امنیت در سیستم‌عامل‌های

یونیکس و لینوکس

مهندس مجید داوری دولت‌آبادی

عضو گروه امنیت GrayHat Hackers

Security Information Assets

انتشارات پندار پارس

سرشناسه : داوری دولت‌آبادی، مجید، ۱۳۵۹ -
عنوان و نام پدیدآور : امنیت در سیستم‌عامل‌های یونیکس و لینوکس / مجید داوری‌دولت‌آبادی.
مشخصات نشر : تهران : پندار پارس، ۱۳۹۳.
مشخصات ظاهری : ۵۵۲ ص.: مصور، جدول.
شابک : ۳۵۰۰۰۰ ریال ۱-۶۵۲۹-۵۶-۶۰۰-۹۷۸
وضعیت فهرست نویسی : فیپا
یادداشت : کتابنامه.
موضوع : سیستم عامل لینوکس
موضوع : سیستم عامل یونیکس
موضوع : سیستم‌های عامل (کامپیوتر)
موضوع : کامپیوترها -- ایمنی اطلاعات
رده بندی کنگره : ۷۶/۷۶Q۸ ۱۳۹۳ ۲د۹۴س/
رده بندی دیویی : ۴۳۲/۰۰۵
شماره کتابشناسی ملی : ۳۴۵۰۲۴۵

انتشارات پندارپارس

دفتر فروش: انقلاب، ابتدای کارگرجنوبی، کوی رشتچی، شماره 14، واحد 16 **www.pendarepars.com**
تلفن: 66572335 - تلفکس: 66926578 همراه: 09122452348 **info@pendarepars.com**

نام کتاب : **امنیت در سیستم عامل‌های یونیکس و لینوکس**
ناشر : **انتشارات پندار پارس**
ترجمه و تالیف : **مهندس مجید داوری دولت آبادی**
چاپ دوم : **اردیبهشت 93**
شمارگان : **500 نسخه**
لیتوگرافی : **ترام‌سنج**
چاپ، صحافی : **فرشیوه، خیام**
قیمت : **35000 تومان** **شابک** : 978-600-6529-56-1

فهرست

سخنی با خوانندگان

این روزها امنیت اطلاعات در هر شاخه‌ای از علم، دارای جایگاه ویژه‌ای می‌باشد که روز به روز بر ارزش آن افزوده می‌شود. هم اینک تقریبا تمامی صنایع و حرفه‌ها به نوعی نیازمند امنیت اطلاعات و ایجاد مکانیزم‌هایی جهت محرمانگی آنها هستند. امنیت در یک سیستم‌عامل نیز جزو موارد اساسی و پایه‌ای در یک شبکه‌ی کامپیوتری است که هر مدیر سیستمی باید به آن توجه ویژه داشته باشد. هم اکنون سیستم‌های‌عامل یونیکس و لینوکس از نظر امنیت نسبت به سیستم‌های‌عامل ویندوز در جایگاه بهتری قرار دارند، اما باز هم دارای نقاط ضعف و آسیب‌پذیری‌های خاص خود می‌باشند که امن‌سازی آنها می‌تواند یک سیستم‌عامل قابل قبول را از دیدگاه امنیتی پدید آورد؛ هرچند همان‌گونه که می‌دانید، امنیت یک موضوع نسبی است و به هیچ‌عنوان ساختاری مطلق به‌خود نخواهد گرفت، اما می‌توان تا حد امکان و قابل قبول و با کمک ساختارها و اصول امنیتی، بستر اصلی آن یعنی سیستم‌عامل را امن ساخت. همان‌گونه که گفته شد، امروزه وجود یک سیستم‌عامل امن و پایدار، از ارکان اصلی و پایه‌ای یک بستر و شبکه‌ی کامپیوتری محسوب می‌شود که هرگونه اختلال در آن حتی جزئی، ممکن است موجب تخریب و سرقت داده‌ها و یا از بین رفتن همه‌ی اطلاعات موجود بر روی آن سیستم گردد. از این‌رو استفاده از یک سیستم‌عامل پایدار، مطمئن و امن می‌تواند در بسترهای مختلف کامپیوتری به کاربران، مدیران و متخصصان فناوری‌اطلاعات کمک به‌سزایی نماید و آنها را در مسیر سرویس‌دهی و نگه‌داری از اطلاعات حساس سازمان‌ها یاری کند.

اینجانب به‌عنوان عضو کوچکی از خانواده‌ی بزرگ سیستم‌های‌عامل کدباز، درصدد گردآوری و تألیف کتابی مرجع به‌منظور آشنایی کامل برخی متخصصان، دانشجویان و مدیران شبکه با اصول امن‌سازی و امنیت سیستم‌های‌عامل مبتنی بر یونیکس و لینوکس بودم تا آنها را با اصول امنیت در سیستم‌های‌عامل کدباز و همچنین اهمیت امنیت در آنها آشنا و آگاه سازم. (گرچه مدیران و متخصصان، حُکم اساتید اینجانب را دارند، امّا به حُکم وظیفه برخود لازم دانستم که این آگاه‌سازی را انجام دهم). پیشاپیش تمام کاستی‌های آن‌را می‌پذیرم و ضمن پوزش از اساتید، متخصصان، دانشجویان و مدیران عزیز، انتقادات و راهنمایی‌های دلسوزانه‌ی آنها را به دیده‌ی منّت پذیرا هستم.

(m_Davari@TOP-co.ir)
(m_Davary@Parshack.zzn.com)

درباره‌ی این کتاب

شیرازه‌ی اصلی این کتاب برگرفته از کتاب‌ها و منابع معتبر درباره‌ی امنیت و امن‌سازی سیستم‌های‌عامل یونیکس و لینوکس است که با تجربیات اینجانب آمیخته شده است، که دخل و تصرفی نیز با آن همراه بوده است. هدف از تألیف این کتاب آشنایی بیشتر کارشناسان و مدیران فناوری‌اطلاعات با آسیب‌پذیری‌های موجود در سیستم‌های‌عامل کدباز (یونیکس و لینوکس) است. همچنین راه‌کارهای مقابله با این نوع آسیب‌پذیری‌ها که موجب بُروز حملات مختلف برعلیه این نوع سیستم‌های‌عامل می‌شوند، در قالب فصل‌های این کتاب بررسی و ارائه شده است. این کتاب اطلاعات مفیدی در اختیار راهبر سیستم‌های یونیکسی و لینوکسی می‌گذارد و وی با کمک این کتاب می‌تواند سیستم‌عامل خود را که معمولا برای مواردی چون سرور و سامانه‌های امنیتی مورد استفاده قرار می‌گیرد، امن نماید تا در مقابل انواع حملات متداول و غیرمتداول ایمن گردد.

پس از سپاس و ستایش به درگاه پروردگار، از همه‌ی دوستان و اساتید عزیزی که مهربانانه دست مرا در انجام اینکار ناچیز فشردند، سپاسگزاری می‌کنم. برخود لازم می‌دانم از زحمات بی‌دریغ استاد محترم سرکار خانم مهندس سیده پونه مرتضویان تشکر و قدردانی نمایم. زحمات خاضعانه‌ی ایشان سهم بزرگی در تهیه و تدوین این کتاب داشته است.

در پایان از جناب آقای مهندس حسین یعسوبی و تمامی همکارانشان که زحمت چاپ کتاب را متقبل شده‌اند، صمیمانه قدردانی می‌نمایم.

یارب چو به وحدتت یقین می‌دارم

ایمان به تو عالم آفرین می‌دارم

دارم لب خشک و دیده‌ی تر بپذیر

کز خشک و تر جهان همین می‌دارم

(مجید داوری دولت آبادی - زمستان 1392)

فصل نخست

مروری بر امنیت در سیستم‌های‌عامل کد باز

سیستم‌عامل، یکی از عناصر چهارگانه در یک سیستم کامپیوتری است که دارای نقشی بسیار مهم و حیاتی در نحوه‌ی مدیریت منابع سخت‌افزاری و نرم‌افزاری می‌باشد. پرداختن به مقوله‌ی امنیت سیستم‌های‌عامل، همواره از بحث‌های مهم در رابطه با ایمن‌سازی اطلاعات در یک سیستم کامپیوتری بوده است که امروزه با گسترش اینترنت، اهمیت آن دوچندان شده است. امروزه در دنیایی متکی بر فناوری اطلاعات زندگی می‌کنیم که هر آن، به خطر افتادن جریان اطلاعات منجر به بُروز خسارت‌های تجاری جبران‌ناپذیری خواهد شد. تمامی شرکت‌ها و متخصصان امنیتی به‌دنبال یک سکوی[1] امن‌تر برای اجرای برنامه‌های‌کاربردی و سرویس‌دهنده‌ها هستند. لینوکس، حرف‌های بسیاری برای گفتن در سمت امنیت دارد. بسیاری از قابلیت‌های امنیتی که در سیستم‌عامل ویندوز وجود ندارند و یا تنها با افزودن نرم‌افزارهای اضافی قابل دسترسی می‌باشند، به‌طور درونی و پیش‌فرض در سیستم‌های‌عامل لینوکس و یونیکس پیاده‌سازی شده‌اند. لینوکس از ابتدا برای محیط‌های شبکه‌ای و چند کاربره طراحی شده است و همین باعث رعایت مسائل امنیتی از ابتدا در آن شده است، درحالی‌که سیستم‌عامل ویندوز اینگونه نبوده و هم اینک نیز از نظر امنیتی دارای کاستی‌های فراوانی است. برای نمونه، یک برنامه‌ی مُخرب با استفاده از همین ضعف‌های امنیتی می‌تواند کل سیستم‌عامل را نابود کند، اما چنانچه مورد مشابهی در لینوکس وجود داشته باشد، حداکثر به دایرکتوری خانگی کاربر اجراکننده آسیب خواهد رسید و کل سیستم‌عامل از خطر در امان می‌ماند.

این‌گونه نیست که لینوکس فاقد هرگونه اشکال امنیتی باشد، بلکه باز بودن کد منبع آن باعث می‌شود تا بسیاری از اشکالات امنیتی پیش از ایجاد خسارت و در مراحل توسعه و برنامه‌نویسی برنامه برملا شده و رفع شوند. اگر اشکالی نیز در برنامه‌های منتشر شده یافت شود، به‌دلیل موجود بودن کد منبع، به‌سرعت برطرف می‌گردد؛ هرچند، در سیستم‌عامل ویندوز، شما باید منتظر شرکت مایکروسافت بمانید. سیستم‌عامل ویندوز دارای اشکال‌های امنیتی بسیاری است که به آسانی هم کشف نمی‌شوند و هنگامی کشف می‌شوند که خسارات جبران‌ناپذیری در اثر حمله از طریق آن ضعف‌های امنیتی رخ دهد که نمونه‌های آن را شاهد هستیم. می‌توان ادعا کرد که شمار بسیار کمی ویروس و برنامه‌ی مُخرب برای لینوکس وجود دارد و این درحالی است که سالیانه بیش از هزار ویروس و برنامه‌ی مُخرب گوناگون برای سیستم‌عامل ویندوز ایجاد می‌شود. این موضوع به‌خاطر گسترده نبودن لینوکس نیست (حدود 70 درصد از سایت‌های وب در جهان بر روی سیستم‌عامل لینوکس و سرویس‌دهنده‌ی وب آپاچی درحال اجرا هستند)، بلکه به‌دلیل وجود حفره‌های امنیتی بی‌شمار ویندوز و سیاست انحصارگرایی شرکت مایکروسافت است. این شرکت طوری رفتار و سیاست‌گذاری کرده است که مشتریان خود را تنها به محصولات خود عادت دهد.

بسیاری از کاربران ویندوز از اینترنت اکسپلورر و ابزار Outlook برای مرور وب و پست‌الکترونیک استفاده می‌کنند. یک ویروس‌نویس، می‌داند که اگر ویروسی را برای کاربران ویندوز بنویسد، بر روی کامپیوترهای 90 درصد آنها اثر خواهد کرد، زیرا بیشترشان از مرورگر IE و ابزار پست‌الکترونیکی Outlook استفاده می‌کنند، اما در سیستم‌عامل لینوکس و یونیکس، شما دامنه‌ی گسترده‌ای از انتخاب‌ها و اجبار نداشتن در آنها دراختیار دارید. برای نمونه، برخی از کاربران لینوکس از مرورگر Mozilla Firefox استفاده می‌کنند، اما برخی مرورگر Konqueror را ترجیح می‌دهند. همچنین درخصوص ابزارهای پست‌الکترونیکی از Kmail، Evolution، Pine، Mutt ویا Mozilla Mail استفاده می‌شود. درنتیجه

[1] Platform

فرد ویروس‌نویس تنها می‌تواند برای یکی از این ابزارها ویروس بنویسد و کد وی بر روی مابقی ابزارها عمل نخواهد کرد و عملا میزان اثر آن اندک خواهد بود. ضمنا هیچ‌یک از ویروس‌هایی که برای سیستم‌عامل ویندوز نوشته شده‌اند، بر روی لینوکس کار نمی‌کنند. البته در مورد کدهای مُخرب دو زیست، موضوع کمی متفاوت است. نخستین و برجسته‌ترین تفاوت در اینجا است که سیستم‌های مبتنی بر یونیکس و لینوکس به معنای کاملا واقعی سیستم‌هایی چندکاربره هستند و برای هر فایل به تنهایی یا یک دایرکتوری می‌توان سطوح دسترسی کاربران و گروه‌های کاربری را تعریف کرد و هر کاربر به‌صورت پیش‌فرض دارای یک محدوده‌ی امن اطلاعاتی از فایل‌های شخصی خانگی است.

موردی که در اینجا درباره‌ی کاربران و گروه‌های کاربری لازم به گفتن است این است که هر کاربر در سیستم یونیکس و لینوکس یک دایرکتوری شخصی به‌نام home خواهد داشت که همه‌ی اختیارات دسترسی فایل در آن برای وی مجاز بوده و می‌تواند در آن به ایجاد و حذف فایل یا عملیات دیگر بپردازد و هیچ‌یک از کاربران یا گروه‌های دیگر به‌جز کاربر ریشه‌ی root درحالت عادی و بدون اجازه‌ی وی نخواهند توانست به اطلاعات او دسترسی یابند. در سیستم‌های مبتنی بر یونیکس و لینوکس، هر کاربری که مالک یا ایجادکننده‌ی یک فایل یا دایرکتوری باشد خواهد توانست با تعیین سطوح دسترسی، فایل یا دایرکتوری مربوط را برای دیگر کاربران یا گروه‌های کاربری آنها، از امکان خواندن و ایجاد تغییر ویا اجرای فایل اجرایی محروم یا بهره‌مند سازد. بدیهی است که مجوزهای تعریف شده برای یک گروه کاربری بر روی همه‌ی اعضای آن گروه اعمال می‌گردد و هر کاربر عضو آن گروه از تمام مزایا یا محدودیت‌های موصوف بهره‌مند خواهند شد.

موضوع تفاوت‌های لینوکس و ویندوز از جمله مواردی است که بسیاری از کاربران را بر سر دو راهی قرار داده است که کدام‌یک از این دو سیستم‌عامل می‌تواند خواسته‌های آنها را برآورده سازد. در این تفاوت‌ها به‌ویژه در حوزه‌ی امنیت سیستم‌عامل، کار را به دقت وافری رهنمون می‌شود، چراکه با توجه به وجود ویروس‌ها و نفوذگران مختلف و حتی وجود اشکال‌های سخت‌افزاری و نوسان‌های برق، نمی‌توان به سادگی از کنار موضوع امنیت سیستم‌عامل گذشت. آزاد و در دسترس بودن کدهای منبع سبب می‌شود تا بتوان از طرز کارکرد دقیق سیستم‌عامل آگاه شد؛ ویژگی که در سیستم‌عامل ویندوز به سادگی در اختیار کاربر قرار ندارد. امنیت برای کاربران عادی نیز در این سیستم‌عامل بسیار کارآمد خواهد بود. برای نمونه، وقتی چندین نفر از یک کامپیوتر استفاده می‌کنند، هر کدام می‌توانند تنها به اطلاعات خود دسترسی داشته باشند و نمی‌توانند برای اطلاعات دیگر کاربران تهدیدی به‌شمار آیند. خبری از کرم‌های اینترنتی، اسب‌های‌تروا و دیگر خرابکارها نیست.

1-1 مکانیزم‌های مختلف در بهبود امنیت لینوکس

بررسی‌های گوناگونی در مورد توزیع‌های مختلف لینوکسی انجام شده است و معمولا از این سو و آن‌سو شنیده می‌شود که فلان توزیع از نظر امنیتی بهتر از توزیع دیگر است. از این‌رو بهتر است امنیت سیستم‌عامل لینوکس را از جنبه‌های مختلف بررسی کنیم. شاید نخستین و اصلی‌ترین بحث امنیت در سیستم، بحث تأمین امنیت فضای کاربر است. حملاتی که به سیستم‌های‌عامل انجام می‌شوند، به‌طور کلی حافظه و مشکلات مدیریت آن‌را هدف می‌گیرند و در واقع، تخریب حافظه یکی از راه‌های معمول برای نفوذ به سیستم‌های کامپیوتری مدرن است. ماجرا از این قرار است که یک سرریزی بافر می‌تواند کار را به حملات پیچیده‌ی تخریب حافظه بکشاند و نظام یکپارچه‌ی پروسه‌ها و حافظه‌ی سیستم را به‌هم بزند. امنیت سیستم‌ها هر چند که در طول سالیان مختلف و از هر نگارش به نگارش بعدی بهبود یافته است، اما تکنیک‌های هجوم به این بخش از سیستم در طول زمان پیچیده‌تر شده و در توزیع‌های مختلف لینوکس برای آزمون امنیت به‌کار می‌رود. بسیاری از کارشناسان معتقدند برخی از مکانیزم‌های امنیت در مقایسه با

دیگر روش‌ها، مقاوم‌تر هستند. افزون بر این، شماری از این روش‌ها تأثیر زیادی روی بازدهی سیستم می‌گذارند و حتی می‌توانند هنگام کامپایل نرم‌افزاری خاص، از نظر تطبیق با سیستم مشکل‌ساز شوند.

می‌توان اشاره کرد که این دو فاکتور، یعنی تأثیر در بازدهی سیستم و امکان از دست رفتن کنترل روی نگه‌داری سیستم، بزرگ‌ترین دغدغه‌هایی است که تمام مکانیزم‌ها و روش‌های مقاومت در برابر حفره‌های امنیتی در توزیع‌های لینوکس با آن مواجهند. در هر نرم‌افزار و توسعه‌ی نرم‌افزاری آن می‌توان بدون هیچ تعجبی دید که این مسئله حتی می‌تواند بحث‌های شدیدتری بین چندین پروژه به‌وجود بیاورد. مکانیزم‌های امنیتی که برای امن‌سازی باید مد نظر قرار گیرد، موارد زیر را دربر می‌گیرد:

- نگاشت حافظه‌ی غیراجرایی (استفاده از بیت سخت‌افزاری NX)
- کد یا برنامه‌ی مستقل از محل (PIC/PIE)
- پیاده‌سازی قابلیت FORTIFY_SOURCE از glibc
- محافظت از نابودی پشته (SSP)
- RELRO یا محکم کردن برنامه‌های ELF با مرتب‌سازی بخش‌هایی از فایل و تبدیل آنها به حالت فقط‌خواندنی

همچنین ابزارهای زیر برای مقایسه استفاده شده است:

- ابزار checksec.sh، با اصلاح کوچکی که وضعیت FORTIFY_SOURCE را نشان دهد.
- اسکریپت پایتونی که قابلیت‌های checksec.sh را افزایش می‌دهد.

هسته‌ی لینوکس، بیت NX را از زمان توزیع 2/6/8 پشتیبانی کرده است. از این‌رو برای سخت‌افزارهای 32 و 64 بیتی که این بیت را در مدار خود دارند، فعال است. برای بهره‌بردن از نگاشت حافظه‌ی غیراجرایی در سخت‌افزارهایی که از این قابلیت استفاده نمی‌کنند، لازم است که از وصله‌ی grsecurity یا بسته‌ی Exec Shield استفاده کرد. همچنین باید اشاره شود که بدون محافظت اضافه، مهاجم به‌سادگی می‌تواند بیت NX را دور بزند و دسترسی‌ها را به‌صورت دستی تنظیم کند. همچنین باید اشاره شود که به‌جز بیت NX، تمام مکانیزم‌های امنیتی که در این بخش به آن اشاره شده است در بخش فضای کاربری حافظه پیاده‌سازی شده‌اند و در سطح هسته نیستند. همچنین مهم است بدانیم حفره‌ای که در هسته‌ی لینوکس وجود دارد، ورای از مکانیزم‌های امنیتی که برای پردازش‌های فضای کاربر فعال شده است، می‌تواند مورد تهاجم قرار گیرد. از این‌رو می‌توان هسته‌های سیستم‌عامل امروزی را با پشتیبانی نابودی پشته کامپایل کرد، هر چند بهبود امنیت هسته، اغلب به نصب وصله‌های زیادی از جمله وصله‌ی grsecurity می‌انجامد. برخی از توزیع‌ها برای خود وصله‌های خاص امنیتی منتشر می‌کنند.

- RELRO: حفاظت ساختمان داده‌ی داخلی یک فایل ELF بر دوش RELRO است تا نگذارد مهاجم، اطلاعات آن‌را مشاهده کند یا جریان اجرای آن‌را کنترل نماید. این امنیت را با اصلاح جدول اتصال روندها (PLT) یا جدول عمومی آفست‌ها (GOT) که بخشی از فایل ELF است، تضمین می‌کنند. زمانی‌که از RELRO استفاده می‌شود، کل بخش GOT به حالت فقط‌خواندنی تغییر پیدا می‌کند و از این‌رو وقتی پردازش اجرا می‌شود، نمی‌تواند توسط هیچ مهاجمی قابل تغییر باشد. اگر RELRO به‌صورت جزئی پیاده‌سازی شود، تنها PLT و GOT به‌عنوان فقط‌خواندنی علامت زده می‌شوند. هر دوی این گزینه‌ها البته ساختمان داده‌ی ELF را از نو مرتب می‌کنند تا مهاجم نتواند این بخش‌ها را با بخش‌های دیگر ELF روی هم بیاندازد. همچنین گفتنی است که RELRO کاملا نیازمند آن است که تمامی سمبل‌ها هنگام اجرای برنامه شناسایی شود تا بتوان کل GOT را به‌صورت فقط خواندنی در آورد. از این‌رو

برنامه‌های بزرگ‌تر با کندی بیشتری هنگام بازشدن روبه‌رو می‌شوند و تعداد زیادی از سمبل‌ها از کتابخانه‌های اشتراک‌یافته باید بارگذاری شود.

- محافظ خرابی پشته: محافظ خرابی پشته، سوءاستفاده از سرریزی بافر را با پیاده‌سازی بررسی‌های امنیتی بیشتر در پردازش پشته دشوارتر می‌کند. افزونه‌ی SSP در کامپایلر GCC از زمان نگارش 4/1 به این برنامه اضافه شد.

1-2 بهره‌گیری از خطاها برای امن‌سازی لینوکس

در این بخش به بررسی خطاهایی می‌پردازیم که درصورت رفع آنها امنیت بیشتری در سیستم‌عامل لینوکس مورد نظر ایجاد می‌گردد. این خطاها شامل موارد زیر می‌باشند:

انتخاب گذرواژه‌ی ساده ویا گذرواژه‌های پیش‌فرض: با توجه به سریع شدن پردازنده‌ها و امکان دسترسی به نرم‌افزارهایی که گذرواژه‌ها را کشف می‌کنند، حتی با انتخاب گذرواژه‌های پیچیده نیز، می‌توان آنها را شکست. با استفاده از ابزارهایی که در سیستم‌عامل یونیکس و لینوکس پیش‌بینی شده است مسئول سیستم می‌تواند اجازه‌ی تولید گذرواژه‌ها و دیگر مسائل مرتبط را کنترل نماید. در برخی از سیستم‌های‌عامل یونیکس، فایلی با نام passwd در شاخه‌ی /etc/default وجود دارد که راهبر یونیکس می‌تواند با ایجاد تغییراتی در آن به کاربر اجازه ندهد که گذرواژه‌های ساده را انتخاب نماید، اما در لینوکس به اندازه‌ی کافی کنترل بر روی گذرواژه‌ها انجام می‌شود و می‌توان تا حدی مطمئن بود که کاربر نمی‌تواند گذرواژه‌های ساده را انتخاب کند. فراموش نگردد که مسئول سایت (راهبر سیستم) این اختیار را دارد که گذرواژه‌های ساده‌ای را برای کاربران تهیه نماید، که این‌کار خطای مسلم راهبر می‌باشد. چراکه هر گذرواژه‌ی ساده، دروازه‌ای برای ورود افراد مهاجم بوده و فرد مهاجم پس از وارد شدن به سیستم می‌تواند با استفاده از نقاط ضعف احتمالی دیگر و به‌وجود آوردن سرریز بافر[1] کنترل سیستم را در دست بگیرد. در بسیاری از سیستم‌های فعلی یونیکس و لینوکس، مجموعه امکانات Pluggable Authentication Modules نصب می‌باشد و توصیه‌ی اکید می‌گردد که کد زیر برای بالا بردن امنیت سیستم، تحت /etc/pam.d و در فایل passwd قرار گیرد:

```
passwd password requisite usr/lib/security/pamcraklib.so retry=3
passwd password required /usr/lib/security/pam_pwdb.so use_authtok
```

در زمان اجرای برنامه‌ی passwd، کتابخانه‌های پویا با نام‌های pamcraklib.so و pam_pwdb.so به برنامه متصل شده و کنترل‌های لازم را انجام می‌دهند. مجموعه نرم‌افزارهای craklib این امکان را به سیستم اضافه می‌کند تا کنترل نماید که آیا گذرواژه‌ی تهیه شده توسط کاربر شکستنی است یا خیر. فراموش نشود که فرمان passwd تابع راهبر سیستم می‌باشد و راهبر سیستم می‌تواند گذرواژه‌ی ساده را انتخاب نماید و این عمل گناهی نابخشودنی را برای مسئول سیستم ثبت خواهد کرد. در مورد گذرواژه‌های پیش‌فرض که درهنگام نصب برخی سوئیچ‌ها و مسیریاب‌ها وجود دارد، راهبر سیستم باید به سرعت (زمان نصب گذرواژه‌های از پیش تعیین شده را تعویض نماید.

- باز گذاشتن پورت‌های شبکه: باز گذاشتن پورت‌هایی که محافظت نشده ویا بدون استفاده می‌باشند، به مهاجمان اجازه می‌دهد به‌نحوی وارد سیستم شوند و امنیت سیستم را مخدوش نمایند. فرمان‌های زیادی مانند finger و rwho و غیره وجود دارند که افراد مهاجم می‌توانند با اجرای آنها در شبکه و قرار دادن آدرس کامپیوتر مقصد، نام‌های کاربران و تعداد زیادی از قلم‌های اطلاعاتی مربوط به کاربران را به‌دست آورند و با حدس زدن گذرواژه

[1] Buffer Overflow

وارد سیستم شوند. به‌وسیله‌ی ابزارهایی که در سیستم‌عامل یونیکس و لینوکس وجود دارد می‌توان پورت‌های باز را پیدا کرد و تمهیدات لازم را انجام داد. یکی از این ابزارها، NMap نام دارد که با اجرای این فرمان و درج سوئیچ‌ها و گزینه‌های لازم و وارد کردن آدرس IP، پورت‌های کامپیوتر مورد نظر را یافت و فعالیت‌های اخلال‌گونه را انجام داد. راهبر سیستم با اجرای فرمان netstat -atuv می‌تواند سرویس‌هایی که در حال اجرا هستند را مشخص کند و با انواع روش‌هایی که وجود دارد، سرویس را غیرفعال نماید و شاید یک روش مناسب برای پاک کردن برنامه‌های سرویس‌دهنده ویا تغییر مجوز آن (با فرمان chmod) باشد. هرچند، می‌توان با فرمان chkconfig اجرای برخی از سرویس‌ها را در زمان راه‌اندازی سیستم متوقف کرد. برای نمونه، با کمک فرمان chkconfig و به‌فرم زیر می‌توان سرویس portmap را بر روی سیستم مورد نظر غیرفعال نمود:

```
# chkconfig -del portmap
```

- استفاده از نرم‌افزارهای قدیمی: توصیه می‌شود که از نرم‌افزارهایی که نسخه‌های جدید آن به‌دلیل وجود اشکالات امنیتی در نسخه‌های قدیمی روانه‌ی بازار شده است، استفاده شود و گناهی بس نابخشودنی است که راهبر سیستم با استفاده از نرم‌افزارهای قدیمی، راه را برای سوءاستفاده کننده‌گان باز بگذارد. برای نمونه، فرمان ls دارای مشکلی است که با قرار دادن آرگومانی خاص می‌توان سرریز بافر را به‌وجود آورد و کنترل سیستم را به‌دست گرفت. چندی پیش مجموعه نرم‌افزارهای مربوط به نمایش نام فایل‌ها و شاخه‌ها (ls، lx، lr و غیره) در سایت‌های مهم و استاندارد قرار داده شد تا استفاده‌کننده‌گان لینوکس آن‌را بر روی سیستم خود نصب نمایند.

- استفاده از برنامه‌های نا امن ویا پیکربندی شده به‌صورت نادرست: به‌دلیل مسائل خاص، برخی از سیستم‌ها نیاز به مجوزهای ویژه دارند و اِعمال مجوزها می‌تواند مسائل غیرقابل پیش‌بینی را به وجود آورد و ضمنا با پیکربندی نامناسب نرم‌افزار، راه برای سوءاستفاده‌کنندگان باز خواهد شد. برای نمونه، نرم‌افزارهایی وجود دارند که برای اجرا شدن، مجوز Set UserID را لازم داشته (مجوز S) و این مجوز درحالتی‌که صاحب فایل اجرایی root باشد، بسیار خطرناک است. فرمانی که این اجازه را دارد با اجرای فراخوان‌های سیستم[1] مانند setid تغییر مالکیت داده و قدرت root را کسب می‌کند و راهبر سیستم باید تاوان این گناه نابخشودنی را نیز پس بدهد. برای نمونه، استفاده از FTP و telnet که اطلاعات را عینا بر روی شبکه منتقل می‌کنند، می‌تواند نگرانی‌هایی را برای مسئول سایت به وجود آورد و شاید راه‌اندازی sshd بتواند کمی از گناهان مسئول سیستم را بکاهد و در مورد پیکربندی نادرست فایل‌ها بتوان نامی از فایل rhosts. برد که مجوز نادرست می‌تواند باعث لو رفتن گذرواژه گردد. بهتر است با فرمان find نام فایل‌هایی که مجوز s را دارند، کنترل کنید تا برنامه‌ی اجرایی با مجوز s در سیستم اضافه نگردد. همچنین، مسئول سیستم در اجرای دستور mount نیز باید دقت فراوان داشته باشد تا برنامه‌هایی که مجوز s بر روی CD/DVD و فلاپی دارند، اجرا نشود.

- ناکافی بودن منابع و یا نامناسب بودن ارجحیت‌ها: کم کردن هزینه‌های مربوط به امنیت و عدم آموزش‌های لازم و تهیه نکردن نرم‌افزارهای بازدارنده می‌تواند شماری مسائل غیرقابل پیش‌بینی به وجود آورد. به‌ویژه جابه‌جایی اولویت‌های هزینه کردن اعتبارات می‌تواند امنیت سیستم را خدشه‌دار نماید. لازم به یادآوری است که این مطلب فنی نیست و مدیریتی می‌باشد، اما راهبر سیستم باید مرتبا نکات لازم را در این زمینه به مقامات مسئول گوشزد نماید تا مدیریت ارشد سازمان بیش از بیش به اهمیتِ امنیت پی ببرد و هزینه‌های لازم را تأمین نماید. عدم

[1]_System call

اطلاع‌رسانی مسئول سایت در این زمینه به مدیریت‌های مافوق که احتمالا در این زمینه تخصصی ندارند نیز اشتباهی بسیار بزرگ به‌شمار می‌رود.

- نگه‌داری UserIDهای قدیمی و غیر لازم و تهیه‌ی شناسه‌های عمومی: نگه‌داری UserIDهای قدیمی و شناسه‌هایی مانند TEST می‌تواند معضلات زیادی را به وجود آورد و امکان سوءاستفاده را بالا ببرد. تهیه‌ی شناسه‌های عمومی نیز به‌دلیل نامشخص بودن هویت اصلی کاربر می‌تواند مشکل‌زا باشد. مسئول سایت باید رویه‌ای را برای کشف UserIDهای غیرفعال اتخاذ نماید و با هر روشی که صلاح می‌داند پس از تهیه‌ی فایل پشتیبان لازم، UserIDهای غیرفعال را در مقاطع معینی متوقف نماید و شاید یکی از بهترین روش‌ها برای انجام این‌کار تعویض گذرواژه باشد. برای نمونه، با دستور زیر می‌توان UserID با نام someone را غیرفعال کرد:

```
# chmod 000 /home/someone
```

تولید UserIDهای عمومی مانند test و guest و غیره که مورد علاقه‌ی بسیاری از مهاجمان است، یکی دیگر از بزرگ‌ترین اشتباه‌های غیرقابل بخشش راهبر سیستم می‌باشد.

- به تعویق انداختن فعالیت‌های مهم در زمینه‌ی ایجاد امنیت: با کم اهمیت دادن مسائل حفاظتی همچون نصب نکردن ترمیم‌ها یا وصله‌ها[1] و تهیه نکردن فایل‌های پشتیبان، می‌توان گفت که مسئول سیستم تیر خلاص را به کامپیوترهای خود شلیک کرده است و خطایی بسیار بزرگ را مرتکب شده است.

1-3 توصیه‌های مهم امنیتی در مورد یونیکس و لینوکس

بررسی و تجزیه و تحلیل امنیت در سیستم‌های‌عامل باید با ظرافت و در چارچوبی کاملا علمی و با در نظر گرفتن تمامی واقعیت‌های موجود، انجام شود تا امکان نگه‌داری و پشتیبانی سیستم‌ها با در نظر گرفتن مجموعه تهدیدات موجود و آتی، به‌سرعت و به‌سادگی میسر گردد. بیشتر حمله‌های موفقیت‌آمیز در اینترنت، به‌دلیل وجود نقاط آسیب‌پذیر در شماری اندک از سرویس‌های سیستم‌های‌عامل متداول است. مهاجمان، با فرصت‌طلبی خاص خود از روش‌های فراوانی به‌منظور سوءاستفاده از نقاط ضعف امنیتی شناخته شده، استفاده می‌کنند و در این راستا ابزارهای متنوع، کارآمد و گسترده‌ای را به‌منظور نیل به اهداف خود، به خدمت می‌گیرند. مهاجمان، در این رهگذر متمرکز بر سازمان‌ها و مؤسسه‌هایی می‌گردند که هنوز مسائل موجود امنیتی (حفره‌ها و نقاط آسیب‌پذیر) خود را برطرف نکرده‌اند و بدون هیچ‌گونه تبعیضی آنها را به‌عنوان هدف، انتخاب می‌کنند. پس با شناسایی و تجزیه و تحلیل این‌گونه نقاط آسیب‌پذیر توسط کارشناسان امنیت اطلاعات، سازمان‌ها و مؤسسات قادر به استفاده از مستندات علمی تدوین شده به‌منظور برخورد منطقی با مشکلات موجود و ایجاد یک دیوار حفاظتی مناسب می‌باشند.

لینوکس و یونیکس، از سیستم‌های‌عامل رایج در جهان هستند که امروزه در سطح بسیار گسترده‌ای استفاده می‌گردند. تاکنون حمله‌های بی‌شماری توسط مهاجمان متوجه سیستم‌هایی بوده است که از یونیکس و لینوکس (نسخه‌های متفاوت) به‌عنوان سیستم‌عامل استفاده می‌کنند. با توجه به حمله‌های متنوع و گسترده‌ی انجام شده، باید سبک مقابله با این حمله‌ها و تهدیدها شناخته شود و راه‌های نفوذ به‌سرعت ترمیم گردد و کاملا مسدود شود. عوامل فراوانی در بُروز این‌گونه حمله‌ها نقش دارد که از آن جمله می‌توان به آگاه نبودن لازم مدیران سیستم درباره‌ی ارتقاء امنیتی سیستم‌هایی که بر روی آنها نرم‌افزارهای مدیریت اطلاعات شبکه نصب یا به‌صورت غیرضروری اجراء می‌گردد و

[1] Patchs

پیکربندی نامناسب برنامه‌ها اشاره کرد. این نمونه‌ها می‌تواند زمینه‌ی یک تهاجم از نوع DoS ویا یک سرریزی بافر را فراهم سازد. به‌منظور حفاظت سیستم و ترمیم سریع‌تر نقاط آسیب‌پذیر، موارد زیر پیشنهاد می‌گردد:

- همیشه آخرین نسخه‌ی نرم‌افزارهای ارائه شده را دریافت و آن‌را بر روی سیستم نصب نمائید. به‌منظور به‌هنگام‌سازی سیستم باید از تمامی وصله‌های ارائه شده توسط تولیدکنندگان استفاده شود و درصورت امکان آن نرم‌افزار را به آخرین نسخه‌ی موجود ارتقاء دهید.
- دیوارآتش موجود بر روی سیستم را دقیقا نصب و با توجه به نیازهای امنیتی خود در سطح مناسب پیکربندی کنید.
- پورت‌های غیرضروری یا در معرض تهدید را در سطح مسیریاب ویا دیوارآتش، با توجه به توصیه‌های امنیتی موجود در بولتن‌های امنیتی مسدود کنید.
- آن دسته از نرم‌افزارهای غیرضروری که به‌صورت پیش‌فرض هنگام نصب سیستم‌های‌عامل نصب می‌گردند و هیچ‌گونه کاربرد عملی ندارند را غیرفعال نمایید.
- به‌منظور پیچیده‌تر کردن حمله‌های خودکار ویا جلوگیری از پویش غیرمجاز سیستم توسط مهاجمان، به توصیه‌های امنیتی تهیه‌کنندگان نرم‌افزارها در سایت‌های مربوط یا دیگر سایت‌های امنیتی لینوکس و یونیکس دقیقا عمل کنید و هیچ نکته‌ای را از قلم نیاندازید.
- اکیدا استفاده از حساب‌های‌کاربری با گذرواژه‌ی ضعیف ویا فاقد گذرواژه را مسدود کنید، زیرا گذرواژه دارای نقشی حیاتی و اساسی در ایجاد نخستین سطح دفاع در یک سیستم اطلاعاتی است و از دست‌رفتن آن ویا ضعف آن می‌تواند سیستم را در معرض تهدیدات جدی قرار دهد. مهاجمان پس از دستیابی به گذرواژه‌ی کاربران تأیید شده (استفاده از مکانیزم‌های متفاوت) قادر به دستیابی به منابع سیستم و حتی تغییر در تنظیمات دیگر حساب‌های‌کاربری تعریف شده و موجود بر روی سیستم خواهند بود، عملیاتی که می‌تواند پیامدهای بسیار منفی را به‌دنبال داشته باشد. چنانچه از حساب‌های‌کاربری استفاده می‌شود که بین کاربران متعدد ویا کارکنان موقت یک سازمان به اشتراک گذاشته شده است ویا کاربران از رمزهای‌عبور به‌درستی حفاظت ننمایند، پتانسیل نفوذ به شبکه توسط یک مهاجم فراهم می‌گردد.
- از یک برنامه‌ی پویش‌گر به‌روز شده که قادر به بررسی دقیق سیستم‌های کامپیوتری به‌منظور تشخیص نقاط آسیب‌پذیر باشد، استفاده کنید. با عضویت در گروه‌های خبری چون Symantec برای آگاهی از آخرین هشدارهای امنیتی اطلاعات، خود را به‌روز نگاه دارید.
- از امکانات رمزنگاری چون OpenSSH در سرویس‌های شبکه مانند Telnet، SMTP، POP3، IMAP، rlogin، HTTP و غیره استفاده کنید.

1-4 مفاهیم پایه‌ای در امنیت سیستم‌های‌عامل کدباز (لینوکس و یونیکس)

در زمینه‌ی امنیت لینوکس و مقایسه‌ی آن با سیستم‌های فراگیر ویندوز، حرف و حدیث‌های بسیاری مطرح است. براساس آمارهای شرکت‌های امنیتی، بدافزارهای بسیاری برای سیستم‌های‌عامل سری ویندوز ساخته شده و گسترش یافته‌اند، درحالی‌که سیستم‌عامل لینوکس تنها هدف تعداد بسیار محدودی از بدافزارها قرار گرفته است. با این اوصاف،

بسیاری معتقدند، بر مبنای اصل امنیت از طریق ابهام[1]، سیستم‌های منبع بسته امنیت بیشتری دارند و به‌دلیل این‌که کد آنها در اختیار همگان نیست، مشکلات امنیتی آنها نیز کمتر مشهود بوده و امنیت بیشتری دارند. در اینجا باید گفت، این ادعا کاملا اشتباه است و ادعایی کاملا نادرست می‌باشد. با اینکه کد سیستم‌عامل لینوکس و برنامه‌های آن باز است و در اختیار همگان قرار دارد، اما فضای اجرایی نرم‌افزار در سیستم‌عامل با استفاده از حقوق دسترسی محافظت‌شده است و دسترسی به بسیاری از امور برای برنامه‌ها فراهم نیست. همچنین، به‌دلیل باز بودن کد برنامه‌ها به‌محض نمایان‌شدن حفره‌های امنیتی در سیستم، هرکسی حتی خود کاربر می‌تواند به برطرف کردن این عیوب اقدام کند و این امر سرعت پاسخ به مشکلات امنیتی را بسیار افزایش می‌دهد. حال این قابلیت و امکانات را با ویندوز و سیستم‌های زیرمجموعه‌ی منبع بسته‌ای مقایسه کنید که در برخی از موارد، ارائه‌ی اصلاحیه‌ی امنیتی برای سیستم تا یک ماه پس از آشکار شدن مشکل طول می‌کشد. در ادامه‌ی این بخش به بررسی برخی از مفاهیم پایه‌ای امنیت در سیستم‌های‌عامل یونیکس و لینوکس می‌پردازیم:

1-4-1 مفهوم TCPwrapper

TCPwrapper یا tcpd به‌صورت واسطه‌ای بین دایمون inetd و سرویسی که inetd فراخوانی می‌کند، اجرا می‌شود. می‌توان TCPwrapperها را برای فیلتر کردن و نیز ثبت وقایع اتصال‌ها به‌ازای هر سرویس تنظیم کرد. همان‌گونه که می‌دانید برای محافظت سرورها می‌توان از دیوارهای‌آتش استفاده کرد. البته در برخی موارد نیاز به محافظت از سیستم توسط TCPwrapper نیز خواهیم داشت. TCPwrapper نیز همانند دیگر سرویس‌های لینوکسی دارای فایل‌های ویژه‌ی پیکربندی است. دو فایل که با این سرویس استفاده می‌شود عبارت‌اند از:

```
/etc/hosts.allow , /etc/hosts.deny
```

زمانی‌که اتصالی به یک پورت خاص برقرار می‌شود، دایمون inetd برای بررسی مجاز بودن درخواست طبق قواعد موجود در فایل‌های مذکور، آن‌را برای دایمون tcpd ارسال می‌کند. درصورت مجاز بودن، tcpd درخواست را برای سرویس مورد نظر ارسال می‌کند، اما درصورتی‌که اتصال غیرمجاز باشد، درخواست مربوط متوقف و مسدود می‌شود. اگر از سیستم لینوکسی خود تنها به‌صورت یک سیستم خانگی استفاده می‌کنید و نیازی به دسترسی از راه دور به آن ندارید، پیشنهاد می‌شود، همه‌ی اتصال‌ها از راه دور را با تنظیمات زیر مسدود کنید. برای مسدودسازی همه‌ی اتصال‌ها از تمام میزبانان راه دور، سطر زیر را به فایل etc/hosts.deny/ بیافزایید:

```
ALL: ALL
```

همچنین برای پذیرفتن همه‌ی اتصال‌ها از میزبانان محلی، سطر زیر را به فایل etc/hosts.allow/ بیافزایید:

```
ALL: LOCAL
```

اکنون فرض کنید قصد داریم اجازه‌ی دسترسی از نوع SSH را برای کلاینت‌های cli1، cli2 و cli3 صادر نماییم. برای انجام این‌کار باید سطر زیر را به فایل etc/hosts.allow/ بیافزاییم:

```
sshd: cli1  cli2  cli3
```

اینک فرض کنید قصد داریم سیستم‌های موجود در یک محدوده از آدرس IP، مجوز دسترسی از نوع SSH را داشته باشند. در این حالت به‌فرم زیر عمل می‌کنیم:

```
sshd: cli1  cli2  cli3  .subnet.example.com
```

---

[1] Security via Obscurity

چنانچه قصد دارید تمام سیستم‌های یک محدوده از آدرس IP مجوز اتصال داشته باشند، اما تنها یک سیستم از این محدوده این اجازه را نداشته باشد، به‌صورت زیر عمل کنید:

```
ALL: .subnet.example.com  EXCEPT  exam.subnet.example.com
```

براساس دستور بالا همه‌ی سیستم‌ها به‌جز سیستم exam، مجوز دسترسی خواهند داشت. دقت داشته باشید که فایل hosts.allow بر hosts.deny حق تقدم دارد. ضمنا این دو فایل قابل خواندن برای همه‌ی کاربران می‌باشند که البته می‌توانید مجوز دسترسی خواندن را از آنها بردارید. با وجود اینکه امکان استفاده از TCPwrapper در کنار xinetd وجود دارد، اما درحالت کلی اهمیت ندارد. دایمون xinetd خصیصه‌های کنترل دسترسی و ثبت وقایع TCPwrapper را با یکدیگر ترکیب می‌کند. نسخه‌ای از TCPwrapper که همراه توزیع Red Hat ارائه می‌شود، گزینه‌های بسیار پیچیده‌تری نیز دارد که قابلیت‌های فوق‌العاده‌ای به آن داده است.

1-4-2 مفهوم دایمون xinetd

xinetd[1]، جایگزین نسل جدید برای دایمون inetd است. این سرویس افزون بر توانایی کنترل دسترسی همانند TCPwrapper، انعطاف‌پذیری بیشتری در تنظیمات و ثبت وقایع دارد. xinetd که همراه بیشتر توزیع‌های لینوکس ارائه می‌شود خود شامل یک TCPwrapper است که در داخل دایمون xinetd گنجانده شده است و به سادگی می‌توان از آن برای فعال یا غیرفعال کردن سرویس‌های شبکه‌ای استفاده کرد. افزون بر فایل اصلی تنظیمات که در مسیر /etc/xinetd.conf جای دارد، به‌ازای هر سرویس، فایل‌های دیگری برای تنظیمات وجود دارد که در شاخه‌ی /etc/xinetd.d/ جای دارند. همان‌گونه که گفته شد، سرویس xinetd جایگزینی برای inetd می‌باشد که همه‌ی سرویس‌های شبکه‌ای را که روی xinetd تنظیم شده‌اند، کنترل و نظارت می‌کند. هم‌اینک برخی از سیستم‌های‌عامل مبتنی بر یونیکس مانند FreeBSD، برای پیکربندی سرویس‌های شبکه‌ای خود از دایمون inetd استفاده می‌کنند. با کمک دستور زیر می‌توانید از فعال بودن دایمون xinetd بر روی سیستم، اطمینان حاصل نمایید:

```
# chkconfig --list xinetd
```

درصورتی‌که قصد دارید شماره فرایند آن‌را نیز مشاهده کنید، از دستور زیر استفاده نمایید:

```
# /etc/init.d/xinetd  status
```

اگر xinetd بر روی سیستم فعال باشد، باید بدانید که چه سرویس‌هایی با این دایمون درحال فعالیت هستند. گفتنی است که به دایمون xinetd، در اصطلاح Super Server نیز گفته می‌شود. با کمک دستور زیر می‌توان فهمید که چه سرویس‌هایی تحت دایمون xinetd پیکربندی شده‌اند و همچنین می‌توان فهمید که پورت‌های کدام‌یک از آنها توسط دایمون xinetd کنترل و نظارت می‌شود:

```
# chkconfig  --list  |  awk  '/xinetd based services/,/""/'
```

اینک اگر تنها قصد دیدن سرویس‌هایی را دارید که پورت‌های آنها با دایمون xinetd کنترل و نظارت می‌شود، از دستور زیر استفاده نمایید:

```
# chkconfig  --list  |  awk  '/xinetd based services/,/""/'  |  grep -v off
```

فرض کنید به‌دلیل وجود آسیب‌پذیری‌های امنیتی موجود در سرویس Telnet، قصد دارید این سرویس را بر روی یک سرور غیرفعال نمایید. گفتنی است که این سرویس نیز در قالب دایمون xinetd می‌باشد. برای انجام این‌کار ابتدا باید

[1] Extended Internet Services Daemon

بررسی کنید که آیا این سرویس بر روی سرور مربوط فعال است یا خیر. برای این منظور می‌توانید از یکی از دستورات زیر استفاده کنید:

```
# chkconfig  --list  telnet
# cat  /etc/xinetd.d/telnet | grep disable
```

با کمک دستور دوم یک سطر از فایل پیکربندی مخصوص سرویس Telnet را مشاهده خواهید کرد که نشان می‌دهد این سرویس تحت xinetd فعال است. در ادامه با کمک دستور زیر سرویس Telnet را غیرفعال می‌کنیم:

```
# chkconfig  telnet  off
```

برای اطمینان بیشتر بهتر است سرور Telnet را به‌طور کامل از روی سرورها و سیستم‌های حساس حذف کنید و به‌جای آن از سرویس SSH استفاده نمایید:

```
# rpm  -e  telnet-server
```

اینک فرض کنید ممکن است یک سرویس تحت xinetd داشته باشید که اطلاعاتی در مورد آن در اختیار ندارید و نمی‌دانید مورد استفاده‌ی آن چیست. برای اینکه بتوانید در مورد یک سرویس خاص اطلاعات کاملی گردآوری کنید و لزوم وجود یا عدم وجود آن‌را مشخص سازید، به‌فرم زیر عمل کنید. فرض کنید با کمک دستور زیر به این نتیجه رسیدید که سرویسی با نام authd تحت دایمون xinetd کار می‌کند، اما نمی‌دانید چه سرویسی است و چه کاری در سیستم انجام می‌دهد:

```
# chkconfig --list xinetd
```

در این حالت ابتدا با کمک دستور زیر، مسیر دقیق دایمون سرویس مذکور را پیدا می‌کنیم:

```
# grep "server"  /etc/xinetd.d/auth
```

خروجی دستور بالا به‌فرم زیر می‌باشد:

```
server = /usr/sbin/in.authd
server_args = -t60 --xerror --os -E
```

در ادامه باید با کمک دستور زیر یک دستور پرس‌و جو از مدیر بسته‌های نرم‌افزاری سیستم دریافت کنیم که ببینیم فایل in.authd در کدام بسته‌ی نرم‌افزاری جای دارد:

```
# rpm  -qf  /usr/sbin/in.authd
```

فرض کنید خروجی دستور بالا به‌فرم زیر باشد:

```
authd-<Version>.rhel3
```

اکنون باید با کمک دستور پرس‌و جوی دیگری، توضیحات مربوط به این بسته‌ی نرم‌افزاری را به‌دست آوریم:

```
# rpm  -qi  authd-<Version>.rhel3 | awk  '/Description/,/""/'
```

چنانچه قصد دارید همه‌ی فایل‌های موجود در این بسته‌ی نرم‌افزاری را نیز مشاهده کنید، دستور زیر را وارد نمایید:

```
# rpm  -ql  authd-<Version>.rhel3
```

اینک با توجه به توضیحات و اطلاعات به‌دست آمده درباره‌ی دایمون مذکور و با در نظر گرفتن سیاست‌گذاری سازمان مربوط می‌توانید در مورد فعال یا غیرفعال کردن سرویس مذکور تصمیم‌گیری کنید. همان‌گونه که می‌دانید برای غیرفعال کردن این سرویس باید به‌فرم زیر عمل کنید:

```
# chkconfig  auth  off
```

3-4-1 مفهوم دایمون identd

این سرویس برای احرازهویت به‌کار می‌رود و صاحب هر اتصال TCP/IP به یک سرور از راه دور، پذیرنده‌ی آن‌را شناسایی می‌کند. زمانی‌که کاربری از راه دور به یک میزبان متصل می‌شود، دایمون inetd بر روی سیستم میزبان راه دور، درخواستی روی پورت شماره 113 با عنوان "چه کسی دارنده‌ی این اتصال خاص به من است؟" برمی‌گرداند، inetd محلی نیز، پاسخ می‌دهد، مثلا "نام‌کاربری Grayhat دارنده‌ی اتصال است". این سرویس نباید به‌عنوان روشی برای احرازهویت استفاده شود، زیرا هر کسی می‌تواند با داشتن دسترسی ریشه، پاسخ دایمون inetd را تغییر دهد. در واقع بر روی بیشتر سیستم‌های‌عامل مانند FreeBSD و ویندوز حتی یک کاربر معمولی نیز می‌تواند پاسخ دایمون identd را مطابق با چیزی که می‌خواهد، مشخص کند. این پروتکل روی سیستم‌های چندکاربره به‌عنوان روشی برای پیگیری کاربران مشکوک سودمند است. اگر یکی از کاربران شما مشکلی روی سیستم دیگری به‌وجود آورد، مدیر سیستم می‌تواند شما را از نام‌کاربری، کاربر مشکل‌زا با خبر کند. آیا لازم است شما سرویس inetd را اجرا کنید؟ بر روی سیستم‌هایی با کاربران زیاد، این‌کار فایده‌ی بسیاری دارد، اما هدف خاصی را روی یک سیستم تک‌کاربره دنبال نمی‌کند. در واقع اجرای آن به‌منزله‌ی باز گذاشتن سرویس برای دنیای خارج با تمام خطرات امنیتی است. نکته‌ی دیگری که باید در نظر گرفت این است که دایمون idnetd می‌تواند به نفوذگران اجازه دهد تا اطلاعات با ارزشی از سیستم شما به‌دست آورند. این اطلاعات می‌تواند شامل سرویس‌های خاصی باشند که با دسترسی ریشه در حال اجرا هستند. همچنین نوع سیستم‌عامل از طریق این دایمون برای نفوذگران قابل تشخیص می‌باشد. یکی از سوئیچ‌های این دایمون، گزینه‌ی n- آن می‌باشد که به‌جای نام‌کاربری، شماره‌کاربری را برای مقصد ارسال می‌کند. همین کار می‌تواند اطلاعات مفیدی در اختیار نفوذگر قرار دهد. برای آگاهی از گزینه‌های مختلف در اجرای این سرویس به فایل پیکربندی آن در مسیر زیر مراجعه نمایید:

```
/etc/identd.conf
```

درحالت کلی تأثیرات غیرفعال کردن دایمون identd شامل موارد زیر می‌باشند:

- میزبانان راه دور ممکن است برای پیگیری حمله‌هایی که از سایت شما نسبت به کاربران خاصی صورت می‌گیرد، دچار مشکل شوند. این مسئله زمانی اهمیت پیدا می‌کند که شما کاربران زیادی روی میزبان خود داشته باشید. غیرفعال کردن این سرویس روی سیستم چندکاربره، نه تنها یک عمل نادرست مدیریتی است، بلکه پیگیری کاربران مشکوک را در سایت شما مشکل خواهد کرد.
- برخی از سرویس‌دهنده‌های FTP و بیشتر سرویس‌دهنده‌های IRC برای احرازهویت کاربران از راه دور به این سرویس نیاز دارند. غیرفعال کردن این سرویس تعداد سرویس‌دهنده‌های IRC و FTP که اتصال‌ها شما را می‌پذیرند، محدود می‌کند، اما چنانچه از IRC استفاده نمی‌کنید، مشکلی ایجاد نمی‌کند.

می‌توانید دسترسی به سرویس identd را با علامت‌گذاری از نوع "#" (Comment) سطر auth در فایل /etc/inetd.conf یا با استفاده از TCPwrapperها ویا نرم‌افزار دیوارآتش، محدود یا غیرفعال کنید. اگر برای اتصال به یک سرور خاص، نیاز دارید سرویس identd فعال باشد، این نکته را در نظر بگیرید که تنها به دسترسی‌هایی که از طرف آن سرور صورت می‌گیرد، مجوز دهید. اگر پورت مربوط به identd را توسط دیوارآتش کنترل می‌کنید، حتما تلاش کنید از سیاست Reject به‌جای deny استفاده کنید. استفاده از deny ممکن است زمان اتصال شما به سرورهایی که از identd استفاده می‌کنند را به شدت افزایش دهد، زیرا پیش از برقراری اتصال، منتظر پاسخ می‌مانند.

1-5 مقاوم‌سازی سرورهای لینوکسی

امن‌سازی سرورهای لینوکس برای محافظت از داده‌های کاربران و مشتریان امری حیاتی و عقلانی است و تأمین امنیت سرورهای لینوکسی، یکی از وظایف مدیران سیستم است. در این بخش راه‌کارهایی برای افزایش امنیت یک سرور لینوکسی تازه نصب شده را عنوان می‌کنیم. این مثال را با یک سرور مبتنی بر Red Hat انجام می‌دهیم، اما برای نصب یا حذف بسته‌ها می‌توانید از ابزار مدیریت بسته‌ی توزیع خود استفاده کنید.

- **رمزگذاری ارتباطات داده**: تمام داده‌های ارسال شده از طریق شبکه قابل شنود است. برای جلوگیری از شنود داده‌ها، داده‌های ارسال شده را تا حد ممکن با گذرواژه یا استفاده از کلیدها و گواهی‌نامه‌های امنیتی رمزگذاری کنید. برای انجام این‌کار موارد زیر پیشنهاد می‌گردد:
  - از سرویس‌های scp، SSH، rsync ویا SFTP برای ارسال داده‌ها استفاده کنید. همچنین می‌توانید سیستم‌فایل سرور راه دور یا پوشه‌خانگی کامپیوتر سرور را با استفاده از ابزار ویژه‌ی sshfs یا ابزار fuse به سیستم بیافزایید و سپس به انتقال داده بپردازید.
  - GnuPG برای شما امکان رمزگذاری و امضا کردن داده‌ها و ارتباطات را فراهم می‌کند. همچنین یک سیستم مدیریت کلید همه‌کاره و ماژول‌های دسترسی به انواع کلید عمومی را به‌همراه دارد.
  - Fugu یک رابط‌گرافیکی برای برنامه‌ی انتقال امن فایل Sftp است. SFTP همانند ftp است، اما برخلاف آن، همه‌ی نشست را رمزگذاری می‌کند و این به آن معنا است که هیچ رمزعبوری به شکل متنی ارسال نخواهد شد. گزینه‌ی دیگر FileZilla است که یک کلاینت مستقل از سکو می‌باشد و از پروتکل FTP، FTP از طریق SSL/TLS یا FTPS و پروتکل انتقال فایل ssh یا همان SFTP، پشتیبانی می‌کند.
  - OpenVPN یک VPN سبک و کم هزینه با پشتیبانی SSL است.
  - پیکربندی و نصب Lighthttpd SSL یا همان HTTPS را در نظر داشته باشید.
  - پیکربندی و نصب Apache SSL یا HTTPS از طریق ماژول mod_ssl آپاچی را نیز فراموش نکنید.
  - در این باره از به‌کار بردن سرویس‌ها و پروتکل‌هایی چون FTP، Telnet و Rlogin/Rsh پرهیز کنید: در بیشتر پیکربندی‌های پیش‌فرض شبکه‌ها، نام‌های‌کاربری، گذرواژه‌ها، دستورهای FTP/Telnet/Rsh و فایل‌های انتقال یافته به سادگی ممکن است توسط کاربری در همان شبکه به کمک برنامه‌های ویژه‌ی استراق‌سمع بسته[1] دریافت شوند. راه‌حل عمومی، استفاده از سرویس‌ها و پروتکل‌های OpenSSH، SFTP و FTPS است. همچنین دستور زیر را اجرا کنید تا NIS، RSH و دیگر سرویس‌های منسوخ را پاک کنید:

```
# yum erase inetd xinetd ypserv tftp-server telnet-server rsh-serve
```

- **کمینه‌سازی برنامه‌ها برای به کاستن از نفوذپذیری**: آیا واقعا به همه‌ی سرویس‌های وبی که روی سیستم نصب شده است، نیاز خواهید داشت؟ برای کاهش نفوذپذیری سیستم، از نصب نرم‌افزارهای اضافی خودداری کنید. از ابزار مدیریت بسته مانند yum، apt-get یا dpkg برای بررسی بسته‌های نصب شده روی سیستم به‌فرم زیر استفاده کنید:

[1] Packet Sniffer

```
# yum list installed
# yum list packageName
# yum remove packageName
```

و یا در نمونه‌های مبتنی بر توزیع Debian:

```
# dpkg --list
# dpkg --info packageName
# apt-get remove packageName
```

- **یک سرویس شبکه‌ای روی هر سیستم یا هر ماشین مجازی**: سرویس‌های شبکه‌ای مختلف را روی سرویس‌دهنده‌های مختلف یا ماشین‌های مجازی مختلف اجرا کنید. این‌کار تعداد سرویس‌هایی را که ممکن است آسیب ببینند، محدود می‌کند. در این‌حالت برای نمونه، اگر یک خرابکار وارد شبکه شود و بتواند در سرویس‌دهنده‌ی وب آپاچی نفوذ کند، قادر نخواهد بود تا در دیگر سرویس‌های شبکه مانند MySQL یا سرویس‌دهنده‌ی پست‌الکترونیکی نفوذ کند.
- **هسته و برنامه‌های سیستم را به‌روز نگه دارید**: اِعمال وصله‌های امنیتی، مهم‌ترین بخش نگه‌داری یک سیستم لینوکسی است. لینوکس همه‌ی ابزارهای لازم برای به‌روزنگه‌داشتن سیستم را فراهم می‌کند و اجازه‌ی ارتقای آسان بین نسخه‌های مختلف را می‌دهد. همه‌ی به‌روزرسانی‌های امنیتی باید بررسی و در اسرع وقت اِعمال شوند و در ادامه از ابزار مدیریت بسته مانند yum یا apt برای به‌روزنگه‌داشتن سیستم استفاده گردد:

```
# yum update
```

یا

```
# apt-get update && apt-get upgrade
```

می‌توانید توزیع Red Hat، CentOS ویا Fedora خود را طوری تنظیم کنید تا خبر انتشار یک به‌روزرسانی برای هریک از بسته‌های yum را از طریق پست‌الکترونیکی به آگاهی شما رساند. گزینه‌ی دیگر، اِعمال همه‌ی به‌روزرسانی‌ها از طریق وظایف ابزار cron است. برای سیستم‌های Debian و Ubuntu از ابزار apticron برای آگاه شدن از وجود به‌روزرسانی‌ها استفاده کنید.

- **از افزونه‌های امنیتی لینوکس استفاده کنید**: لینوکس از وصله‌های امنیتی مختلفی برای محافظت در قبال تنظیم‌های نادرست یا برنامه‌های مشکل‌ساز استفاده می‌کند. اگر ممکن است، از SELinux یا دیگر ابزارهای امنیتی اضافی لینوکس برای تشدید محدودیت‌های برنامه‌های شبکه و دیگر برنامه‌ها استفاده کنید. برای نمونه، SELinux از سیاست‌های امنیتی گوناگونی برای هسته‌ی لینوکس استفاده می‌کند. استفاده از SELinux که یک کنترل دسترسی اجباری (MAC)[1] را فراهم می‌کند، پیشنهاد می‌شود. در حالت استاندارد کنترل دسترسی احتیاطی (DAC)[2]، برنامه یا پردازه‌ای که به‌عنوان یک کاربر (UID یا SUID) اجرا می‌شود، تمام مجوزهای کاربر را برای دسترسی به اشیائی مانند فایل‌ها، سوکت‌ها و دیگر پردازه‌ها دارد. اجرای کنترل دسترسی اجباری، سیستم را از بدافزارهایی که ممکن است آن‌را تخریب یا نابود کنند، جلوگیری می‌کند. در فصل دوم کتاب به بررسی و تشریح کامل پیکربندی SELinux خواهیم پرداخت.

---

[1] Mandatory Access Control
[2] Discretionary Accesc Control

- **سیاست حساب‌های‌کاربری و رمزهای عبور قوی**: از دستور useradd و usermod برای افزودن یک کاربر یا نگه‌داری یک حساب‌کاربری استفاده کنید. مطمئن شوید که از یک سیاست رمزعبور مناسب و قدرتمند استفاده می‌کنید. برای نمونه، یک گذرواژه‌ی مناسب دست‌کم شامل هشت حرف و ترکیبی از حروف الفبای کوچک و بزرگ، اعداد و کاراکترهای خاص است. از همه مهم‌تر گذرواژه‌ای انتخاب کنید که بتوانید آن‌را به‌خاطر بسپارید. از ابزارهایی چون John the ripper برای پی‌بردن به گذرواژه‌های ضعیف کاربران روی سرور استفاده کنید. pam_cracklib.so را برای اجباری کردن یک سیاست گذرواژه‌ی قوی، پیکربندی کنید.

  - طول عمر رمزهای‌عبور: دستور chage تعداد روزهای بین تغییرات گذرواژه و تاریخی که آخرین‌بار گذرواژه تغییر کرده است را تغییر می‌دهد. این اطلاعات به‌وسیله‌ی سیستم استفاده می‌شود تا مشخص کند یک کاربر چه زمانی باید رمزعبورش را عوض کند. فایل /etc/login.defs پیکربندی رمزهای‌عبور مربوط به سایت، چون طول عمر آنها، را برعهده دارد. برای غیرفعال‌کردن تعیین طول عمر رمزعبور، دستور زیر را وارد کنید:

```
# chage -M 99999 userName
```

در آخر می‌توانید فایل /etc/shadow را طبق الگوی کلی زیر ویرایش نمایید:

```
:{password}:{lastpasswdchanged}:{Minimum_days}:{Maximum_days}:{Warn}:{Inactive}
:{expire}
```

گزینه‌های بالا با توجه به نام عبارت‌ها نیازی به توضیح ندارند، اما پیشنهاد می‌شود به‌جای ویرایش فایل /etc/shadow از دستور chage طبق الگوی کلی زیر استفاده کنید:

```
# chage -M 60 -m 7 -W 7 userName
```

  - محدودکردن استفاده از گذرواژه‌های پیشین: می‌توان کاربران را از استفاده‌ی دوباره از گذرواژه‌ی پیشین منع کرد. پارامتر remember در ماژول pam_unix می‌تواند برای پیکربندی تعداد گذرواژه‌های پیشین که نمی‌توانند دوباره استفاده شوند، مورد استفاده قرار گیرد.

  - قفل‌کردن یک کاربر پس از ورود ناموفق: در سیستم‌عامل لینوکس می‌توان از دستور faillog برای دیدن گزارش‌های ورود ناموفق یا اِعمال محدودیت‌های ورود ناموفق استفاده کرد. faillog محتویات فایل ویژه‌ی ثبت وقایع و گزارش (log) را از گزارش‌های خطای ورود بانک اطلاعات /var/log/faillog به‌روز می‌کند. همچنین faillog برای نگه‌داری تعداد خطاهای ورود و اِعمال محدودیت استفاده می‌شود. برای دیدن تلاش‌های ناموفق در ورود به سیستم، از دستور زیر استفاده نمایید:

```
# faillog
```

همچنین برای بازکردن یک حساب‌کاربری قفل شده پس از تلاش در ورود ناموفقش، از دستور زیر استفاده کنید:

```
# faillog -r -u userName
```

توجه داشته باشید که می‌توان از دستور passwd نیز برای قفل کردن و باز کردن یک حساب‌کاربری استفاده کرد. برای قفل‌کردن یک حساب‌کاربری به‌فرم زیر عمل می‌شود:

```
# passwd -l UserName
```

برای بازکردن یک حساب‌کاربری نیز از دستور زیر استفاده می‌شود:

```
# passwd -u UserName
```

- چگونه حساب‌های‌کاربری بدون رمز را پیدا کنیم؟ برای انجام این‌کار باید از دستور زیر استفاده نمایید:

```
# awk -F: '($2 == "") {print}' /etc/shadow
```

در ادامه با کمک دستور Passwd، حساب‌های بدون گذرواژه را قفل کنید.

- مطمئن شوید شناسه‌ی هیچ کاربری به‌جز کاربر ریشه، عدد صفر نباشد: تنها کاربر ریشه با شناسه کاربری صفر امکان دسترسی کامل به سیستم را دارد. برای پیدا کردن تمام حساب‌های‌کاربری با شناسه‌ی صفر دستور زیر را وارد کنید:

```
# awk -F: '($3 == "0") {print}' /etc/passwd
```

پس از اجرا باید تنها یک سطر مانند نمونه‌ی زیر را مشاهده کنید:

```
root:x:0:0:root:/root:/bin/bash
```

چنانچه حساب‌های دیگری از شناسه‌ی کاربری صفر استفاده می‌کنند، آنها را حذف کنید یا مطمئن شوید که آنها را می‌شناسید.

- **ورود کاربر ریشه را غیرفعال کنید**: هیچ وقت با کاربر ریشه وارد سیستم نشوید. باید از دستور sudo برای اجرای دستورهای در سطح کاربر ریشه استفاده کنید. دستور sudo به‌شدت امنیت سیستم را بالا می‌برد، به‌دلیل اینکه دیگر لازم نیست رمز کاربر ریشه را در اختیار دیگر مدیران سیستم قرار دهید. افزون بر این sudo قابلیت پیگیری دستورهای استفاده شده را دارد.

- **امنیت فیزیکی سرور**: باید سرور را از نظر دسترسی فیزیکی به کنسول محافظت کنید. بایوس را برای غیرفعال کردن امکان راه‌اندازی سیستم از طریق DVD، CD یا حافظه‌های USB تنظیم کنید. همچنین روی بایوس و Grub، گذرواژه‌ای را تنظیم کنید.

- **سرویس‌هایی را که نیاز ندارید، غیرفعال کنید**: سرویس‌ها و دایمون‌هایی (سرویس‌هایی که در پس‌زمینه اجرا می‌شوند) را که نیاز ندارید، غیرفعال کنید. همچنین باید همه‌ی این سرویس‌های ناخواسته را از بخش راه‌اندازی سیستم خارج نمایید. دستور زیر را برای فهمیدن اینکه چه دستورهایی در سطح سه اجرا[1] خواهند شد، اجرا کنید:

```
# chkconfig --list | grep '3:on
```

برای غیرفعال‌کردن این سرویس‌ها می‌توان از دستورهای زیر استفاده کرد:

```
# service serviceName stop
# chkconfig serviceName off
```

همچنین پورت‌هایی که به شبکه گوش می‌دهند، را بیابید. از دستور زیر برای یافتن پورت‌های باز شبکه و برنامه‌های مربوط به آن استفاده کنید:

```
# netstat -tulpn
```

یا

```
# nmap -sT -O localhost
```

[1]_Run Level 3

```
# nmap -sT -O server.example.com
```

از iptables برای بستن پورت‌های باز یا از دستورهای service و chkconfig برای توقف سرویس‌هایی که به این پورت‌ها گوش می‌کنند، استفاده کنید.

- **محیط x را پاک کنید**: به‌طور معمول در سرورها به محیط X نیازی نیست. دلیلی برای استفاده از محیط X در سرور اختصاصی پست‌الکترونیکی ویا سرور وب آپاچی وجود ندارد. می‌توان محیط X را غیرفعال یا حذف کرد تا امنیت و بازدهی سرور بالا رود. فایل /etc/inittab را ویرایش کنید و سطح پیش‌فرض را به سه تغییر دهید. سرانجام محیط X را با کمک دستور زیر حذف کنید:

```
# yum groupremove  "X Window System"
```

- **پیکربندی iptables و TCPWrappers**: ابزار iptables برنامه‌ای است که در فضای کاربر User Spare اجرا می‌شود و به کاربر اجازه‌ی پیکربندی دیوارآتش گنجانده شده در هسته‌ی لینوکس (Netfilter) را می‌دهد. از دیوارآتش برای پالایش ترافیک و صدور مجوز تنها برای ترافیک لازم، استفاده می‌شود. از TCPWrappers برای فیلترکردن دسترسی شبکه به اینترنت استفاده کنید. از بسیاری از حمله‌های نوع DoS می‌توان به کمک ابزار iptables جلوگیری کرد.

- **مقاوم‌سازی /etc/sysctl.conf**: فایل /etc/sysctl.conf برای پیکربندی هسته‌ی لینوکس در زمان اجرا استفاده می‌شود. هسته، تنظیمات صورت گرفته در این فایل را هنگام راه‌اندازی سیستم می‌خواند و پیاده می‌کند. یک فایل نمونه به‌صورت زیر است:

```
#Turn on execshield
kernel.exec-shield=1
kernel.randomize_va_space=1
#Enable IP spoofing protection
net.ipv4.conf.all.rp_filter=1
#Disable IP source routing
net.ipv4.conf.all.accept_source_route=0
#Ignoring broadcasts request
net.ipv4.icmp_echo_ignore_broadcasts=1
net.ipv4.icmp_ignore_bogus_error_messages=1
#Make sure spoofed packets get logged
net.ipv4.conf.all.log_martians =1
```

- **پارتیشن‌های جداگانه**: جداسازی سیستم‌های‌فایلی سیستم‌عامل از فایل‌های کاربران، می‌تواند سیستم بهتر و امن‌تری را برای کاربر به ارمغان بیاورد. مطمئن شوید که سیستم‌های‌فایلی usr/، home/، var/ و tmp/ در پارتیشن‌های جداگانه‌ای در اصطلاح Mount شده‌اند. پارتیشن‌های جداگانه برای مسیر ریشه‌ی آپاچی و سرویس‌دهنده‌ی FTP بسازید. فایل /etc/fstab را ویرایش کنید و مطمئن شوید که گزینه‌های پیکربندی زیر را در آن فایل پیاده کرده‌اید:
  - noexec: اجازه‌ندادن به اجرای فایل‌های دودویی در این پارتیشن برای جلوگیری از اجرا شدن فایل‌های موجود در آن پارتیشن به‌جز فایل‌های اسکریپت.
  - nodev: اجازه‌ندادن به کاراکتر یا دستگاه‌های ویژه در این پارتیشن برای پیش‌گیری از استفاده از فایل دستگاه‌هایی چون zero، sda و غیره.
  - nosuid: اجازه‌ندادن به دسترسی به SUID و SGID در این پارتیشن برای جلوگیری از فعال‌شدن بیت setuid.

یک مثال از فایل etc/fstab/ برای محدودکردن دسترسی کاربری بـه پارتیشـن dev/sda5/ کـه پارتیشـن ریشـه‌ی سرویس‌دهنده‌ی FTP است، به‌فرم زیر می‌باشد:

```
/dev/sda5   /ftpdata        ext3    defaults,nosuid,nodev,noexec 1 2
```

همچنین عملیات مربوط به سهمیه‌بندی دیسک را به‌دقت انجام دهید. مطمئن شوید که سهمیه‌بندی دیسک برای تمام کاربران فعال شده باشد. برای پیاده‌سازی سهمیه‌بندی دیسک گام‌های زیر را دنبال کنید:

- سهمیه‌بندی برای هر سیستم‌فایل را با تغییر etc/fstab/ اِعمال کنید.
- سیستم‌های‌فایل را دوباره Mount نمایید.
- پایگاه‌داده سهیمه‌ها را ایجاد کنید و جدول استفاده از دیسک را بسازید.
- سیاست‌های سهمیه‌بندی را پیاده نمایید.

- **قابلیت IPv6 را غیرفعال کنید**: IP نسخه شش، یک لایه اینترنتی جدید از پروتکل TCP/IP است که جـایگزین IP نسخه چهار خواهد شد و مزایای زیادی نیز به‌همراه خواهد داشت. هم‌اکنون ابزارهای مناسـبی بـرای بررسـی مشکلات امنیتی یک سیستم IPv6 موجود نیست. شمار زیادی از توزیع‌ها اقـدام بـه فعـال‌کـردن IPv6 بـه‌طـور پیش‌فرض کرده‌اند. به‌دلیل اینکه بیشتر مدیران سیستم IP نسخه شش را نظارت نمی‌کنند، نفـوذگران مـی‌تواننـد یک ترافیک بد را از طریق IPv6 وارد شبکه کنند. بنابراین، یا IPv6 را غیرفعال کنید ویا دیوارآتش IPv6 لینوکس را پیکربندی نمایید.
- **فایل‌های SUID و SGID ناخواسته را غیرفعال کنید**: تمام فایل‌هایی که بیت SUID/SGID آنهـا فعـال شـده، می‌توانند درصورت وجود مشکل امنیتی، مورد سوءاستفاده قرار گیرند. همه‌ی کاربران محلی یا راه‌دور می‌تواننـد از این فایل‌ها استفاده کنند. ایده‌ی خوبی است که همه‌ی این فایل‌ها یافت شود. از دستور find بـرای ایـن منظـور استفاده کنید:

  یافتن همه‌ی فایل‌هایی که بیت user id آنها فعال شده:

```
# find / -perm +4000
```

  پیداکردن تمام فایل‌هایی که بیت group id آنها فعال شده:

```
# find / -perm +2000
```

  و یا برای پیداکردن هر دوی آنها:

```
# find / \( -perm -4000 -o -perm -2000 \) -print
# find / -path -prune -o -type f -perm +6000 -ls
```

باید در مورد هر فایل گزارش شده، تحقیق شود. برای کسب اطلاعات بیشتر راهنمای آن فایل‌ها را بخوانید.

- فایل‌های قابل نوشتن عمومی: هر فایل قابل نوشتن عمومی می‌تواند توسط هر کسـی مـورد اسـتفاده قـرار گیرد و این باعث به‌وجود آمدن مشکلات امنیتی خواهد شد. با کمک دستور زیـر فایـل‌هـای قابـل نوشـتن عمومی و با بیت‌های sticky فعال شده را بیابید:

```
# find /dir -xdev -type d \( -perm -0002 -a ! -perm -1000 \) -print
```

نیاز است که درباره‌ی هر فایل گزارش شده تحقیق کنید و سپس بیت SUID و GUID آنها را تنظیم یا پاک کنید.

- فایل‌های بدون مالک: فایل‌های بدون مالک می‌توانند مشکل امنیتی به‌وجود بیاورند. با کمک دستور زیر، این فایل‌ها را که به هیچ کاربر یا گروه معتبری تعلق ندارند، پیدا کنید:

```
# find /dir -xdev \( -nouser -o -nogroup \) -print
```

پس از آن باید در مورد گزارش‌های خروجی این دستور تحقیق کنید و مالکیت فایل‌ها را به کاربران ویا گروه‌های شناخته شده انتقال دهید.

- **از یک سرویس تأیید هویت مرکزی استفاده کنید**: بدون یک سیستم هویت‌سنجی متمرکز، داده‌های هویت‌سنجی کاربران، ناپایدار و غیرقابل اعتماد می‌شوند. این امر باعث به‌وجود آمدن امکان دسترسی برای حساب‌های تاریخ گذشته یا حساب‌های فراموش شده‌ای می‌شود که باید از روی سیستم پاک می‌شدند. یک سرویس تأیید هویت مرکزی به کاربر اجازه می‌دهد تا امکان کنترل و نگه‌داری حساب‌های یونیکس و لینوکس و داده‌های هویت‌سنجی آنها را به‌صورت متمرکز در اختیار داشته باشد. همچنین می‌توان داده‌ها را بین سرورهای خود همسان[1] کرد. از سرویس NIS برای این سیستم تأیید هویت مرکزی استفاده نکنید و به‌جای آن OpenLDAP را برای سرورها و کاربران به‌کار ببرید. Kerberos، هویت‌سنجی را به‌صورت یک سرویس طرف سوم قابل‌اعتماد و با استفاده از رمزگذاری و با این فرض انجام می‌دهد که بسته‌های شبکه در فضایی ناامن منتقل می‌شوند و ممکن است توسط هر کسی خوانده، ویرایش و دستکاری شوند. Kerberos، براساس رمزنگاری با کلید متقارن کار می‌کند و به یک مرکز توزیع کلید نیاز دارد. به کمک این ابزار می‌توان از راه دور وارد سیستم شد، فایل‌ها را کپی کرد و کپی فایل درون سیستمی و دیگر عملیات با خطر بالا را امن‌تر نمود. به این ترتیب زمانی‌که کاربرانی با استفاده از Kerberos وارد یکی از سرویس‌های شبکه می‌شوند، کسانی که سعی می‌کنند با پایش اطلاعات شبکه گذرواژه‌ها را بدزدند، ناکام می‌مانند.
- **گزارش‌گیری و حسابرسی**: باید سیستم گزارش‌گیری و حسابرسی را برای جمع‌آوری اطلاعات مربوط به تلاش‌های نفوذ تنظیم کنید. به‌صورت پیش‌فرض گزارش‌های سیستمی در مسیر /var/log ذخیره می‌شوند. این‌کار همچنین برای پی‌بردن به پیکربندی نامناسب نرم‌افزار که سیستم را برای حمله آماده می‌سازد، مفید است. خواندن راهنماهای مربوط به دایمون syslogd و فایل پیکربندی syslog.conf نیز در این باره مفید خواهد بود.

- گزارش‌های مشکوک را با کمک ابزارهای logwatch و logcheck بررسی کنید. این ابزارها بررسی گزارش‌ها را ساده‌تر می‌سازد و درباره‌ی موارد نامعمول، توضیحات جزئی‌تری فراهم می‌نماید. نمونه‌ای از سبک استفاده و خروجی ابزار logwatch درقالب شکل (1-1) نمایش داده شده است.

- حسابرسی سیستم با استفاده از auditd: دایمون auditd برای انجام حسابرسی سیستم ایجاد شده است. auditd مسئول نوشتن گزارش‌های audit بر روی دیسک است. در حین فرآیند راه‌اندازی سیستم قواعد موجود در /etc/audit.rules توسط سرویس پس‌زمینه‌ی auditd خوانده می‌شود. می‌توان فایل /etc/audit.rules را باز و تغییرات مورد نظر مانند تنظیم محل فایل گزارش و دیگر گزینه‌ها را تنظیم نمود. توسط دایمون auditd می‌توان از موارد زیر آگاه شد:
  - رخدادهای هنگام راه‌اندازی و خاموش شدن سیستم

[1] _Sync

- تاریخ و زمان یک رخداد
- نام‌کاربری که مسئول یک رخداد است
- نوع رخداد (ویرایش، دسترسی، پاک کردن، نوشتن و غیره)
- موفقیت یا شکست یک رخداد
- ثبت رخدادهایی که تاریخ و زمان را تغییر می‌دهند.
- آگاهی از اینکه چه کسی تنظیمات شبکه سیستم را تغییر داده است.
- ثبت رخدادهایی که اطلاعات کاربر/گروه را تغییر می‌دهند.
- آگاهی از اینکه چه کسی یک فایل را تغییر داده است.

```
# logwatch --service sshd --range=Today --detail=High

 ---------- SSHD Begin ----------

 Illegal users from:
    192.168.1.83: 12 times
       bob/password: 6 times
       george/password: 3 times
       raphael/password: 3 times

 **Unmatched Entries**
 pam_succeed_if(sshd:auth): error retrieving information about user raphael :
3time(s)
 pam_succeed_if(sshd:auth): error retrieving information about user bob : 6
time(s)
 PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser=
rhost=192.168.1.83  : 4 time(s)
 pam_succeed_if(sshd:auth): error retrieving information about user george : 3
time(s)

 ---------- SSHD End ----------
```

شکل (1-1) شمایی از نحوه‌ی استفاده و خروجی ابزار logwatch

- **سرور OpenSSH خود را امن کنید**: استفاده از پروتکل SSH برای ورود و انتقال فایل‌ها از راه دور پیشنهاد می‌شود. با توجه به اینکه SSH در معرض انواع حمله‌ها قرار دارد، تلاش کنید آن‌را تا حد امکان امن کنید. پروتکل SSH و سرور OpenSSH و همچنین مکانیزم‌های امن‌سازی آنها در فصل دوازدهم کتاب به‌طور کامل بررسی خواهد شد.
- **سیستم‌های تشخیص نفوذ را نصب کرده و به‌کار ببرید**: سیستم تشخیص نفوذ شبکه (NIDS)[1] سیستمی برای کشف فعالیت‌های مُخرب چون حمله‌های DoS، پویش پورت‌ها و هر تلاشی برای شکستن سیستم، از طریق کنترل و نظارت ترافیک شبکه است. بهتر است تمام نرم‌افزارهای کنترل سلامت و یکپارچگی سیستم را پیش از فعال شدن سیستم در محیط کاری نصب و آزمایش کنید. حتی پیشنهاد می‌شود نرم‌افزار AIDE را پیش از اتصال سرور به هر شبکه‌ای نصب کنید. IDE یک سیستم تشخیص نفوذ مبتنی بر میزبان است. این برنامه می‌تواند سازوکار درونی سیستم را پایش کند. Snort نیز نرم‌افزاری است که می‌تواند تحلیل بی‌درنگ بسته‌ها در شبکه‌ی IP را برای کشف نفوذ، به انجام برساند. فصل دهم کتاب به بررسی سیستم‌های تشخیص نفوذ، Honeypot و مانیتورینگ شبکه خواهیم پرداخت.

[1]_Network Intrusion Detection System

- **محافظت از پوشه‌ها، فایل‌ها و نامه‌های‌الکترونیکی**: لینوکس محافظت‌های عالی در برابر دستیابی به اطلاعات برای افراد احرازهویت نشده دارد و با کمک سطوح دسترسی فایل‌ها و MAC از دستیابی افراد ناشناس به فایل‌ها جلوگیری می‌کند؛ هر چند مجوزهای لینوکس، چنانچه حمله‌کننده به سیستم دسترسی فیزیکی داشته باشد و بتواند دیسک‌سخت آن‌را به کامپیوتر دیگری منتقل کند، هیچ سودی نخواهد داشت، اما با ابزارهای دیگری می‌توان ضریب امنیت را بالا برد. این ابزارها شامل موارد زیر می‌باشند:
  - استفاده از gpg برای رمزنگاری و رمزگشایی فایل‌ها به‌وسیله‌ی گذرواژه
  - گذرواژه‌های لینوکسی و یونیکسی از فایل‌ها درهنگام استفاده از OpenSSL یا دیگر ابزارهای مشابه محافظت می‌کنند.
  - encryptfs می‌تواند برای رمزنگاری پوشه‌ها استفاده شود.
  - True Crypt یک برنامه‌ی رمزنگاری دیسک کدباز برای سیستم‌های لینوکسی، ویندوزی و MacOS است.

می‌توان از گواهی‌های SSL و کلیدهای gpg برای امن‌سازی ارتباطات پست‌الکترونیکی بین هر دو طرف سرور و کلاینت استفاده کرد.

1-6 مهم‌ترین نقاط آسیب‌پذیر یونیکس و لینوکس

در این بخش به‌طور کلی به معرفی مهم‌ترین نقاط آسیب‌پذیر یونیکس و لینوکس خواهیم پرداخت. در این راستا، در فصل‌های بعدی کتاب به بررسی کامل هریک از نقاط آسیب‌پذیر اشاره شده، علت وجود ضعف امنیتی، سیستم‌های‌عامل در معرض تهدید، روش‌های تشخیص آسیب‌پذیری سیستم و سبک مقابله ویا پیشگیری در مقابل هریک از نقاط آسیب‌پذیر می‌پردازیم. همان‌گونه که بخش‌های پیشین به آن اشاره شد، بیشتر تهدیدها و حمله‌ها، متأثر از وجود نقاط آسیب‌پذیر در سیستم‌های‌عامل می‌باشد که زمینه‌ی تهاجم را برای مهاجمان فراهم می‌آورد. شناسایی و تجزیه و تحلیل نقاط آسیب‌پذیر در هریک از سیستم‌های‌عامل، نتیجه‌ی تلاش و پردازش ده‌ها کارشناس امنیتی ورزیده در سطح جهان است و باید مدیران سیستم و شبکه در یک سازمان به‌سرعت با آنها آشنا شوند و اقدام‌های لازم را انجام دهند. نقاط آسیب‌پذیر موجود در هر سیستم‌عامل مبتنی بر یونیکس و لینوکس که در ادامه به آنها اشاره می‌گردد، سندی پویا و شامل دستورالعمل‌های لازم برای برخورد مناسب با هریک از نقاط آسیب‌پذیر می‌باشد. همان‌گونه که می‌دانید یونیکس و لینوکس، سیستم‌عامل‌هایی رایج در جهان هستند که امروز در سطح بسیار گسترده‌ای استفاده می‌شوند. تاکنون حمله‌های فراوانی توسط مهاجمان متوجه این نوع سیستم‌های‌عامل بوده است. با توجه به حمله‌های متنوع و گسترده‌ی انجام شده، می‌توان مهم‌ترین نقاط آسیب‌پذیر یونیکس و لینوکس را به ده گروه عمده‌ی زیر تقسیم نمود:

- BIND Domain Name System
- Remote Procedure Calls (RPC)
- Apache Web Server
- General UNIX Authentication Accounts with No Passwords or Weak Passwords
- Clear Text Services
- Sendmail
- Simple Network Management Protocol (SNMP)
- Secure Shell (SSH)
- Misconfiguration of Enterprise Services NIS/NFS
- Open Secure Sockets Layer (SSL)

1-7 امنیت سیستم‌عامل یونیکس

یونیکس به‌عنوان وارث ساده و کاراتر مالتیکس، ویژگی‌های امنیتی فراوانی مانند گذرواژه‌های قوی، استفاده از حلقه‌های حفاظت، لیست کنترل دسترسی و غیره را از مالتیکس اقتباس کرد، اما این ویژگی‌ها ساده‌تر و کاراتر بودند. به‌دلیل اینکه یونیکس مانند سیستم‌عامل مالتیکس پروژه‌ای دولتی نبود، با اهداف امنیتی متفاوتی طراحی شد. هدف یونیکس، ایجاد سکوی مشترکی (برای نمونه، دستگاه‌ها و سیستم‌فایل) بود که می‌توانست توسط کاربران مختلف به اشتراک گذاشته شود. درنتیجه، مسئله‌ی امنیت به محافظت تبدیل شد که در آن هدف، محافظت از داده‌های کاربران در برابر خطاهای غیرعمدی در برنامه‌هایشان می‌باشد. هرچند، محافظت تضمین نمی‌کند که می‌توان به اهداف محرمانگی و جامعیت (یعنی امنیت) دست یافت. اجرای اهداف امنیتی مستلزم این است که مکانیزم‌های امنیتی سیستم بتوانند اهداف امنیتی را حتی هنگامی‌که تمام نرم‌افزارهای بیرون از پایگاه محاسبه، معتمدی مغرض باشند، اجرا کنند. درنتیجه، وقتی سیستم‌های یونیکس از طریق اینترنت به کاربران نامعتبر متصل شدند، بسیاری از تصمیم‌های طراحی که برای محافظت اتخاذ شده بودند، کاربرد چندانی نداشتند. مکانیزم‌های امنیتی یونیکس، قابلیت برآورده ساختن نیازمندی‌های یک سیستم‌عامل امن را ندارند. برای گسترش یا جایگزینی مکانیزم‌های ناامن در سیستم‌های یونیکس متداول با مکانیزم‌هایی که امکان برآورده ساختن نیازمندی‌های یک سیستم‌عامل امن را دارند، تلاش‌های زیادی انجام شده است. در ادامه به بررسی برخی از این مکانیزم‌ها درباره‌ی امنیت یونیکس می‌پردازیم.

1-7-1 امنیت یونیکس

پیش از بررسی جزییات امنیتی یونیکس، توصیف کوتاهی از سیستم یونیکس ارائه می‌کنیم. یک سیستم یونیکس درحال اجرا، از یک هسته‌ی سیستم‌عامل و شمار زیادی فرآیند که هریک از آنها برنامه‌ای را اجرا می‌کنند، تشکیل شده است. یک مرز حلقه‌ی حفاظت، هسته‌ی یونیکس را از فرآیندها ایزوله می‌کند. هر فرآیند، فضای آدرس ویژه‌ی خود را دارد که آدرس‌هایی از حافظه را که آن فرآیند می‌تواند به آنها دسترسی داشته باشد، مشخص می‌کند. سیستم‌های یونیکس نوین، اصولا فضاهای آدرس را برحسب مجموعه صفحات حافظه که فرآیندها می‌توانند به آنها دسترسی داشته باشند، تعریف می‌کنند. یونیکس از مفهوم فایل برای تمام اشیای پایای سیستم مانند دستگاه‌های I/O، شبکه و ارتباطات بین فرآیندی استفاده می‌کند. به هر فرآیند یونیکس، بر اساس کاربر مربوطه، یک شناسه تخصیص داده می‌شود و دسترسی به فایل‌ها با استفاده از شناسه‌ی فرآیند، محدود می‌شود.

در امنیت یونیکس، هدف، محافظت کاربران از یکدیگر و محافظت برپایه‌ی محاسبه‌ی معتبر (TCB[1]) از تمام کاربران می‌باشد. به بیانی غیر رسمی، TCB یونیکس، از هسته و شمار زیادی فرآیند که با شناسه‌ی کاربر ممتاز (root یا کاربر مافوق) اجرا می‌شوند، تشکیل شده است. فرآیندهای root، سرویس‌های مختلفی مانند راه‌اندازی سیستم، احرازهویت کاربر، مدیریت، سرویس‌های شبکه و غیره را فراهم می‌سازد. هم هسته و هم فرآیندهای root، مجوز دسترسی کامل به سیستم را دارند. تمام فرآیندهای دیگر، بر اساس شناسه‌ی کاربر مربوط، دارای دسترسی محدودی می‌باشند.

1-7-2 سیستم حفاظت یونیکس

یونیکس یک سیستم حفاظت کلاسیک و نه یک سیستم حفاظت امن پیاده‌سازی می‌کند. سیستم حفاظت یونیکس از یک وضعیت حفاظت و مجموعه‌ای از عملیات که امکان تغییر وضعیت را فراهم می‌سازند، تشکیل شده است. درنتیجه، یونیکس یک سیستم *کنترل دسترسی اختیاری* (DAC) است. با این وجود، یونیکس برخی از جنبه‌های سیستم حفاظت

[1]_Trusted Computing Base

امن را دارا می‌باشد. نخست اینکه، سیستم حفاظت یونیکس یک وضعیت گذار تعریف می‌کند که مشخص می‌سازد که فرآیندها چگونه بین حوزه‌های حفاظت تغییر می‌کنند. دوم اینکه، وضعیت برچسب زنی معمولا برای موارد خاص استفاده می‌شود. سرویس‌های معتبر، شناسه‌های کاربری را به فرآیندها تخصیص می‌دهند، اما کابران می‌توانند تخصیص مجوزها به منابع سیستم (یعنی فایل‌ها) را کنترل کنند. به‌عنوان تحلیل پایانی، این مکانیزم‌ها و سیستم حفاظت اختیاری، برای ایجاد سیستمی که نیازمندی‌های یک سیستم‌عامل امن را برآورده سازد، ناکافی هستند. وضعیت حفاظت، عملیاتی را که نهادهای سیستم می‌توانند روی اشیای سیستم انجام دهند، مشخص می‌کند. وضعیت حفاظت یونیکس، دسترسی نهادها به فایل‌ها (اشیا) را به شناسه‌های فرآیند (نهادها) تخصیص می‌دهد. هر شناسه‌ی فرآیند یونیکس، از یک شناسه‌ی کاربر (UID) یک شناسه‌ی گروه (GID) و یک مجموعه گروه‌های پیوستی، تشکیل شده است. از ترکیب این موارد برای تعیین حق دسترسی استفاده می‌شود.

همه‌ی منابع یونیکس، به‌عنوان فایل نشان داده می‌شوند. وضعیت حفاظت، نهادهایی را مشخص می‌کند که می‌توانند عملیات خواندن، نوشتن و اجرا کردن را (با معنای استاندارد این عملیات) روی فایل‌ها انجام دهند. با اینکه دایرکتوری‌ها فایل نیستند، اما به‌عنوان فایل نشان داده می‌شوند، هرچند عملیات، معناهای متفاوتی دارند (مثلا برای یک دایرکتوری، عمل اجرا به معنی جست‌وجو است). همچنین به فایل‌ها یک UID و یک GID نسبت داده می‌شود که امتیازهای خاص فرآیندهایی با این شناسه‌ها را نشان می‌دهند. یک فرآیند با UID مالک، می‌تواند هر بُعدی از وضعیت حفاظت را برای این فایل تغییر دهد. فرآیندهایی با هریک از شناسه‌های UID مالک یا GID گروه، می‌توانند حقوق اضافه‌تری در دسترسی به فایل داشته باشند که در ادامه به تشریح آن می‌پردازیم. مجموعه‌ی محدود اشیا و عملیات، طراحان یونیکس را قادر می‌ساخت تا از یک لیست کنترل دسترسی فشرده به نام بیت‌های حالت یونیکس برای مشخص کردن حقوق دسترسی شناسه‌ها نسبت به فایل‌ها، استفاده کنند. بیت‌های حالت، حقوق سه نوع نهاد را مشخص می‌کردند که شامل موارد زیر می‌باشند:

1. UID مالک فایل
2. GID گروه فایل
3. دیگر نهادها

با استفاده از بیت‌های حالت، مجوزدهی به‌طریق زیر انجام می‌شود:

نخست، مکانیزم مجوزدهی یونیکس بررسی می‌کند که آیا UID شناسه‌ی فرآیند، برابر UID مالک می‌باشد؟ اگر چنین بود، از بیت‌های حالت مالک برای صدور مجوز دسترسی استفاده می‌کند. اگر GID شناسه‌ی فرآیند یا گروه‌های پیوستی، برابر GID گروه فایل بودند، آن‌گاه از بیت‌های حالت مجوزهای گروه، استفاده می‌شود؛ وگرنه، از مجوزهای تخصیص داده شده به دیگران استفاده می‌شود. همان‌گونه که می‌دانید، بیت‌های حالت یونیکس به شکل {بیت‌های دیگران، بیت‌های گروه، بیت‌های مالک} می‌باشند که هر عنصر این چندتایی، از یک بیت خواندن، یک بیت نوشتن و یک بیت اجرا تشکیل شده است. همان‌گونه که به آن اشاره شد، سیستم حفاظت یونیکس، یک سیستم کنترل دسترسی اختیاری است. به‌ویژه، این بدین معناست که بیت‌های حالت یک UID مالک، یا GID گروه می‌توانند با هر فرآیندی که توسط مالک فایل اجرا می‌شود (یعنی، فرآیندی که UID آن با مالک فایل یکسان است) تغییر داده شوند. اگر اعتماد داشته باشیم که تمام فرآیندهای کاربران، هر کاری که انجام می‌دهند به سود کاربر است، آن‌گاه اهداف امنیتی کاربر می‌توانند اجرا شوند. هرچند، این فرض یک فرض منطقی نیست.

امروز، کاربران انواع مختلفی از فرآیندها را اجرا می‌کنند که برخی از آنها ممکن است متعلق به یک حمله‌کننده باشد یا نسبت به به‌مخاطره افتادن توسط حمله‌کننده‌ها آسیب‌پذیر باشد؛ پس کاربر نمی‌تواند هیچ تضمینی داشته باشد که این فرآیندها با اهداف امنیتی تناقض نخواهند داشت. درنتیجه، یک سیستم‌عامل امن، برای اجرای اهداف امنیتی کاربر، نمی‌تواند از کنترل دسترسی اختیاری استفاده کند. چراکه کنترل دسترسی اختیاری، به کاربران اجازه می‌دهد که افزون بر بیت‌های حالت UID مالک، GID گروه مربوط به فایل‌هایشان را نیز تغییر دهند. در این‌حالت برچسب‌زنی فایل‌ها نیز اختیاری می‌باشد.

در یک سیستم حفاظت امن، وضعیت برچسب‌زنی باید اجباری باشد؛ پس این دلیلی دیگر برای این است که سیستم یونیکس نیازمندی‌های امنیتی یک سیستم‌عامل امن را برآورده نمی‌سازد. فرآیندهای یونیکس، با استفاده از سرویس‌های معتمد با برچسب‌هایی از مجموعه‌ای از برچسب‌ها (یعنی UIDها و GIDها) که توسط مدیران معتمد تعریف شده‌اند، برچسب‌زنی می‌شوند و فرآیندهای فرزند، شناسه‌ی فرآیندیشان را از والد خود به ارث می‌برند. چنین رهیافت اجباری در برچسب‌زنی فرآیندها با شناسه‌ها، نیازمندی‌های امنیتی یک سیستم‌عامل امن را برآورده می‌کند (گرچه، این رهیافت نسبتا نامنعطف است). همان‌گونه که می‌دانید، بیت‌های حالت در یونیکس، خصوصیتی را نیز برای گذارهای حوزه‌ی حفاظت دارد که بیت setuid نامیده می‌شود. زمانی‌که این بیت برای یک فایل یک می‌شود، هر فرآیندی که فایل را اجرا کند، به‌طور خودکار یک گذار حوزه‌ی حفاظت به UID مالک فایل و GID گروه انجام خواهد داد. برای نمونه، اگر یک فرآیند root بیت setuid را روی فایلی که آن فرآیند مالک آن می‌باشد یک کند، آنگاه هر فرآیندی که آن فایل را اجرا کند، با UID ریشه اجرا خواهد شد. به‌دلیل اینکه بیت UID یک بیت حالت است، مالک فایل می‌تواند آن‌را یک کند، بنابراین به روشی اختیاری مدیریت می‌شود. یک وضعیت حفاظت امن، نیاز به یک وضعیت گذار اجباری دارد که تمام گذارهای حوزه حفاظت را مشخص می‌کند، پس استفاده از بیت‌های اختیاری setuid ناکافی می‌باشد.

3-7-1 تحلیل امنیتی یونیکس

درصورتی یک سیستم‌عامل امن است که بتواند نیازمندی‌های امنیتی سیستم‌عامل امن را برآورده کند. یونیکس قادر به برآورده کردن هیچ‌یک از این نیازمندی‌ها که شامل موارد زیر می‌باشند، نیست:

- وساطت کامل: واسط ناظر مرجع چگونه تضمین می‌کند که تمام عملیات حساس امنیتی به‌درستی وساطت می‌شوند؟ واسط ناظر مرجع یونیکس از قلاب‌ها تشکیل شده است که مجوزهای دسترسی به فایل یا inode را در برخی فراخوان‌های سیستمی بررسی می‌کنند. واسط ناظر مرجع یونیکس، دسترسی به اشیایی را که هسته از آنها در عملیاتش استفاده می‌کند، مجوزدهی می‌نماید. مشکلی که وجود دارد این است که عملیات یونیکس (خواندن، نوشتن و اجرا) برای بیان کنترل دسترسی به اطلاعات، به اندازه‌ی کافی بامعنی و رسا نیستند. در یونیکس می‌توان بدون نیاز به داشتن مجوز نوشتن، فایل‌ها را تغییر داد (برای نمونه، با استفاده از fcntl).
- وساطت کامل: آیا واسط ناظر مرجع، تمام عملیات حساس امنیتی را روی تمام منابع سیستم وساطت می‌کند؟ مجوزدهی در یونیکس، وساطت کامل تمام منابع سیستم را فراهم نمی‌کند. برای برخی اشیا مانند ارتباطات شبکه‌ای، خود یونیکس هیچ مجوزدهی را فراهم نمی‌سازد.
- وساطت کامل: چگونه اثبات می‌کنیم که واسط ناظر مرجع، وساطت کامل را فراهم می‌کند؟ به‌دلیل اینکه واسط ناظر مرجع در جایی قرار گرفته است که در آنجا تمام عملیات حساس امنیتی انجام می‌شوند، دانستن اینکه تمام

عملیات شناسایی شده‌اند و تمام مسیرها وساطت شده‌اند، کار مشکلی است. از هیچ رهیافتی برای اثبات وساطت کامل استفاده نشده است.

- مقاوم بودن در برابر دستکاری: سیستم چگونه از ناظر مرجع خود (شامل سیستم حفاظتش) محافظت می‌کند؟ ناظر مرجع و سیستم حفاظتش در هسته نگه‌داری می‌شوند، اما این‌کار، مقاوم بودن در برابر دستکاری را تضمین نمی‌کند. نخست اینکه سیستم حفاظت اختیاری است، بنابراین هر فرآیند درحال اجرا می‌تواند آن‌را دستکاری کند. فرآیندهای غیرقابل اعتماد کاربر می‌توانند مجوزهای داده‌های کاربران خود را به‌طور دلخواه تغییر دهند، بنابراین اجرای اهداف امنیتی روی داده‌های کاربران امکان‌پذیر نیست. دوم اینکه هسته‌ی یونیکس به شیوه‌ای که هسته‌ی مالتیکس محافظت می‌شد، در برابر فرآیندهای غیرقابل اعتماد محافظت نمی‌شود. هر دو از حلقه‌های حفاظت برای ایزوله‌سازی استفاده می‌کنند، اما سیستم مالتیکس آشکارا دروازه‌هایی را مشخص می‌کند که برای بررسی قانونی بودن آرگومان‌های گذارهای حلقه استفاده می‌شوند. گرچه هسته‌های یونیکس بیشتر رویه‌هایی را برای بررسی صحت آرگومان‌های فراخوان‌های سیستمی فراهم می‌سازند، اما چنین رویه‌هایی احتمالا در مکان‌های اشتباهی قرار گرفته‌اند. در آخر، فرآیندهای سطح کاربر از واسطهای مختلفی، بالاتر و فراتر از فراخوان‌های سیستمی، برای دسترسی و تغییر خود هسته استفاده می‌کنند. گستره‌ی این واسطها از توانایی در نصب پیمانه‌های هسته در سیستم‌های‌فایل (مانند proc/ یا sysfs) تا واسطهایی میان سوکت‌های netlink برای دسترسی مستقیم به حافظه‌ی هسته (مانند از طریق دستگاه فایل dev/kmem/) می‌باشد. تضمین اینکه تنها کدهای معتمد می‌توانند به این واسطها دسترسی داشته باشند، ناممکن است.
- مقاوم بودن در برابر دست‌کاری: آیا مکانیزم حفاظت سیستم از برنامه‌های نوع TCB محافظت می‌کند؟ TCB یونیکس افزون بر هسته، تمام فرآیندهای root (شامل تمام فرآیندهایی که هنگام وارد شدن کاربر به‌عنوان کاربر root اجرا می‌شوند) را شامل می‌شود. به‌دلیل اینکه این فرآیندها می‌توانند هر نوع برنامه‌ای را اجرا کنند، محافظت TCB در برابر دست‌کاری را نمی‌توان تضمین کرد. حتی بدون در نظر گرفتن کاربران root، حجم کد TCB بسیار زیاد است و با تهدیدهای بسیاری مواجه است، از اینرو نمی‌توان ادعا کرد که TCB در برابر دست‌کاری مقاوم است. برای نمونه، فرآیندهای root فراوانی وجود دارند که پورت‌های شبکه بازی دارند که راه‌هایی برای به مخاطره انداختن این فرآیندها می‌باشند. اگر هر کدام از این فرآیندها به مخاطره بیافتد، سیستم یونیکس به مخاطره می‌افتد، زیرا هیچ محافظتی بین فرآیندهای root وجود ندارد. چنانکه در ادامه بیان می‌شود، به فرآیندهای root یونیکس نمی‌توان به اندازه‌ی کافی اعتماد داشت و محافظت شده نیستند، بنابراین به‌طور کلی امکان تغییر غیرمجاز سیستم حفاظت وجود دارد. درنتیجه، نمی‌توانیم در یونیکس یک سیستم حفاظت مقاوم در برابر دست‌کاری داشته باشیم.
- قابل اثبات بودن: چه چیزی مبنای درستی TCB سیستم است؟ هر مبنایی برای اثبات درستی سیستم یونیکس، غیررسمی است. اندازه‌ی مرزبندی نشده‌ی TCB، مانع از هرگونه اثبات رسمی می‌شود. افزون بر این، اندازه و ماهیت گسترش‌پذیری هسته (مثلا از طریق راه‌اندازهای دستگاه‌ها و دیگر ماژول‌های هسته‌ای جدید)، اثبات درستی هسته را ناممکن می‌سازد.
- قابل اثبات بودن: آیا سیستم حفاظت، اهداف امنیتی را اجرا می‌کند؟ به‌دلیل عدم وساطت کامل و مقاوم نبودن در برابر دست‌کاری، نمی‌توان اجرای اهداف امنیتی را اثبات کرد. به‌دلیل اینکه نمی‌توانیم سیاستی به اندازه‌ی کافی غنی تعریف کنیم که از افشا یا تغییر غیرمجاز اطلاعات جلوگیری کند، نمی‌توانیم اهداف امنیتی محرمانگی و