مجلـه‌ی علمی پژوهش در علوم رایانه ، شماره ۱۷، بهار ۱۳۹۹، ص ۹۸- ۸۵

ISSN: 2745-3673
http://www.Rcsj.ir

مجلـه‌ی علمی پژوهش در علوم رایانه (سال پنجم)
شماره ۱۷ / بهار ۱۳۹۹ / ص ۹۸-۸۵

پیشنهاد یک روش تایید سه گانه برای جلوگیری از سوء استفاده ها از حفره های نرم افزار های کاربردی توسط کاربران غیر مجاز در سطح کاربر

محمد هادیان[۱]، رامین نصیری[۲]، افشین سلاجقه[۳]

[۱] دانشجوی دکتری دانشگاه آزاد تهران جنوب.

[۲] دانشیار دانشگاه آزاد تهران مرکز.

[۳] استادیار دانشگاه آزاد تهران جنوب.

نام نویسنده مسئول:
محمد هادیان

تاریخ دریافت: ۱۳۹۹/۱/۲۳
تاریخ پذیرش: ۱۳۹۹/۳/۱۳

چکیده

امروزه با پیشرفت روز افزون تکنولوژی و استفاده روز افزون جوامع از سیستم های رایانه ای نقصان حفظ تمامیت حریم خصوصی کاربران این فضا احساس میگردد ، نمونه ای از خملات به حفره های نرم افزارهای کاربردی به اکسپلویت مشهور می باشد لذا در این پژوهش رویکرد محققین و پژوهشگران بر این بود سیستمی ارائه گردد تا توانایی مقابله با اینگونه حملات را دارا باشد در این پژوهش محققین توانایی سیستم مذکور را مورد ارزیابی قرار داده و به پاسخ ۹۹,۹٪ تشخیص در نفوذ دست یافتند.

واژگان کلیدی: اکسپلویت- شلکد-تایید سه گانه.

ISSN: 2745-3673
http://www.Rcsj.ir

مقدمه

Exploit به عنوان یکی از مخربترین اما محبوب ترین نوع حملات هکرها محسوب می شود. که دلیل آن میتواند نتیجه ضعف ساختار نرم افزاری و برنامه نویسی ضعیف باشد. با این حمله که ممکن است در انواع مختلف شکل بگیرد، هکر تلاش می کند تا کد مخرب را در هسته سیستم هدف قرار دهد که شامل آسیب پذیری است. در واقع، این نوع حمله به عنوان یک نوع تزریق کد محسوب میگردد. امروزه سیستم های نرم افزاری مختلفی جهت جلوگیری از ضعف های امنیتی نرم افزارهای کاربردی طراحی و ارائه شده اند ، مانند Dep، ASLR، SE H SAFE، و غیره که هر کدام از آنها می توانند بخش هایی از آسیب پذیری سیستم را پوشش دهند. به عنوان مثال، DEP یک سیستم برای محافظت از پشته است.در ادامه چند روش جلوگیری از استفاده حفره های نرم افزاهای کاربردی را مورد بررسی قرار خواهیم داد.

ASLR

یکی از محبوب ترین موانع امنیتی برای هکرها ASLR است. در حقیقت، این مانع امنیتی یک نقش دفاعی بازی نمیکند، اما برای دفاع از انواع سرریز بافر استفاده میشود. این یک مکانیزم اجرایی ساده است، به نحوی که بعد از اجرای برنامه هربار یک آدرس جدید به برنامه داده خواهد شد با این کار هکر توانایی تشخیص آدرس اجرای شلکد را از دست میدهد. بدیهی است، این باعث می شود که Exploit دیگر پاسخ ندهد که این مسئله برای هکرها مطلوب نیست.. ([۲۷]).

DEP

این مانع برای محافظت در مقابل حملات سرریز بافر ارائه شده است که مانع اجرای دستورات در بخش های غیر اجرایی حافظه می شود. این مانع به اصطلاح تخصصی NX BIT شناخته می شود. ([۱۹]).

SEH SAFE

هر استثنا معتبر SEH در جدول SafeSEH DLL موجود است. از سوء استفاده از پرونده های SEH رونویسی شده جلوگیری می کند ([۱۸]).

SEHOP

یک کوکی در انتهای زنجیره SEH قرار می دهد. Dispatcher استثنا پیاده سازی زنجیره و تایید اینکه آیا با کوکی به پایان می رسد اگر یک رکورد SEH رونویسی شود، زنجیره SEH شکسته خواهد شد و با کوکی پایان نخواهد یافت ([۱۸])

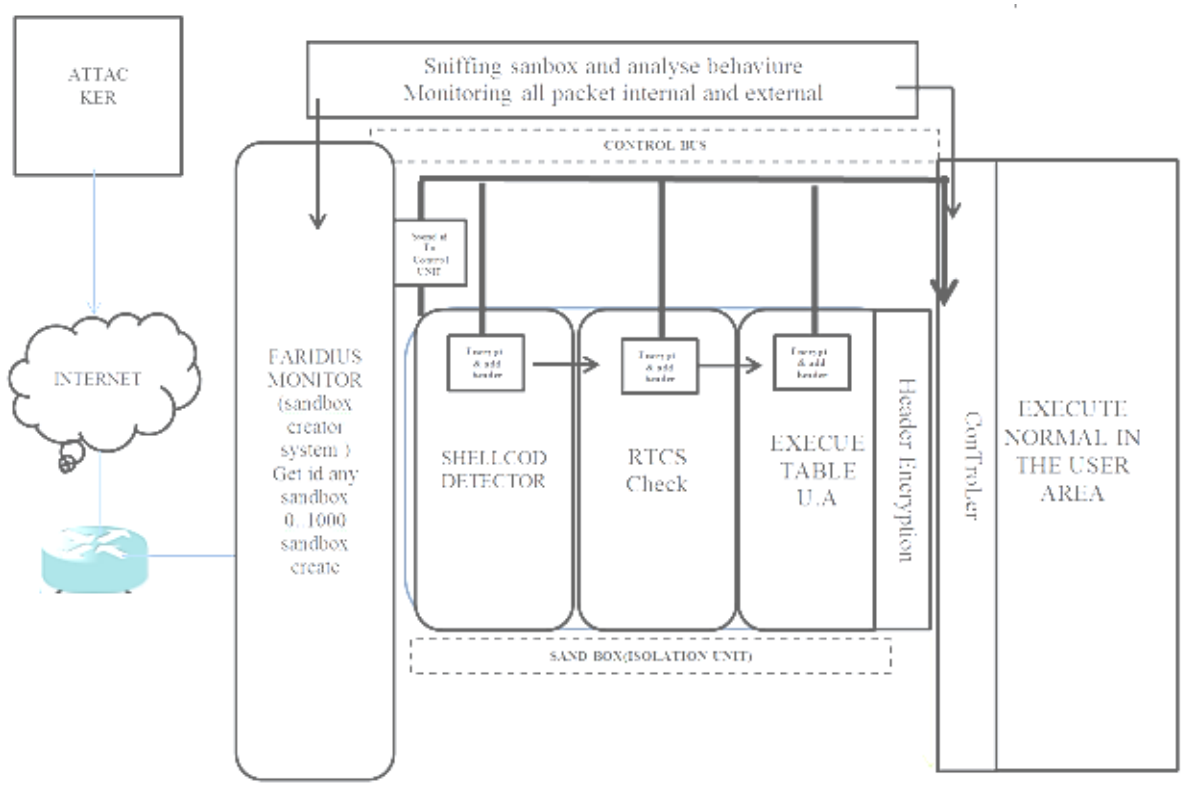
روش دفاع بر اساس آمار

این روش بر اساس کشف سوء استفاده از حملات Z-Day در زمان اجرا استوار است که بر اساس هویت حمله که بر اساس داده های قبلی ساخته شده است. این رویکرد با انتقال Z-Day و بهره برداری از مدل داده های قبلی که با داده های معمول همخوانی ندارد استوار است. هر گونه حملات Z-Day، رویکرد جدیدی را ایجاد خواهد کرد. برای شناسایی کرم های چند شکلی، تمرکز اصلی بر روی امضای مبتنی بر شناسایی است. امضای مبتنی بر شناسایی وابسته به امضای است که توسط امضای مشخص شده ساخته شده است. این امضا ها علیه برخی تغییرات در امضای کلیدی دفاع و یا استفاده از بسته ها توسط مهاجمان برای پنهان کردن احراز هویت شناخته شده اعمال می شود. این روش شناسایی مبتنی بر آمار یا امضا می باشد باعث می شود آسیب پذیری شکسته مورد هجوم قرار نگیرد([۱۶])

ISSN: 2745-3673
http://www.Rcsj.ir

طرح پیشنهادی

در شکل ذیل پکت ها اول وارد بخش SandBoxcreator شده سپس sandbox ۱برای آن فرآیند ایجاد شده سپس آن sandbox کد شنسایی خاص خود را به بخشواحد کنترل ارسال و فضایی را درخواست میکند و سپس در ادامه در داخل sandbox ابتدا در لحظه ورود به بخش اول سیستم یک کد شناسایی خاص به آن داده می شود بصورت هدر و در بخش اول محتوا چک شده اگر شل موجود بود برنامه Fail شده در غیر این صورت به بخش بعدی منتقل می شود یک کد شناسایی خاص به هدر مربوطه اضافه می شود همان کد از طریق خط کنترل به بخش کنترل می رود و در آنجا به وسیله الگوریتم مربوطه کد شده و به هدر قبلی افزوده می شود در مرحله run time error check ۲ چک میشود اگر خروجی آن تهی بود برنامه مربوطه Fail می گردد اگر مشکلی نبود به مرحله سوم میرود در آنجا بصورت بخش اول و دوم یک id به هدر بصورت رمزشده افزوده شده و سپس به جدول مربوطه در بخش موارد مورد ارزیابی رفته و تمامی قسمت ها مورد اعمال قرار می گیرد در انتها کل هدر شامل ۳ بخش می باشد که بصورت رمز شده با هدر موجود در Unit Control مقایسه شده اگر یکسان نبود برنامه Fail شده و اگر موردی نداشت برنامه وارد محیط کاربر می گردد در همه موارد یک سیستم مانیتورینگ برنامه ما را مورد ارزیابی لجظه ای قرار می دهد در هر حمله هکری یک امضای حمله ۲وجود دارد که می توان با شنود روی سیستم آن را مشاهده نمود و حمله را تشخیص داد همان کاری که ids های سخت افزاری انجام می دهند.



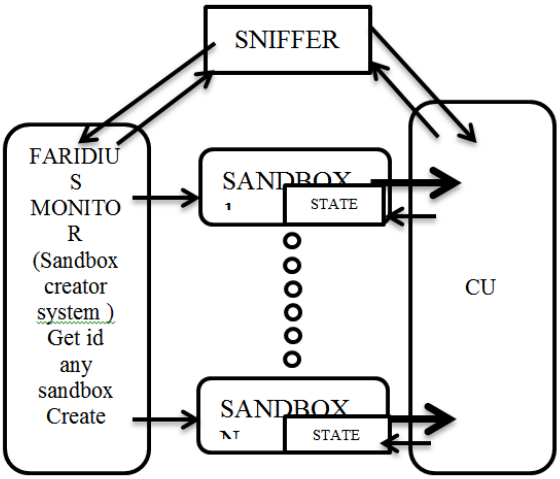
شکل ۱- طرح کلی سیستم پیشنهادی

[1] Isolation unit
[2] Signature attack

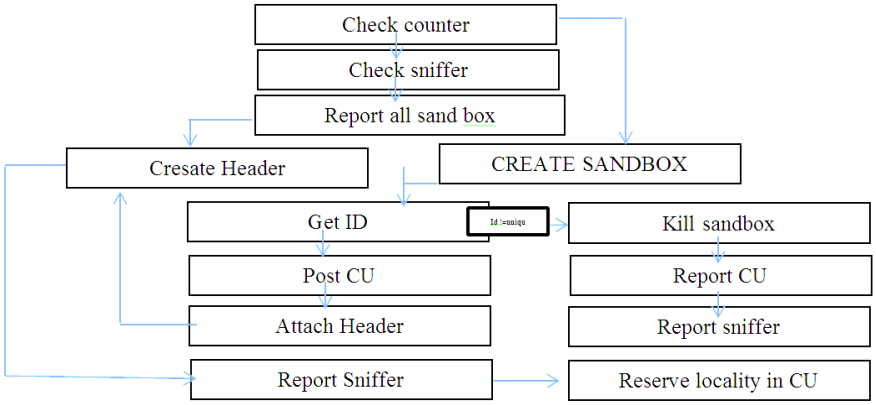
ISSN: 2745-3673
http://www.Rcsj.ir

FV Monitor[3]

این سیستم برای ساخت sandbox کارآیی دارد که می تواند sandbox تولید کند و با واحد کنترل و واحد شنود پایدار در ارتباط کامل می باشد هر sandbox که تولید می گردد شماره منحصر به فرد خود را دارد که در همان ابتدا آن را به واحد کنترل می فرستد و فضایی برای آن برنامه رزرو می کندسپس header سه گانه برنامه به واحد کنترل فرستاده شده سپس آنالیز در آن صورت می پذیرد شماتیک واحد fv monitor به شکل زیر می باشد:



شکل ۲- طرح واحد FV MONITOR



شکل ۳- طرح واحد SANDBOX CREATOR

تشخیص ShellCode[4]

در این مرحله ما با دانش این که هر اکسپلویتی یک امضای بخصوصی دارد و باید به syscall های خاصی دسترسی داشته باشد آنها را شنود و رهگیری می نماییم و بر طبق امضای آنها شناسایی را انجام میدهیم در ذیل چندین امضای ShellCode را بصورت سطحی آورده ایم و طریقه ارزیابی آن را میگوییم در این مرحله یک id خاص به آن داده می شود اگر برنامه امضای شل را یافت آن را Fail میکند در غیر این صورت در انتهای این مرحله آن id را با الگوریتم رمزنگاری SHA2-

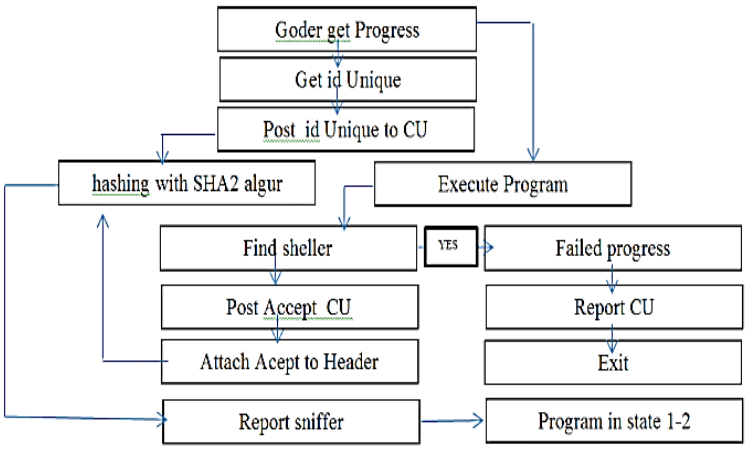
[3] Sand box creator unit
[4] Excecute bad code

ISSN: 2745-3673
http://www.Rcsj.ir

256 بیتی رمز شده و به مرحله بعدی میرود در ابتدای این مرحله وقتی ID به فرآیند حامل داده میشود از طریق CB [5]به بخش CU فرستاده شده و در آنجا طبق همان الگوریتم به رمز در خواهد آمد و مرحله اول پشت سر گذاشته شده و یک ACk به هدر مربوطه داده می شود و طبق آن می گوید مرحله اول Negative بوده است و سپس به مرحله دوم میرویم در ذیل تعدای sh signature آمده است:

```
1:
x31\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69"
"\x6e\x89\xe3\x50\x53\x89\xe1\xb0\x0b\xcd\x80";
```

همانطور که مشاهده می شود باید به دنبال آپکد دستورات فراخوانی Suid [6]و یا زنجیره های s.e.h یا rop بگردیم برای اینکار به یک پایگاه حاویشده اند می بیاشد داشته باشیم.



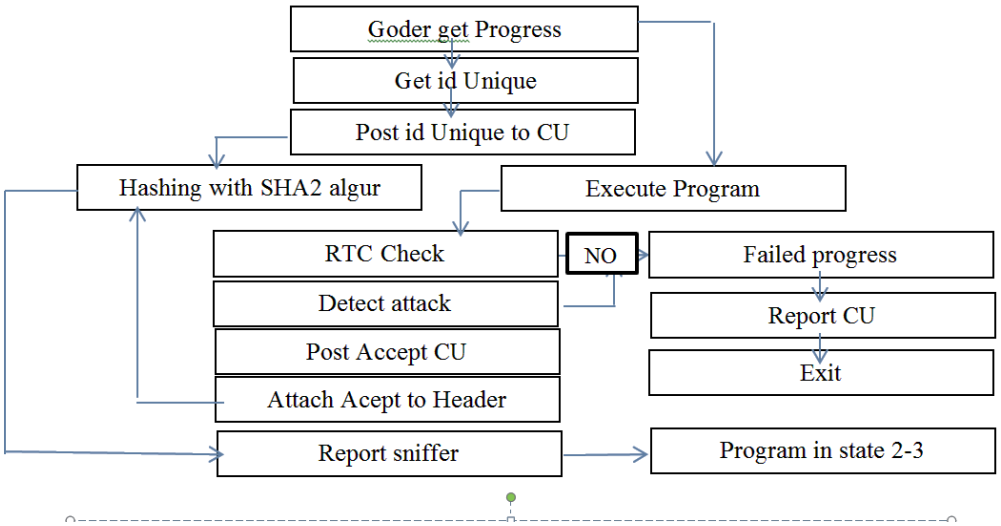
شکل۴- الگوریتم بخش shellcode detector

بخش دوم محیط قرنطینه(RTC)

RTC (بررسی خطای زمان اجرا) هنگامی که RTC مورد بررسی قرار گرفت، پس اگر پاسخ این بود که هیچ خطایی وجود نداشته است، این برنامه با ارسال یک گزارش به واحد کنترل و نظارت به پایان خواهد رسید،اگر پاسخ سازگار باشد، هدر به بخش کنترل فرستاده می شود و سپس همان هدر که قبل از اتصال به هدر بخش قبلی با الگوریتم رمزگذاری شده به بخش کنترل رفت، و سپس یک گزارش به Sniffer داده می شود و به بخش سوم میرود.

[5] Control Bus
[6] Set user ID

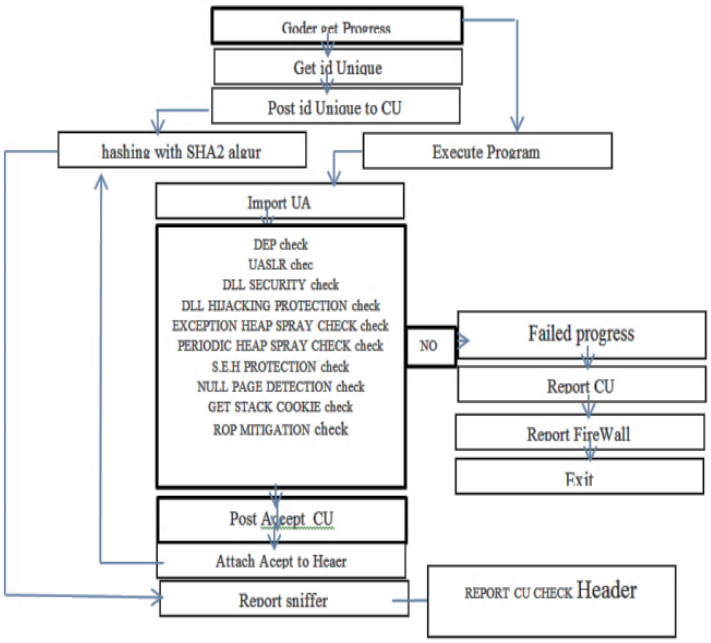
ISSN: 2745-3673
http://www.Rcsj.ir



شکل ۶- الگوریتم بخش دوم محیط قرنطینه(RTC Checker)

بخش سوم sandbox

در این بخش که بخش اصلی این واحد محسوب می شود تمامی مکانیزم های نفوذ بر روی برنامه پیاده سازی میشوند در هر بار ایجاد یک SB[7] این فرآیندها یک بار چک می گردند در صورتی که تاییدیه در یافت نشود اجرا متوقف می گردد.



شکل ۷- بخش سوم sandbox

[7] Sand box

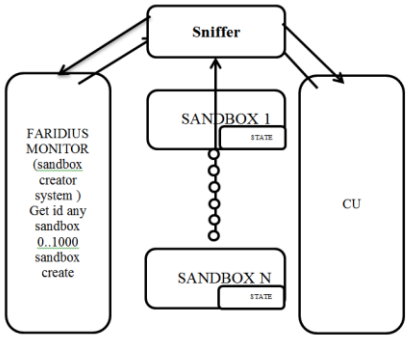
ISSN: 2745-3673
http://www.Rcsj.ir

جدول ۱- مواردی که در بخش سوم محیط قرنطینه باید چک شوند

DEP
UASLR[36]
DLL SECURITY
DLL HIJACKING PROTECTION[29]
EXCEPTION HEAP SPRAY CHECK[30]
PERIODIC HEAP SPRAY CHECK[31]
S.E.H PROTECTION[32]
NULL PAGE DETECTION[33]
GET STACK COOKIE [34]
ROP MITIGATION[35]

واحد شنود پایدار[8]

این واحد دقیقا با واحد های CU[9] و FV Monitor[10] در تماس مستقیم می باشد تمامی SandBox ها را شنود و در حالت های bypass هر یک ازsandbox ها شماره شناسه آنها را به واحد CU گزارش میدهد ، واحد CU آن برنامه را مسدود مینماید دومین کار ومهمترین کار این واحد گزارش خطا در FV Monitor می باشد که باعث جلوگیری از ادامه فرآیند ورود و مسدود کردن تمامی ورودی ها میگردد با این کار اگر احیانا واحد های تابعه دچار خلل گردند آنها را گزارش میدهد شماتیک کلی این بخش به شکل زیر می باشد



شکل ۸- واحد شنود پایدار

همانطور که در شکل بالا مشاهده می نمایید تمامی مراحل و تمامی ابزارها تحت نظارت SNIFFER که برای گزارش دهی به CU تعبیه شده است می باشد بدون استفاده از شنود پایدار برنامه مذکور خطاهای در عین اجرا را نمی تواند مشاهده نماید بطور مثال ممکن است نفوذگر بخواهد خود FV MONITOR را سرریز کند که منطقی می باشد مسلما اگر FVM ۱۰دور زده شود دیگر کا برنامه برنامه تمام است اما با وجود واحد شنود پایدار برنامه گزارشی بر این اساس که FVM از کار افتاده را به CU میدهد CU آن واحد را مسدود و تمامی تقاضا ها را از آن پس منقضی میکنددر مرحله بعد تمامی SANDBOX ها باید به این واحد گزارش دهند در صورتی که گزارشی در CU موجود ولی در SNIFFER نبود منقضی

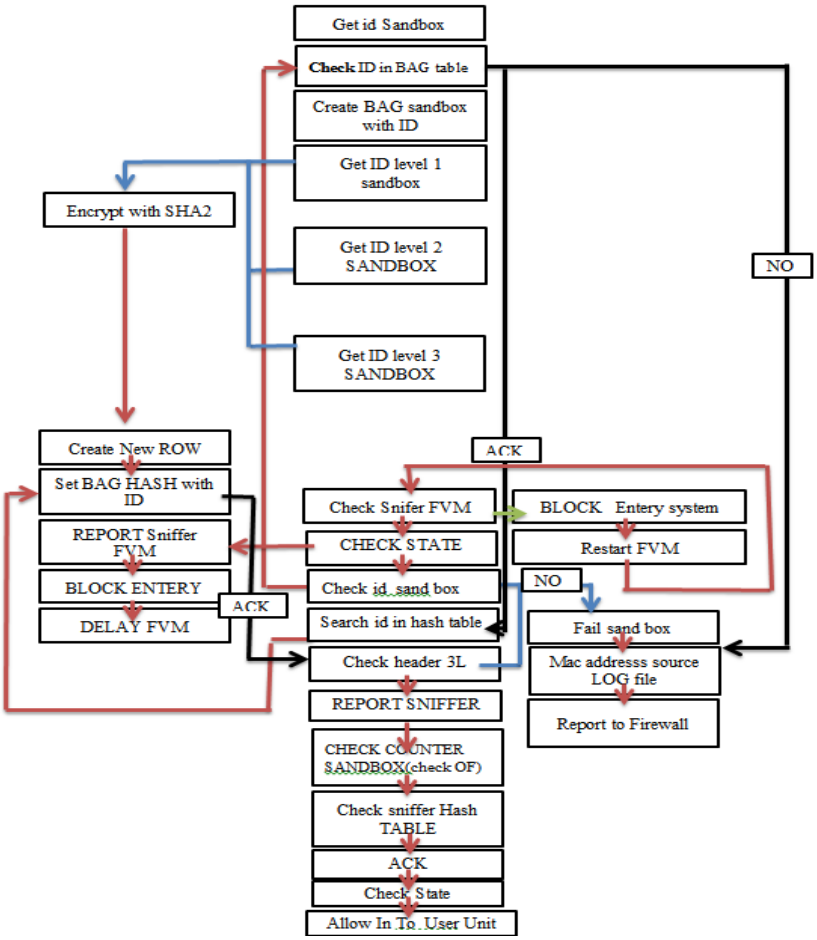
[8] Sniffing unit
[9] Control unit
[10] FV Monitor

ISSN: 2745-3673
http://www.Rcsj.ir

می گردد بطور مثال یکی از SANDBOX ها گزارشش در CU موجود می باشد اما اطلاعی از آن در واحد شنود پایدار نیست آن فرآیند باید مسدود گردد وآدرس فرستند آن توسط LOGER ثبت میگرددو ورودی های دیگر این مبدا مسدود می گردد نظارت پایدار نکته دیگری نیز در خود دارد و آن این است که درخواستهای ورودی به دستگاه و تمامی پورت ها را تحت نظارت خود دارد و در کل مانند یک دیواره آتش عمل می نماید بخش سوم که مهمترین بخش این واحد است نظارت و گزارش به CU می باشد تمامی گزارشاتی که واحد شنود دریافت می نماید به واحد CU داده شده و تصمیمات نهایی در آن واحد گرفته خواهد شد اگر احیانا CU دچار اختلال شد کل ورودی ها و خروجی ها و ارتباطات خارجی و داخلی مسدود و تا راه اندازی مجدد واحد CU منتظر میماند

واحد کنترل CU

بخش کنترلی این نرم افزار با تمامی اجزای سیستم در ارتباط و تعامل می باشد زیرا بایداز تمامی واحدها گزارشات لحظه ای دریافت و آن را آنالیز کرده و تصمیمات متناظر با هر یک را می گیرد و انجام می دهد تنها واحدی که به طور منظم CU راچک می کند واحد شنود پایدار است این واحد توانایی از کار انداختن کل فرآیندها و پکت های ورودی و sand box ها را دارا می باشد برای این که احیانا اگر CU به مشکلی برخورد کرد از واحد شنود پایدار استفاده می کنیم که با از کار افتادن CU به طور کل سیستم را راه اندازی مجدد می نماید درز کل واحد CU قلب این نرم افزار به حساب می آید در این بخش CU را به سه بخش تقسیم می نماییم سپس شماتیک آن را نشان می دهیم ابتدا بحث ارتباط آن با واحد شنود پایدار و FVM را شرح میدهید.



شکل ۹- الگوریتم بخش کنترل

ISSN: 2745-3673
http://www.Rcsj.ir

ارزیابی

به منظور تست و ارزیابی روش مطرح شده در این تحقیق، یک سری داده هایی را به روش مطرح شده تزریق نموده تا بتوان نتایج بدست آمده را سنجش نمود. در این پژوهش برای شبیه سازی راهکارهای ارایه شده از مجموعه داده MCFU۱۱ استفاده شده است این مجموعه داده نسخه های متفاوت با حجم های مختلفی ارایه شده است که سعی شده است تا از آخرین نسخه برای شبیه سازی جهت جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربر استفاده گردد. بدین منظور از نسخه ۱۰ استفاده شده است. این مجموعه داده دارای حجمی معادل5GB می‌باشد که بعلت عدم قابلیتهای سخت افزاری برای پردازش داده مزبور، از ۱۰ درصد مجموعه داده استفاد شد. یکی از مهمترین علتهای انتخاب این داده ها در این پژوهش سازگار بودن با استانداردهای امنیتی از قبیل ISO15446 , ISO2700x و CEM V3.0 می باشد. این مجموعه داده دارای ۱۵ ویژگی با مقادیر اکثرا متنی است که در مرحله پیش پردازش داده‌ها لازم به تغییر به مقادیر عددی دارد. که بدین منظور ابتدا مقادیر کلیه ویژگی‌ها شناسایی و مقادیر عددی نسبت داده شد.در جدول (۴-۱) و (۴-۲) تعدادی ویژگی‌های تغییر یافته ارایه شده است.

مشخصات سیستم مورد استفاده

روش پیشنهادی در این پژوهش با استفاده از شبیه ساز متلب پیاده سازی شده است. همچنین در جدول زیر مشخصات مربوط به سیستمی که پیاده سازی روش پیشنهادی و ارزیابی نتایج در آن انجام شده نشان داده می شود.

جدول ۲- مشخصات سیستم جهت شبیه سازی و ارزیابی نتایج

سخت افزار / نرم افزار	مشخصات
سیستم عامل	ویندوز 7
نوع سیستم عامل	سیستم عامل 32 بیتی و ۶۴ بیتی
حافظه RAM	4 گیگابایت – 3.06 گیگ قابل استفاده
پردازنده	پردازنده اینتل – تعداد هسته ها 7(Core™)i7 CPU Q 720 @ 1.60GHz 1.60 GHz

بنابراین با توجه به یک سیستم با مشخصات فوق شبیه سازی مربوطه انجام شده و نتایج ارزیابی شده است. لذا در همین بخش به صورت کامل نتایج بدست آمده تشریح می گردد.

نتایج شبیه سازی روش پیشنهادی

در این قسمت با توجه به شبیه سازی انجام شده بر روی منبع داده معرفی شده دقت و خطای جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربر نیز محاسبه گردیده است که در نهایت با سایر روشها مورد مقایسه قرار می گیرد. بطور کلی در طی دو مرحله داده ها شبیه سازی شده و نتایجی محاسبه گردیده است:

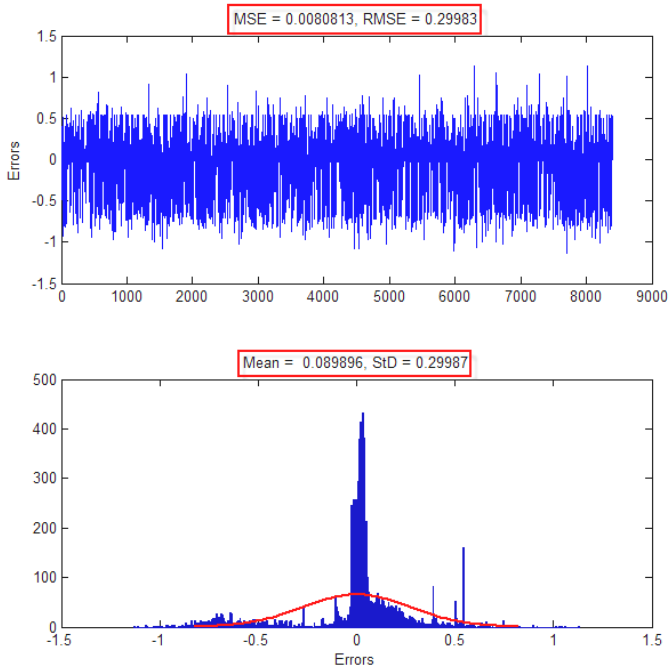
یکی در حالت آموزش

در حالت آزمایش

در شکل (۴-۱) نتایج مربوط به اجرای روش پیشنهادی در مرحله آموزش نشان داده شده است. در شکل (۴-۱) نیز مرحله آموزش روش پیشنهادی و تست بر روی ۸,۰۰۰ داده صورت گرفته است.

[11] [http://social.technet.microsoft.com/wiki/contents/articles/4399.private-cloud-reference-model.aspx.

ISSN: 2745-3673
http://www.Rcsj.ir



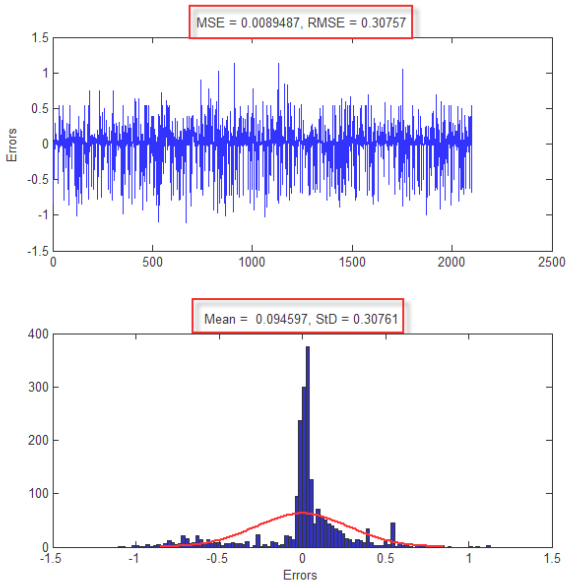
شکل ۱۰- مرحله آموزش روش پیشنهادی و محاسبه نتایج(الف)

شکل (۱۰) دارای دو بخش بوده که بطور کلی میزان خطای آموزش را نشان می دهد. در بخش بالا میزان خطای جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربر برای ۸۰۰۰ نمونه محاسبه می گردد که محور افقی بیانگر تعداد نمونه های آموزش و محور عمودی نیز میزان خطای تشخیص به ازای هر نمونه را نشان می دهد.

همانطور که از شکل (۱۰) دیده می شود میزان خطای روش پیشنهادی در مرحله آموزش بر روی داده های تله نرم افزاری و غیره تله نرم افزاری در حدود ۰,۰۸۹۸ است. دقت تولید مدل مربوطه جهت جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربر در مرحله آموزش تقریبا برابر با ۹۹,۹٪ است که این دقت نسبت به سایر روشها در حدود ۰,۸ تقریبا بهتر می باشد. از طرفی معیار های RMSE ۱۲و MSE ۱۳نیز محاسبه شده اند که از این معیار ها نیز می توان نتایج مختلفی استنباط گردد. انحراف معیار روش پیشنهادی در مرحله آموزش نیز برابر با ۰,۲۹۹ است که نشانگر عملکرد بسیار خوب روش پیشنهادی در مرحله آموزش با استفاده از منطق تصمیم گیرنده بر روی داده های اولیه موجود در شبکه اجتماعی است. در شکل (۴-۲)نیز مراحل آزمایش بر روی مجموعه داده های ۲,۰۰۰ نمونه ای صورت گرفته است.

[12] Root Mean Squared Error
[13] Mean Squared Error

ISSN: 2745-3673
http://www.Rcsj.ir

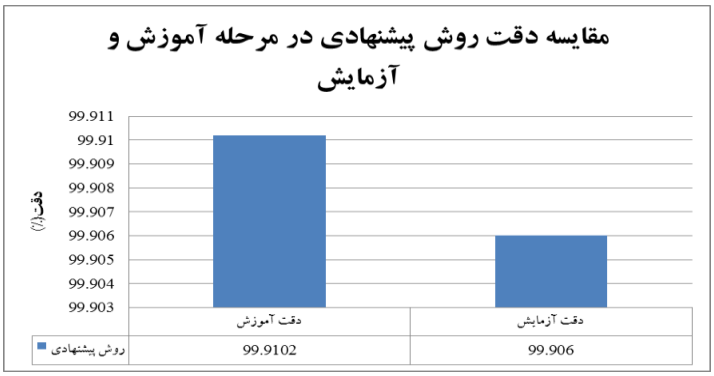


شکل۱۱- مرحله آزمایش روش پیشنهادی و محاسبه نتایج(ب)

شکل (۱۱) دارای دو بخش بوده که بطور کلی میزان خطای آزمایش را نشان می دهد. در بخش بالا میزان خطای جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربربرای ۲۰۰۰ نمونه محاسبه می گردد که محور افقی بیانگر تعداد نمونه های آزمایشی و محور عمودی نیز میزان خطای تشخیص به ازای هر نمونه را نشان می دهد.

همانطور که از شکل (۱۱) دیده می شود میزان خطای روش پیشنهادی در مرحله آموزش بر روی داده های تله نرم افزاری و غیره تله نرم افزاری در حدود ۰,۰۹۴۵ است. دقت تولید مدل مربوطه جهت جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربردر مرحله آزمایش تقریبا برابر با ۹۹,۹٪ است که این دقت نسبت به سایر روشها در حدود ۰,۸ تقریبا بهتر می باشد. از طرفی معیار های RMSE و MSE نیز محاسبه شده اند که از این معیار ها نیز می توان نتایج مختلفی استنباط گردد. انحراف معیار روش پیشنهادی در مرحله آموزش نیز برابر با ۰,۳۰۷ است که نشانگر عملکرد بسیار خوب روش پیشنهادی در مرحله آزمایش با استفاده از منطق تصمیم گیرنده بر روی داده های اولیه موجود در شبکه اجتماعی است. از نکات برجسته اجرای این الگوریتم‌ها تعداد زیاد رکوردهای بخش آموزش و آزمایش بود که منجر به ارایه خروجی دقیق‌تری گردیده است.

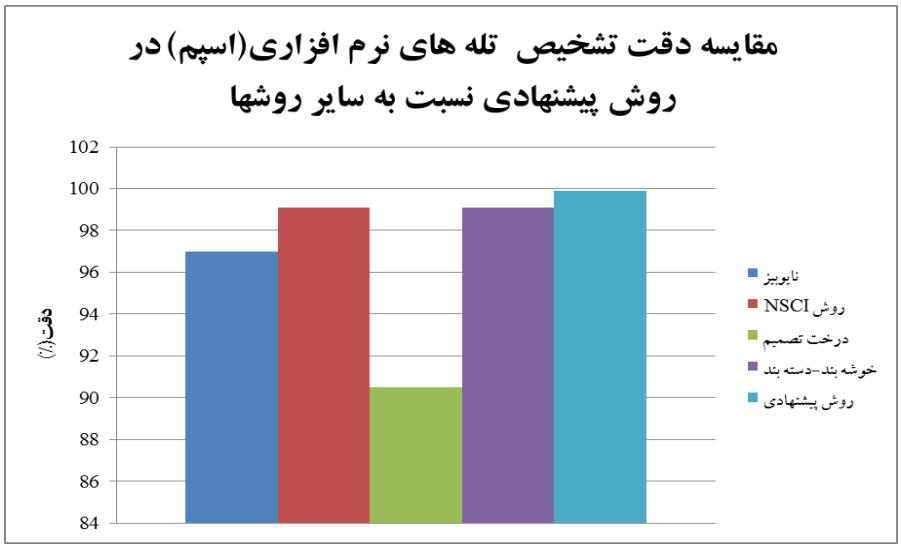
با توجه به شبیه سازی روش پیشنهادی و مشاهده نتایج بدست آمده، قسمت هایی از نتایج روش پیشنهادی از لحاظ میزان خطا و دقت در مرحله تست نسبت به سایر روش ها در قسمت زیر بحث می گردد. در شکل (۱۲) مقایسه دقت مرحله آزمایش و آموزش داده های تله نرم افزاری و غیره تله نرم افزاری در روش پیشنهادی نشان داده شده است.



شکل ۱۲- مقایسه دقت مرحله آزمایش و آموزش داده های تله نرم افزاری و غیره تله نرم افزاری

ISSN: 2745-3673
http://www.Rcsj.ir

همانطور که دیده می شود میزان دقت در مرحله تست مطرح است که در حدود ۰,۰۰۴ با هم تفاوت دارند که این میزان قابل مطرح نیست. بنابراین با این میزان دقت در جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربرمراحل مقایسه با سایر روشها صورت می گرد. با توجه به مقایسه دقت روش پیشنهادی با مقاله پایه[2] در شکل(۱۴) زیر میزان بهبود روش مطرح شده در این پژوهش نشان داده شده است.



شکل ۱۳- :مقایسه میزان دقت جلوگیری از سوء استفاده از حفره های نرم افزارهای کاربردی در سطح کاربر در روش پیشنهادی نسبت به سایر روشها

بنابراین با توجه به شبیه سازی مسئله مطرح شده در این پژوهش با کمک منطق تصمیم گیرنده و روش استعماری مبتنی بر تایید سه مرحله ای مشاهده گردید که دقت روش پیشنهادی جهت تشخیص حفره های نرم افزاری در حدود ۹۹,۹٪ بوده است. همچنین با بررسی مقاله پایه دیده شده است که دقت آنها در حدود ۹۹,۱ در بهترین حالت نیز بوده است. بطور کلی در مقایسه با روشهای بحث شده، میزان بهبود دقت روش پیشنهادی نسبت به روشها خوشه بندی، نایو بیز، درخت تصمیم و روش NSCI۱۴ به ترتیب در حدود ۱,۰۰۸٪، ۱,۰۲۹٪، ۱,۱۰۳، و ۱,۰۰۸٪ بهبود داشته است.

نتیجه‌گیری

در این مقاله پژوهشی بطور تقریبی تمامی حملاتی که منجر به اکسپلویت میشود را شناسایی و تشخیص داده شد سیستم فوق الذکر توانایی ۹۹,۹٪ در شناسایی حملات اکسپلویت های شناخته شده را بدست بیاورد ولی مشکلی در طرح مذکور وجود دارد سربار ارتباطی بین واحد کنترل –واحد ساخت SandBox و واحد شنود پایدار می باشد که سربار بالایی را دارد و از طرفی به دلیل اینکه ممکن است تعدادی از پکت ها دارای زمان انقضاء باشند نیز ممکن است تعدادی پکت را از دست بدهیم به هرحال سیستم مذکور دارای ایراداتی می باشد که محققین در تلاش های آتی برای رفع موارد مذکور می باشند و در آینده باید سربار این سیستم را کاهش دهیم، لازم بذکر است نقطه قوت و شاید نوآوری این طرح در روش تاییدیه سه گانه برای ورودی های سیستم می باشد که باعث می شد در برابر حملات اکسپلویت مقاومت خوبی از خود نشان دهد.

ISSN: 2745-3673
http://www.Rcsj.ir

منابع و مراجع

[1] hairul Anuar Ishak, "Matlab Tutorial of Fundamental Programming", Department of Electrical, Electronic & System Engineering Faculty of Engineering University Kebangsaan Malaysi, 2012.
[2] Greg Hoglund, Gary McGraw , Exploiting Software: How to Break code. 2011,
[3] Jon Erickson, the Art of exploit software. 2008,
[4] McReynolds, Joren Bartley. "Systems and methods to detect and neutralize malware infected electronic communications." U.S. Patent 9,710,645, issued July 18, 2017.
[5] Sunder, Divya Naidu Kolar, David M. Durham, and Hormuzd M. Khosravi. "Providing authenticated anti-virus agents a direct access to scan memory." U.S. Patent 9,087,188, issued July 21, 2015.
[6] Largman, Kenneth, Anthony B. More, and Jeffrey Blair. "Computer system and method of controlling communication port to prevent computer contamination by virus or malicious code." U.S. Patent 7,849,360, issued December 7, 2010.
[7] Shukla, Jayant. "Application Sandbox to Detect, Remove, and Prevent Malware." U.S. Patent Application 11/769,297, filed June 27, 2007..
[8] Pavlyushchik, Mikhail A. "Method and system for antimalware scanning with variable scan settings." U.S. Patent 7,725,941, issued May 25, 2010.
[9] S. Honiden, "Goal Model Elaboration for Software Evolution," Engineering of Complex Computer Systems (ICECCS), 2013 18th International Conference on, Singapore, 2013, pp. 3-3.
[10] Gupta, R. Kaushal, "Improving Spam Detection in Online Social Networks", *Springer*, pp. 1-2, 2015.
[11] Sallam, Ahmed Said. "Systems and methods for identifying hidden processes." U.S. Patent 8,549,648, issued October 1, 2013.
[12] Gupta, Rajarshi, Soorgoli Ashok Halambi, Sudha A. Gathala, and Vinay Sridhara. "Communicating behavior information in a mobile computing device." U.S. Patent 9,690,635, issued June 27, 2017.
[13] Gupta, Rajarshi, Xuetao Wei, Anil Gathala, and Vinay Srishara. "Architecture for Client-Cloud Behavior Analyzer." U.S. Patent Application 13/776,414, filed February 25, 2013.
[14] T. Parker, J. Johnson, M. Tummala, J. McEachen and J. Scrofani, "Dynamic state determination of a software-defined network via dual basis representation," Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on, Gold Coast, QLD, 2014, pp. 1-7.
[15] D. M. Stanley, D. Xu and E. H. Spafford, "Improved kernel security through memory layout randomization," 2013 IEEE 32nd International Performance Computing and Communications Conference (IPCCC), San Diego, CA, 2013, pp. 1-10.
[16] David Hammarberg. “The Best Defenses Against Zero-day Exploits for Various-sized Organizations” .Sans. https://www.sans.org/reading-room/whitepapers/bestprac/defenses-zero-day-exploits-various-sized-organizations-35562 .2014 [ACCESS AVAILABLE[1 sept 2017]
[17] Microsoft develop,2015 https://msdn.microsoft.com/en-us/library/8wtf2dfz.aspx[ACCESS AVAILABLE[22 apr 2017]
[18] Alexander Sotirov , 2009 https://www.usenix.org/legacy/event/sec09/tech/slides/sotirov.pdf" ACCESS AVAILABLE[15 sept 2017]
[19] A detailed description of the Data Execution Prevention (DEP) feature in Windows XP Service Pack 2, Windows XP Tablet PC Edition 2005, and Windows Server 2003. http://support. microsoft. com/kb/875352/EN-US/
[20] Timm, Kevin, Intrusion Detection FAQ: How does and attacker evade IDS with Session Splicing, SANS. http://www. sans. org/securityresources/idfaq/sess- splicing. php.\
[21] R. Roemer, E. Buchanan, H. Shacham, and S. Savage. Return-oriented programming: Systems, languages, and applications. Manuscript, 2009. Online: https://cseweb. ucsd. edu/~hovav/papers/rbss09. Html

ISSN: 2745-3673
http://www.Rcsj.ir

[22] H. Shacham. The geometry of innocent flesh on the bone: Return-intolibc without function calls (on the x86). In S. De Capitani di Vimercati and P. Syverson, editors, Proceedings of CCS 2007, pages 552-61. ACM Press, Oct. 2007.
[23] Welch, X Gao, P Komisarczuk Computer Networks "Detecting Heap-Spray Attacks in Drive-by Downloads: Giving Attackers a Hand" 2013 - ieeexplore.ieee.org
[24] Dr. Anatoliy S. Gordonov . "The Cost of Preventing a Buffer Overflow". conferace paper .2014
[25] Scarfone, Karen and Mell, Peter. Guide to Intrusion Detection and Prevention Systems, Computer Security Resource Center (National Institute of Standards and Technology) (800-94) February 2007.
[26] Serna, Fermin J., Polymorphic shellcodes vs. application IDSs, Next Generation Security Technologies. January 2002.
[27] PaX Team. PaX Address Space Layout Randomization. http://pax. grsecurity. net/docs/aslr. txt
[28] Encapsulation Security Payload, http://social.technet.microsoft.com/wiki/contents/articles/4399.private-cloud-reference-model.aspx.
[29] Dynamic-Link Library Hijacking ,Max "RIVAL" ,https://www.exploit-db.com/docs/english/31687-dynamic-link-library-hijacking.pdf
[30] Ashfaq Ansari, HEAP SPRAYING – ACTIVEX CONTROLS UNDER ATTACK , https://www.exploit-db.com/docs/english/31019-heap-spraying---activex-controls-under-attack.pdf
[31] Paruj Ratanaworabhan, Benjamin Livshits, Benjamin Zorn, NOZZLE: A Defense Against Heap-spraying Code Injection Attacks,2016
[32] SEH overwrite and its exploitability, Shuichiro Suzuki, https://www.ffri.jp/assets/files/research/research_papers/SEH_Overwrite_CanSecWest2010.pdf,2010
[33] Detecting and preventing null pointer errors with pluggable type-checking, CSE 331, University of Washington, https://pdfs.semanticscholar.org/presentation/f2ad/0e46065a31ca2501258a616b692e6373130d.pdf, [access available 2018]
[34] Stack Smashing as of Today A State-of-the-Art Overview on Buffer Overflow Protections on linux_x86_64, Hagen Fritsch, Black Hat Europe,2009
[35] rop&dep, Modern Binary Exploitation , Markus Gaasedelen, http://security.cs.rpi.edu/courses/binexp-spring2015/lectures/11/07_lecture.pdf ,[access available 2015]
[36] Advanced Attacks: How One Exploited Endpoint Leads to Total Datacenter Breach, Nati Davidi, Sebastian Goodwin ,RSA Conference 2015.