Erişim Denetim Düzenekleri (Access Control Mechanisms)

Yrd. Doç. Dr. Özgü Can

Erişim Denetim Listeleri (Access Control Lists, ACL)

- Erişim Denetim Matrisi → Kullanıcının dosyalar üzerindeki haklarını matris içerisinde tanımlar.
- Problemler:
  - Özne ve nesne sayısını arttıkça matris boyutunun kullanacağı bellek
  - Matris girdileri:
    - Boş (empty) → Erişim yok
    - Aynı → Uygulamanın sağladığı varsayılan ayar
  - Özne ve nesnelerin silinmesi durumunda → Matrisin dikkatli bir şekilde yönetimi gereklidir.

Erişim Denetim Listeleri (Access Control Lists, ACL)

Erişim Denetim Listesi

Erişim denetim matrisini temel almakta

ve

erişim denetim matrisinde ortaya çıkan problemleri

gidermeye çalışmaktadır.

Erişim Denetim Listeleri (Access Control Lists, ACL)

- **S:** Özneler kümesi
- **R:** Haklar kümesi
- **l:** Erişim denetim listesi, $l = \{(s, r): s \in S, r \subseteq R\}$
- **acl:** *l* listesinin belirli bir *o* nesnesi ile ilişkili olduğunu belirten fonksiyon

$$acl(o) = \{(s_i, r_i) : 1 \leq i \leq n\}$$

s_i öznesi *o* nesnesine r_i içerisindeki herhangi bir hakkı kullanarak erişebilir.

Erişim Denetim Listeleri (Access Control Lists, ACL)

	file 1	file 2	process 1	process 2
process 1	read, write, own	read	read, write, execute, own	write
process 2	append	read, own	read	read, write, execute, own

acl(file1) = {(process1, {read, write, own}), (process2, {append})}

acl(file2) = {(process1, {read}), (process2, {read, own})}

acl(process1) = {(process1, {read, write, execute, own}), (process2, {read})}

acl(process2) = {(process1, {write}), (process2, {read, write, execute, own})}

Erişim Denetim Listeleri (Access Control Lists, ACL)

- **Varsayılan İzin (Default Permission)** → ACL içerisinde bir özneden bahsedilmiyorsa, öznenin nesne ile ilişkili herhangi bir izni yoktur.
- Birçok öznenin bulunduğu bir sistemde, ACL çok büyük olabilir.
  - Aynı dosya üzerinde birçok öznenin aynı hakkı varsa;
    - "***Wildcard***" tanımlanarak isimlendirilmemiş özneler için varsayılan haklar verilebilir.

Erişim Denetim Listeleri (Access Control Lists, ACL)

ÖR:

(user, group, rights) formundaki ACL girdisinde:

(Ayşe, ogrenci, r) → Ayşe'nin nesne üzerinde okuma (**r**) hakkı sadece Ayşe **ogrenci** grubunda olduğunda vardır.

(Ayşe, *, r) → Ayşe'nin nesne üzerinde okuma (**r**) hakkı içinde bulunduğu gruptan bağımsızdır.

(*, ogrenci, r) → **ogrenci** grubunda ki herhangi bir kullanıcının nesne üzerinde okuma (**r**) hakkı vardır.

ACL'in Kısaltılması

- UNIX → İzinler **üçlüler (triplets)** ile ifade edilir.
- Kullanıcı (user), grup (group) ve diğerleri (others)
- Haklar:
  - `r` ***read***
  - `w` ***write***
  - `x` ***execute***

```
% ls -l /etc/passwd
-rw-r--r--    1 root     sys      1306 Jan 31 17:07  /etc/passwd
uuugggooo      owner    group     size  date   mod   name
```

UNIX Erişim İzinleri

Number	Octal Permission Representation	Ref
0	No permission	---
1	Execute permission	--x
2	Write permission	-w-
3	Execute and write permission: 1 (execute) + 2 (write) = 3	-wx
4	Read permission	r--
5	Read and execute permission: 4 (read) + 1 (execute) = 5	r-x
6	Read and write permission: 4 (read) + 2 (write) = 6	rw-
7	All permissions: 4 (read) + 2 (write) + 1 (execute) = 7	rwx

Value	Permission	Directory Listing
0	No read, no write, no execute	`---`
1	No read, no write, execute	`--x`
2	No read, write, no execute	`-w-`
3	No read, write, execute	`-wx`
4	Read, no write, no execute	`r--`
5	Read, no write, execute	`r-x`
6	Read, write, no execute	`rw-`
7	Read, write, execute	`rwx`

ACL'in Kısaltılması

ÖR:

UNIX sisteminde 5 kullanıcı olması durumunda

Kullanıcılar: *Can-Aslı-Selin-Cem-Selim*

Can; Aslı'ya okuma, Selin'e yazma, Cem'e yürütme, Selim'e okuma ve yazma izinleri vermek istiyorsa;

Klasik UNIX erişim denetimi bütün erişim izinlerini gerçekleştirmek için yeterli değildir.

ACL'in Kısaltılması

- UNIX erişim denetimi
  - "*Cem dışındaki bütün kullanıcılar*"

  şeklinde bir tanıma izin vermemektedir.

- Grupları sadece sistem yöneticisi yaratabilir.

ACL'in Kısaltılması

- Birçok sistem ACL kısaltmalarına gelişmiş ACL'ler kullanarak ekleme yapmaktadır.
  - ACL kısaltmaları varsayılan (default) izin ayarları olmakta
  - Gerekli durumlarda belirtilmiş (explicit) ACL varsayılan ACL üzerine yazmaktadır.

IBM AIX

ÖR:

- AIX → UNIX'in IBM versiyonu
  - Klasik UNIX ACL kısaltması → **Taban İzinler (Base Permissions)**
  - Taban izinleri genişleten izinlerden oluşan bir ACL kullanır. → **Genişletilmiş İzinler (Extended Permissions)**

IBM AIX

- Grup ve kullanıcı kimliği eşleşmesini temel alır.
- Kullanıcı, kullanıcı kümesine eklenecek ya da silinecek izinleri belirler.

IBM AIX Algoritması

1. Kullanıcının taban izinlerden hangi *S* izin kümesine sahip olduğunun belirle
2. Eğer genişletilmiş izinler etkisiz (disabled) bırakılmış ise dur. *S* kümesi kullanıcının izin kümesidir.
3. Genişletilmiş izinler içerisinde bir sonraki girdiyi al. Yoksa, dur. *S* kümesi kullanıcının izin kümesidir.
4. Eğer girdi erişilmek istenen süreç ile aynı kullanıcı ve gruba sahip ise girdinin erişimi reddedip reddetmediğini belirle. Reddediyorsa dur. Erişim engellenmiştir.
5. S'yi girdideki izinlere göre düzenle.
6. 3.adıma git.

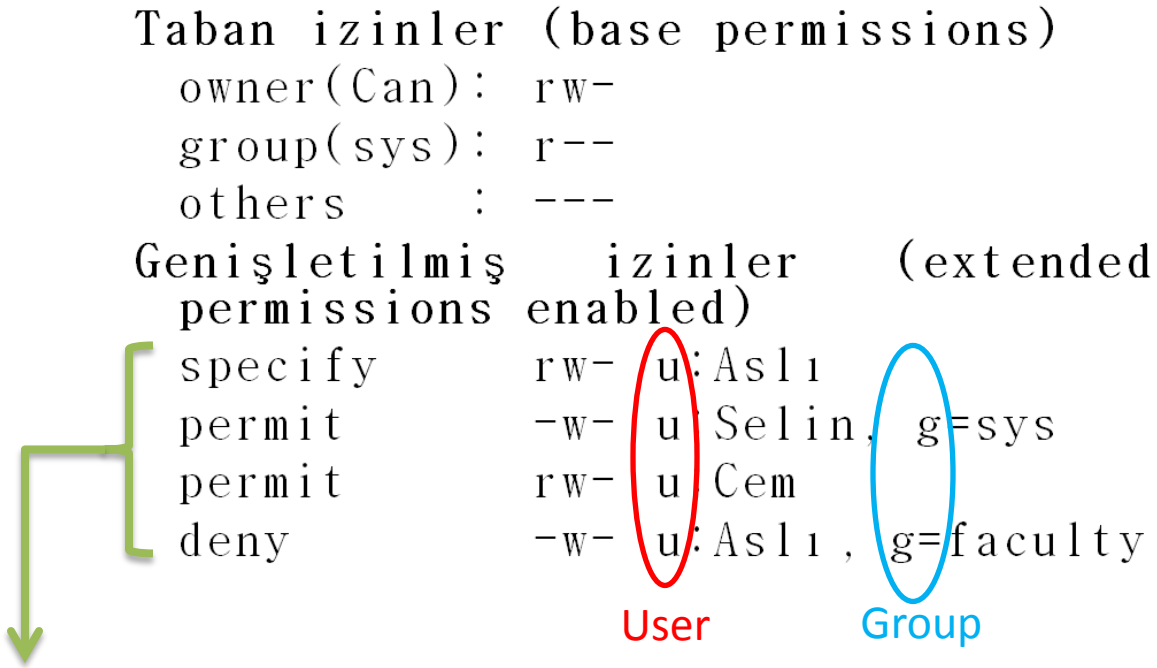
IBM AIX Algoritması

Taban izinler (base permissions)

owner(Can)	:	rw-
group(sys)	:	r--
others	:	---

Genişletilmiş izinler (extended permissions enabled)

specify	rw-	u:Aslı
permit	-w-	u:Selin, g=sys
permit	rw-	u:Cem
deny	-w-	u:Aslı, g=faculty



- **Specify** → Taban izinlerin üzerine yazılması
- **Permit** → Hakların eklenmesi
- **Deny** → Hakların silinmesi

```
Taban izinler (base permissions)
  owner(Can):  rw-
  group(sys):  r--
  others     :  ---
Genişletilmiş izinler (extended
  permissions enabled)
  specify      rw-  u:Aslı
  permit       -w-  u:Selin, g=sys
  permit       rw-  u:Cem
  deny         -w-  u:Aslı, g=faculty
```

- Aslı;
  - Dosyayı okuyabilir ve dosyaya yazabilir.
  - Eğer **`faculty`** grubunda ise dosyaya yazamaz.
- **`sys`** grubunda yer alan Selin, dosyayı okuyabilir (taban izinlerde ki grubundan dolayı) ve dosyaya yazabilir.
- Genişletilmiş izinler, taban izinlere ilave yapmaktadırlar.

ACL'lerin Yaratılması ve Yönetimi

ACL'lerin gerçekleştiriminde izlenen adımlar:

1. Hangi özne nesnenin ACL'ini güncelleyebilir?
2. Eğer ayrıcalıklık bir kullanıcı varsa (UNIX'te root gibi) ACL'ler bu kullanıcıya uygulanıyor mu?
3. ACL *grup*ları ve *wildcard*'ları destekliyor mu?
4. Tutarsız erişim denetim izinleri nasıl ele alınıyor? Özneye bir dosya üzerinde hem izin hem yasak verilmiş ise, özne hangi hakka sahiptir?
5. Eğer varsayılan ayara izin veriliyorsa, ACL izinleri bunu güncelleyebiliyor mu ya da ACL içinde özne ile ilgili bir tanım yoksa varsayılan ayar mı kullanılıyor?

Hangi özne nesnenin ACL'ini güncelleyebilir?

- Nesne yaratıldığında ACL'i yaratılmakta ve haklar oluşturulmaktadır.
- Klasik biçimde, **own** hakkına sahip olan ACL'i güncelleyebilir.
- Ancak, bazı sistemler diğer kişilerin de haklar üzerinde değişiklik yapmasına izin vermektedir.

ACL'ler ayrıcalıklı kullanıcıya uygulanıyor mu?

- Birçok sistemde ayrıcalıklı kullanıcılar bulunmaktadır.
  - UNIX → root
- Genellikle, ACL'ler ayrıcalıklı kullanıcılara sınırlı bir şekilde uygulanmaktadır.

ACL *grup*ları ve *wildcard*'ları destekliyor mu?

- Klasik biçimde, ACL'ler grupları ve wildcard'ları desteklememektedir.

- Pratikte, sistemler ACL'lerin işlenmesini kolaylaştırmak için bunları desteklemektedir.

Çelişmeler (Conflicts)

Aynı ACL içerisinde bulunan iki girdi,
özneye **farklı** izinler veriyorsa

Çelişme meydana gelir.

Çelişmeler (Conflicts)

- Herhangi bir girdi erişim hakkı veriyorsa → Sistem erişime izin verir.

- Herhangi bir girdi erişim hakkını reddediyorsa → Sistem erişime izin vermez.

ya da

- Özne ile ilgili ilk girdi geçerli olur.

Çelişmeler (Conflicts) - Örnek

AIX ACL'de;

Herhangi bir girdi erişime izin vermiyorsa

Erişim yapılamaz.

Aksi takdirde, öznenin erişimi onaylanır.

Reddetme önceliği (Denial precedence)

Çelişmeler (Conflicts) - Örnek

CISCO yönlendiricileri (routers);

Gelen paket ile ilgili ACL'deki ilk girdiyi dikkate alır.

Gelen paket ile ilgili hiçbir girdi yoksa
paket reddedilir.

Herhangi bir girdi erişim hakkını reddediyorsa → Sistem erişime izin vermez.

ACL ve Varsayılan İzinler

- Varsayılan erişim hakları mevcut ise;

1. Varsa özelleştirilmiş ACL uygulanır, özelleştirilmiş ACL yoksa varsayılan ACL girdisi uygulanır.

2. Varsayılan izinlere özelleştirilmiş ACL girdileri eklenir.

ACL ve Varsayılan İzinler - Örnek

AIX, taban izinleri güncellediğinden

Varsayılan izinlere

özelleştirilmiş ACL girdileri eklenir.

ACL ve Varsayılan İzinler - Örnek

CISCO yönlendiricisinde, paket yönlendirilmesi için ACL'de pakete izin veren bir girdi yoksa paket reddedilir.

Varsa özelleştirilmiş ACL uygulanır,

özelleştirilmiş ACL yoksa

varsayılan ACL girdisi uygulanır.

Hakların Geri Alımı

- Öznenin nesne ile ilgili haklarının ACL'den silinmesidir.
- Nesnenin ACL'inden özne ile ilgili girdi silinir.
- Sadece belirli haklar silinecekse, ACL'de özne ile ilgili girdiden bu haklar silinir.

Hakların Geri Alımı - Örnek

- Can → Cem'e *R* dosyası üzerinde okuma izni verir.
- Cem → Selin'e *R* dosyası üzerinde okuma izni verir.
- Can, Cem'in *R* dosyasını okuma iznini sildiğinde Selin'in izni de silinecektir.

Hakların Geri Alımı - Örnek

- Can → Cem'e *R* dosyası üzerinde okuma izni verir.
- Cem → Selin'e *R* dosyası üzerinde okuma izni verir.
- Aslı → Selin'e *R* dosyası üzerinde okuma izni verir.
- Can, Cem'in *R* dosyasını okuma iznini sildiğinde Selin'in izni de silinecektir. Ancak, Aslı tarafından verilen okuma izini silinmediğinden *R* dosyası üzerinde okuma izni devam edecektir.

Yetenekler (Capabilities)

- Yetenek, erişim denetim matrisinin bir satırı gibidir.

- **C-List** → Yetenek Listesi (Capability List)

- Bu liste ile ilişkilendirilmiş özne, belirtilen nesneye belirtilen haklar aracılığı ile erişebilir.

Yetenekler (Capabilities)

- **O**: Nesneler Kümesi
- **R**: Haklar Kümesi
- **c**: Yetenek Listesi, $c=\{(o, r): o \in O, r \subseteq R\}$
- **cap**: *c* listesini belirli bir *s* öznesi ile ilişkilendiren fonksiyon.

$$cap\ (s) = \{(o_i, r_i): 1 \leq i \leq n\}$$

s öznesi o_i nesnesine r_i içerisindeki herhangi bir hakkı kullanarak erişebilir.

Yetenekler (Capabilities) - Örnek

	file 1	file 2	process 1	process 2
process 1	read, write, own	read	read, write, execute, own	write
process 2	append	read, own	read	read, write, execute, own

Her özne ile ilgili C-List vardır:

cap(process1) = {(file1, {read, write, own}), (file2, {read}),
(process1, {read, write, execute, own}), (process2, {write})}

cap(process2) = {(file1, {append}), (file2, {read, own}),
(process1, {read}), (process2, {read, write, execute, own})}

Yetenekler (Capabilities)

- Yetenekler nesne kimliğini sarmalar (encapsulate).

- Nesnenin bellekteki yeri, yetenek içerisinde sarmalanmaktadır.

Yetenekler (Capabilities)

- Süreç, nesneye erişmek istediğinde, işletim sistemi nesneyi ve istenilen erişim hakkını belirlemek için yeteneği incelemelidir.

- Yetenek olmadan, süreç, nesneyi istenilen erişimi elde edecek şekilde belirtemez.

Yetenekler (Capabilities) - Örnek

UNIX'te bir dosyaya erişmek isteyen süreç:

1. Kernel'a dosyanın adını belirtir.
2. Kernel, dosya hiyerarşisinden ilgili dosyanın **inode** numarasını belirler.
3. **inode** numarası belirlendikten sonra, sistem erişim denetim izinlerini belirler.
4. Erişim onaylanırsa, işletim sistemi *dosya açıklayıcısı* (file descriptor) adında bir yetenek döndürür.
   - Yetenek, dosyaya bağlıdır.
   - Dosya silinip, aynı isimle tekrar yaratıldığında Dosya açıklayıcısı bir önceki dosyayı belirtmektedir.