Лабораторна робота
Контроль цілісності файлової системи Linux утилітою Tripwire

Шляхом обчислення контрольної суми можна не тільки визначити справжність дистрибутива перед встановленням, а й регулярно перевіряти цілісність системних файлів в процесі роботи, щоб уникнути несанкціонованого проникнення в систему.

Хоча ліцензія Linux/Free BSD дозволяє використовувати дистрибутив, куплений на радіоринку, з тим же успіхом, що і придбаний у офіційного дистриб'ютора, але довіряти веб-сервер вартістю понад тисячу у. е. диску за кілька десятків гривень особисто я б не ризикнув. Звичайно, цей диск - це, швидше за все, просто «злитий» з сайту і записаний на болванку ISO-образ. Але гарантувати цього ніхто не зможе - що заважає замінити деякі пакети на допрацьовані під свої потреби з «трояном» всередині? Причому самому і вигадувати щось не треба, все давно є в Мережі.
Якщо ж іншого виходу немає, або просто потрібно встановити програму, що не входить в основний дистрибутив, то завжди потрібно перевірити на збіг *контрольної суми.*
У більшості FTP-архівів разом з файлом можна знайти інформацію про контрольну сумі. Ця інформація знаходиться або в окремому файлі на кшталт CHECKSUM.MD5, або безпосередньо на веб-сторінці. Така перевірка за алгоритмом MD5 (SHA, SHA1, SHA256, тощо) по можливості гарантує, що в файлі немає змін і перед вами дійсно оригінал.
Дізнатися контрольну суму завантаженого файлу дуже просто:

$ md5sum mysql-max-3.23.55-unknown-freebsd4.7-i386.tar.gz
9b543fbe12c66d365ca68e819b0

Порівнявши обидва значення контрольних сум, можна зробити висновок про справжність файлу. Перевірка контрольної суми повинна стати звичкою при кожній установці програмного забезпечення. З цієї ж причини ніколи не варто брати файл з першого-ліпшого сервера. Найкраще зайти або на домашню сторінку, або на спеціальні сайти на кшталт

http://rpmfind.net/ и *http://www.freshports.org/.*

За допомогою контрольної суми можна не тільки перевірити програми, що встановлюються. Якщо зловмисник проникне в мережу, то першою його дією, швидше за все, буде установка або зміна деяких програм. Наприклад, він може замінити стандартну і досить часто використовувану програму *ps* на іншу, з «трояном» всередині, а команда *ls* може не помітити створених ним каталогів.

Щоб уникнути цього і мати інструмент, що дозволяє проконтролювати цілісність системних файлів, застосовуються утиліти, які автоматично перевіряють значення контрольних сум файлів і каталогів і заносять їх у свою базу даних. Згодом системний адміністратор може в автоматичному режимі звірити оригінальну, створену при установці базу даних з наявними в системі файлами - і при розбіжності контрольних сум видати попередження по електронній пошті. Це дозволить відразу ж дізнатися про вторгнення і оцінити наслідки.

Слід зазначити, що всі ці заходи не виключають, а, скоріше, доповнюють інші методи захисту. Причому засоби, що контролюють стан файлів, мають деяку перевагу перед програмами, які виявляють вторгнення за наявністю певних сигнатур, хоча ніщо не заважає використовувати комплексний підхід.

Найбільш популярний засіб для вирішення цього завдання - утиліта **Tripwire**. Завантажити її разом з документацією tripwire-docs можна з *http: //download.sourceforge.net/tripwire/.*

Після закінчення установки необхідно виконати початкову ініціалізацію бази даних. Але для того, щоб не довелося робити це двічі, краще відразу відредагувати файли конфігурації і політик. У каталозі */etc/tripwire* (в FreeBSD - */usr/local/etc/tripwire*) є два шаблони у вигляді текстових файлів. У файлі ***twcfg.txt*** міститься інформація про розміщення файлів бази даних (її конкретний зміст залежить від системи), а також який буде використовуватися редактор, поштовий клієнт і їх параметри.

У файлі ***twpol.txt*** міститься політика, що складається з серії правил, за виконанням яких стежить Tripwire, дані об'єктів контролю, важливість контрольованого об'єкта і для кожної групи контролю - адреса електронної пошти *(emailto =)*, на яку надсилається повідомлення про порушення. При необхідності можна вказати відразу декілька адрес через кому.

Тут же бажано замінити *TEMPDIRECTORY=/tmp* на каталог з нормальними правами доступу (в */tmp* для всіх *rwx*). Щоб уникнути несанкціонованої зміни, всі важливі файли Tripwire зберігаються на диску в закодованій і підписаній формі. Сама база даних, політики, конфігурація і іноді файл звіту захищені асиметричною криптографією ElGamal з 1024-розрядної сигнатурою. ElGamal використовує шифрування з одним відкритим і одним закритим ключем. У криптографічній системі Tripwire ця пара ключів зберігається в двох файлах.

Файли конфігурації і політики захищені від запису відкритим ключем, а база даних і звіти - локальним ключем. Для читання досить відкритого ключа, але для запису потрібно закритий ключ, захищений паролем. Тому після їх редагування відповідно до політики системи, необхідно запустити скрипт twinstall.sh, що знаходиться в цьому ж каталозі. Після цього програма запросить фрази ***site*** і ***local*** (бажано, не менше 8-ми символів) для генерації відповідних ключових пар. Потім в каталозі утворюються зашифровані файли конфігурації і пополітікі ***tw.cfg*** і ***tw.pol***, а також файли ключів */etc/tripwire/site.key* і */etc/tripwire/host-local.key.* З метою безпеки рекомендується після закінчення інсталяції видалити файли шаблонів з даного каталогу.

Проведемо початкове налаштування і ініціюємо базу даних:

 # /etc/tripwire/twinstall.sh
 # /usr/sbin/tripwire --init

Після запиту локального пароля програма створює «відбиток» важливих системних файлів (розмір, контрольна сума, права, час доступу, тощо) і записує її в файл в каталозі */var/lib/tripwire/$(HOSTNAME).twd*. При створенні бази даних зверніть увагу на повідомлення про помилки, наприклад:

Filename: /bin/bash ### No such file or directory ### Continuing...
Warning: File system error.
Filename: /bin/ash.static ### No such file or directory ### Continuing...

Це означає, що в конфігураційний файл занесена зайва інформація. Бажано відразу внести виправлення, інакше ці повідомлення в подальшому постійно будуть вас переслідувати і перевантажувати лог-файл.
Отримана база даних щодня порівнюється з поточним станом файлової системи. При цьому видаються досить докладні звіти. Це дозволяє виявити додані, змінені і вилучені файли. Таке розташування файлу зручно при перевірці узгодженості файлової системи за допомогою демона cron, що дозволяє повністю автоматизувати даний процес, включаючи інформування за вказаною електронною адресою про результати перевірки.

Але потенційний зловмисник, виявивши присутність утиліти, при отриманні певних прав

може замінити (оновити) базу даних або повністю перевстановити Tripwire (що набагато простіше). Після такої модифікації утиліта перевірки не помітить каверзи - і ви будете отримувати поштою повідомлення про те, що з системою все нормально, поки в один прекрасний день не знайдете розбіжність парольних фраз. Такий варіант також передбачений розробниками. Щоб уникнути цього, необхідно взяти отформатовану дискету (флешку) і присвоїти змінній *TRIPWIRE_FLOPPY=YES* значення з командою *make install*

$ make install TRIPWIRE_FLOPPY=YES

Або просто створити резервну базу даних:

$ make floppy

В результаті, на флешку буде скопійовано все, що необхідно для відновлення системи і автономної роботи утиліт: копія початкової бази даних системи, утиліти Tripwire, Twcheck і Gunzip і копія файлів налаштувань tripwire.

Звичайно, все це може не поміститися на 1,44 Мб навіть після архівування. На жаль, єдиний вихід у цьому випадку - зменшити кількість об'єктів бази даних. Як би там не було, на всяк випадок бажано резервувати всю систему разом з Tripwire і періодично звіряти оригінальну і робочу бази даних.

Для перевірки системи запускається *tripwire --check*. Можна явно вказати ім'я файлу створюваного звіту:

$ tripwire --check -r файл_звіту.twr

Tripwire порівнює поточний стан системи зі збереженим в базі даних і при невідповідності видає повідомлення. Програма звертає увагу навіть на такі дрібниці, як зміна часу модифікації. Аналогічно виконує щоденну перевірку демон *cron* за допомогою */etc/cron.daily /tripwire-check.*
Якщо вам ніколи чекати, поки система все перевірить, можна вказати опцію *--email-report*:

$ tripwire --check --email-report

Тоді повідомлення буде послано всім одержувачам, зазначеним у файлі політики, в форматі, визначеному в EMAIL-REPORTLEVEL. Але перед початком експлуатації бажано перевірити роботу електронної пошти за допомогою команди:

$ tripwire --test --email user@domain.com

Якщо потрібно перевірити не всю систему, а тільки окремі її об'єкти, їх можна відразу вказати так:

$ tripwire --check object1 object2 object3

Системні файли розбиті по групах важливості, що дозволяє більш гнучко будувати звіти. Завдяки цьому для скорочення звіту можна проігнорувати некритичні для системи файли. Для того, щоб перевірити файли, починаючи з середнього ступеня важливості і вище, можна ввести таку команду:

 # tripwire --check --severity 66

Або використовувати перевірку правил по іменам, які прописані у файлі tw.pol. Для перевірки системних файлів використовується команда:

 # tripwire --check --rule-name "File System and Disk Administraton Programs"

При необхідності частих перевірок певного типу замініть довгі імена за замовчуванням на більш короткі.

Ключ *--interactive* дозволяє змінити БД в діалоговому режимі. Можна також відредагувати форму зміни БД, в якій кожен файл, що додається, відзначається хрестиком в секції *Object Summary*, в текстовому редакторі, який запускається за допомогою команди *--visual <ім'я-редактора>* або прописується в файлі конфігурації за допомогою змінною *EDITOR* або змінних оточення *VISUAL або EDITOR*. Для такого редагування БД потрібно знати пароль.

В процесі роботи деякі утиліти оновлюються або замінюються іншими. Тому базу даних доводиться оновлювати за допомогою опції *--update*. Для оновлення всієї системи використовується команда:

$ tripwire --update –r файл_звіта.twr

(Оновлення полягає в актуалізації бази даних відповідно до стану, зафіксованого в файлі звіту). Якщо ж потрібно оновити тільки відомості про певну програму, то:

$ tripwire --update /usr/sbin/sshd

Після всіх перевірок створюється звіт. За замовчуванням це файл
/var/lib/tripwire/report/(HOSTNAME)-$(DATE).twr.
Відразу після створення він називається приблизно так: *localhost-20160327-071039.twr.* Але за допомогою команди

 # tripwire --check --twrfile /var/lib/report/myreport.twr

можна задати інше ім'я, яке зручніше використовувати в скриптах.

За замовчуванням, при генерації звітів використовується *REPORT-LEVEL = 3 (Concise Report),* що виводить досить докладну інформацію, в тому числі і про очікувані та спостережувані значення. Цей режим можна змінити у файлі *tw.cfg*. Tripwire дозволяє гнучко задавати параметри звітів, залежно від потреби в тій чи іншій інформації. Для цього по ходу роботи змінюється report-level в межах від 0 до 4:

 # twprint --print-report --report-level 1 --twr-file /var/lib/report/report.twr

Якщо знадобиться внести глобальні зміни, можна отримати розшифровану копію конфігураційного файлу за допомогою команди:

 # twadmin --print-cfgfile > /etc/tripwire/twcfg.txt

Після внесення та збереження змін зашифрований файл створюється командою:

 # twadmin --create-cfgfile --site-keyfile /etc/tripwire/site.key /etc/tripwire/twcfg.txt

У міру додавання нових файлів, які ви бажаєте контролювати, а також зі зміною рівня значущості і видалення старих файлів, зміни адрес електронної пошти і засміченням файлу звітів помилковими повідомленнями, файл політик починає потребувати змін. Перш ніж щось міняти, потрібно створити копію файлу політик за допомогою команди:

 # tripwire --print-profile > /etc/tripwire/twpol.txt

Після внесення змін модифікуємо політики в такий спосіб:

 # tripwire --update-policy /etc/tripwire/twpol.txt

Ключові файли *site* і *local* утворюються під час першого запуску, але іноді виникає необхідність в їх повній заміні. Пароль змінюється тільки при зміні ключів, тому, для того щоб змінити його, треба попередньо дешифрувати всі файли, згенерувати нові ключі, а потім зашифрувати файли з новими ключовими даними.

Якщо при цьому ви забудете пароль або видалите файли з ключами, то всі зашифровані файли (конфігурація, політики, база даних і звіти) після шифрування стануть недоступні. Кращим виходом з такої ситуації є перевстановлення всієї системи.
До речі, зазначимо: шифрування не завадить хакеру, якщо той отримає певні права, просто видалити будь-який файл, включаючи саму базу даних Tripwire. Тому використання Tripwire не скасовує обов'язкового резервного копіювання.

Дізнатися, який ключ був використаний (і чи був використаний взагалі) для шифрування файлу, можна за допомогою такої команди:

 # twadmin --examine filel file2

Шифрування скасовується (при цьому запитується пароль, сам файл залишається в бінарному форматі) так:

 # twadmin --remove-encryption filel file2

Потім створюємо новий ключ:

 # twadmin --generate-keys --local-keyfile /etc/tripwire/localkey.key

або:

 # twadmin --generate-keys --site-keyfile /etc/tripwire/sitekey.key

та шифруємо файл:

 # twadmin --encrypt --local-keyfile /etc/tripwire/localkey.key filel file2
або:

 # twadmin --encrypt --site-keyfile /etc/tripwire/sitekey.key filel file2

База даних зберігається в закодованому бінарному форматі. Для роздруківки її у вигляді текстового файлу можна скористатися додатковою утилітою, що входить в комплект Twprint. При використанні бази даних за замовчуванням команда виглядає так:

 # twprint --print-dbfile > db.txt

А для довільної бази даних - так:

- twprint --print-dbfile --dbfile otherfile.twd > db.txt

Аналогічно робимо для файлу зі звітом:

 # twprint --print-report --twrfile имя_файла_с_отчетом

У комплект пакета входить допоміжна програма, що обчислює контрольну суму методами CRC-32, MD5, HAVAL і SHA в реалізації Tripwire, що дозволяє порівняти продуктивність різних методів.
При необхідності повторити установку Tripwire, видаліть в каталозі */etc/tripwire* всі файли, крім *twcfg.txt*, *twinstall.sh і twpol.txt*, а також файли бази даних (.twd) і звітів (.twr) в каталогах */var/lib/tripwire* і */var/lib/tripwire/report.*
Звичайно, Tripwire - далеко не панацея. Але використання цієї утиліти істотно ускладнить життя бажаючим покопатися в чужому комп'ютері. Вона вчасно попередить адміністратора

про те, що відбувається.

Формати конфігураційних файлів

Конфігураційний файл */etc/tripwire/tw.cfg* (***twcfg.txt)*** містить шляхи до допоміжних програм, адреси електронної пошти та каталоги для розміщення конфігураційних файлів Tripwire. Файл структурований як список пар *ім'я_змінної=значення*. Всі рядки, які починаються з символу #, вважаються коментарями.

До деяких змінних можна звертатися як *$(ім'я_змінної)*. Дві змінні мають стандартні значення, які не можна змінювати. Це *HOSTNAME* (просте ім'я хоста) і *DATE* (представлення часу в форматі 20160427-111033). Обов'язкові змінні такі:

- POLFILE = */etc/tripwire/tw.pol* - ім'я файлу політик;
- DBFILE = */var/lib/tripwire/$(HOSTNAME) .twd* - ім`я файлу з базою даних;
- REPORTFILE = */var/lib/tripwire/report/$(HOSTNAME)-$(DATE).twr* - шаблон імен файлів зі звітами;
- SITEKEYFILE = */etc/tripwire/site.key* - відкритий ключ;
- LOCALKEYFILE = */etc/tripwire/$(HOSTNAME)-local.key* - закритий ключ.

Крім обов'язкових, є і додаткові змінні:

- EDITOR = */bin/vi* - ім'я текстового редактора, який використовується в інтерактивному режимі. Якщо не визначено, використовується програма, зазначена в системній змінній $VISUAL або $EDITOR;
- TEMPDIRECTORY = */tmp* — каталог для тимчасових файлів;
- LATEPROMPTING = *false* - запитувати пароль в останню чергу, щоб він якомога менше знаходився в системній пам'яті;
- SYSLOGREPORTING = *true* - видавати для syslog рівні user і notice - повідомлення про створення бази даних, перевірках, зміни бази даних і політик будуть мати рівень деталізації 0;
- LOOSEDIRECTORYCHECKING = *false* - при значенні true не видає зайвих повідомлень про зміни *atim*e
- REPORTLEVEL = 3 - рівень деталізації звіту *Twprint*, який використовується за умовчанням.

Параметри повідомлення, що відправляється по електронній пошті, визначаються наступними змінними:

- GLOBALEMAIL = список_адрес - адресати, зазначені в списку, отримують повідомлення за замовчуванням;
- MAILMETHOD = SMTP або SENDMAIL - використовуваний Tripwire протокол для передачі повідомлень;
- SMTPHOST = *імя_або_адреса* - ім'я домену або IP-адреса сервера SMTP. Ігнорується, якщо попередня змінна не встановлена в SMTP;
- SMTPPORT = 25 - номер порту, який використовується з SMTP. Ігнорується, якщо MAILMETHOD — не є SMTP;
- MAILPROGRAM - */usr/sbin/sendmail -oi -t* - шлях до поштової програмі. Якщо MAILMETHOD = SENDMAIL, то це може бути будь-яка програма, яка відповідає RFC 822;
- EMAILREPORTLEVEL = 3 — рівень деталізації звіту для поштових повідомлень;
- MAILNOVIOLATIONS = *false* - не посилати листи, якщо не зафіксовано порушень політик. При значенні true в цьому випадку будуть приходити повідомлення про відсутність порушень. Корисно для контролю функціонування системи, але збільшує кількість поштових повідомлень.

Конфігураційний файл політик */etc/tnpwire/tw.pol* (***twpol.txt***) складається з серії правил, що визначають для кожного об'єкта, який реквізит повинен бути зібраний і збережений у файлі бази даних. Кожен об'єкт у файлі політик пов'язаний з маскою властивості, яка визначає, що саме повинна контролювати утиліта Tripwire. Налаштовуючи різні параметри файлу політик, адміністратор системи може гнучко визначити способи перевірки цілісності системи за допомогою Tripwire.

Основним компонентом файлу політик є правила (rules), в яких визначаються властивості, що перевіряються для кожного об'єкта. Файли, розташовані в дочірніх каталогах, успадковують правила батьківського каталогу - за умови, що внутрішні каталоги знаходяться на одному і тому ж дисковому пристрої. В іншому випадку правила визначаються окремо.

Як і в файлі конфігурації, в файлі політик можливе використання коментарів і змінних. Крім об'єктів, що перевіряються, можуть додатково зазначатися ігноровані об`єкти, що дозволяє виключити "зайві" файли, наприклад:

object name -> property mask [attribute=value ...];

object name;

Пробіли в імені об'єкта ігноруються. Щоб обійти це правило, імена з пробілами беруть в лапки. Те ж стосується і самих символів лапок, а також інших спеціальних символів, які використовуються в C ++ як керуючі.
Властивості, що перевіряються, задаються маскою, де кожній властивості відповідає певна буква. При відсутності заданих властивостей об'єкт не перевіряється. Якщо перед буквою стоїть "мінус", властивість ігнорується, якщо "плюс" - перевіряється. Якщо знака перед буквою немає, мається на увазі попередній заданому правилу знак або, при відсутності знаків - "плюс". Перевіряються наступні властивості (формат *[+ -] * [pinugtsldbamcrCMSH]*):

- *p* - права доступу;
- *i* - inode;
- *n* - число жорстких посилань;
- *u* - uid;
- *g* - gid;
- *t* - тип файлу;
- *s* - розмір;
- *l* - вважається, що файл буде рости; якщо він зменшився, фіксується порушення (корисно для ведення журналу);
- *d* - номер пристрою, на якому зберігається відповідний inode;
- *b* - число блоків;
- *a* - час доступу (несумісний з + CMSH, так як для обчислення суми необхідно прочитати файл; доступ до вмісту директорії змінює час останнього доступу до неї. Цього можна уникнути за допомогою recurse = false, LOOSEDIRECTORYCHECKING = true і правила -a);
- *m* - час останньої модифікації;
- *c* - час створення/модифікації inode;
- *r* - номер пристрою для файлів пристроїв;
- *C* - контрольна сума по CRC-32, POSIX 1003.2 (найшвидший, але найменш захищений з представлених варіант);
- *M* - контрольна сума по MD5 RSA Data Security;
- *S* - контрольна сума по SHS/SHA-алгоритму;
- *H* - контрольна сума по HAVAL.

Рекомендується використовувати властивості C, M, S, H для файлів парами, а не всі чотири відразу, так як визначення всіх чотирьох істотно знизить продуктивність.

Атрибути правил визначають поведінку об'єктів і надають додаткову інформацію про них. Атрибути записуються у вигляді списку, кожен елемент якого має вигляд ім'я = значення і відділений від інших елементів комами. Атрибути можуть застосовуватися як до окремого правила, так і до групи правил. Атрибут для окремого правила записується так:

/usr/lib -> $(Readonly) (emailto = admin@foo.com);

Група правил записується в фігурні дужки. Список атрибутів для групи правил записується в дужках перед нею:

(attribute = value) {rulel; rule2; ... }

Індивідуальні атрибути мають більший пріоритет, ніж групові, за винятком атрибута *emailto* - в цьому випадку атрибут додається до попередніх, наприклад:

(emailto = adminl@foo.com, severity = 90) { /etc/dog-> +pingus (severity = 75);
/etc/cat-> $(Dynamic) (emailto = admin2@foo.com); }

Це можуть бути такі атрибути:

- *rulename* - просте ім'я файлу, що використовується за замовчуванням в звітах, для групування правил, для перевірки частини правил. Рекомендується вибирати коротке і зрозуміле ім'я;
- *emailto* - адреса електронної пошти, на яку по замовчуванню висилаються повідомлення. Кілька адрес перераховуються через крапку з комою і беруть в лапки;
- *severity* - рівень значущості правила (від 0 до 1000000, за замовчуванням - 0}. Чим більше значення, тим серйозніше рівень. Перевірка виконується тільки для правил зазначеного рівня і вище. В командному рядку можуть використовуватися ключі: *high* - рівень 100, *medium* - рівень 66, *low* - рівень 33;
- *recurse* - рекурсивна перевірка директорій. За замовчуванням має значення true (-1), що відповідає режиму повного перегляду. Можливо також значення false (0) або інше додатне число (максимальний рівень вкладеності до 1000000).

Tripwire підтримує невеликий набір preprocessor-подібних директив, які групують правила, забезпечують умовне виконання, виконують найпростішу діагностику і налагодження. Первинне призначення цього механізму полягає в забезпеченні спільного використання файлу політики декількома машинами. Директиви мають наступний синтаксис: *@@ directive [arguments]*. При цьому сама директива не може бути отримана як значення змінної, але змінні можуть використовуватися в якості параметрів. Обробляються наступні директиви:

- *@@ section ім'я* - призначена, в основному, для роботи з Windows NT. В Unix обробляються тільки правила в першій секції з ім'ям FS: якщо секцій немає, то обробляються всі правила, інтерпретують як правила UNIX. Тому файл політики, розроблений для Windows NT, може викликати проблеми при використанні в UNIX. Глобальні змінні визначаються в секції GLOBAL;
- *@@ ifhost ім'я_хоста* {|| *ім'я_хоста*} - використовувати правила тільки для зазначеного хоста (хостів}. Умовм можуть бути вкладеними:
- *@@ else і @@ endif* - обробка умовних виразів разом з *@@ ifhost @@ ifnost machine1 || machine2 /usr/bin -> + pinug; @@ eise /usr/bin -> + pinugsmC; @@ endif;*
- *@@ print* "рядок в лапках" - виведення на *stdout;*
- *@@ error* "рядок в лапках" - виведення на *stdout* і завершення зі статусом 1;
- *@@ end* (весь наступний текст у файлі ігнорується, не може використовуватися всередині групи або умовного блоку).

У файлі політики застосовуються два типи змінних. Область дії глобальних змінних поширюється на весь файл політики. Область дії локальних змінних, визначених у розділі UNIX, обмежена тільки

цим розділом. Якщо глобальна і локальна змінні мають однакові імена, локальна змінна отримує більший пріоритет в своєму розділі, тимчасово маскуючи глобальну змінну. Значення змінної присвоюється командою *ім'я_змінної = значення*, підстановка значення здійснюється конструкцією *$ (variable)*. Існують наступні наперед задані змінні:

- *ReadOnly* = + *pinugsmtdbCM-ractSH* (передбачається, що файли доступні тільки для читання);
- *Dynamic* = + *pinugtd-rsacmblCMHS* (передбачається часта зміна вмісту файлів, наприклад /home, /var);
- *Growing* = + *pinugtdl-rsacmbCMSH* (передбачається, що файли (наприклад, журнали) будуть тільки рости);
- *IgnoreAll* = *-pinusgamctdrblCMSH* (перевіряється тільки наявність або відсутність файлу):
- *IgnoreNone* = + *pinusgamctdrbCMSH -l* (перевіряються всі властивості; змінна призначена для конструювання власної маски, наприклад, $ (IGnoreNone) -ar - для попередження порушення часу доступу);
- *Device* = + *pugsdr-intlbamcCMSH* (для пристроїв і файлів, які не можна відкривати).

Рівні звітів:

- Рівень 0. Звіт одним рядком. З'являється завжди в рядку Subject повідомлення електронної пошти і має такий формат: *TWReport ім'я-хоста дата-і-час V: число-порушень S: макс-рівень A: додано R: видалено C: змінено TWReport LIGHTHOUSE 19991021134026 V: 45 S: 100 A: 2 R: l C: 6*
- Рівень 1. Список імен змінених файлів у вигляді, що легко розбирається програмою автоматичного відновлення або подібної. Кожен рядок складається з ключового слова (Added Modified), двокрапки й імені файлу: *Added: / usr / bin / bash Modified: / usr / bin*
- Рівень 2. Зведений звіт. Включає список порушень із зазначенням правил, а також список доданих, змінених і видалених файлів. У розділі *Object Summary* міститься докладний список змінених файлів.
- Рівень 3. Зведений звіт. Включає список порушень із зазначенням правил, а також список доданих і видалених файлів, очікувані і реальні властивості змінених файлів і додаткові подробиці.
- Рівень 4. Максимально детальний зведений звіт, що містить список порушень із зазначенням правил, список доданих, змінених і видалених файлів, детальний звіт по кожному доданому (всі властивості), кожному зміненому (все перевіряються, очікувані і реальні властивості) і кожному видаленому файлу (все перевіряються і очікувані властивості). Відмінні параметри позначаються зірочкою на початку рядка.

Завдання роботи:

1. Встановіть tripwire (twinstall.sh) та ініціалізувати базу даних.

2. Перезапустіть систему. Які файли при цьому зміняться. Перевірте зміни, зафіксовані tripwire і збережіть файл звіту.

3. Оновіть базу Tripwire. Вимкніть перевірку файлів, які відсутні в даній системі.

4. Переконайтеся, що база відповідає поточному стану файлової системи (здійсніть перевірку файлової системи і порівняйте з результатами попередньої перевірки).

5. Для кількох файлів налаштуйте перевірку незмінності прав доступу, числа жорстких посилань, uid, gid, розміру файлу, часу створення і часу доступу. Потім змініть одну з властивостей і перевірте, що Tripwire помітить ці зміни.