

W sprawozdaniu zamieść zrzuty ekranu z istotnych etapów oraz odpowiedzi na pytania.



Taki symbol oznacza, że trzeba w sprawozdaniu dodać zrzut ekranu (najczęściej 1) z wyniku działania polecenie.



Taki symbol oznacza, że należy dodać opis (najczęściej 1 zdanie) z wyniku działania polecenia.

Wstęp

Współczesne systemy operacyjne to złożone konstrukcje, które z jednej strony oferują szeroki zakres funkcji i dużą uniwersalność, lecz z drugiej – stwarzają liczne możliwości ataku. Termin *security hardening*, czyli „utwardzanie” systemów operacyjnych, odnosi się do zestawu działań mających na celu podniesienie i utrzymanie wysokiego poziomu bezpieczeństwa danej instancji systemu. Mówiąc kolokwialnie: co zrobić, aby utrzymać system linuksowy w stanie bezpiecznym przez cały okres jego „życia”.

Proces utwardzania bywa często pomijany, zwłaszcza przez mniej doświadczonych użytkowników, którzy wierzą w popularne mity na temat bezpieczeństwa Linuksa:

- Linux jest bezpieczny „z automatu” – wystarczy go zainstalować i nie trzeba podejmować dodatkowych działań.
- Na Linuksa nie ma wirusów.
- Mój system jest stabilny i bezpieczny, bo pracuje nieprzerwanie od 750 dni (uptime) bez problemów.
- W razie incydentu zawsze można sięgnąć do logów, by namierzyć przyczynę i sprawcę.
- „Działa? Nie ruszaj” – lepsze jest wrogiem dobrego!

Najczęstsze zagrożenia dla systemów linuksowych można pogrupować w kilka kategorii:

- przejęcie istniejącego konta użytkownika,
- podatne lub niewłaściwie skonfigurowane usługi sieciowe,
- nieautoryzowany dostęp i eskalacja uprawnień,
- ataki typu Denial of Service (DoS).

Celem *hardeningu* nie jest stworzenie systemu całkowicie odpornego na ataki (co jest de facto niemożliwe – patrz Dijkstra), lecz maksymalne utrudnienie pracy potencjalnym atakującym – nie mylić z hakerami* – przy jednoczesnym zachowaniu funkcjonalności systemu.



Edsger W. Dijkstra (1930-2002)

*Testowanie programu może wykazać obecność błędów,
ale nigdy ich brak!*

* Słowo **haker** ma szczególne znaczenie w środowisku open source. Od czasu powstania pierwszej wersji Linuksa mianem hakera określano ekspertów od architektury systemów i oprogramowania – osoby wyróżniające się zarówno wiedzą, jak i wyjątkową kreatywnością. Wielu specjalistów do dziś oburza fakt, że określenie haker zostało przerzucone na cyberprzestępców.

I. Weryfikacja obrazów ISO

Choć jest to dość zaskakujące, pobierając obraz ISO danego systemu operacyjnego (i nie tylko), warto upewnić się, czy jednocześnie na dysk nie jest ściągany... koń trojański. Wbrew pozorom jest to sytuacja jak najbardziej realna – taki przypadek miał miejsce w 2016 roku, kiedy na oficjalnej stronie dystrybucji Linux Mint dwa razy (!!!) podmieniono link do pliku ISO na złośliwy, prowadzący do obrazu zawierającego malware. Aby się przed tym zabezpieczyć, po pobraniu obrazu należy sprawdzić jego autentyczność. Większość dystrybucji publikuje zarówno sumy kontrolne ISO (np. SHA-512), jak i podpisy cyfrowe (zazwyczaj oparte na standardzie OpenPGP). Pewność daje weryfikację tych drugich.

1. Uruchom maszynę wirtualną z systemem Kali Linux.
2. Otwórz terminal w Kali Linux.
3. Stwórz katalog o nazwie odpowiadającej Twojemu numerowi indeksu:

```
mkdir nr_indeksu
```

4. Przejdź do tego katalogu:

```
cd nr_indeksu
```

5. Uruchom przeglądarkę internetową.
6. Wejdź na stronę Debiana: <https://www.debian.org/download>



Downloading Debian

This is Debian 12, codenamed *bookworm*, netinst, for 64-bit PC (amd64) [debian-12.7.0-amd64-netinst.iso](#) **1**

Download checksum: **2** [SHA512SUMS](#) **3** [Signature](#)

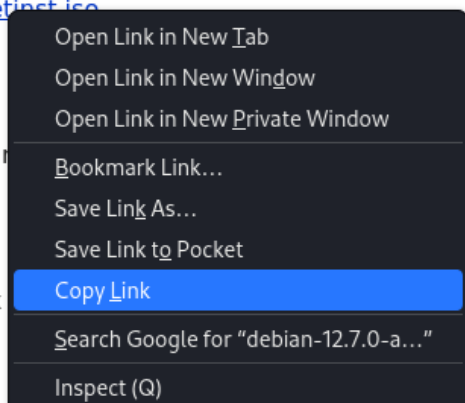
Debian installer ISOs are hybrid images, which means they can be written directly to CD/DVD/BD media OR to [USB](#)

7. Kliknij prawym przyciskiem myszy na link do pobrania obrazu ISO z Debianem (**1**), następnie wybierz opcję Copy Link.

) [debian-12.7.0-amd64-netinst.iso](#)

en directly to CD/DVD/BD r

ystems without a network



8. Wróć do terminala i pobierz obraz ISO:

```
wget skopiowany_link
```

9. Pobierz również plik sumy kontrolnej (2) oraz plik z podpisem (3).



10. Wyświetl zawartość katalogu, aby upewnić się, że wszystkie pliki zostały pobrane.

```
(kali@kali)-[~/debian]
$ ls -l
total 646156
-rwxrw-rw- 1 kali kali 661651456 Sep 14 06:17 debian-12.7.0-amd64-netinst.iso
-rw-rw-r-- 1 kali kali 494 Aug 31 15:20 SHA512SUMS
-rw-rw-r-- 1 kali kali 833 Aug 31 18:01 SHA512SUMS.sign
```

11. Jak zostało wspomniane, pierwszy etap to weryfikacja sum kontrolnych. Oczywiście możesz to zrobić ręcznie. Oblicz sumę dla pliku iso:

```
sha512sum plik_iso
```

12. Następnie wyświetl zawartość pliku z sumami kontrolnymi:

```
cat SHA512SUMS
```

13. Porównaj obie sumy kontrolne, aby sprawdzić ich zgodność.

```
(kali@kali)-[~/debian]
$ sha512sum debian-12.7.0-amd64-netinst.iso
e0bd9ba03084a6fd42413b425a2d20e3731678a31fe5fb2cc84f79332129afca2ad4ec897b4224d6a833afaf28a5d938b0fe5d680983182944162c6825b135ce  debian-12.7.0-amd64-netinst.iso

(kali@kali)-[~/debian]
$ cat SHA512SUMS
e0bd9ba03084a6fd42413b425a2d20e3731678a31fe5fb2cc84f79332129afca2ad4ec897b4224d6a833afaf28a5d938b0fe5d680983182944162c6825b135ce  debian-12.7.0-amd64-netinst.iso
915ab697472fd9a25a6b7b5d4988ee59fed61cd6dc6cd990435971af5894fca82426f213913fd95c0e4de8d10e0ee709023b677d02d5c48062208ff5ab3112  debian-edu-12.7.0-amd64-netinst.iso
d9480c2d765f3b1ebe8e7d06b1cf6ecf30b95146d5c2036f20904957db6139a440f9f8e7f4f901da6a02f810f2b3ab660aea56d99778c647c62386a2082c9407  debian-mac-12.7.0-amd64-netinst.iso
```

Sumy wydają się zgodne, jednak łatwo o pomyłkę (np. mała litera 'l' i wielka litera 'i' – $l \neq I$), co może stanowić poważne ryzyko. Dlatego skorzystajmy z oprogramowania GnuPG.

14. Aby zweryfikować podpisy, należy zaimportować odpowiednie klucze publiczne do bazy GnuPG. Wróć do przeglądarki na stronę Debiana, z której pobrałeś obraz i pliki weryfikacyjne, i kliknij na [ISO Verification Guide](#).

The screenshot shows the Debian website's 'Downloading Debian' page. At the top, there are links for 'Blog', 'Micronews', and 'Planet'. Below the page title, it states 'This is Debian 12, codenamed bookworm, netinst, for 64-bit PC (amd64)' and provides a link to 'debian-12.7.0-amd64-netinst.iso'. It also includes a 'Download checksum: SHA512SUMS Signature' link. Further down, it mentions 'Debian installer ISOs are hybrid images, which means they can be written directly to CD/DVD/BD media OR to USB sticks.' There are sections for 'Other Installers' and 'Related Links'. The 'Related Links' section contains links for 'Installation Guide', 'Release Notes', 'ISO Verification Guide' (which is circled in red), and 'Alternate Download Sites'.

15. Znajdziesz tam trzy klucze — interesują nas te z lat 2009 i 2011 (stabilne), natomiast klucz opisany jako Testing pomijamy.

```
pub  rsa4096/988021A964E6EA7D 2009-10-03
    Key fingerprint = 1046 0DAD 7616 5AD8 1FBC  0CE9 9880 21A9 64E6 EA7D
uid      Debian CD signing key <debian-cd@lists.debian.org>

pub  rsa4096/DA87E80D6294BE9B 2011-01-05 [SC]
    Key fingerprint = DF9B 9C49 EAA9 2984 3258  9D76 DA87 E80D 6294 BE9B
uid      Debian CD signing key <debian-cd@lists.debian.org>

pub  rsa4096/42468F4009EA8AC3 2014-04-15 [SC]
    Key fingerprint = F41D 3034 2F35 4669 5F65  C669 4246 8F40 09EA 8AC3
uid      Debian Testing CDs Automatic Signing Key <debian-cd@lists.debian.org>
```

16. Wróć do terminala i zaimportuj pierwszy klucz:

```
gpg --keyserver keyring.debian.org --recv-keys 988021A964E6EA7D
```

```
(kali@kali)-[~/debian]
$ gpg --keyserver keyring.debian.org --recv-keys 988021A964E6EA7D
gpg: key 988021A964E6EA7D: public key "Debian CD signing key <debian-cd@lists.debian.org>" imported
gpg: Total number processed: 1
gpg:   new keys imported: 1
```

17. Następnie zaimportuj również drugi klucz.



18. Wyświetl listę kluczy, aby potwierdzić ich obecność w bazie GnuPG:

```
gpg --list-keys
```

19. Co ważne, pobrane klucze są dostępne nie tylko na serwerach Debiana, ale także na wielu innych, takich jak keys.openpgp.org czy keyserver.ubuntu.com. Wykonaj dodatkową weryfikację, importując klucz z innego serwera:

```
gpg --keyserver domena_serwera --recv-keys 988021A964E6EA7D
```

```
(kali@kali)-[~/debian]
$ gpg --keyserver keys.openpgp.org --recv-keys 988021A964E6EA7D
gpg: key 988021A964E6EA7D: "Debian CD signing key <debian-cd@lists.debian.org>" not changed
gpg: Total number processed: 1
gpg:   unchanged: 1
```

20. Ponieważ nie zostały wprowadzone żadne zmiany, oznacza to, że dany klucz jest poprawny. Przeprowadź taką samą weryfikację dla drugiego klucza.

21. Mając pewność, że zaimportowaliśmy właściwe klucze, możemy teraz przeprowadzić weryfikację pobranego obrazu (oraz wszystkich przyszłych wersji podpisanych przez Debiana). Sprawdź zgodność hasha pliku ISO z plikiem z sumami kontrolnymi

```
sha512sum -c SHA512SUMS
```

```
(kali㉿kali)-[~/debian]
$ sha512sum -c SHA512SUMS
debian-12.7.0-amd64-netinst.iso: OK
sha512sum: debian-edu-12.7.0-amd64-netinst.iso: No such file or directory
debian-edu-12.7.0-amd64-netinst.iso: FAILED open or read
sha512sum: debian-mac-12.7.0-amd64-netinst.iso: No such file or directory
debian-mac-12.7.0-amd64-netinst.iso: FAILED open or read
sha512sum: WARNING: 2 listed files could not be read
```

Jak zauważyłeś pobrany obraz z Debianem ma poprawną sumę (OK). Pozostałe dwa (-edu- oraz -mac-) nie zostały pobrane, więc komunikaty o ich nieznalezieniu są prawidłowe i nie powinny niepokoić.

22. Wróćmy do sprawdzenia autentyczności naszych sum kontrolnych. W tym celu zweryfikujemy podpis cyfrowy pliku z sumami. Wykonaj poniższą komendę:

```
gpg --verify SHA512SUMS.sign SHA512SUMS
```

Powiniś otrzymać podobny wynik:

```
gpg: Signature made Sat 31 Aug 2024 06:01:11 PM EDT
gpg: using RSA key DF9B9C49EAA9298432589D76DA87E80D6294BE9B
gpg: Good signature from "Debian CD signing key <debian-cd@lists.debian.org>"
[unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the owner.
Primary key fingerprint: DF9B 9C49 EAA9 2984 3258 9D76 DA87 E80D 6294 BE9B
```

Komunikat *Good signature from Debian CD signing key* dowodzi, że podpis jest poprawny. Widać także, że złożono go tym samym kluczem, którego identyfikator (a w zasadzie jego fragment) został podany przy imporcie. Teraz dopiero mamy pewność, że możemy bezpiecznie instalować naszego Debiana.

II. Źródła i aktualność oprogramowania

Najprostszym sposobem, aby uczynić system podatnym na ataki, jest... dostatecznie długo nic nie robić. Choć jest to frapujące, prędzej czy później w aktualnie używanej wersji jądra lub jednym z kilkuset zainstalowanych pakietów odkryte zostaną podatności. Instalacja aktualizacji to jedno z podstawowych zadań administratora, gdyż prościej jest wytłumaczyć osobom podejmującym decyzje biznesowe potrzebę wyłączenia serwera z użytku wynikającą z zaplanowanego okna serwisowego, niż z powodu nagłego ataku bądź awarii. Niestety, utrzymanie aktualności całego oprogramowania w systemie jest znacznie bardziej skomplikowane niż okresowe uruchamianie polecenia *apt update* && *apt upgrade*.

W przypadku Kali Linux (opartego na Debianie) podstawowym źródłem oprogramowania są repozytoria.

1. Przejrzyj swoje repozytoria.

```
nano /etc/apt/sources.list
```

2. Zauktualizuj lokalną listę dostępnych pakietów.

```
sudo apt update
```

Operacja powinna przebiegać bez problemów, jednak jaką mamy pewność, że nasze repozytoria są autentyczne? Odpowiedzią jest mechanizm automatycznej weryfikacji integralności pakietów, który opiera się na podpisach GPG.

- Wyświetl listę wszystkich kluczy GPG zainstalowanych w systemie, które są używane do weryfikacji autentyczności repozytoriów pakietów:

```
apt-key list
```

Jeśli jakieś repozytorium nie zostało prawidłowo zweryfikowane kluczem, podczas aktualizacji listy pakietów zostaniesz o tym poinformowany.

- Zweryfikujmy to. Otwórz ponownie plik `sources.list`.
- Dodaj repozytorium Ubuntu: `deb http://archive.ubuntu.com/ubuntu focal main restricted universe multiverse`

```
GNU nano 8.1 /etc/apt/sources.list *
# See https://www.kali.org/docs/general-use/kali-linux-sources-list-repositories/
deb http://http.kali.org/kali kali-rolling main contrib non-free non-free-firmware
deb http://archive.ubuntu.com/ubuntu focal main restricted universe multiverse
# Additional line for source packages
# deb-src http://http.kali.org/kali kali-rolling main contrib non-free non-free-firmware
```

- Zapisz zmiany, a następnie opuść plik.



- Wykonaj ponownie `apt update`.

```
(kali@kali) ~
$ sudo apt update
Get:1 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Hit:2 http://http.kali.org/kali kali-rolling InRelease
Err:3 http://archive.ubuntu.com/ubuntu focal InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0821F32 NO_PUBKEY 871928D1991BC93C
Warning: GPG error: http://archive.ubuntu.com/ubuntu focal InRelease: The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0821F32 NO_PUBKEY 871928D1991BC93C
Error: The repository 'http://archive.ubuntu.com/ubuntu focal InRelease' is not signed.
Notice: Updating from such a repository can't be done securely, and is therefore disabled by default.
Notice: See apt-secure(8) manpage for repository creation and user configuration details.
```

Jak widać, tym razem otrzymaliśmy komunikat, że system nie może zweryfikować podpisów GPG pakietów z repozytorium, ponieważ brakuje odpowiednich kluczy publicznych.

- Edytuj ponownie plik `sources.list` i usuń dodane przed chwilą repozytorium.
- Zaktualizuj listę dostępnych pakietów komendą `apt update`.
- Zanim przystąpimy do aktualizacji pakietów, zainstalujmy przydatne narzędzie `aptitude`:

```
sudo apt install aptitude -y
```

- Program `aptitude` posiada także tryb interaktywny, który umożliwia zarządzanie pakietami za pomocą interfejsu tekstowego. Aby uruchomić interfejs, wpisz:

```
sudo aptitude
```

```
Actions Undo Package Resolver Search Options Views Help
C-T: Menu ?: Help q: Quit u: Update g: Preview/Download/Install/Remove Pkgs
aptitude 0.8.13 @ kali
-- Upgradable Packages (2039)
-- Installed Packages (1232)
-- Not Installed Packages (66099)
-- Obsolete and Locally Created Packages (38)
-- Virtual Packages (61102)
-- Tasks (29)
```

12. Przeglądaj funkcje programu, a następnie wyjdź, naciskając klawisz `q`.

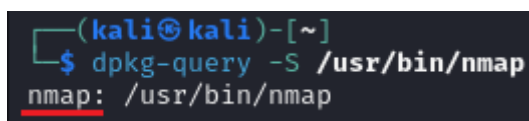
13. Zaktualizuj wszystkie zainstalowane pakiety (może to potrwać kilka minut):

```
sudo apt upgrade
```

Może się zdarzyć, że nie będziesz pewny, z jakiego pakietu pochodzi dany plik. Aby dowiedzieć się, skąd pochodzi określony plik w systemie, użyj polecenia `dpkg`.

14. Przykładowo, aby zidentyfikować pakiet zawierający plik `/usr/bin/nmap`, wpisz:

```
dpkg-query -S /usr/bin/nmap
```



```
(kali㉿kali)-[~]  
$ dpkg-query -S /usr/bin/nmap  
nmap: /usr/bin/nmap
```

Jak widać, plik ten pochodzi z pakietu `nmap`, co jest zgodne z oczekiwaniami.

15. Możesz również sprawdzić spójność tego pakietu (jest to przydatne, aby wykryć nieautoryzowane zmiany w plikach systemowych, które mogą świadczyć o problemach z bezpieczeństwem):

```
sudo dpkg --verify nmap
```

Jeśli po wykonaniu komendy nie zostaną zwrócone żadne komunikaty, oznacza to, że wszystko jest w porządku.

16. Na koniec sprawdź, czy nie występują pakiety, które mogą być niekompletnie zainstalowane lub usunięte, co może powodować błędy w systemie:

```
sudo dpkg --audit
```

Tym razem również nie powinien wystąpić żaden komunikat. zostały pobrane, więc komunikaty o ich nieznalezieniu są prawidłowe i nie powinny niepokoić.

III. Konta, użytkownicy, uprawnienia

Kluczową kwestią w systemie Linux jest nadawanie użytkownikom i programom tylko takich uprawnień, jakie są niezbędne... i niczego więcej. Należy bezwzględnie unikać stosowania `chmod -R 777` jako „uniwersalnego sposobu” na rozwiązywanie problemów z dostępem. Dla formalności poniżej umieszczono tabelę wartości numerycznych uprawnień:

Ósemkowy	0	1	2	3	4	5	6	7
Dwójkowy	000	001	010	011	100	101	110	111
Uprawnienia	---	--x	-w-	-wx	r--	rx-	rw-	rwx

Niestety, nie istnieje uniwersalna metoda bezpiecznego nadawania uprawnień – każdy użytkownik musi być świadomy podejmowanych działań i ich konsekwencji. Poniżej omówione zostanie znaczenie tego zagadnienia.

Przykład 1 (SUID oraz SGID)

Jak pamiętasz, aby dodać repozytorium w pliku `sources.list`, musiałeś użyć `sudo`. **UWAGA:** na koncie `root` należy przebywać tylko tymczasowo – nigdy nie pracować na nim stale. Każdą operację, która nie wymaga najwyższych uprawnień, należy wykonywać z konta zwykłego użytkownika, a `sudo` stosować tylko wtedy, gdy jest to absolutnie konieczne.

1. Sprawdźmy uprawnienia pliku `sources.list` oraz kto jest jego właścicielem:

```
ls -l /etc/apt/sources.list
```

```
(kali㉿kali)-[~]  
$ ls -l /etc/apt/sources.list  
-rw-r--r-- 1 root root 374 Nov 10 10:47 /etc/apt/sources.list
```

Jak widać, właścicielem pliku jest `root`, natomiast pozostali użytkownicy mają jedynie prawo do odczytu. Użycie `sudo` było więc w pełni uzasadnione. Istnieją jednak specjalne uprawnienia, takie jak `setuid (u+s)`, z którymi należy być szczególnie ostrożnym. Uprawnienie to pozwala uruchomić dany plik binarny z uprawnieniami jego właściciela, a nie użytkownika, który go uruchomił. Identycznie działa `setgid(g+s)`, ale w kontekście grupy. Sprawdźmy to w praktyce.

2. Sprawdź lokalizację binarki edytora `nano`:

```
which nano
```

```
(kali㉿kali)-[~]  
$ which nano  
/usr/bin/nano
```

3. Wyświetl szczegóły dotyczące uprawnień `nano`:

```
ls -l /usr/bin/nano
```

```
(kali㉿kali)-[~]  
$ ls -l /usr/bin/nano  
-rwxr-xr-x 1 root root 291672 Jul 15 18:25 /usr/bin/nano
```

4. Teraz kluczowa operacja. Dodaj uprawnienia `setuid`:

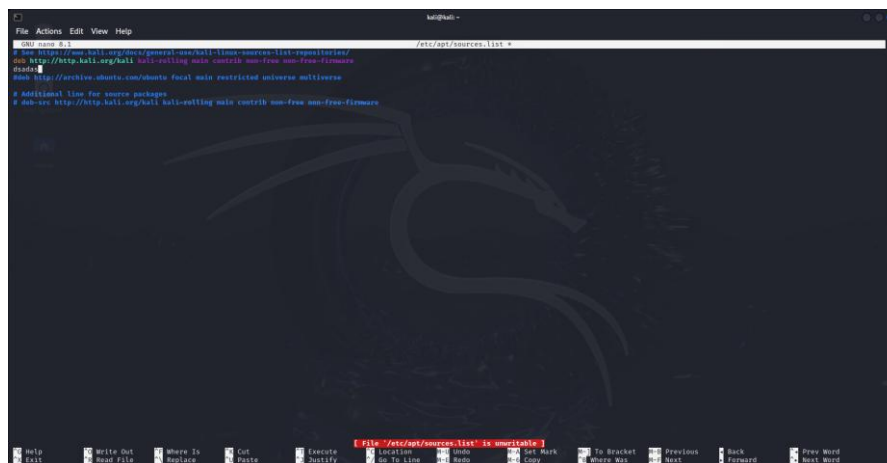
```
sudo chmod u+s /usr/bin/nano
```

-  5. Ponownie wyświetl szczegóły dotyczące uprawnień `nano`.

```
(kali㉿kali)-[~]  
$ ls -l /usr/bin/nano  
-rwsr-xr-x 1 root root 291672 Jul 15 18:25 /usr/bin/nano
```

Jak widzisz, zmiany wydają się kosmetyczne, ale w tym momencie system jest całkowicie skompromitowany. Teraz każdy, kto uruchomi edytor `nano`, uruchomi go z uprawnieniami konta `root` (ponieważ `root` jest właścicielem pliku wykonywalnego).

6. Aby to zademonstrować, spróbuj edytować plik `sources.list`, tym razem bez użycia `sudo`.



7. Dodaj losowy tekst, a następnie spróbuj zapisać plik (zignoruj komunikat informujący, że plik jest tylko do odczytu). Operacja powinna zakończyć się sukcesem! Innymi słowy, obecnie **każdy** użytkownik może czytać i modyfikować dowolny plik systemowy!

8. Usuń wprowadzone zmiany w pliku.



9. Przywróć poprzednie uprawnienia `nano` i załącz zrzut ekranu wyniku komendy `ls -l` jako dowód.

Jak widzisz, manipulowanie uprawnieniami może być niebezpieczne. W przypadku praw `setuid` i `setgid` najskuteczniejszym sposobem ochrony jest regularne sprawdzanie, które pliki mają te uprawnienia, oraz upewnienie się, że wiemy, dlaczego zostały one nadane!

10. Wykonaj poniższą komendę, aby sprawdzić jakie pliki mają nadane wspomniane dwa uprawnienia:

```
sudo find / -type f -perm /u+s,g+s
```

Przykład 2 (Sticky bit)

Choć intuicja podpowiada, że jeden użytkownik nie może usunąć pliku należącego do innego użytkownika (oczywiście poza użytkownikiem root), w rzeczywistości jest to sytuacja jak najbardziej możliwa. Aby taka sytuacja miała miejsce, wystarczy, że dany plik znajduje się w katalogu, który ma nadane prawo zapisu dla innych użytkowników. Usuwanie pliku jest bowiem de facto modyfikacją spisu zawartości katalogu. Udowodnijmy to.

11. Przejdź do katalogu `tmp`:

```
cd /tmp
```

12. Stwórz katalog o nazwie `test`:

```
mkdir test
```

13. Sprawdź jego uprawnienia:

```
ls -l
```

```
drwxrwxr-x 2 kali kali 40 Nov 10 15:23 test
```

Jak widać, aktualnie folder ma wyłączone uprawnienia zapisu dla innych użytkowników. Wyobraźmy sobie jednak sytuację, w której jest to folder współdzielony, a pozostali użytkownicy również powinni mieć uprawnienia zapisu.

14. Dodaj uprawnienia zapisu dla pozostałych użytkowników.

15. Sprawdź uprawnienia po modyfikacji, aby upewnić się, że zmiany zostały zastosowane.

```
drwxrwxrwx 2 kali kali 40 Nov 10 15:23 test
```

16. Przejdź do katalogu:

```
cd test
```

17. Stwórz plik o nazwie plik:

```
touch plik
```

18. W celach testowych utwórz nowego użytkownika o nazwie nowy:

```
sudo adduser nowy
```

19. Zaloguj się na użytkownika:

```
su nowy
```

20. Odczytaj właściciela pliku:

```
ls -l
```

```
(nowy@kali)-[/tmp/test]
$ ls -l
total 0
-rw-rw-r-- 1 kali kali 0 Nov 10 15:38 plik
```

Jak widać, właścicielem jest użytkownik kali, a pozostali użytkownicy mają uprawnienia tylko do odczytu.



21. Usuń plik (zatwierdź operację wpisując: yes):

```
rm plik
```

```
(nowy@kali)-[/tmp/test]
$ rm plik
rm: remove write-protected regular empty file 'plik'? yes

(nowy@kali)-[/tmp/test]
$ ls

(nowy@kali)-[/tmp/test]
$
```

Jak widać, użytkownik `nowy` usunął bez problemu plik należący do użytkownika `kali`. Jest to zasadniczy problem, ponieważ foldery współdzielone powinny umożliwiać użytkownikom tworzenie własnych plików tymczasowych, ale nie usuwać cudze! Rozwiązaniem tego problemu jest tzw. `sticky bit` (`o+t`), czyli prawo ograniczonego usuwania. W katalogu, w którym jest ustawiony `sticky bit`, tylko właściciel pliku może go usunąć lub zmienić jego nazwę.

22. Przełoguj się ponownie na użytkownika `kali`:

```
su kali
```

23. Wyjdź z folderu `test`:

```
cd ..
```

24. Dodaj `sticky bit` do katalogu `test`:

```
chmod o+t test
```

25. Sprawdź uprawnienia katalogu po zmianie (zwróć uwagę na literę `t`).

```
drwxrwxrwt 2 kali kali    40 Nov 10 15:41 test
```

26. Ponownie wejdź do katalogu `test` i stwórz plik:

```
cd test && touch plik
```

27. Przełoguj się na użytkownika `nowy`.



28. Wykonaj kolejną próbę usunięcia pliku.

```
(nowy@kali)-[/tmp/test]
$ rm plik
rm: remove write-protected regular empty file 'plik'? yes
rm: cannot remove 'plik': Operation not permitted
```

Jak widzisz, tym razem usunięcie pliku było niemożliwe!