ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Maestría En Seguridad Informática Aplicada

"DESARROLLO DE UN ESQUEMA DE SEGURIDAD Y UN FIREWALL DE BORDE PARA EL SISTEMA WEB DE UNA EMPRESA DE SALUD"

EXAMEN DE GRADO (COMPLEXIVO)

Previo a la obtención del Título de:

MAGISTER EN SEGURIDAD INFORMÁTICA APLICADA

Presentado por:

HERNÁN EDUARDO CUEVA DELGADO

GUAYAQUIL - ECUADOR

AÑO: 2015

AGRADECIMIENTO

Agradezco a Dios por la salud y la fuerza que me ha dado, a mi familia por todo el respaldo obtenido durante este proceso, a mis instructores por el tiempo dedicado para impartir sus conocimientos.

DEDICATORIA

Dedico este trabajo a mi familia, en especial a mis padres, quienes estuvieron pendientes de mis estudios desde muy pequeño y que siempre me inculcaron valores que me permitan crecer como persona y como profesional. Este trabajo es una meta mas en mi vida a la que he podido llegar gracias a ellos.

TRIBUNAL DE SUSTENTACIÓN

_____

Ing. Lenin Freire.

DIRECTOR DE LA MAESTRÍA EN

SEGURIDAD INFORMÁTICA APLICADA

_____

Mgs. Gonzalo Luzardo.

PROFESOR DELEGADO POR LA

MAESTRÍA EN SEGURIDAD

INFORMÁTICA APLICADA

_____

Mgs. Roky Barbosa.

PROFESOR DELEGADO POR LA

MAESTRÍA EN SEGURIDAD

INFORMÁTICA APLICADA

RESUMEN

El objetivo de este trabajo consiste en aplicar algoritmos de cifrado de datos en la base de datos y archivos que utiliza el Aplicativo Web de una Empresa de Medicina Pre-pagada, con el fin de garantizar la confidencialidad de la información que pertenece a los afiliados de la Empresa de Salud.

Adicional se implementará un Firewall de borde dado que el Servidor Web donde se encuentra alojado el Sistema esta conectado directamente al internet y esto podría a futuro perjudicar a la seguridad de la información dado que hackers puedan vulnerar los sistemas y hacer mal uso de la misma.

Para prevenir intentos de acceso externo se integrará el Firewall “IPTABLES” con un IDS “Fail2Ban”, así mismo un sistema de notificaciones que alertará al Administrador de los Sistemas.

ÍNDICE GENERAL

ABREVIATURAS Y SIMBOLOGÍA

AES: Advanced Encryption Standard

DMZ: Demilitarized Zone

DNS: Domain Name Server

DOS: Denial of Service

FW: Firewall

IDS: Intrusion Detection System

IP: Internet Protocol

LAN: Local Area Network

RSA: Rivest, Shamir y Adleman

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La cantidad de delitos informáticos contra empresas han aumentado últimamente, cada vez los ataques son más sofisticados y existen un sin número de herramientas que ayudan agilizar el trabajo de los hackers de una manera sencilla.

Hoy en día la información está considerada como un activo fundamental en una empresa, la cual se debe proteger contra todo tipo de eventos que causen la perdida de la operatividad de la misma, es por esa razón que se debe implementar mecanismos de seguridad sobre toda infraestructura tecnológica.

La Empresa de Salud a la cual se debe mejorar la seguridad utiliza actualmente un Sistema Web, por medio del cual se realizan todas las operaciones y el manejo de información de los afiliados. Cada afiliado tiene al menos un plan de asistencia médica y otros servicios adicionales en los cuales contienen exámenes médicos y demás expedientes médicos.

Existen leyes que penalizan el mal uso de información médica, por tal motivo es necesario aplicar una solución que prevenga a futuro este tipo de problemas.

La solución propuesta en este documento mejora que la información de los afiliados permanezca integra y no legible para aquel que la sustraiga de manera ilegal. Además se plantea la incorporación de un firewall filtrador de paquetes que ayudará a evitar que intrusos puedan ingresar al Servidor de Web.

En este proyecto se definirá la metodología usada para la solución del problema, cada paso esta descrito en el Capítulo 2, así mismo que, el análisis de resultados y la pruebas realizadas están descritas en el Capítulo 3.

CAPÍTULO 1

GENERALIDADES

1.1 Descripción del Problema

La empresa de salud se dedica a ofrecer servicio de medicina pre-pagada, para ello cuenta con un Sistema Web. Este tipo de aplicación tiene automatizado los procesos de contratos, cobranzas, facturación, punto médico, entre otros. El sistema web cuenta con una base de datos en la que se almacena información como son cuentas bancarias, datos de tarjetas de crédito, datos personales de los afiliados y sobre todo el historial médico. Hacemos un énfasis sobre la manipulación del Historial Médico dado que por políticas gubernamentales hay leyes que penalizan el mal uso de este

tipo de información. Al igual que todo tipo de información bancaria o financiera debe cumplir con políticas de seguridad.

Actualmente la inseguridad que presentan las organizaciones que utilizan sistemas web es sumamente agravantes, es por ello que se ha procedido a realizar un análisis con una breve auditoría en la base de datos, en la cual se observó que la información de tarjetas de crédito como son el número, la fecha de caducidad y el código de verificación están almacenadas en texto plano indicando además los nombres a quién pertenece la tarjeta. Este tipo de información es necesaria dado que la empresa realiza sus cobros a través de débitos en cuentas bancarias y en tarjetas de crédito, las cuales deben generar archivos con dicha información para enviar a los bancos por la respectiva autorización de débito.

El sistema web además permite subir archivos de las autorizaciones (aprobaciones bancarias), emitida por cada entidad bancaria para efectuar los cobros en el sistema. Los pagos son realizados mensualmente por el uso del servicio de medicina pre-pagada y demás servicios que el afiliado (persona cliente) ha adquirido.

Además la base de datos contiene la información de la historia clínica o historial médico como son atenciones, medicamentos administrados, diagnósticos, indicaciones médicas, resultados de exámenes médicos de cada afiliados (paciente). Se entiende que este tipo de información debe ser de carácter confidencial puesto que solo deberá conocer el médico tratante y el afiliado. Los factores de riesgos e impactos que pueden ocurrir son:

Riesgo:

- Robo de información de tarjetas de crédito.
- Robo de información de pacientes y usuarios de sistema.
- Robo de historia clínica de pacientes.
- Seguridad de servidores débil.

Impacto:

- Problemas judiciales por no proveer un soporte.
- Historia Clínica.
- Información de tarjetas de crédito.
- Información de cuentas bancarias
- Fácil penetración y acceso a la información por parte de hackers.

La aplicación web está desarrollada en Mysql como base de datos utilizando un framework PHP llamado “Yii”.

Para la empresa de salud es necesario brindar a sus afiliados los servicios a través de medios digitales como es el Internet, por lo que la seguridad es necesaria implementarla, dado al gran magnitud de información que se procesa, ya que no solo corresponde a la empresa sino a los afiliados.

Se conoce que toda Aplicación Web, cuando está a disposición de un grupo de usuarios a través de una red de comunicaciones, está expuesta múltiples riesgos que debe afrontar. Esto es porque un servicio puede tener muchas vulnerabilidades que permite a un atacante conseguir información que no le corresponde.

1.2 Solución Propuesta

Se propone mejorar la seguridad de la Aplicación Web agregando soporte de encriptación de datos de alto nivel, para así asegurar la confidencialidad

e integridad de los datos, para el caso de tarjetas de crédito, historial médico y otro tipo de información de gran importancia para las personas afiliadas a la empresa de seguros.

Características:

- Encriptación de datos de alto nivel.
- Seguridad en servidores y protocolos de comunicación.

Beneficios:

- Encriptación de la información en la base de datos.
- Servidores seguros ante todo tipo de riesgos de robo de información.

Solución:

Las metodologías a usar serían encriptación AES-256 bits y RSA-512 bits. Se utilizará AES de 256 bits para encriptación de archivos, con una clave maestra dentro un archivo cuyo contenido estará encriptado por medio de una llave privada RSA de 512 bits perteneciente al servidor. Estas llaves

RSA de 512 bits deben ser únicas. También se implementará un firewall de borde el cual limitará las conexiones del exterior para evitar ataques externos.

El firewall debe configurarse de tal manera que solo se habiliten los puertos y servicios que son necesarios y para evitar ataques de denegación de servicios. Además se implementará fail2ban para bloquear los intentos de accesos no autorizados, así como el envío de notificaciones al administrador del servidor y se integrará con la solución del firewall (Iptables).

CAPÍTULO 2

ANÁLISIS DE IMPLEMENTACIÓN

2.1 Cifrado de Datos

El cifrado de datos se realizará para información almacenada en la base de datos así como para los archivos que se almacenarán en el servidor y que cuyo contenido tiene información importante.

2.1.1 Metodología de cifrado

El mecanismo de cifrado se realizará por medio del Lenguaje de

Programación PHP que está desarrollado el Framework “Yii”, plataforma en la cual está desarrollado el aplicativo web de la empresa de salud. Para más información del tipo de cifrado RSA [1] y AES [2].

En la figura 2.1 se indica el proceso generación del archivo que contendrá la clave maestra.

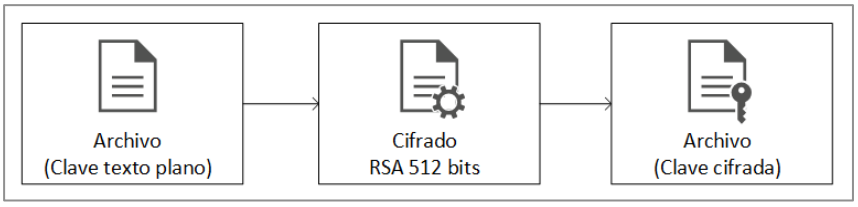


Figura 2.1: Modelo creación de Archivo de Clave Maestra

El proceso de cifrado de datos se puede visualizar mediante la figura 2.2 utilizando el archivo que contiene la clave maestra.

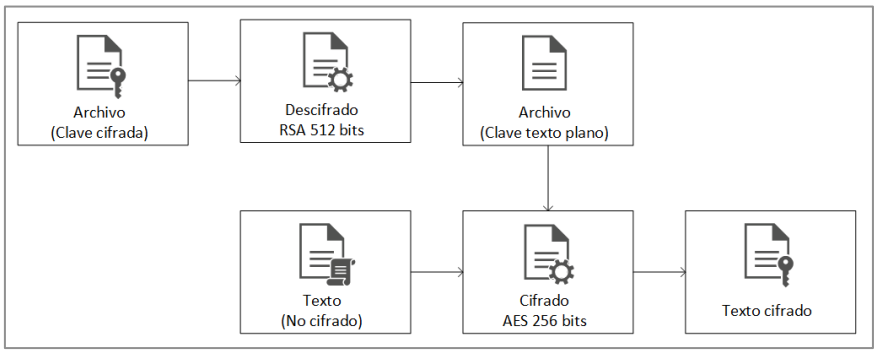


Figura 2.2: Modelo de cifrado de datos

El proceso de descifrado de datos se puede visualizar mediante la figura 2.3 utilizando el archivo que contiene la clave maestra.

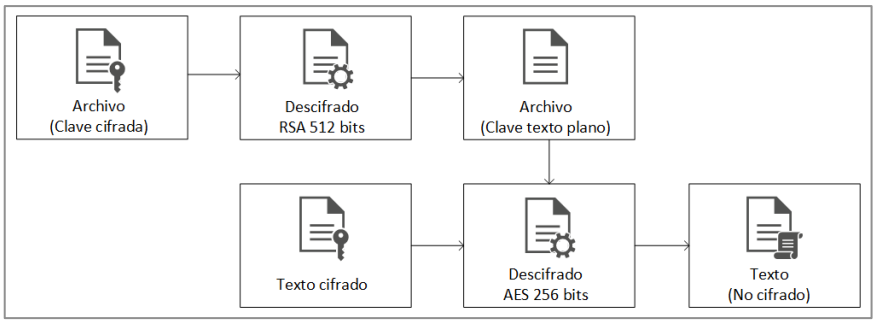
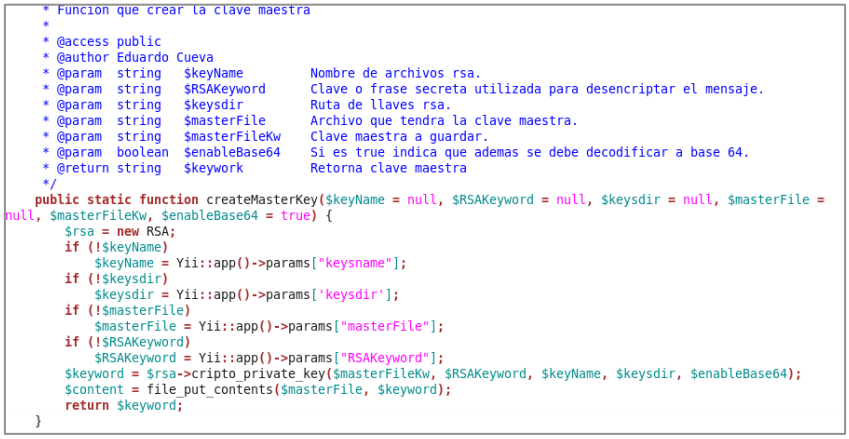


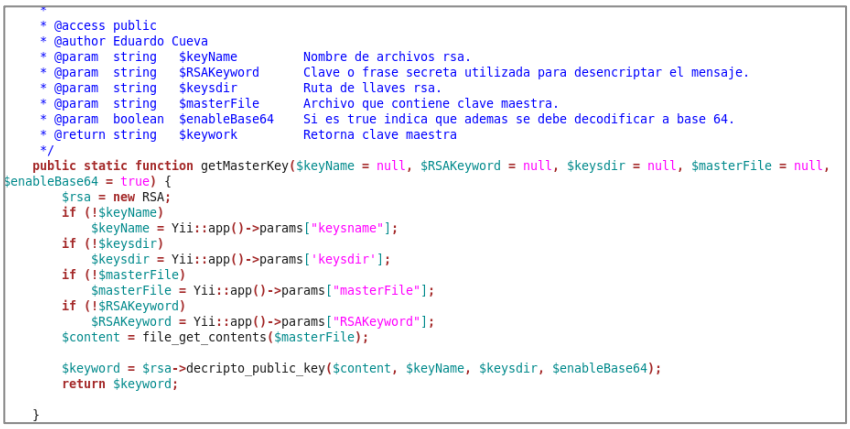
Figura 2.3: Modelo de descifrado de datos

Dado el modelo propuesto anteriormente las siguientes figuras muestran los métodos principales que permiten la generación y consulta de la clave maestra.



```
    * Funcion que crear la clave maestra
    *
    * @access public
    * @author Eduardo Cueva
    * @param  string   $keyName         Nombre de archivos rsa.
    * @param  string   $RSAKeyword      Clave o frase secreta utilizada para desencriptar el mensaje.
    * @param  string   $keysdir         Ruta de llaves rsa.
    * @param  string   $masterFile      Archivo que tendra la clave maestra.
    * @param  string   $masterFileKw    Clave maestra a guardar.
    * @param  boolean  $enableBase64    Si es true indica que ademas se debe decodificar a base 64.
    * @return string   $keywork         Retorna clave maestra
    */
   public static function createMasterKey($keyName = null, $RSAKeyword = null, $keysdir = null, $masterFile =
null, $masterFileKw, $enableBase64 = true) {
       $rsa = new RSA;
       if (!$keyName)
           $keyName = Yii::app()->params["keysname"];
       if (!$keysdir)
           $keysdir = Yii::app()->params['keysdir'];
       if (!$masterFile)
           $masterFile = Yii::app()->params["masterFile"];
       if (!$RSAKeyword)
           $RSAKeyword = Yii::app()->params["RSAKeyword"];
       $keyword = $rsa->cripto_private_key($masterFileKw, $RSAKeyword, $keyName, $keysdir, $enableBase64);
       $content = file_put_contents($masterFile, $keyword);
       return $keyword;
   }
```

Figura 2.4: Método de creación de archivo con clave maestra



```
    *
    * @access public
    * @author Eduardo Cueva
    * @param  string   $keyName         Nombre de archivos rsa.
    * @param  string   $RSAKeyword      Clave o frase secreta utilizada para desencriptar el mensaje.
    * @param  string   $keysdir         Ruta de llaves rsa.
    * @param  string   $masterFile      Archivo que contiene clave maestra.
    * @param  boolean  $enableBase64    Si es true indica que ademas se debe decodificar a base 64.
    * @return string   $keywork         Retorna clave maestra
    */
   public static function getMasterKey($keyName = null, $RSAKeyword = null, $keysdir = null, $masterFile = null,
$enableBase64 = true) {
       $rsa = new RSA;
       if (!$keyName)
           $keyName = Yii::app()->params["keysname"];
       if (!$keysdir)
           $keysdir = Yii::app()->params['keysdir'];
       if (!$masterFile)
           $masterFile = Yii::app()->params["masterFile"];
       if (!$RSAKeyword)
           $RSAKeyword = Yii::app()->params["RSAKeyword"];
       $content = file_get_contents($masterFile);

       $keyword = $rsa->decripto_public_key($content, $keyName, $keysdir, $enableBase64);
       return $keyword;

   }
```

Figura 2.5: Método de consulta de archivo con clave maestra

2.1.2 Base de Datos

El objetivo es cifrar la información de Historia Clínica e información bancaria de los afiliados que reciben el servicio de salud.

Las siguientes figura 2.6 y 2.7 se muestra los métodos realizados para el cifrado de información con AES a 256 bits.

```php
    /**
     * crypt AES 256
     * @param string $password      Clave para cifrar el contenido de $data
     * @param data $data            Mensaje a cifrar
     * @param boolean $baseEncode   Permite identificar si el texto hay que codificarlo al final con Base64
     * @param string  $aes_cbc      Tipo de codificacion
     * @return encrypted data
     */
    public function cryptAES($data, $password, $baseEncode = true, $aes_cbc = 'aes-256-cbc') {
        // Set a random salt
        $salt = openssl_random_pseudo_bytes(8);
        $salted = '';
        $dx = '';
        // Salt the key(32) and iv(16) = 48
        while (strlen($salted) < 48) {
            $dx = md5($dx . $password . $salt, true);
            $salted .= $dx;
        }
        $key = substr($salted, 0, 32);
        $iv = substr($salted, 32, 16);
        $encrypted_data = openssl_encrypt($data, $aes_cbc, $key, true, $iv);
        if ($baseEncode)
            return base64_encode('Salted__' . $salt . $encrypted_data);
        else
            return 'Salted__' . $salt . $encrypted_data;
    }
}
```

Figura 2.6: Método de cifrado AES 256 bits

```php
/**
 * decrypt AES 256
 * @param string $password      Clave para descifrar el contenido de $edata
 * @param data $edata           Mensaje a descifrar
 * @param boolean $baseDecode   Permite identificar si el texto hay que decodificarlo a Base64
 * @param string  $aes_cbc     Tipo de codificacion
 * @return dencrypted data
 */
public function decryptAES($edata, $password, $baseDecode = true, $aes_cbc = 'aes-256-cbc') {
    $data = $edata;
    if ($baseDecode)
        $data = base64_decode($edata);
    $salt = substr($data, 8, 8);
    $ct = substr($data, 16);
    $rounds = 3;
    $data00 = $password . $salt;
    $md5_hash = array();
    $md5_hash[0] = md5($data00, true);
    $result = $md5_hash[0];
    for ($i = 1; $i < $rounds; $i++) {
        $md5_hash[$i] = md5($md5_hash[$i - 1] . $data00, true);
        $result .= $md5_hash[$i];
    }
    $key = substr($result, 0, 32);
    $iv = substr($result, 32, 16);
    $decrypto = openssl_decrypt($ct, $aes_cbc, $key, true, $iv);
    if ($decrypto)
        return $decrypto;
    else {
        return false;
    }
}
```

Figura 2.7: Método de descifrado AES 256 bits

Con los métodos anteriores se puede realizar el cifrado y descifrado de cualquier contenido para ser ingresado en la base de datos.

2.1.3 Archivos de Aplicación

Los archivos que sean subidos mediante la aplicación web serán encriptados, archivos como los comprobantes de débito o con información bancaria.

En la siguiente figura se puede visualizar como se utiliza los métodos anteriores así como la creación del nuevo archivo con la función llamada “file_put_contents” de PHP.

```php
Yii::import("ext.EAjaxUpload.qqFileUploader");
$allowedExtensions = array("jpg", "jpeg", "gif", "png", "pdf");
$sizeLimit = 10 * 1024 * 1024; // maximum file size in bytes
$uploader = new qqFileUploader($allowedExtensions, $sizeLimit);
if (Utilities::verificarDirectorio($folder)) {
    $result = $uploader->handleUpload($folder);
    if (file_exists($folder . $result['filename'])) {
        $nombarch = $folder . $result['filename'];
        $path_parts = pathinfo($nombarch);
        $sizeFile = filesize($nombarch);
        // renombramos el archivo
        $nameFileNew = Utilities::getTokenToFileName() . "." . $path_parts["extension"];
        Utilities::renameFileUpload(basename($nombarch), $nameFileNew, $folder);
        // proceso de codificacion del archivo
        $password = Utilities::getMasterKey();
        $message = file_get_contents($folder . $nameFileNew);
        $rsa = new RSA;
        $cripto = rsa->cryptAES($message, $password);
        file_put_contents($folder . $nameFileNew, $cripto);
    }
}
```

Figura 2.8: Método de cifrado AES 256 bits en archivos

```
// proceso de decodificacion del archivo
$password = Utilities::getMasterKey();
$message = file_get_contents($folder . $nameFileNew);
$rsa = new RSA;
$decripto = rsa->decryptAES($message, $password);
file_put_contents($folder . $nameFileNew, $decripto);
```

Figura 2.9: Método de descifrado AES 256 bits en archivos

2.2 Implementación de Firewall de Borde

La implementación del esquema de red será como se visualiza en la figura 2.10.

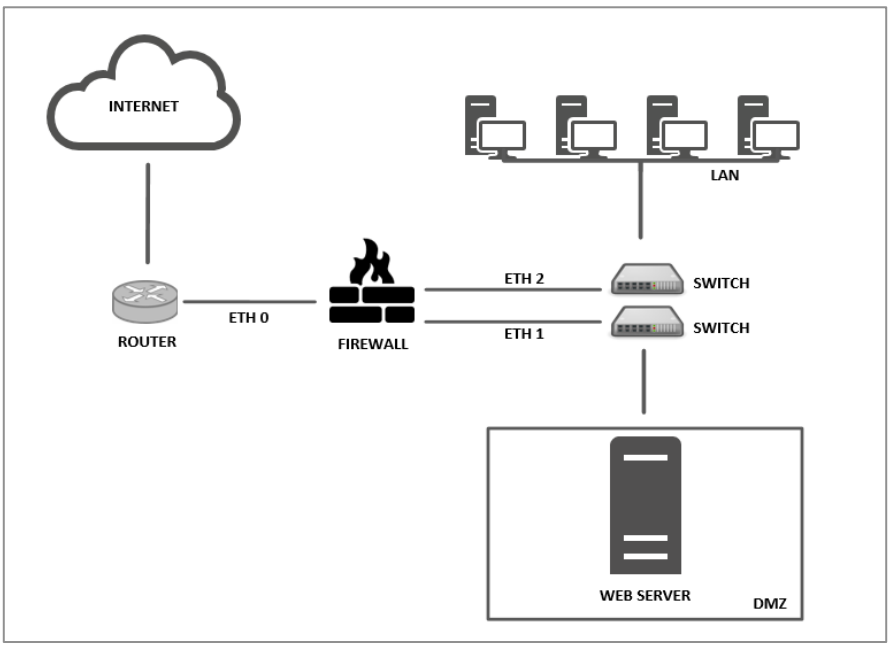


Figura 2.10: Esquema de red

Para implementar el firewall [7] se debe cumplir el esquema mencionado en la figura 2.10 debido a que actualmente el web server está conectado directamente al router del proveedor de internet.

2.2.1 Instalación de IPTABLES

Antes de empezar la instalación se debe mencionar que todo programa será instalado en el equipo Firewall con Sistema Operativo Linux Centos 6 de 64 bits.

En la distribución Centos viene instalado por defecto pero en el caso de que no tenga instalado el paquete de iptables la instalación es sencilla por medio del siguiente comando **“yum install iptables”**.

Si ya tiene instalado el paquete iptables, entonces se debe realizar es la actualización del mismo en el caso de que no tenga la última versión por medio del siguiente comando “**yum update iptables”**.

2.2.2 Configuración de IPTABLES

El siguiente paso es crear un archivo con extensión **".sh"** el cual tendrá todas las reglas y normativas que debe aplicar al Firewall. Por defecto se va a negar todo trafico entrante y saliente, después se abrirá los puertos necesarios para la conexión con aplicaciones así como las redirecciones a los servicios respectivos.

Los puertos que deben estar abiertos son:

- http - 80 (Al Servidor Web de la DMZ)
- https - 443 (Al Servidor Web de la DMZ)
- ssh - 12022 (Cambio del defecto 22 al 12022 del Firewall)
- ssh - 22 (Al Servidor Web de la DMZ)
- http - 80 (Navegación web para DMZ y LAN)
- Pop, Imap, Smtp (25, 110, 143, etc. Para estaciones de trabajo)

Las siguientes figuras detallan cada sección de la configuración realizada en un archivo script en **"bash"**. Se utiliza el esquema de la figura 2.10.

```
#!/bin/bash

#VARIABLES
eth_EXT=eth0
eth_DMZ=eth1
eth_LAN=eth2

RED_LAN=192.168.1.1/25
RED_DMZ=192.168.1.128/28
RED_FW=192.168.1.144/30

WEB_SERVER=192.168.1.129
IP_ADMIN=192.168.1.10

## Borrar cadenas anteriores
  iptables -F
  iptables -X
  iptables -Z
  iptables -t nat -F
```

Figura 2.11: Seteo de variables de script

```
## Condiciones por defecto
## Politica por defecto: DROP
  iptables -P INPUT DROP
  iptables -P OUTPUT DROP
  iptables -P FORWARD DROP
  iptables -t nat -P PREROUTING ACCEPT
  iptables -t nat -P POSTROUTING ACCEPT

# Con esto permitimos hacer forward de paquetes en el firewall,
# para que otras máquinas puedan salir a traves del firewall.
  echo 1 > /proc/sys/net/ipv4/ip_forward
```

Figura 2.12: Reglas Drop por defecto

```
# Permitir navegacion en FW
  iptables -A OUTPUT -p tcp -o $eth_EXT --dport 80 -j ACCEPT
  iptables -A INPUT -p tcp -i $eth_EXT --sport 80 -j ACCEPT
  iptables -A OUTPUT -p tcp -o $eth_EXT --dport 443 -j ACCEPT
  iptables -A INPUT -p tcp -i $eth_EXT --sport 443 -j ACCEPT

# Permitir Consulta Servidor Web a FW
  iptables -A OUTPUT -p tcp -o $eth_DMZ -d $WEB_SERVER --dport 80 -j ACCEPT
  iptables -A INPUT -p tcp -i $eth_DMZ -d $WEB_SERVER --sport 80 -j ACCEPT
  iptables -A OUTPUT -p tcp -o $eth_DMZ -d $WEB_SERVER --dport 443 -j ACCEPT
  iptables -A INPUT -p tcp -i $eth_DMZ -d $WEB_SERVER --sport 443 -j ACCEPT

# Permitir consulta de DNS para FW
  iptables -A OUTPUT -p udp -o $eth_EXT --dport 53 -j ACCEPT
  iptables -A INPUT -p udp -i $eth_EXT --sport 53 -j ACCEPT

# Permitir consulta de hora
  iptables -A OUTPUT -p udp -o $eth_EXT --dport 123 -j ACCEPT
  iptables -A INPUT -p udp -i $eth_EXT --sport 123 -j ACCEPT
```

Figura 2.13: Permiso de puertos al Firewall

```
# Permitir el trafico de conexiones ya establecidas
# (el control de tráfico se hace al inicar las conexiones)
  iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
  iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
  iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

# Limitar tráfico ICMP (permitir hasta un maximo de 5 peticiones/segundo)
  iptables -A INPUT -p icmp -m limit --limit 5/second -j ACCEPT
  iptables -A OUTPUT -p icmp -m limit --limit 5/second -j ACCEPT
  iptables -A FORWARD -p icmp -m limit --limit 5/second -j ACCEPT

# Redireccion de trafico de la DMZ a la LAN y viceversa
  iptables -A FORWARD -i $eth_LAN -o $eth_DMZ -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -o $eth_LAN -m state --state ESTABLISHED,RELATED -j ACCEPT

# Redireccion de trafico de la LAN a la DMZ para consulta de servicio Web
  iptables -A FORWARD -i $eth_LAN -d $WEB_SERVER -p tcp --dport 80 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -d $WEB_SERVER -p tcp --dport 443 -j ACCEPT

# Permitir Acceso a servicios a Servidor Web de manera externa
  iptables -A FORWARD -i $eth_EXT -d $RED_DMZ -p tcp --dport 80 -j ACCEPT
  iptables -A FORWARD -i $eth_EXT -d $RED_DMZ -p tcp --dport 443 -j ACCEPT
```

Figura 2.14: Redirección de tráfico entre la DMZ y la LAN

```
# Permitir Navegacion Web en LAN y DMZ
  iptables -A FORWARD -s $RED_DMZ -o $eth_EXT -p tcp --dport 80 -j ACCEPT
  iptables -A FORWARD -s $RED_DMZ -o $eth_EXT -p tcp --dport 443 -j ACCEPT
  iptables -A FORWARD -s $RED_LAN -o $eth_EXT -p tcp --dport 80 -j ACCEPT
  iptables -A FORWARD -s $RED_LAN -o $eth_EXT -p tcp --dport 443 -j ACCEPT

# Permitir consulta de DNS para LAN y DMZ
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 53 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p udp --dport 53 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 53 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p udp --dport 53 -j ACCEPT

# Permitir consulta de FTP para la LAN y DMZ de servicios externos
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 20 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 21 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 20 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 21 -j ACCEPT

# Permitir que LAN y DMZ se conecten a Servidores SSH Externos
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -o $eth_EXT -p tcp --dport 22 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -o $eth_EXT -p tcp --dport 22 -j ACCEPT
```

Figura 2.15: Permiso de navegación para la DMZ y la LAN

```
# Permitir Correos en LAN y DMZ
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 25 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 25 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 109 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 109 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 110 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 110 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 143 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 143 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 220 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p udp --dport 220 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 220 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p udp --dport 220 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 465 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 465 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 563 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 563 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 587 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 587 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 992 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 992 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 993 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 993 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 995 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 995 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p udp --dport 995 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p udp --dport 995 -j ACCEPT
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 2525 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 2525 -j ACCEPT
```

Figura 2.16: Permiso de consulta de correo

```
# Consulta de servidor Web desde la LAN
  iptables -A FORWARD -i $eth_DMZ -s $RED_DMZ -p tcp --dport 80 -j ACCEPT
  iptables -A FORWARD -i $eth_LAN -s $RED_LAN -p tcp --dport 80 -j ACCEPT

# SNAT para enmascarar la salida al internet
  iptables -t nat -A POSTROUTING -s $RED_DMZ -o $eth_EXT -j MASQUERADE
  iptables -t nat -A POSTROUTING -s $RED_LAN -o $eth_EXT -j MASQUERADE

# DNAT para redirigir trafico a la DMZ en este caso el servidor Web
  iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 80
      -j DNAT --to-destination $WEB_SERVER:80
  iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 443
      -j DNAT --to-destination $WEB_SERVER:443
```

Figura 2.17: Enmascaramiento de la LAN y DMZ al internet

```
# Establecer logs
  iptables -N LOGGING
  iptables -A INPUT -j LOGGING
  iptables -A OUTPUT -j LOGGING
  iptables -A FORWARD -j LOGGING
  iptables -A LOGGING -m limit --limit 2/min
     -j LOG --log-prefix "IPTables-Dropped: " --log-level 4
  iptables -A LOGGING -j DROP
```

Figura 2.18: Escritura de Log de iptables

```
# Se guardan las reglas
  service iptables save
```

Figura 2.19: Guardar reglas de Iptables

2.3 Implementación de FAIL2BAN

Fail2Ban es una herramienta muy utilizada para detectar intentos de login de posibles atacante que utilicen alguna herramienta de ataque de fuerza bruta. Es por eso que el objetivo de esta herramienta es bloquear la ip del equipo que está tratando de conectarse algún servicio dentro de la intranet y que no tenga los accesos necesarios para realizarlo.

2.3.1 Instalación de FAIL2BAN

Se instalará el paquete Fail2Ban en el equipo que hace de Firewall para ello se deben seguir los siguientes pasos:

1) Descargar el paquete **EPEL**, a través del comando "**wget http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm"**.
2) Instalar el paquete **EPEL**, a través del comando "**rpm -ivh epel-release-6.8.noarch.rpm"**.
3) Instalar el paquete **Fail2Ban**, a través del comando "**yum install fail2ban"**.

2.3.2 Configuración de FAIL2BAN

Primero se realizar una copia del archivo donde estarán las reglas con el siguiente comando **“cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local”**.

Dentro del archivo encontraremos algunas variables de importancia como son:

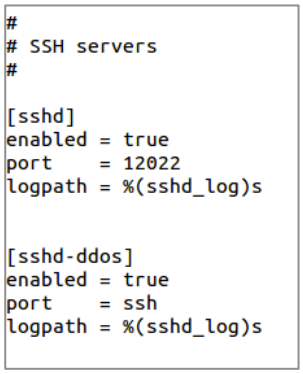
- Ignoreip: Listado de ip’s que estarán en lista blanca. Default: 127.0.0.1.
- Bantime: Número de segundos que una ip será baneada. Default: 600. La variable **“Bantime”** se cambiará para que el bloqueo sea de una hora, es decir 3600 segundos.
- Maxretry: Número de intentos fallidos antes de banear o bloquear a un ip.

2.4 Integración de IPTABLES con FAIL2BAN

Se prosigue la configuración de Fail2Ban pero en este caso se va a incluir iptables, dado que Fail2Ban creará las reglas de baneo a un listado de ips bloqueadas.

2.4.1 Configuración en FAIL2BAN

Dentro del archivo /etc/fail2ban/jail.local se debe configurar la sección de sshd y sshd-ddos. En la figura 2.20 se puede visualizar la configuración.



```
#
# SSH servers
#

[sshd]
enabled = true
port    = 12022
logpath = %(sshd_log)s


[sshd-ddos]
enabled = true
port    = ssh
logpath = %(sshd_log)s
```

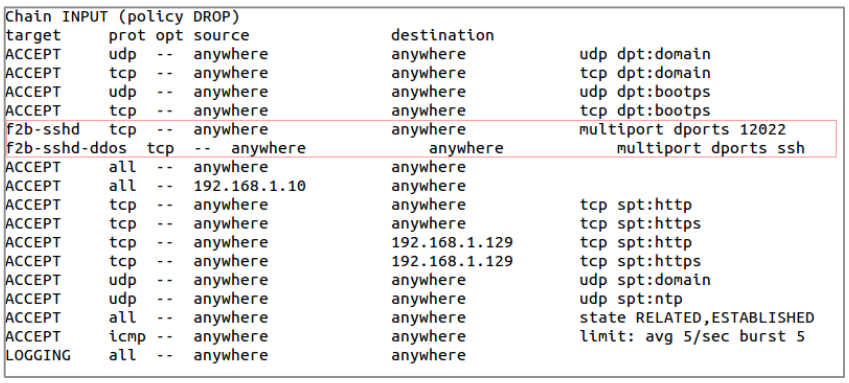
Figura 2.20: Configuración de jail.local

Realizada la configuración lo que falta es reiniciar el servicio de fail2ban con el comando **"service fail2ban start"**.

2.4.2 Configuración en IPTABLES

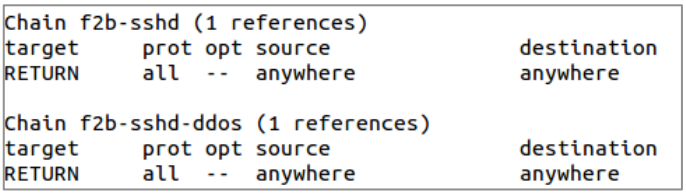
Para verificar las nuevas reglas que fail2ban ingreso se debe ejecutar el comando **"iptables –L"**.

El resultado muestra la inclusión de las nuevas reglas de baneo en las figuras 2.21 y 2.22.



```
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere            udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:bootps
f2b-sshd   tcp  --  anywhere             anywhere            multiport dports 12022
f2b-sshd-ddos  tcp  --  anywhere             anywhere            multiport dports ssh
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  192.168.1.10         anywhere
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:http
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:https
ACCEPT     tcp  --  anywhere             192.168.1.129       tcp spt:http
ACCEPT     tcp  --  anywhere             192.168.1.129       tcp spt:https
ACCEPT     udp  --  anywhere             anywhere            udp spt:domain
ACCEPT     udp  --  anywhere             anywhere            udp spt:ntp
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere            limit: avg 5/sec burst 5
LOGGING    all  --  anywhere             anywhere
```

Figura 2.21: Reglas Iptables incluidas por fail2ban



```
Chain f2b-sshd (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere

Chain f2b-sshd-ddos (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere
```

Figura 2.22: Cadenas agregadas por fail2ban en iptables

Por último es necesario agregar al final del script de reglas de iptables para que cada vez que se ejecute reinicie el fail2ban de tal manera que se agreguen las nuevas reglas de baneo:

```
# Se guardan las reglas
  service iptables save

# Se reinicia fail2ban
  service fail2ban restart
```

Figura 2.23: Ingreso a script de reglas iptables para Fail2ban

2.5 Implementación de Notificaciones

Todo administrador de servidores debe estar pendiente de lo que sucede en la red corporativa, no es suficiente la instalación y configuración de un IDS como Fail2Ban, además es necesario realizar controles para conocer cuando un intruso quiere acceder a los sistemas. Las notificaciones permiten alertar al administrador cuando un evento relacionado a un servicio o sistema está ocurriendo de manera incorrecta, de esa forma tomar las medidas necesarias para que infraestructura tecnológica no sea afectada.

2.5.1 Correo Electrónico

El mecanismo de envió de correos electrónicos es el más utilizado por los administradores, es por ello que esta herramienta ayuda mucho en el control y auditorias de servidores.

Se va a realizar un script en **“bash”** para comunicar por medio de un correo electrónico cuando exista un evento anormal sobre un servicio, en este caso el SSH por medio de Fail2Ban, cabe indicar que Fail2Ban tiene la posibilidad de enviar notificaciones por correo electrónico pero para ello es necesario que exista un servidor de correo lo cual para el caso expuesto no existe alguno dentro de la infraestructura de red de la empresa.

Primero es necesario instalar un programa llamado SendEmail, el cual es un cliente de correo mediante consola y que ayudará a enviar todos los emails que necesitemos enviar. La figura 2.24 se presenta el proceso de instalación en nuestro FW.

```
scripts]# wget caspian.dotconf.net/menu/Software/SendEmail/sendEmail-v1.56.tar.gz
scripts]# tar -xvzf sendEmail-v1.56.tar.gz
scripts]# cd sendEmail-v1.56
sendEmail-v1.56]# cp -a sendEmail /usr/local/bin
sendEmail-v1.56]# chmod +x /usr/local/bin/sendEmail
```

Figura 2.24: Instalación de comando SendEmail

La configuración para enviar el correo será la que se indica en la tabla 1.

Tabla 1: Configuración de cuenta de correo electrónico

CUENTA DE CORREO ELECTRONICO	
Cuenta	admin@media.com
Clave	12345
Servidor SMTP	mail.media.com
Puerto	26

Para el proceso de envío de correos desde FW, es necesario habilitar la salida al puerto 26. Es decir que se debe agregar una nueva regla al script donde residen todas las reglas creadas en el script del FW.

En la figura 2.25 se visualiza las nuevas reglas agregadas al script.

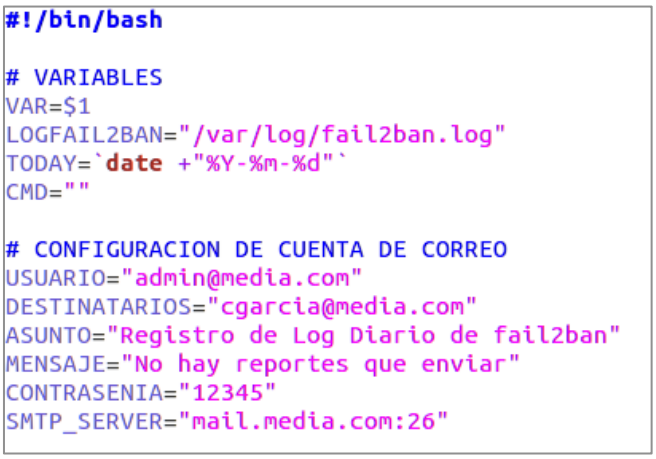
```
# Acceso a envio de correos por script de notificacion
  iptables -A OUTPUT -p tcp -o $eth_EXT --dport 26 -j ACCEPT
```

Figura 2.25: Regla para permitir salida en FW al puerto 26

El script escrito en “bash” se deberá ubicar en el directorio **“/opt/scripts”**. Los siguientes comandos se deben ejecutar a continuación:

1) **mkdir -p /opt/scripts**
2) **touch /opt/scripts/notificacion.sh**
3) **chmod +x /opt/scripts/notificacion.sh**

El contenido del archivo se puede visualizar en las siguientes figuras.



```
#!/bin/bash

# VARIABLES
VAR=$1
LOGFAIL2BAN="/var/log/fail2ban.log"
TODAY=`date +"%Y-%m-%d"`
CMD=""

# CONFIGURACION DE CUENTA DE CORREO
USUARIO="admin@media.com"
DESTINATARIOS="cgarcia@media.com"
ASUNTO="Registro de Log Diario de fail2ban"
MENSAJE="No hay reportes que enviar"
CONTRASENIA="12345"
SMTP_SERVER="mail.media.com:26"
```

Figura 2.26: Configuración de variable en script de notificaciones

En la figura 2.27 se puede apreciar una función cuya tarea es de dar formato al log que devuelve fail2ban.

```
# DECLARACION DE FUNCIONES
function convertStringLogToArray(){
        read -a arr <<< $1
        linea=""
        dataTotal="<html><head><style>table{border-collapse:collapse;}table,td,th
{border: 1px solid black}</style><title>Reporte</title><body><table
border=1><tr><td><strong>ID</strong></td><td><strong>DIA</strong></
td><td><strong>HORA</strong></td><td><strong>SERVICIO</strong></td><td><strong>IP</
strong></td></tr>"
        for index in "${!arr[@]}"
        do
                n=$((index%5))
                size=$(expr $index + 1)
                # Se debe crear una nueva linea
                if [ $n = 0 ]; then
                        if [ -n "$linea" ]; then
                                dataTotal="$dataTotal<tr>$linea</tr>"
                        fi
                        linea=""
                fi
                # Se debe concatenar el valor
                linea="$linea<td>${arr[index]}</td>"
                if [ $size = ${#arr[@]} ];then
                        dataTotal="$dataTotal<tr>$linea</tr>"
                fi
        done
        echo -n -e "$dataTotal</table></body></html>"
}
```

Figura 2.27: Función que formatea el contenido de log

En la figura 2.28 se visualiza una función que permite filtrar la salida del log de Fail2Ban con el objetivo de devolver solo la información que se requiera. En este caso el script debe poder enviar cada 5 minutos los eventos nuevos que se reportan en los logs.

```
# DECLARACION DE FUNCIONES
function filtradorDeLogs(){
        read -a arr <<< $1
        dato1="${arr[1]}"
        dato2="${arr[2]}"
        datetosearch=`date --date "-5 minute" +%s`
        newdata=""
        for index in "${!arr[@]}"
        do
                n=$((index%5))
                if [ $n = 0 ]; then
                        pos0=$index
                        pos1=$index+1
                        pos2=$index+2
                        pos3=$index+3
                        pos4=$index+4
                        datelog=`date --date "${arr[pos1]} ${arr[pos2]}" +"%Y-%m-%d %
H:%M:%S"`
                        diferencia=$(expr $(expr $datetosearch - $(date --date
"$datelog" +%s)) / 60)
                        if [ $diferencia -le 5 ]; then
                                newdata="$newdata ${arr[pos0]} ${arr[pos1]} ${arr
[pos2]} ${arr[pos3]} ${arr[pos4]}"
                        fi

                fi
        done
        echo -n -e "$newdata"
}
```

Figura 2.28: Declaración de función que segmenta logs antiguos

Existen 2 tipos de notificaciones:

1) “Notificación Diaria” enviará un correo electrónico cada 5 minutos si hubiera actividad en los logs.
2) La segunda “Notificación Nocturna” enviará un correo electrónico todos los días a las 23:45 con el reporte de lo acontecido durante el día. El comando awk [12] como se visualiza dentro de la figura 2.29 es quien segmenta el log para poder procesar el contenido del mismo que será enviado en la notificación.

En la figura 2.29 se indica el tipo de notificación que se va a realizar, en este caso se van a realizar 2.

```
# Evaluacion de tipo de reporte
case $VAR in
        diaria)
                CMD=$(awk '($(NF-1) = /Ban/){print $1,$2,$(NF-2),$NF}' $LOGFAIL2BAN
| grep `date +%Y-%m-%d` | sort | uniq -c | sort -n)
                CMD=$( filtradorDeLogs "$CMD" )
                MENSAJE=$( convertStringLogToArray "$CMD" )
        ;;
        nocturna)
                CMD=$(awk '($(NF-1) = /Ban/){print $1,$2,$(NF-2),$NF}' $LOGFAIL2BAN
| grep `date +%Y-%m-%d` | sort | uniq -c | sort -n)
                ASUNTO="Registro de Log del fail2ban a la fecha de $TODAY"
                MENSAJE=$( convertStringLogToArray "$CMD" )
        ;;
        *)
                echo "ejecucion de comandos"
        ;;
esac
```

Figura 2.29: Evaluación de tipo de notificación

En la figura 2.30 se indica el proceso de envió del correo electrónico con el reporte respectivo sea “diario” o “nocturno”. Aquí se utiliza la aplicación sendEmail que fue instalada anteriormente.

```
# Envio de reportes
if [[ -n "$CMD" ]]; then
        echo "Enviando reportes"
        # Se envia el correo con el mensaje Seteado
        /usr/local/bin/sendEmail -f "$USUARIO" -t "$DESTINATARIOS" -u "$ASUNTO" -m
"$MENSAJE" -o message-content-type=html -o message-charset=UTF-8 -s "$SMTP_SERVER" -
o tls=no -xu "$USUARIO" -xp "$CONTRASENIA"
else
        echo "No hay reportes que enviar"
fi
```

Figura 2.30: Envío de notificaciones por correo electrónico

Una vez realizado el script lo único que falta es crear la tarea programada que ejecute el script en las horas indicadas.

Se debe crear un archivo llamado “notificaciones” en el directorio **“/etc/cront.d/”** por medio del siguiente comando **“touch /etc/cron.d/notificaciones”**. El contenido que tendrá el siguiente archivo es el que se muestra en la figura 2.31:

```
45 23 * * * root /bin/bash /opt/scripts/notificacion.sh "nocturna"
*/5 * * * * root /bin/bash /opt/scripts/notificacion.sh "diaria"
```

Figura 2.31: Contenido de archivo cron de notificaciones

En figura anterior 2.31 los 5 primeros parámetros de la primera línea quieren decir: Minuto, Hora, Día del Mes, Mes, Día de la Semana, Usuario que ejecuta el comando, script o comando a ejecutar [13]. Es decir para la primera línea la tarea se debe ejecutar a las 23 Horas con 45 minutos todos los días. En la segunda línea se indica que se debe ejecutar el comando cada 5 minutos todos los días. Además del tiempo la diferencia radica en el parámetro que se agrega al comando “nocturna” o “diaria” que significa la acción que debe tomar el script para obtener la información que será enviada por correo.

CAPÍTULO 3

ANÁLISIS DE RESULTADOS

3.1 Reducción de Ataques Informáticos

La seguridad de un firewall reduce el índice de ataques pero no garantizan que un atacante persista en tratar de acceder al servidor que desea acceder. En este caso Fail2Ban nos ayuda a bloquear intentos de conexión fallidas bloqueando la ip origen del atacante. En la figura 3.1 se trata de acceder al FW a través de la ip 192.168.1.145.

```
[root@localhost ~]# ssh root@192.168.1.145 -p12022
root@192.168.1.145's password:
Permission denied, please try again.
root@192.168.1.145's password:
Permission denied, please try again.
root@192.168.1.145's password:
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
[root@localhost ~]# ssh root@192.168.1.145 -p12022
ssh: connect to host 192.168.1.145 port 12022: Connection refused
```

Figura 3.1: Bloqueo de ip por intentos fallidos de conexión

En la figura 3.2 se puede visualizar que la ip 192.168.1.129 ha sido bloqueada. Esa regla fue agregada por fail2ban al haber superado el número de intentos de conexión permitidos.

```
Chain f2b-sshd (1 references)
target     prot opt source               destination
REJECT     all  --  192.168.1.129        anywhere            reject-with icmp-port-unreachable
RETURN     all  --  anywhere             anywhere
```

Figura 3.2: Bloqueo de ip en reglas de Iptables

En la figura 3.3 se puede ver el log de fail2ban donde indica que la ip 192.168.1.129 fue bloqueada.

```
2015-07-26 01:08:29,002 fail2ban.filter         [2825]: INFO    [sshd] Found 192.168.1.129
2015-07-26 01:08:34,511 fail2ban.filter         [2825]: INFO    [sshd] Found 192.168.1.129
2015-07-26 01:08:51,764 fail2ban.filter         [2825]: INFO    [sshd] Found 192.168.1.129
2015-07-26 01:08:51,912 fail2ban.actions        [2825]: NOTICE  [sshd] Ban 192.168.1.129
2015-07-26 01:08:53,589 fail2ban.filter         [2825]: INFO    [sshd] Found 192.168.1.129
2015-07-26 01:18:52,367 fail2ban.actions        [2825]: NOTICE  [sshd] Unban 192.168.1.129
```

Figura 3.3: Mensaje de bloqueo a través del log de fail2ban

Si se revisa de nuevo las reglas de Iptables por medio del comando “iptables -L” se podrá visualizar en la figura 3.4 que ya no existe bloqueo para la ip 192.168.1.129.

```
Chain f2b-sshd (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere

Chain f2b-sshd-ddos (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere
```

Figura 3.4: Regla del Iptables sin ip bloqueada

Como se puede verificar las pruebas realizadas garantizan que un equipo pueda ser bloqueado por tratar de acceder a un servidor sin tener los accesos.

3.2 Alerta de Intento de Conexiones

Este proceso de notificaciones le permite a un administrador conocer lo que esta pasando en los servidores de tal manera que pueda ejecutar acciones sobre la información obtenida.

Basándose en el IDS Fail2Ban se elaboró un script para alertar a un administrador de los sucesos que ocurren en un equipo, poniendo en practica lo elaborado se realizaron pruebas y se obtuvo un listado de ip's que fueron bloqueadas por el IDS. En la figura 3.5 se puede visualizar el listado emitido a las 23:45 (notificación nocturna).

ID	DIA	HORA	SERVICIO	IP
1	2015-07-27	01:06:54,395	[sshd]	192.168.1.129
1	2015-07-27	01:27:09,249	[sshd]	192.168.1.129
2	2015-07-27	01:27:20,249	[sshd]	192.168.1.130
3	2015-07-27	01:27:29,249	[sshd]	192.168.1.140
4	2015-07-27	01:27:19,249	[sshd]	192.168.1.12
5	2015-07-27	01:28:09,245	[sshd]	192.168.1.100
6	2015-07-27	01:30:00,200	[sshd]	192.168.1.19
7	2015-07-27	01:37:00,229	[sshd]	192.168.1.10
8	2015-07-27	02:27:00,149	[sshd]	192.168.1.20
9	2015-07-27	03:00:09,119	[sshd]	192.168.1.120
10	2015-07-27	03:28:09,243	[sshd]	192.168.1.40
11	2015-07-27	03:30:00,139	[sshd]	192.168.1.24
12	2015-07-27	04:00:00,219	[sshd]	192.168.1.11
13	2015-07-27	05:30:00,249	[sshd]	201.183.54.54

Figura 3.5: Recepción de notificación enviada a correo electrónico

Cabe indicar que con esta información el administrador puede decidir si desea bloquear definitivamente una IP, debido a que esta dirección externa o interna ha sido bloqueada una cantidad de veces considerable, por lo que se puede decir que se trata de un ataque de fuerza bruta y están tratando de explotar el servidor.

3.3 Confidencialidad de la Información

Se puede decir que la información es confidencial dada que se ha implementado un método para mantenerla segura y visible solo para quien tenga los accesos necesarios.

Las siguientes figuras se pueden visualizar el antes y después de aplicar el cifrado de datos.


```
{
    "data":
    [
        {
            "PER_NOMBRE": "Juan",
            "PER_APELLIDO": "Piguave Martinez",
            "PER_CEDULA": "1234567890",
            "TPER_NUM_TARJETA": "502020202020",
            "TPER_CODE_VERIFICACION": "123",
            "TPER_MES_EXPIRACION": "MARZO",
            "TPER_ANIO_EXPIRACION": "2017",
        }
    ]
}
```


Figura 3.6: Consulta base de datos antes de implantar método de cifrado

```
{
    "data":
    [
        {
            "PER_NOMBRE": "Juan",
            "PER_APELLIDO": "Piguave Martinez",
            "PER_CEDULA": "1234567890",
            "TPER_NUM_TARJETA": "U2FsdGVkX1+2XKOTSe8p+NTbEG0jgFALyq2AzQjdyV0=",
            "TPER_CODE_VERIFICACION": "U2FsdGVkX19K1DJ0VBr3U+urah6Kq2T2jJwIGeEsSko=",
            "TPER_MES_EXPIRACION": "U2FsdGVkX1/IoZJ3+oPjwQUus8dAuSTPCoXz20akZ1A=",
            "TPER_ANIO_EXPIRACION": "U2FsdGVkX18Xw5vCBVr2BOqOOlbQB/zdq0ta4qsachI=",
        }
    ]
}
```


Figura 3.7: Consulta base de datos después de implantar método de cifrado

La información esta cifrada de tal manera, que si un hacker ingresa y roba la base de datos necesitaría la clave maestra para descifrar el contenido de las tablas.

3.4 Pruebas y Resultados con Servidores

Los ataques de fuerza bruta hacia un servidor aumentan el ancho de banda y consumen recursos de CPU es por eso que con un sistema como Fail2Ban el bloqueo de intrusos ahorraría mucho el ancho de banda utilizado y CPU.

Para este caso se utilizará un programa para realizar el ataque de fuerza bruta llamado Hydra con un listado de posibles claves, luego analizará utilizando Iptables y Fail2Ban el consumo de CPU.

Para empezar en la tabla 2 se muestra el listado de posibles claves para realizar la prueba de ataque de fuerza bruta.

Tabla 2: Ejemplo de claves para Hydra

EJEMPLO DE CLAVES
P4$$w0rd
abc123
12345
1234567890
qwerty
love
god
admin
password
4dm1n

Para realizar la prueba es necesario tener instalado Hydra. El comando para realizar las pruebas es **“hydra -l root -P passwordlist.txt -u ssh://192.168.0.2:12022 -t 4”.**

Una vez ejecutado lo anterior utilizamos en el servidor la herramienta top por consola para visualizar los procesos. Cabe indicar que esta primera prueba se la realiza sin Firewall ni con Fail2Ban. En la figura 3.8 se puede visualizar la salida del comando top.

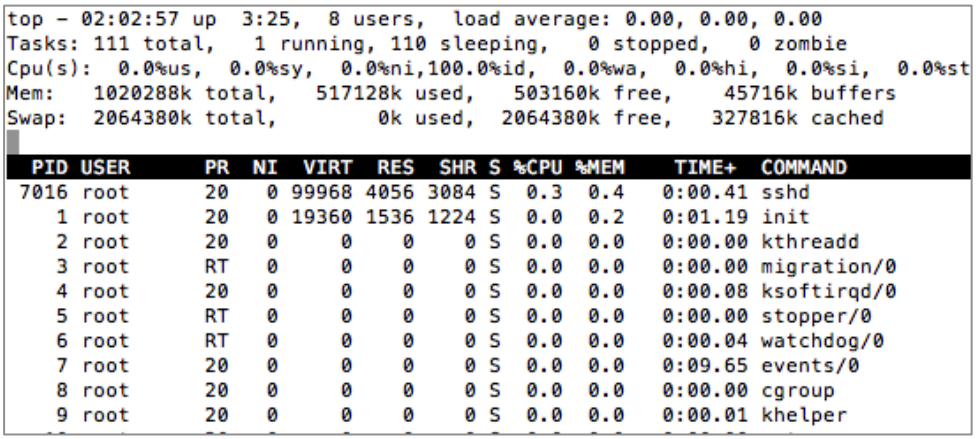


Figura 3.8: Sesión ssh sin establecer ejecutándose en FW sin iptables ni IDS

Se puede visualizar que existe un consumo de 0.4% en memoria y 0.3% en cpu por parte del servicio sshd.

En la figura 3.9 se visualiza la salida del comando top pero en este caso el Firewall y Fail2Ban están ejecutándose.

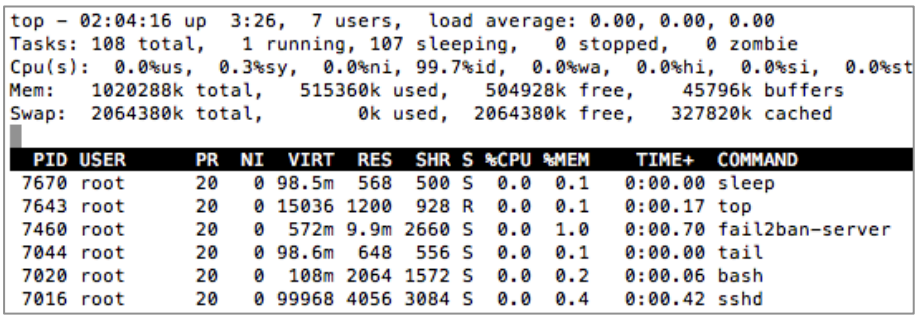


Figura 3.9: Proceso Fail2Ban ejecutándose en el FW

Como se puede visualizar el servicio fail2ban-server tiene un consumo de casi 0% de cpu y de memoria 1.0% mientras que el servicio sshd tiene casi 0% de cpu y 0.2% en memoria.

En la figura 3.10 se puede apreciar que la sesión ssh que quería establecer a través de Hydra ya no aparece dentro de los procesos.

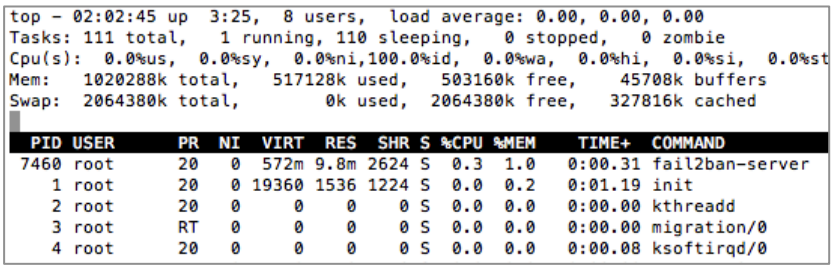


Figura 3.10: Sesión ssh bloqueada

En la figura 3.11 se visualiza el log de Fail2Ban que notifica que una ip ha sido bloqueada.

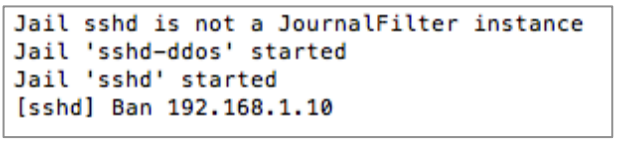


Figura 3.11: Bloqueo de ip atacante

Se puede decir que para el proceso sshd para una sesión, se puede evitar el consumo de memoria y de cpu dado que una vez que el origen ha sido baneado el servicio sshd no escuchara peticiones de ese origen por lo que se evita el uso de cpu y memoria. También controla un ataque de DoS cortando las sesiones que sobrepasen el número de intentos fallidos de conexión.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES:

1. Iptables es una herramienta muy poderosa y además gratuita de la cual se puede sacar provecho al máximo si se sabe administrar correctamente debido a su simplicidad y fácil integración con otros programas a través de la consola de comandos.
2. En este proyecto Iptables fue implementado con reglas por defecto DROP, lo cual para el administrador significa una mayor protección de la red con respecto a herramientas de escaneo, pero al implementar reglas de ese tipo existirá un problema con los servicios que requieran abrir puertos hacia el internet o para comunicar de una red a otra.
3. Esta implementación no controla la infección de malware en las

estaciones de trabajo ya sea por descargas de programas desde el internet, pero si bloquea intento de conexiones externas a puertos no permitidos en el firewall a diferencia del puerto 80 el cual no debe ser bloqueado dado que significa que las estaciones de trabajo no puedan navegar al internet.

4. El cifrado AES garantiza la confidencialidad de la información pero si se usa una clave débil un atacante puede fácilmente recuperar la información utilizando herramientas especializadas y romper la seguridad de la misma.
5. Fail2Ban es una herramienta muy versátil y la cual permite cambiar las acciones que se realicen sobre un objeto que cumpla con la regla de filtrado en logs lo que permite al administrador configurarlas como prefiera.

RECOMENDACIONES:

1. Se debe agregar al Fail2Ban más servicios que se utilicen por ejemplo la lectura de logs del Servidor Web.
2. Utilizar una clave extremadamente fuerte como clave maestra esta debe ser como mínimo de 8 caracteres, entre letras mayúsculas, minúsculas, números y caracteres especiales.
3. Utilizar otros sistemas de detección y prevención de intruso como por ejemplo SNORT y tratar de integrarlo con los servicios expuestos en este

proyecto.

4. Tener en cuenta el listado de puertos que se deben abrir dentro del firewall para comunicación ya sea entre redes internas o hacia el internet.
5. Siempre mantener actualizado los servidores con los últimos parches y actualizaciones de programas para evitar tener puertas traseras que puedan ser usadas para vulnerar un servidor o una red.

BIBLIOGRAFÍA

[1] Wikipedia, RSA, https://es.wikipedia.org/wiki/RSA, fecha de consulta Julio 15 del 2015

[2] Wikipedia, AES, https://es.wikipedia.org/wiki/Advanced_Encryption_Standard, fecha de consulta Julio 15 del 2015

[3] Wikipedia, Criptografía Simétrica, https://es.wikipedia.org/wiki/Criptograf%C3%ADa_sim%C3%A9trica, fecha de consulta Julio 17 del 2015

[4] Wikipedia, Criptografía Asimétrica, https://es.wikipedia.org/wiki/Criptograf%C3%ADa_asim%C3%A9trica, fecha de consulta Julio 18 del 2015

[5] Astudillo, K. ,Como hackear profesionalmente en 21 días o menos, CreateSpace Independent Publishing Platform 1nd Ed, Octubre 25 del 2013

[6] Juancarlosmolinos, Configuración Firewall Iptables, http://juancarlosmolinos.net/2012/03/08/configuracion-firewall-iptables-red-hatcentos-6-desde-linea-de-comandos, fecha de consulta Julio 18 del 2015

[7] TLDP-ES/LuCAS, Iptables manual práctico, http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf, fecha de consulta Julio 19 del 2015

[8] Neeonez, Instalar Fail2Ban en Centos 6, http://www.neeonez.com/instalar-failt2ban-en-centos-6-para-proteger-los-accesos-ssh-y-ftp, fecha de consulta

Julio 20 del 2015

[9] Wikipedia, Bash, https://es.wikipedia.org/wiki/Bash, fecha de consulta Julio 21 del 2015

[10] TLDP-ES/LuCAS, Programación en Bash, http://es.tldp.org/COMO-INSFLUG/COMOs/Bash-Prog-Intro-COMO, fecha de consulta Julio 22 del 2015

[11] Wikipedia, List of TCP and UPD port numbers, https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers, fecha de consulta Julio 24 del 2015

[12] The Art of Web, Monitoring Fail2Ban Logs, http://www.the-art-of-web.com/system/fail2ban-log, fecha de consulta Julio 25 del 2015

[13] Wikipedia, Cron (Unix), https://es.wikipedia.org/wiki/Cron_%28Unix%29, fecha de consulta Julio 27 del 2015