

Implementacja i zastosowanie POSIX Capabilities w systemie plików



Michał Karbowańczyk

Politechnika Łódzka
Samodzielny Zakład Sieci Komputerowych

mak@zsk.p.lodz.pl



- POSIX Capabilities – POSIX 1003.1e
- Implementacja w jądrach 2.2-2.6.23
 - Domyślne przydzielanie uprawnień
 - Procesy z UID=0
- Implementacja w jądrach od 2.6.24
 - Uprawnienia dla pliku wykonywalnego
 - Uprawnienia dla użytkownika
- Globalna maska uprawnień
- Podsumowanie



- Tzw. „prawa administratora” to w rzeczywistości zbiór poszczególnych uprawnień (ang. capabilities)
- Uprawnienia te są zdefiniowane w drafcie POSIX 1003.1e określającym mechanizmy kontroli uprawnień w systemach zgodnych z POSIX (również m. in. podstawowe prawa dostępu do plików, ACL)
- Prace nad POSIX 1003.1e zostały przerwane, ale dokument jest traktowany jako standard



- Linux – od wersji jądra 2.2 na poziomie jądra, od wersji 2.6.24 również w systemie plików (pełna implementacja)
- Sun Trusted Solaris 8
- SGI Irix 6.5



- Definicje uprawnień znajdują się w kodzie źródłowym jądra w pliku *include/linux/capability.h*
- Każda definicja określa nazwę i numer uprawnienia, definicje są też opatrzone komentarzem wyjaśniającym znaczenie danego uprawnienia
- Przykład:

```
/*      In a system with the
[_POSIX_CHOWN_RESTRICTED] option defined, this
overrides the restriction of changing file
ownership and group ownership.      */
#define CAP_CHOWN 0
```



Przykłady capabilities

- `CAP_CHOWN` 0
Uprawnienie do zmiany właściciela pliku (chown)
- `CAP_DAC_OVERRIDE` 1
Uprawnienie do ignorowania praw dostępu do plików
- `CAP_KILL` 5
Uprawnienie do wysyłania sygnałów do procesów innych użytkowników
- `CAP_SETPCAP` 8
Uprawnienie do przydzielania i odbierania uprawnień innym procesom
- `CAP_NET_BIND_SERVICE` 10
Uprawnienie do otwierania tzw. niskich (<1024) portów TCP/UDP



Przykłady capabilities

- **CAP_NET_BROADCAST** 11
Uprawnienie do korzystania z transmisji broadcastowej
- **CAP_NET_ADMIN** 12
Uprawnienie do zmiany ustawień sieciowych (konfiguracja interfejsów, tablic routingu, reguł filtrowania itp)
- **CAP_NET_RAW** 13
Uprawnienie do korzystania z „surowych” gniazd sieciowych (z pominięciem wyższych warstw modelu ISO/OSI)
- **CAP_SYS_MODULE** 16
Uprawnienie do zarządzania modułami jądra
- **CAP_SYS_CHROOT** 18
Uprawnienie do wykonania operacji przeniesienia katalogu głównego (chroot)



- **CAP_SYS_ADMIN** 21
Uprawnienie do „administracji systemem”, w skład tego pojęcia wchodzi zarządzaie m. in: urządzeniami blokowymi, systemami plików, zużyciem energii, pamięcią wirtualną, limitami dyskowymi. Takie połączenie uprawnień jest poddawane krytyce.
- **CAP_SYS_NICE** 23
Uprawnienie do zarządzania priorytetami procesów
- **CAP_SYS_TIME** 25
Uprawnienie do zarządzania czasem systemowym



Implementacja w jądrach 2.2 – 2.6.23

Domyślne przydzielanie uprawnień



- Uprawnienia na poziomie jądra systemu są zaimplementowane w jego głównej linii od wersji 2.2
- „Implementacja na poziomie jądra” oznacza tu, że każdemu procesowi przypisany jest zbiór uprawnień i uprawnienia te są sprawdzane przy wykonywaniu przez proces operacji



Implementacja w jądrach 2.2 – 2.6.23

Domyślne przydzielanie uprawnień



- Procesy uruchamiane z UID równym 0 otrzymują wszystkie uprawnienia; jest to postrzegane jako „uprawnienia administratora” albo żargonowo „prawa root'a”
- Procesy uruchamiane z UID różnym niż 0 nie otrzymują żadnych uprawnień



- Przy tak określonych zasadach, jedyną możliwością uzyskania uprawnień przez „zwykłego użytkownika” jest uruchomienie przezeń procesu z UID równym 0
- Jest to możliwe dzięki zastosowaniu prawa SUID do pliku wykonywalnego. Powoduje to, iż UID procesu jest równy UID właściciela pliku wykonywalnego
- Wystarczy więc, aby właścicielem pliku był root i plik nadane miał prawo SUID, aby „zwykły użytkownik” mógł uruchomić proces ze wszystkimi uprawnieniami



- Najpopularniejsze polecenia korzystające z prawa SUID to: `passwd` i `ping`
- W przypadku polecenia `passwd` niezbędna jest możliwość zapisu do `/etc/shadow`
- W przypadku polecenia `ping` prawa administracyjne są potrzebne do uzyskania dostępu do „surowych” gniazd sieciowych
- Podobnie wiele serwerów usług sieciowych jest uruchamianych z uprawnieniami administratora po to, aby móc otworzyć do nasłuchu port TCP/UDP o numerze niższym niż 1024



Implementacja w jądrach 2.2 – 2.6.23

Procesy z UID=0



- Każdy proces działający z UID=0 jest ze względu na swoje uprawnienia potencjalnym zagrożeniem dla bezpieczeństwa systemu
- Dotyczy to szczególnie procesów korzystających z komunikacji sieciowej jako narażonych na ataki
- Problemem jest to, że na skutek bardzo zgrubnego rozdzielania uprawnień poszczególnych procesów (wszystko albo nic), wiele procesów posiada uprawnienia które normalnie nie są im potrzebne, ale mogą być wykorzystane przez atakującego



Implementacja w jądrach 2.2 – 2.6.23

Procesy z UID=0



- Istnieje kilka sposobów unikania tych zagrożeń:
 - Zmiana uprawnień przez sam proces po wykonaniu czynności, dla których niezbędne było posiadanie uprawnień – wymaga właściwego napisania programu
 - Uruchomienie procesu w izolowanym systemie plików (chroot)
 - Ograniczanie dostępu procesów do zasobów na bazie atrybutów procesu innych niż UID (SELinux)



- Poczynając od wersji jądra 2.6.24, uprawnienia są jednym z atrybutów plików wykonywalnych; można zatem konkretnemu programowi przydzielić dokładnie takie uprawnienia, jakich on potrzebuje
- Przykład: polecenie `ping` wymaga jedynie uprawnienia `CAP_NET_RAW`; jeżeli nadamy mu dokładnie to uprawnienie, to nawet w przypadku przejęcia tego procesu przez atakującego potencjalne szkody będą ograniczone



Implementacja w jądrach od 2.6.24

Zbiory uprawnień



- Każdy plik wykonywalny oraz każdy proces posiada trzy zbiory uprawnień
 - **Uprawnienia dozwolone (permitted set)** – zbiór wszystkich uprawnień, z jakich może korzystać proces
 - **Uprawnienia efektywne (effective set)** – zbiór uprawnień, z jakich proces korzysta w danej chwili
 - **Uprawnienia dziedziczne (inheritable set)** – zbiór uprawnień, które są przekazywane procesom potomnym



Implementacja w jądrach od 2.6.24 Zarządzanie uprawnieniami plików



- Aby zarządzać uprawnieniami plików wykonywalnych niezbędne są:
 - biblioteka *libcap*
 - polecenie `setcap` – ustawia uprawnienia pliku
 - polecenie `getcap` – odczytuje uprawnienia pliku
 - polecenie `getpcap` – odczytuje uprawnienia procesu



Implementacja w jądrach od 2.6.24

Nadawanie plikom uprawnień efektywnych



- W najprostszym przypadku, aby proces po uruchomieniu mógł korzystać z uprawnienia bez wykonywania dodatkowych czynności, należy umieścić to uprawnienie w zbiorze uprawnień dozwolonych i efektywnych pliku
- Przykład: odebranie poleceniu ping prawa SUID i nadanie uprawnienia CAP_NET_RAW:

```
# chmod u-s /bin/ping  
# setcap cap_net_raw=ep /bin/ping
```
- Dzięki posiadaniu uprawnienia CAP_NET_RAW proces ping działa mimo, że UID tego procesu nie jest równy 0



Implementacja w jądrach od 2.6.24

Uprawnienia efektywne a dozwolone



- Proces może w każdym momencie dodawać/ujmować ze zbioru uprawnień efektywnych te uprawnienia, które znajdują się w jego zbiorze dozwolonym
 - Wymaga to użycia specjalnego wywołania systemowego w kodzie programu
 - Jeżeli program potrafi w ten sposób zarządzać uprawnieniami („capabilities-aware”), to wystarczy dla niego określić zbiór uprawnień dozwolonych
- Nadawanie uprawnień efektywnych dla pliku wykonywalnego jest niezbędne tylko w przypadku programów, które nie potrafią same zarządzać swoimi uprawnieniami



- Umieszczenie uprawnienia w zbiorze uprawnień dziedzicznych (nie zaś dozwolonych) powoduje, że uprawnienie to znajdzie się w zbiorze uprawnień dozwolonych nowego procesu, tylko wtedy, jeśli znajduje się w zbiorze uprawnień dziedzicznych procesu uruchamiającego
- Inaczej mówiąc, aby proces mógł korzystać z uprawnienia dziedzicznego, musi ono być uprawnieniem dziedzicznym zarówno pliku wykonywalnego, jak i procesów tego użytkownika



- Ustalanie zbiorów uprawnień dla procesów użytkownika odbywa się poprzez moduł PAM *pam_cap*, którego wywołanie należy umieścić w konfiguracji PAM systemu
- *pam_cap* korzysta z pliku */etc/security/capability.conf*, w którym każda linia zawiera nazwę użytkownika oraz listę numerów uprawnień, które będą przez ten moduł nadawane jako dziedziczne procesowi uruchamianemu na rzecz użytkownika po jego zalogowaniu



- Przykład:

Nadanie poleceniu `ping` uprawnienia `CAP_NET_RAW` w zbiorze uprawnień dziedzicznych:

```
setcap cap_net_raw=ei /bin/ping
```

Konfiguracja modułu `pam_cap`:

```
cap_net_raw      sample
none             *
```

- Przy takiej konfiguracji proces `ping` będzie mógł korzystać z uprawnienia `CAP_NET_RAW` tylko wtedy, gdy będzie uruchomiony na rzecz użytkownika `sample` (lub `root`)



Implementacja w jądrach od 2.6.24

Szczegóły techniczne



- Nie zmieniły się zasady domyślnego przydzielania uprawnień, tzn. procesy z UID=0 mają wszystkie uprawnienia w zbiorze dozwolonym i efektywnym, natomiast procesy innych użytkowników domyślnie nie mają uprawnień w żadnym zbiorze
- Informacje o zbiorach uprawnień dla plików są przechowywane w formie atrybutów rozszerzonych (ang. *Extended Attributes*, xattrs), podobnie jak listy ACL czy konteksty SELinux.



Implementacja w jądrach od 2.6.24

Szczegóły techniczne



- Atrybut przechowujący informacje o uprawnieniach pliku wykonywalnego nosi nazwę `security.capability`
- Nie jest możliwa manipulacja wartością tego atrybutu poprzez użycie polecenia `setfattr`, podobnie jak w przypadku innych atrybutów z gałęzi `security`.
- Należy zwrócić uwagę na korzystanie z narzędzi plikowych obsługujących atrybuty rozszerzone



Globalna maska uprawnień



- Oprócz przydzielania uprawnień „per użytkownik” i „per plik” istnieje możliwość globalnego ograniczania uprawnień WSZYSTKICH procesów (także tych o UID=0)
- Możliwość ta dotyczy także implementacji w jądrach 2.2 – 2.6.23
- Globalna maska uprawnień jest dostępna jako pseudoplik */proc/sys/kernel/cap_bound*



- Globalna maska jest zrealizowana w reprezentacji binarnej ze znakiem, gdzie każdemu uprawnieniu odpowiada bit o numerze zgodnym z numerem uprawnienia
- Aby wyłączyć uprawnienie w masce globalnej, należy od jej aktualnej wartości odjąć wagę bitu o stosownym numerze i zapisać nową wartość
- Nie jest możliwe ponowne włączenie wyłączonego uprawnienia w masce globalnej – potrzebne jest ponowne uruchomienie systemu



- Implementacja uprawnień POSIX 1003.1e w systemie plików umożliwia zlikwidowanie poważnego zagrożenia dla systemu, jakim jest uruchamianie procesów z UID=0, a więc wszystkimi uprawnieniami
- Warto podkreślić, że stosowanie uprawnień nie wymaga zmian w kodzie programów
- Czekamy na wprowadzenie nowych polityk bezpieczeństwa przez wielkich dystrybutorów



- Dziękuję za uwagę i zapraszam do dyskusji
- Więcej informacji:
 - <http://www.friedhoff.org/posixfilecaps.html>
 - Linux Magazine, sierpień 2008

