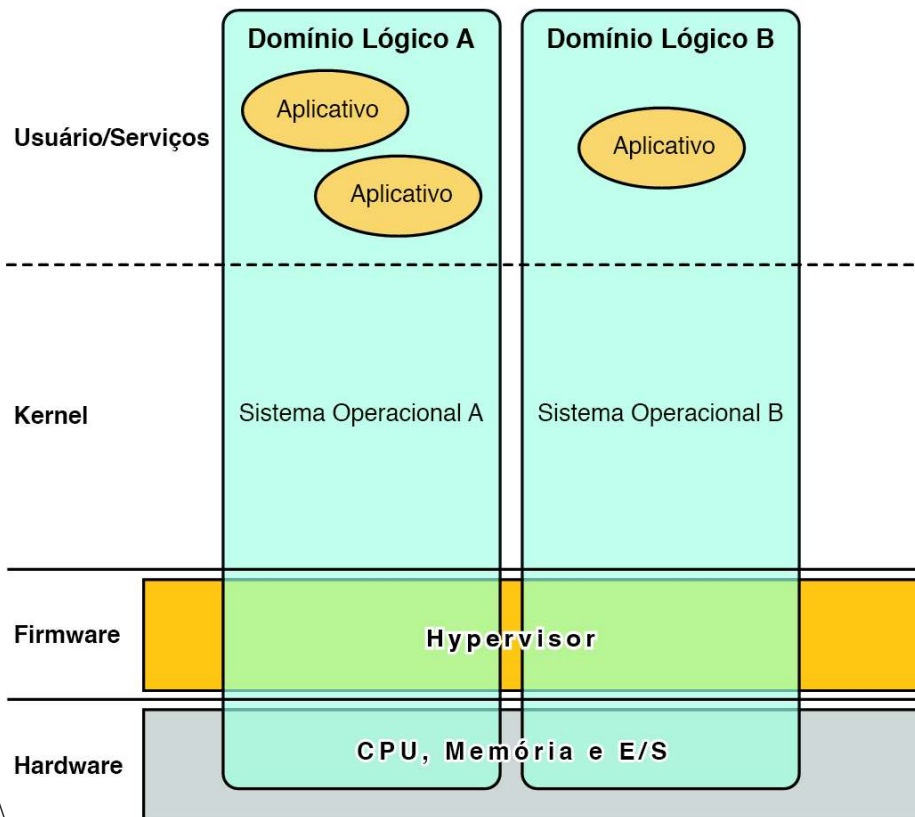
k-Means – Um Exemplo



Fonte: LADEIRA, 2016.

Na criptoanálise o algoritmo *k-Means*, usado em análise de potência de ataque horizontal tem colaborado em diversos estudos que buscam garantir a segurança de algoritmos de RSA e ECC. Estes estudos têm como resultados formas de contramedidas de segurança a serem implementadas tanto em *hardware* como *software*, um exemplo pode ser visto na figura 9 onde existe a separação por camadas entre os níveis.

Figura 9 – Segurança da aplicação



Fonte: Oracle, 2017.

Dentre estas contramedidas, o *NIST* tem encorajado desenvolvimento de métodos de teste e ferramentas para avaliar a eficiência de mitigações de contra-ataques não-invasivos a módulos criptográficos.

Em um momento em que sistemas de *Smart Grid* estão cada vez mais próximos do nosso cotidiano, a preocupação com o ataque de análise de potência vem se tornando motivo de preocupação e está motivando diversos estudos nesta área.

Uma outra tecnologia emergente é a computação em *fog*, que é basicamente trazer a computação em *cloud* para a borda da rede o que torna este ambiente susceptível a ataques cibernéticos por análise de potência e que também requer novos conceitos quanto ao paradigma da segurança.

REFERÊNCIAS

LADEIRA, L. Simpósio Brasileiro em segurança da informação e de sistemas computacionais. Disponível em: <<http://sbseg2016.ic.uff.br/pt/files/minicursos.pdf>>. Acesso em 16/11/2016.

MATLAB. Disponível em: <<https://www.mathworks.com/help/stats/k-means-clustering.html/>>. Acesso em 19/11/2016.

MATTEUCCI, M. Tutorial de Algoritmos Clustering. Disponível em: <https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html/>. Acesso em 27/11/2016.

MOORE, A. “K-Means Hierarquia”. Disponível em: <[http://www.autonlab.org/tutorials/index.html?s\[\]=means/](http://www.autonlab.org/tutorials/index.html?s[]=means/)>. Acesso em 27/11/2016.

NOGARE, D. Engenharia do Conhecimento e Sistemas Especialistas. Disponível em: <<http://www.diegonogare.net/2015/08/entendendo-como-funciona-o-algoritmo-de-cluster-k-means/>>. Acesso em 30/11/2016.

ORACLE. Guia de Segurança. Disponível em: http://docs.oracle.com/cd/E60518_01/html/E60506/productoverview.html/. Acesso em 15/08/2017.

STALLINGS, W. Criptografia e segurança de redes princípios e práticas. 4 ed. São Paulo: Pearson Prentice Hall, 2010.