Desconexión temprana de equipos comprometidos: Descon2, la propuesta de la Universidad Carlos III de Madrid

ENFOQUES

Early Disconnection of Compromised Computers: Descon2, the University Carlos III Approach

Rafael Calzada Pradas

Resumen

Nuestra red es un entorno abierto, en el que hemos reducido al mínimo las restricciones de conectividad para facilitar las labores de docencia e investigación. Esta situación aumenta la exposición a ataques de los equipos conectados a nuestra red. Por ello, hemos configurado de la forma más robusta posible los puntos de acceso a la red, para repeler los ataques desde el exterior y protegido los servidores corporativos, siguiendo las guías de buenas prácticas. Pero, quién protegerá al usuario de otros usuarios, y también de sí mismo. No todos los usuarios tienen el mismo nivel de consciencia sobre la importancia de la seguridad informática y este tipo de redes son propensas a la propagación de virus/gusanos. Para evitar esta situación, en UC3M (la Universidad Carlos III) hemos comenzado el desarrollo del proyecto Descon2, que pretende la agregación de información de seguridad, para la detección temprana de equipos comprometidos y su posterior puesta en cuarentena.

Palabras clave: Redes auto-defendidas, detección temprana, cuarentena de equipos comprometidos, agregación de información de seguridad.

Summary

Our network is a wide open environment, in which connectivity restrictions have been minimized in order to facilitate research and teaching activities. This situation increases the attack exposure level of computers connected to the network. For that, we have fortified the network access points to repel the most common attacks from outside and protect the core servers, with well-know best practices. But not all users have the same security awareness level, and this kind of networks are prone to network outbreaks due virus/worms expansion. We have started the development of Descon2 to deal with this situation, summarizing security information in order to detect compromised computers and quarantine them as soon as possible.

Keywords: Self defending networks, early detection, quarantine compromised computers, summarizing security information

El proyecto Descon2, pretende la agregación de información de seguridad, para la detección temprana de equipos comprometidos y su posterior puesta en cuarentena

1.- Introducción

La seguridad de un sistema depende siempre del elemento más débil del mismo. Tratándose de un sistema informático, los elementos principales son los servidores corporativos, los ordenadores de los usuarios y la red que los interconecta. Si deseamos incrementar la seguridad del sistema, tendremos que incidir en estas tres áreas. Pero los entornos académicos y de investigación tienen peculiaridades que los hacen especiales.

La seguridad de un sistema depende siempre del elemento más débil del mismo y en un sistema informático, los elementos principales son los servidores corporativos, los ordenadores de los usuarios y la red que los interconecta

Generalmente se trata de entornos abiertos, en los que el personal docente e investigador demanda un acceso completo y sin restricciones al mismo, para poder realizar su trabajo. Cualquier restricción que se desee aplicar debe ser justificada y no debe interferir con su labor investigadora.

En UC3M hemos sufrido varias veces infecciones masivas, provocadas por gusanos que rápidamente se extienden por una red en la que los equipos no tienen instaladas las actualizaciones de seguridad, configurado el cortafuegos y/o instalado el anti-virus corporativo. La justificación a esta situación se encuentra en que la universidad desarrolla proyectos de investigación que requieren el desarrollo de herramientas/uso de las mismas en entornos muy concretos en los que a veces no es posible tener el sistema completamente asegurado.

Tras analizar el problema, determinamos que hay dos formas de afrontarlo:

1.- Enfoque preventivo, que consiste en prevenir la infección/propagación mediante la obligación de disponer de los elementos de seguridad (actualizaciones, anti-virus y cortafuegos) actualizados y correctamente configurados para permitir el acceso a la red.

2.- Enfoque reactivo, desde el que cualquier equipo tiene permitido el acceso a la red, pero si se detecta un comportamiento anómalo, se denegará el acceso al equipo comprometido.

Lo obvio en esta situación es que más vale prevenir que curar, por lo que siempre que se pueda debe prevenirse, pero, como ya se ha mencionado, el entorno académico y de investigación no facilita el enfoque preventivo.

Más vale prevenir que curar, por lo que siempre que se pueda debe prevenirse, pero el entorno académico y de investigación no facilita el enfoque preventivo

2.- Productos evaluados

Durante el segundo semestre de 2004, evaluamos diferentes aproximaciones para reducir la propagación de infecciones masivas. Las dos primeras son soluciones comerciales, basadas en el enfoque preventivo, mientras que la última es una solución abierta, basada en *software* libre y con un enfoque reactivo.

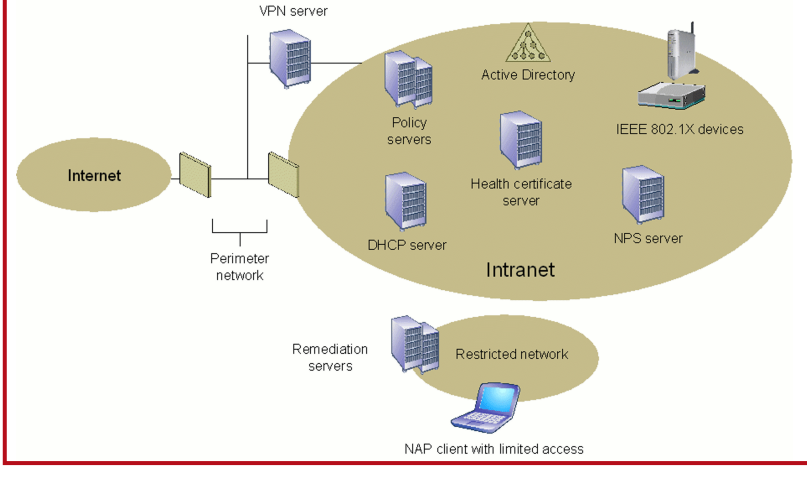
2.1.- Microsoft Connection Manager

Se trata de un plataforma gratuita de Microsoft [1], que ahora se distribuye bajo la denominación Network Access Protection [2], cuyo objetivo es obligar a que los sistemas que se conectan a la red cumplan con una política predeterminada, a efectos de elementos de seguridad. El nivel de acceso a la red, se basa en el nivel de cumplimiento de dicha política. Así, podemos configurar que todos los sistemas con las actualizaciones y el patrón de anti-virus actualizado a día de ayer, pueden acceder a la red sin restricciones, puesto que se pondrán al día en un período de tiempo breve y no suponen una amenaza seria a la seguridad del entorno informático de la institución. Los equipos que no cumplan con la política son conectados a una red de cuarentena, en la que sólo tienen permiso para acceder a las actualizaciones del sistema y de anti-virus. Periódicamente se evalúa su nivel de cumplimiento de la política para determinar si deben ser conectados a la red de explotación.

Hemos evaluado varias aproximaciones para reducir la propagación de infecciones masivas: dos comerciales, basadas en el enfoque preventivo y una solución abierta, basada en *software* libre y con un enfoque reactivo

Se trata pues de un sistema preventivo, que está orientado a uno de los más conflictivos, si tenemos el número de ataques recibidos, que son los sistemas Microsoft Windows. Además Microsoft está invirtiendo mucho en este sistema, lo que aporta un respaldo notable.

FIGURA 1: ARQUITECTURA DE MICROSOFT NAP



Por desgracia, Microsoft se ha centrado especialmente en su plataforma (Windows Vista y Longhorn para Network Access Protection, y Windows 2000 y posteriores para Connection Manager) lo que hace que no sea factible su implantación, teniendo en cuenta que habrá sistemas sobre los que no pueda comprobarse su nivel de riesgo.

Además, en nuestro caso, su implantación exigiría la migración del directorio LDAP a Active Directory, o en su defecto, a instalar un sistema intermedio que hiciese las labores de pasarela. Si a esto se añade que el sistema es responsable de permitir el acceso a la red y por lo tanto debería tener redundancia para prevenir contingencias, el número de servidores a mantener aumenta.

2.2.- Cisco Network Admission Control

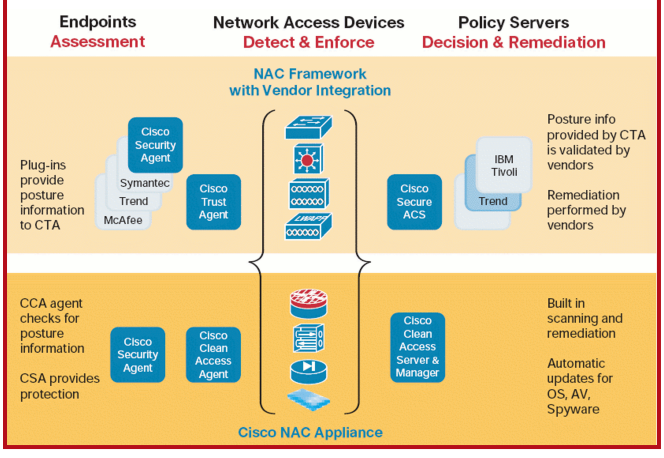
La propuesta de Cisco [3], está basada en el estándar 802.1x [4]. La autenticación se realiza empleando un servidor Radius/Ldap y para la comprobación del nivel de riesgo se emplea el Cisco Security Agent, que permite la integración de módulos de terceros.

Hoy día Cisco ha desarrollado el sistema NAC para prácticamente todos los productos soportados

Cisco ha establecido acuerdos con los principales fabricantes de anti-virus y otros productos de seguridad, asegurando que su plataforma tiene el respaldo necesario para competir con otros sistemas.

Los sistemas potencialmente peligrosos son confinados en una red de cuarentena, implementada como una VLAN con restricciones adicionales en el encaminador.

FIGURA 2: ARQUITECTURA DE CISCO NAC



Cuando nosotros evaluamos el producto de Cisco, sólo estaba disponible para equipos de gama alta o muy recientes, a día de hoy, tras dos años, Cisco ha desarrollado el sistema NAC para prácticamente todos los productos soportados. Pero el principal inconveniente que hemos visto estriba en la necesidad de instalar el Cisco Security Agent en el equipo del usuario y que Cisco sólo ofrece el soporte para sistemas Microsoft Windows, dejando el resto de los sistemas en manos de terceros.

La idea principal del sistema Netsquid consiste en ubicar un sistema Linux entre los conmutadores y el encaminador de cada subred

2.3.- Netsquid

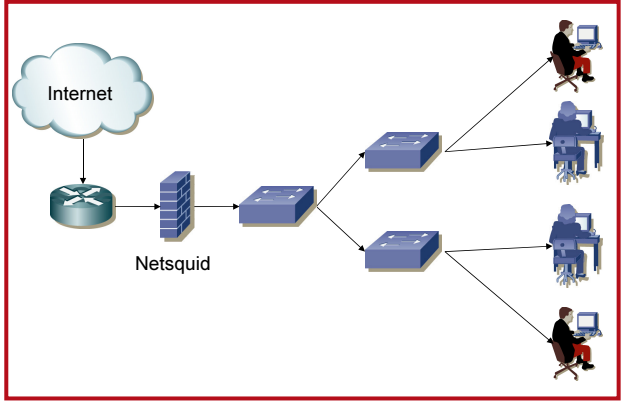
Netsquid [5] surgió como respuesta de la Texas A&M University a los gusanos Nachi/Whelchia, que azotaron Internet en octubre de 2003. La idea principal del sistema consiste en ubicar un sistema Linux entre los conmutadores y el encaminador de cada subred, este sistema Linux actuará como conmutador, pero también ejercerá de cortafuegos empleando iptables.

La detección de equipos comprometidos se basa en el Sistema de Detección de Intrusiones (IDS) *snort* [6], utilizando un conjunto de reglas denominado *bleeding-rules* [7]. Estas reglas suelen ser muy recientes, lo que permite la detección de virus/gusanos de nueva aparición, aunque también son

susceptibles de dar falsos positivos, indicando que un sistema está comprometido, cuando realmente está generando tráfico legítimo.

El sistema puede configurarse para que sólo bloquee de forma temporal aquellos sistemas que hayan generado una alerta perteneciente a una de las clases de alertas de *snort.* Una vez bloqueado, todo su tráfico web es redirigido al servidor web que se haya indicado, habitualmente ese sitio web contiene las herramientas necesarias para desinfectar/limpiar el sistema.

FIGURA 3: ARQUITECTURA DE NETSQUID



También permite enviar un mensaje de Windows (WinPopUp) al sistema tras recibir la primera alerta de infección, proporcionando un mecanismo adicional de comunicación con los usuarios de sistemas Microsoft Windows.

Dado el carácter abierto de nuestra red, Netsquid se adecua mucho más que los anteriores a nuestra filosofía, pero también presenta inconvenientes

Dado el carácter abierto de nuestra red, este sistema se adecua mucho más que los anteriores a nuestra filosofía, pero también presenta inconvenientes. El sistema Linux se convierte en un punto único de fallo, menos fiable que los conmutadores de red, ya que es más sensible a vulnerabilidades, fallos de configuración y sobre todo a problemas con el suministro eléctrico.

Por otro lado, el sistema sólo recoge información de *snort* y tiene poca granularidad, ya que una alerta puede provocar la desconexión de la dirección IP implicada, lo que además hace que el sistema sea vulnerable a ataques de negación de servicio basados en falsificación de la dirección IP origen *(ip-spoofing)*.

Desde nuestro punto de vista, un sistema comprometido debe ser puesto en cuarentena lo antes posible

3.- Descon2

Tras la evaluación de los productos anteriormente citados y llegar a la conclusión de que ninguno de ellos se adaptaba realmente a nuestras necesidades, decidimos dedicar un periodo de tiempo a idear estrategias para abordar el problema que supone tener un equipo comprometido conectado a la red.

3.1.- Antecedentes

En los Grupos de Trabajo de 2005, Víctor Pérez Roche presentó TrackUZ [8], un desarrollo realizado por la Universidad de Zaragoza (UNIZAR), para detectar equipos comprometidos por *botnets* [9] basándose en el comportamiento detectado en el equipo PacketShaper [10] de Packeteer. TrackUZ evalúa cada 15 minutos el nivel de compromiso de los equipos sospechosos de estar comprometidos y en caso de determinarse que el sistema tiene un comportamiento similar a un sistema comprometido, se le restringe el acceso a Internet y se redirecciona su tráfico web a un servidor en el que se le informa al usuario de la situación de su equipo, las causas que han determinado la desconexión y ayuda a corregir el estado del sistema.

Desde nuestro punto de vista, un sistema comprometido debe ser puesto en cuarentena lo antes posible y tener acceso exclusivamente a aquellos recursos necesarios para solventar el compromiso.

3.2.- Versión inicial

Con todas estas ideas, nosotros abordamos el desarrollo de un nuevo proyecto, denominado DESCONexiones 2, que permitiese aunar los aspectos positivos de los sistemas anteriores, solventar los problemas que habíamos detectado y fuera adaptable a diferentes organizaciones.

Nuestro entorno de ejecución se limita a conmutadores y encaminadores Cisco y servidores Linux, operados por el Servicio de Informática. Por la rapidez de desarrollo, determinamos desarrollar el sistema integramente en Perl [11].

Los objetivos que inicialmente se plantearon para el proyecto fueron:

- Crear un sistema que permita automatizar el proceso de desconexiones previo.
- Emplear las alertas de *snort*.
- El sistema debe permitir la utilización de otras fuentes de alertas.
- Cada alerta tendrá asignado un coste y un período de vida.
- Cada cierto tiempo se evaluará el coste total de las alertas correspondientes a cada dirección IP implicada.

El sistema se diseñó con una arquitectura distribuida. Por un lado existen los detectores, programas simples y especializados, que monitorizan algunos eventos y notifican las alertas correspondientes. Los detectores utilizan información de trazas de iptables, conexiones simultáneas en el sistema de gestión de Ancho de Banda, trazas de acceso a servidores web, sistemas de detección de intrusiones, etc. Cada alerta generada, tiene un peso o coste, y tiempo de vida. Los valores de estos atributos son configurados en cada detector para dar respuesta a los diferentes eventos.

Uno de los objetivos iniciales del proyeco fue crear un sistema que permita automatizar el proceso de desconexiones previo

Cada alerta generada, tiene un peso o coste, y tiempo de vida

```
use strict;
use Alarm;
use Detector;

# Creamos el detector
my $detector = new Detector;
open (LOG, "tail –f –retry –n0 –follow=name /var/log/kern.log|");
while (<LOG>) {
    #Iteramos sobre el fichero de log
        if (/incubadora/){
                &log_incubadora($_);A
        }
}

sub reconnect{
        $detector = new Detector;
}

sub log_incubadora{
        my $al = new Alarm;
        my $line= $_;
        my $now=time();
        my $expire=$now+43200;
        $al->alarmtime($now);
        $al->expiretime($expire);
        $_=$line;
        if (/ (\w+\.uc3m\.es).*SRC=(\d+\.\d+\.\d+\.\d+) DST=(163\.117\.\d+\.\d+).*PROTO=(\w+).*SPT=(\d+) DPT=(\d+)/){
        # Es una alerta de escaneo
                $al->ipsrc($2);
                $al->ipdst($3);
                $al->proto($4);
                $al->portsrc($5);
                $al->portdst($6);
                $al->description("Escaneo detectado en $1");
        $al->cost(500);
        $al->url("none");
        reconnect() unless ($detector->send($al->serialize));
    }
}
```

Tabla I. Ejemplo de detector, implementado en PERL

Por ejemplo, el detector de iptables obtiene información de las trazas de un cortafuegos y genera una alerta por cada intento fallido de conexión hacia una red protegida por dicho cortafuegos. Entonces, para detectar escaneos lentos, hemos configurado estas alertas con un peso muy bajo, pero con un tiempo de vida elevado. De este modo, un intento erróneo de acceso no provoca la desconexión de un usuario legítimo.

Debemos destacar que los detectores son la pieza fundamental de nuestro sistema, son los vigilantes que avisan de todos aquellos eventos que pueden desencadenar una desconexión. Por ello, aunque sean programas sencillos, se les debe prestar una atención especial, configurándolos de modo que sólo envíen alertas procedentes de eventos completamente fiables, garantizando el origen de la alerta. Un error en este punto haría que el sistema fuese susceptible de un ataque por negación de servicio.

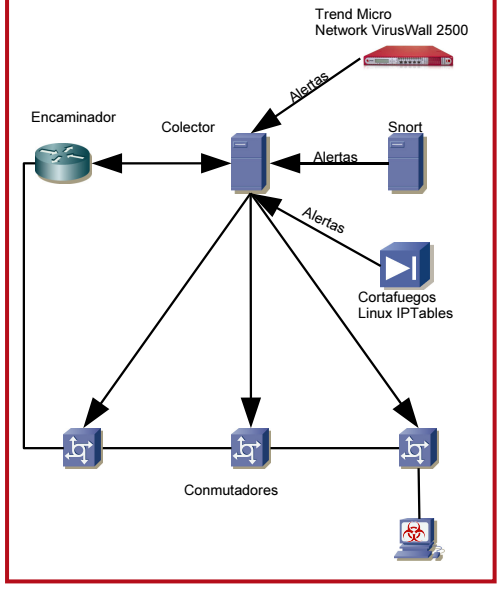
Los detectores son la pieza fundamental de nuestro sistema, son los vigilantes que avisan de todos aquellos eventos que pueden desencadenar una desconexión

Por otro lado, el colector de alarmas recoge las alertas enviadas desde los detectores y organiza las alertas en función de la dirección IP implicada. Periódicamente suma los pesos de las alertas correspondientes a cada dirección IP cuyo tiempo de vida no haya expirado, y si se ha superado el peso máximo, el ordenador es desconectado temporalmente de la red. Este programa es también responsable de la reconexión de los equipos comprometidos, una vez haya expirado su periodo de cuarentena.

Si el sistema que debe ser desconectado de la red pertenece a la red UC3M, el colector consulta al router de la subred a la que pertenece el sistema la dirección Ethernet. Tras ello emplea el protocolo Cisco Discovery Protocol (CDP), para establecer la cadena de conmutadores entre el encaminador y el equipo comprometido, el conmutador y puerto que da servicio al equipo comprometido y la lista de direcciones Ethernet detectadas en dicho puerto.

Si el sistema que debe ser desconectado de la red pertenece a la red UC3M, el colector consulta al router de la subred a la que pertenece el sistema la dirección Ethernet

FIGURA 4: ARQUITECTURA DESCON2 VERSIÓN INICIAL



Si por el contrario, el sistema comprometido no pertenece a la red UC3M, el sistema no efectúa ninguna acción.

Durante el periodo de cuarentena o desconexión, el colector ignora todas las alertas recibidas referentes a sistemas desconectados.

En esta versión inicial, Descon2 envía un mensaje de correo electrónico con toda la información recolectada y las alertas que provocaron la desconexión.

Desde el 23 de noviembre de 2005 hemos tenido en explotación esta versión, con unos resultados satisfactorios, hemos recibido más de 10.000 mensajes de aviso, lo que arroja una media de 60 mensajes diarios. El sistema nos ha permitido detectar más de 260 sistemas internos a UC3M comprometidos. También hemos tenido falsos positivos, que nos han permitido ajustar la configuración del sistema y mejorar su fiabilidad. Descon2 ha sido especialmente útil tras la aparición de un nuevo virus, para el que el fabricante de nuestro anti-virus corporativo tardó dos días en disponer del patrón de detección y unos días más en proporcionar el patrón de desinfección. Gracias a Descon2, tuvimos un máximo de nueve infecciones/día, cuando en casos anteriores habíamos llegado a más de 200.

La versión inicial, además permite bloquear las tramas de la dirección Ethernet de los sistemas comprometidos pertenecientes a UC3M en el conmutador central de cada campus. Esta funcionalidad no se ha utilizado, debido a que impide que el usuario pueda emplear cualquier mecanismo de recuperación.

La reconexión a la red se puede realizar manualmente o tras transcurrir un periodo de tiempo predeterminado (sin haber realizado ningún tipo de verificación del estado de seguridad del sistema afectado). Por ello la recuperación de un equipo afectado en esta versión se hace de forma manual, bien por parte del usuario o por parte del personal de Servicio de Informática.

La versión inicial, permite bloquear las tramas de la dirección Ethernet de los sistemas comprometidos pertenecientes a UC3M en el conmutador central de cada campus

```
-> DISCONNECTED HOSTS
   · 163.117.xxx.yyy - MAC: 0004.76d1.ghij
Switch: xxxxxx.uc3m.es Port: 5/6
MACs on this port: 0004.76d1.ghij
* VLAN 9, cost 2050
* Disconnected from xxxxxx.uc3m.es
Total alarms: 41
-<>-<>
Alarm Time: 17/05/2006 15:17:28  Expire Time: 18/05/2006 03:17:28
-> Cost:50 From 163.117.xxx.yyy:4997 to 163.117.aaa.bbb:139
-> Escaneo detectado en xxxxx.uc3m.es (none)
-<>-<>
Alarm Time: 17/05/2006 15:17:31  Expire Time: 18/05/2006 03:17:31
-> Cost:50 From 163.117.xxx.yyy:4997 to 163.117.aaa.bbb:139
-> Escaneo detectado en xxxxx.uc3m.es (none)
-<>-<>
Alarm Time: 17/05/2006 15:17:37  Expire Time: 18/05/2006 03:17:37
-> Cost:50 From 163.117.xxx.yyy:4997 to 163.117.aaa.bbb:139
-> Escaneo detectado en xxxxx.uc3m.es (none)
-<>-<>
....
```

Tabla II. Notificación de desconexión de la versión inicial de Descon2

3.3.- Versión en desarrollo

La versión inicial nos ha servido para comprobar la factibilidad del sistema, pero sobre todo, nos ha mostrado cuáles eran sus principales deficiencias. Entre las que destacan:

Una de las principales deficiencias de la versión inicial es la carencia de un mecanismo rápido y fiable de comunicar al usuario la desconexión de su sistema

- *Carencia de mecanismos para la corrección del sistema comprometido*. Está aceptado generalmente que un sistema comprometido no puede restaurarse de forma fiable si no se efectúa una reinstalación completa del sistema. Si bien la afirmación anterior es cierta, no es menos cierto, que a veces, el compromiso del sistema se reduce a un virus o gusano, que no alcanzó los privilegios del administrador del sistema, y que con una versión actualizada del anti-virus, puede ser recuperado satisfactoriamente por el propio usuario, evitando la tediosa tarea de guardar sus datos en una copia de respaldo, reinstalación del sistema y aplicaciones del usuario, y finalmente restauración de los datos.
- *Carencia de un mecanismo fiable y rápido de comunicación al usuario de la desconexión de su sistema.* Si bien la notificación por correo electrónico es uno de los mecanismos más eficaces en la transmisión de información por su característica asíncrona, para poder enviar dicho mensaje, es necesario disponer de la dirección del correo del destinatario. Podríamos disponer de una relación IP-Usuario responsable, pero es difícil mantenerla actualizada, y a veces, se emplean protocolos de asignación dinámica de direcciones IP, como DHCP, o bien el usuario es

malintencionado y usurpa una dirección IP que no le pertenece. En estos casos la información no es fiable y debe buscarse un método alternativo.

- *Dificultad para obtener el estado del sistema.* En la versión inicial, toda la información se almacena en memoria volátil y no es posible tener acceso a esa información, la única fuente de información son los mensajes de correo electrónico con las desconexiones, lo que hace que la información se encuentre en un formato poco manejable.

- *Imposibilidad de gestionar la desconexión en los entornos más conflictivos.* Desde el punto de vista de la seguridad, puede que la mayor de las amenazas provenga de aquellos equipos móviles (p.e. PDA y portátiles) y de las conexiones VPN externas. Las características principales de estos sistemas son: que se conectan a Internet empleando otros proveedores, aparte de la universidad y que en el caso de VPN, no tienen dirección Ethernet que bloquear.

La nueva versión utiliza XML para el intercambio de alertas entre los detectores y el colector de alertas

- *Un sistema comprometido debe ser aislado del resto de la red* de UC3M, bloquear el tráfico en el conmutador central no es el mejor método, ya que tiene accesibles a todos los sistemas conectados a su VLAN.

- *Ligado al lenguaje de programación PERL.* El intercambio de información se basa en un módulo de PERL, que empaqueta los datos en formato binario, lo cual dificulta la integración de detectores desarrollados en otros lenguajes de programación.

- *No existe control de acceso a la información,* todas las desconexiones son notificadas a todos los receptores de correo, con independencia del sistema desconectado.

Con estas deficiencias abordamos la nueva versión de Descon2, que incluye las siguientes mejoras con respecto a la anterior:

- Utilización de XML para el intercambio de alertas entre los detectores y el colector de alertas. Para ello hemos definido una DTD sencilla, que incluye los atributos que estamos empleando hasta hoy.

La nueva versión utiliza una base de datos sobre MySQL, que permite realizar consultas sobre cualquier atributo de las alertas o desconexiones

- Utilización de una base de datos sobre MySQL, permitiendo realizar consultas sobre cualquier atributo de las alertas o desconexiones. También se ha empleado la base de datos para mantener los usuarios del sistema y los grupos de redes que gestionan, implementando el control de accesos. Además, se generan informes periódicamente y son almacenados en formato XML en la base de datos, para su posterior consulta. Estos informes pueden ser enviados por correo electrónico y utilizados para detectar tendencias o posibles patrones de ataque.

```
<?xml version="1.0"?>
<!ELEMENT alarm (cost, proto, ipsrc, portsrc?, ipdst?,
        portdst?, description, alarmtime, expiretime, url?)>
<!ELEMENT cost (#PCDATA)>
<!ELEMENT proto (#PCDATA)>
<!ELEMENT ipsrc (#PCDATA)>
<!ELEMENT portsrc (#PCDATA)>
<!ELEMENT ipdst (#PCDATA)>
<!ELEMENT portdst (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT alarmtime (#PCDATA)>
<!ELEMENT expiretime (#PCDATA)>
<!ELEMENT url (#PCDATA)>
```

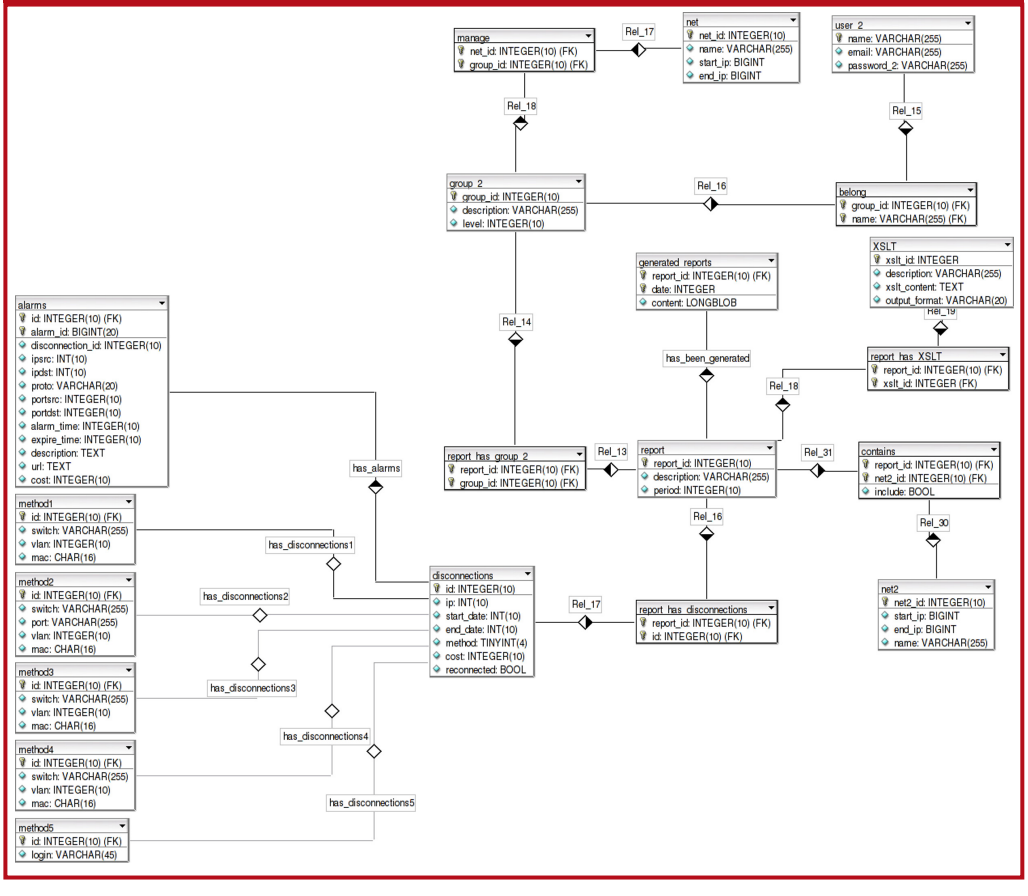
Tabla III. DTD de las alertas intercambiadas en el sistema

- Proporcionar interfaz web, que permita al personal de Centro de Atención a Usuarios y a los del Área de Seguridad y Comunicaciones, consultar el estado en tiempo real del sistema y hacer búsquedas sobre las desconexiones realizadas.

- Proporcionar al usuario un mecanismo eficaz de cuarentena, que permita notificarle la desconexión de su sistema y le permita acceder a los servicios mínimos establecidos (p.e. correo electrónico a través de Webmail) y a las herramientas necesarias para corregir el compromiso de

su sistema (p.e. anti-virus, actualizaciones, etc.), para ello vamos a utilizar una VLAN de cuarentena y servidor Linux con Squid y Transparent Proxy [12] que simulará la dirección IP del encaminador de los equipos desconectados. Todo el tráfico estará bloqueado excepto el explícitamente permitido. El tráfico web circulará a través del proxy y cualquier acceso web no explícitamente permitido, devolverá una página en la que se indicará al usuario que su sistema está en cuarentena por motivos de seguridad.

FIGURA 5: ESQUEMA DE LA BASE DE DATOS DE DESCON2



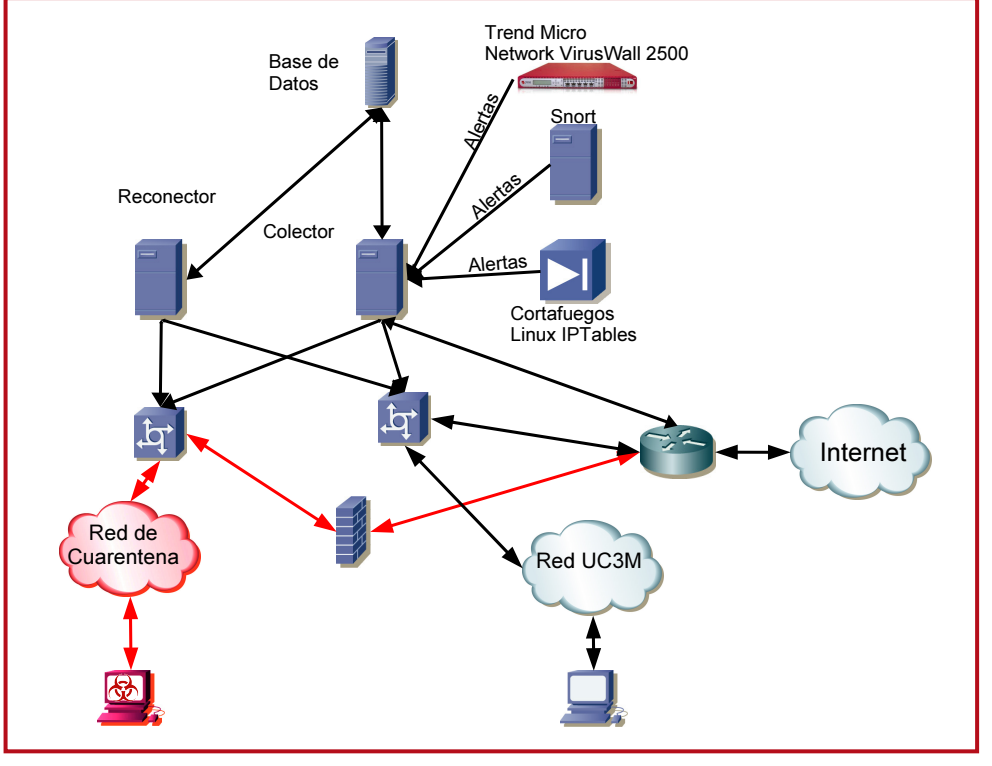
Las reconexiones no se realizarán automáticamente, sino que un nuevo elemento, Reconector, se encargará de reconectar a aquellos equipos comprometidos que ya no generen alertas

- Las reconexiones no se realizarán automáticamente, sino que un nuevo elemento, se encargará de reconectar a aquellos equipos comprometidos que ya no generen alertas.
- Adaptarnos a los métodos de conexión empleados por nuestros usuarios, para ello hemos pensado varios métodos de desconexión:
- *Método 0*. Empleado para sistemas externos a la red UC3M, consistirá en bloquear el acceso a la red UC3M, empleando para ello un cortafuegos perimetral. Además, utilizando la información contenida en la base de datos, podemos notificar al IRIS-CERT los sistemas pertenecientes a la red académica española que hemos bloqueado porque están presuntamente comprometidos.
- *Método 1*. Empleado para sistemas conectados a la red UC3M a través de un concentrador o conmutador no gestionado por el Área de Seguridad y Comunicaciones. En este caso se bloquearán el tráfico del equipo comprometido en el conmutador más cercano al mismo, gestionado por AsyC.

- *Método 2*. Empleado para sistemas conectados a la red UC3M a través un conmutador gestionado por ASyC. En este caso, el sistema comprometido es el único que accede a la red a través del puerto en el que está conectado y podemos cambiar su VLAN a la de cuarentena. Este es el método preferido ya que ofrece todas las funcionalidades.
- *Método 3*. Empleado para sistemas conectados a UC3M y que no se han podido bloquear empleando los métodos 1 y 2, consiste en bloquear en el conmutador central el tráfico del sistema comprometido.
- *Método 4*. Empleado para sistemas conectados a través de la red WiFi, en la que no es posible aplicar cuarentena, lo hemos desligado del método 3, por si en un futuro fuera posible.
- *Método 5*. Empleado para las conexiones eduroam [13] y VPN. En este caso, la desconexión consiste en bloquear el acceso en el servidor Radius que permite la conexión eduroam o los túneles VPN.

Entre algunas de las mejoras para incorporar al sistema se trata de mantener en la base de datos la configuración de los detectores, y permitir la gestión de dichas configuraciones a través de un interfaz web

FIGURA 6: ARQUITECTURA DE LA VERSIÓN DESCON2 EN DESARROLLO



3.4.- Futuras mejoras

Aun cuando no hemos terminado el desarrollo de la segunda versión, ya estamos tratando de ver cuáles pueden ser las mejoras para incorporar al sistema. Entre las detectadas se encuentran:

- Mantener en la base de datos la configuración de los detectores, y permitir la gestión de dichas configuraciones a través de un interfaz web.

- Desarrollo de nuevos detectores, p.e. trazas del servidor apache, de SSH, alertas enviadas por NFSen[14], etc.
- Identificación de los detectores al conectarse al colector.
- Cifrado de las comunicaciones entre los detectores y el colector.
- Mejorar el algoritmo de agregación de alertas.
- Incluir el concepto de reputación, para ponderar el coste de las alertas en función de la fiabilidad del detector, basándonos en la tesis doctoral de Agustín Orfila[15].

Otra posible mejora a incluir es la identificación de los detectores al conectarse al colector

4.- Conclusiones

En un entorno como el nuestro, es necesario aislar un sistema comprometido tan pronto como se tenga noticia de su existencia, de la rapidez y eficacia con la que se haga este trabajo, se puede deducir la diferencia entre incidente aislado e infección masiva.

Además, no podemos restringirnos única y exclusivamente a la red interna, desde el exterior se reciben a diario multitud de ataques que si son bloqueados a tiempo no llegarán a transformarse en intrusiones.

Nuestro sistema, es una aproximación a la automatización del proceso de aislamiento de sistemas comprometidos, probablemente muy ligada a nuestro entorno, pero que hemos intentado hacer lo más abierta posible. Durante más de seis meses, la versión inicial ha demostrado su validez y esperamos hacer disponible la versión 2 en breve.

Nuestro sistema, es una aproximación a la automatización del proceso de aislamiento de sistemas comprometidos, probablemente muy ligada a nuestro entorno, pero que hemos intentando hacer lo más abierta posible

Hemos desarrollado un sistema abierto, adaptable en mayor o menor medida a otros entornos, que puede ampliarse con todas aquellas fuentes de información de seguridad que deseemos, gracias a la utilización de un mecanismo de comunicación sencillo, basado en XML, por lo que cualquiera podría crear su propio detector y ponerlo a disposición de la comunidad.

5.- Agradecimientos

Al personal de RedIRIS, por su apoyo y permitirnos presentar nuestro trabajo en los Grupos de Trabajo de 2006 y en esta publicación.

A Víctor Barahona, por sus ánimos y el proyecto ACRI, fuente de alertas de Descon2.

A Chelo Malagón, por sus críticas y sugerencias, que han sido incluidas en la versión en desarrollo. También por su apoyo y entusiasmo.

A todos los miembros del Área de Seguridad y Comunicaciones del Servicio de Informática y Comunicaciones de la Universidad Carlos III, sin su esfuerzo y dedicación, Descon2 no habría pasado de ser un sueño.

Referencias

Todas los enlaces web estaban operativos a fecha 19 de junio de 2006

[1] *www.microsoft.com/windows2000/es/server/help/sag_cmaktopnode.htm*
[2] *www.microsoft.com/nap*
[3] *www.cisco.com/en/US/netsol/ns466/networking_solutions_package.html*
[4] *www.ieee802.org/1/pages/802.1x.html*
[5] *http://netsquid.tamu.edu*
[6] *www.snort.org*
[7] *www.bleedingsnort.com*
[8] *http://sicuz.unizar.es/trackuz/index.html*
[9] *http://es.wikipedia.org/wiki/Botnet*
[10] *www.packeteer.com/products/packetshaper*
[11] *www.cpan.org*
[12] *http://squid.visolve.com/squid/trans_caching.htm*
[13] *www.eduroam.org*
[14] *http://nfsen.sourceforge.net*
[15] Orfila Díaz-Pabón, Agustín: *Eficacia de los sistemas de detección de intrusiones: un análisis orientado a la decisión*, 2005.

Rafael Calzada Pradas
(rafael.calzada@uc3m.es)
Inés Tovar Martínez
(itovar@di.uc3m.es)
Juan Manuel Canelada Oset
(jcanelad@di.uc3m.es)
Javier Chamizo Aguado
chaminet@di.uc3m.es)
David Gutiérrez Rueda
(david@di.uc3m.es)
Área de Seguridad y Comunicaciones
Servicio de Informática
Universidad Carlos III de Madrid
https://asyc.uc3m.es/index.php?Id=1