

Bölüm 2: İşletim Sistemi Yapıları



BİL309 İşletim Sistemleri

Doç.Dr. Ahmet ZENGİN



Bölüm 2: İşletim Sistemlerinin Yapısı

- İşletim Sistemi Servisleri
- Kullanıcı Arayüzü
- Sistem Çağrılar
- Sistem Çağrısı Tipleri
- Sistem Programları
- İşletim Sistemi Tasarımı ve Uygulaması
- İşletim Sistemi Yapısı
- Sanal Makineler
- İşletim Sistemlerinde Hata Ayıklama
- İşletim Sistemi Üretimi
- Sistem Önyükleme

BİL309 İşletim Sistemleri

2.2



Doç.Dr. Ahmet ZENGİN



Hedefler

- Bir işletim sisteminin, kullanıcılara, processlere, ve diğer sistemlere sağladığı hizmetleri / servisleri tanıtmak
- İşletim sistemlerinin farklı yöntemlerle yapılandırılmasının tartışılması
- İşletim sistemlerinin kurulum, özelleştirilme ve önyükleme işlemlerinin açıklanması

BİL309 İşletim Sistemleri

2.3

Doç.Dr. Ahmet ZENGİN



İşletim Sistemi Servisleri

- İşletim sistemleri, programların çalışması için kullanıcılara ve programlara uygun bir çevre sağlar.
- İşletim sistemleri kullanıcılara yardımcı olacak servisler sunar. Bunlardan bazıları:
 - **User interface (Kullanıcı Arayüzü)**- Neredeyse tüm işletim sistemleri kullanıcı arayüzüne sahiptir(UI). Kullanıcının işletim sistemine komut vermesini sağlar.
 - Komut Satırı (CLI), Grafiksel Kullanıcı Arayüzü(GUI), Çalıştırılabilir Dosyalar(Batch) vb.
 - **Program execution (Uygulama Çalıştırma)**- İşletim sistemi, programları belleğe yükleyerek çalıştırabilir ve çalışmakta olan programları normal ya da anormal durumlarda (hata belirterek) sonlandırabilir.
 - **I/O operations (Giriş/Çıkış İşlemleri)**- Çalışan bir program girdi/çıkış işlemleri yapmak isteyebilir (dosyadan ya da bir giriş/çıkış aygıtından).
 - **File-system manipulation (Dosya-Sistem Değişiklikleri)**- Programlar, dosya ve dizinlerde okuma, yazma, silme, oluşturma, arama yapma, bilgi listeleme ve erişim izinlerini yönetme gibi işlemler gerçekleştirmek isteyebilir. Dosya sistemleri özellikle bu konuyla ilgilidir.

BİL309 İşletim Sistemleri

2.4



Doç.Dr. Ahmet ZENGİN



İşletim Sistemi Servisleri(Devam)

- **Communications (İletişim)** – Sistemdeki bir *process*(işlem) network üzerinden başka bir işlem ya da bilgisayarla iletişime geçmek isteyebilir.
 - › Bu iletişim, *shared memory* veya *message passing* yöntemlerinden biri ile gerçekleştirilir.
- **Error detection (Hata Tespiti)**– İşletim sistemi, oluşabilecek hatalara karşı sürekli tetikte olmalıdır.
 - › İşlemcide, fiziksel bellekte, bağlı durumdaki bir giriş/çıkış aygıtında ya da kullanıcı programında hata ile karşılaşılabilir.
 - › İşletim sistemi her türlü hataya karşı önlem almalıdır.
 - › Hata ayıklama araçları, kullanıcı ve programcının sistemi etkin olarak kullanabilmesine çok büyük katkı sağlar.

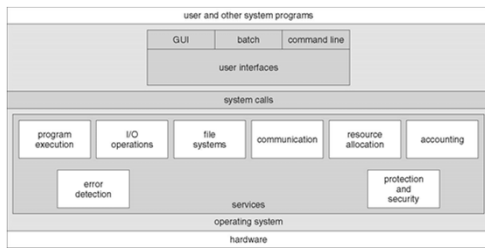


İşletim Sistemi Servisleri(Devam)

- Bazı işletim sistemi servisleri de sistemin kendi kendine etkin bir biçimde kaynak paylaşımı yapabilmesini sağlar:
 - **Resource allocation (Kaynak Tahsis)** - Birçok kullanıcı veya görevin (*task*) aynı anda çalıştığı sistemlerde, her biri için kaynak ayrılmasını sağlar.
 - (İşlemci, bellek ve sabit disk gibi) Pek çok aygıt için özel bir kaynak ayırma işlemi uygulanırken, (giriş/çıkış aygıtları gibi) bazı aygıtlarda ise daha genel bir kaynak ayırma işlemi uygulanır.
 - **Accounting** – Kullanıcıların ne kadar kaynak kullandığını takip edilmelidir.
 - **Protection and security (Koruma ve Güvenlik)** - Çok sayıda işlem eş zamanlı işletilirken, bir işlemin diğer işlemin işleyişine veya işletim sisteminin işleyişine karışmıyor olması gerekir. Bu yüzden bir işlem, başka bir işlemin bellek bölgesine erişemez.
 - › **Koruma**, sistem kaynaklarına erişim kontrollü biçimde gerçekleşmesidir.
 - › **Güvenlik**, dışarıdan gelen izinsiz talepleri engelleme gibi konuları içerir.
 - › Bir sistemin güvenlik ve koruma durumu sürekli olmalıdır. Zincir en zayıf halkası kadar güçlüdür.



İşletim Sistemi Servislerine Bakış



Kullanıcı Arayüzü - CLI

- Komut satırı arayüzü ya da **komut yorumlayıcısı** doğrudan sisteme komut verme imkanı sağlar.
 - › Komut satırından doğrudan ya da sistem servisleri vasıtasıyla işletim sistemi çekirdeğine (kernel) ulaşılır.
 - › Çoğu kez çoklu yapılar kullanılır – kabuk(**shells**)
 - › Temel olarak, kullanıcıdan komut alınır ve çalıştırılır.
 - Bazen bu girdi bir program adı bazen de yerleşik bir komut olabilir.
 - › Komutlar parametrelerle ile değişik davranış gösterir.



-

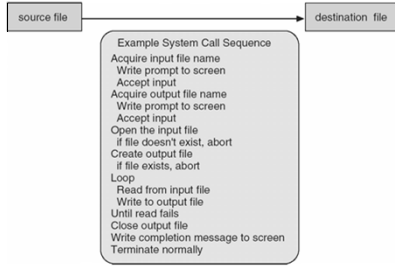


-



Sistem Çağrısı Örneği

- Bir dosyadan diğerine içerik kopyalamak için yapılan sistem çağrısı.



BIL309 İşletim Sistemleri

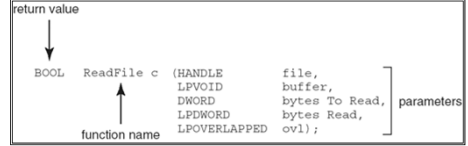
2.13

Doç.Dr. Ahmet ZENGİN



Standard API Örneği

- ReadFile() fonksiyonunu göz önüne alalım
- Win32 API—dosyadan okuma yapan bir fonksiyon



- ReadFile()'a geçen parametrelerin bir tanımı
 - HANDLE file—okuma yapılacak dosya
 - LPVOID - okunan verinin yazılacağı bellek alanı
 - DWORD bytesToRead- okunacak veri boyutu
 - LPDWORD bytesRead— en son okunan bayt miktarı
 - LPOVERLAPPED overl—çakışmalı(overlapped) I/O kullanılıp kullanılmadığını gösterir

BIL309 İşletim Sistemleri

2.14

Doç.Dr. Ahmet ZENGİN



Sistem Çağrısı Yapmak

- Her sistem çağrısının kendine ait bir numarası vardır.
 - Sistem çağrısı arayüzü, bu numaraları indexlenmiş bir tablo halinde tutar.
- Sistem çağrı arayüzü, işletim sistemi kernelindeki ilgili sistem çağrısını çağırır ve sistem çağrısının durumunu ve dönüş değerini döndürür.
- Sistem çağrısında bulunan program ya da istemci, sistem çağrısının nasıl gerçekleştiğiyle ilgilenmez.
- Yalnızca API ile iletişim kurar ve işletim sisteminin döndüreceği sonucu bekler.
 - Bu sürecin bir çok detayı API sayesinde programcıdan gizlenmiştir.
 - İşlemler, çalışma zamanı destek kütüphanesi tarafından yönetilir.(Gerekli derleyici ve fonksiyonlar bu kütüphanelerde yerleşik olarak bulunur.)

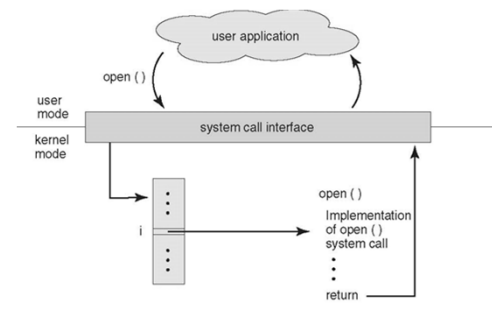
BIL309 İşletim Sistemleri

2.15

Doç.Dr. Ahmet ZENGİN



API – Sistem Çağrısı– OS ilişkisi



BIL309 İşletim Sistemleri

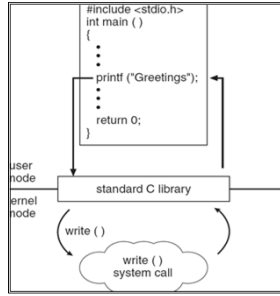
2.16

Doç.Dr. Ahmet ZENGİN



Standard C Kütüphanesi Örneği

- C programı, printf() fonksiyonu ile C kütüphanesine "write() sistem çağırısı" yapması için çağrıda bulunuyor.



BIL309 İşletim Sistemleri

2.17

Doç.Dr. Ahmet ZENGİN



Sistem Çağrılarına Parametre Geçişi

- Genellikle, istenilen sistem çağırısına ulaşmak için basit bir kimlik numarasından daha fazlası gerekir.
 - Gereken bilgi türü ve miktarı işletim sistemine ve çağrıya göre değişir.
- İşletim sistemine parametre göndermek için 3 temel metod kullanılır:
 - Basit: parametreleri register a yazma.
 - Bazı durumlarda, parametre miktarı register dan fazla olabilir.
 - Parametreler, bellek içerisinde blok ya da tabloda tutulur , ve blok adresleri register a yazılır.
 - Bu yaklaşım Linux ve Solaris'ten alınmıştır.
 - Parametreler, program tarafından, yığın (*stack*) üzerine eklenir (*push*) ve işletim sistemi yığın üzerinden alınır (*pop*).
 - Blok ve yığın metotları, sayılara ya da geçilen parametre uzunluklarına bir limit koymaz.

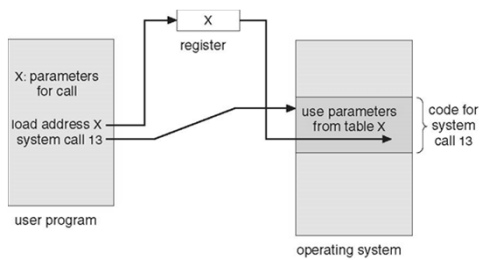
BIL309 İşletim Sistemleri

2.18

Doç.Dr. Ahmet ZENGİN



Tablo ile Parametre Geçişi



BIL309 İşletim Sistemleri

2.19

Doç.Dr. Ahmet ZENGİN



Sistem Çağrı Tipleri

- Process Kontrolü
 - Bitirme, iptal etmek
 - Yükleme, çalıştırma
 - create process (işlem oluşturma), terminate process (işlem sonlandırma)
 - get process attributes (işleme değerlerini getirme), set process attributes (işleme değer atama)
 - Belirli bir süre bekleme
 - Bekleme event'i (olayı), sinyal event'i
 - Bellek ayırma, belleği serbest bırakma
- File Management (Dosya yönetimi)
 - Dosya oluşturmak, silmek
 - Dosya açmak, kapatmak
 - Dosyadan okuma, dosyaya yazma, konum değiştirme
 - Dosya özelliklerini (attribute) değiştirmek

BIL309 İşletim Sistemleri

2.20

Doç.Dr. Ahmet ZENGİN



Sistem Çağrı Tipleri (Devam)

- Device Management (Aygıt yönetimi)
 - Aygıt talep etme, aygıtı serbest bırakma
 - Okuma, yazma, konum değiştirme
 - Aygıt değerlerini getirme, aygıt değerlerini değiştirme
 - Aygıt ekleme, kaldırma
- Information maintenance (Bilgilendirme Hizmeti)
 - Zamanı ya da tarihi getirme, değiştirme
 - Sistem verisini getirme, değiştirme
 - İşlem, dosya ya da aygıt değerlerini getirme, değer atama
- Communications (İletişim)
 - Bağlantı oluşturmak, silmek
 - Mesaj alma, gönderme
 - Durum bilgisi transferi
 - Uzak aygıtla bağlanmak, bağlantıyı sonlandırmak

BIL309 İşletim Sistemleri

2.21

Doç.Dr. Ahmet ZENGİN



Windows ve Unix Sistem Çağrı Örnekleri

	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

BIL309 İşletim Sistemleri

2.22

Doç.Dr. Ahmet ZENGİN



Örnek: MS-DOS

- Aynı anda tek bir işlem yapılabilir (Single-tasking)
- Sistem başlatılırken shell çağırılır.
- Programı başlatmak için basit bir yöntem kullanılır.
 - İşlem oluşturulmaz.
- Bellek tek bölümden oluşur.
- Loads program into memory, overwriting all but the kernel
- Programı kapat -> shell reloaded

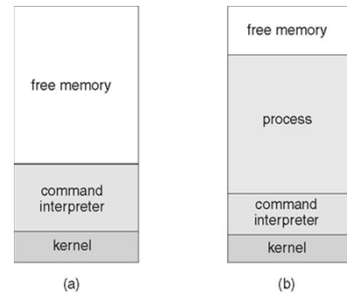
BIL309 İşletim Sistemleri

2.23

Doç.Dr. Ahmet ZENGİN



MS-DOS Yürütme



- (a) Sistem ilk başlatıldığında
(b) Çalışmakta olan program

BIL309 İşletim Sistemleri

2.24

Doç.Dr. Ahmet ZENGİN



Örnek: FreeBSD

- Unix türevi
- Aynı anda birden fazla işlem yapabilme (Multitasking)
- Kullanıcı girişi -> kabuk
- Kabuk (Shell) prosesi oluştururken fork() sistem çağrısını çalıştırır
 - Prosese programı yüklerken exec() sistem çağrısını çalıştırır.
 - Kabuk prosenin kullanıcı komutlarıyla sonlanmasını yada devam etmesinin bekler
- Prosesler 0 koduyla(hatasız) veya > 0 koduyla (hata kodu) sonlanır

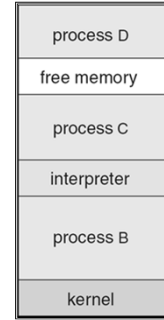
BIL309 İşletim Sistemleri

2.25

Doç.Dr. Ahmet ZENGİN



FreeBSD Üzerinde Birden Fazla İşlem Çalışırken



BIL309 İşletim Sistemleri

2.26

Doç.Dr. Ahmet ZENGİN



Sistem Programları

- Sistem programları program geliştirme ve çalıştırma için ortam sağlar. Sistem programlarını şu başlıklara ayırabiliriz:
 - Dosya işleme
 - Durum bilgisi
 - Dosya değişiklikleri
 - Programlama dili desteği
 - Program yükleme ve çalıştırma
 - İletişim
 - Uygulama programları
- Kullanıcıların çoğu gerçek sistem çağrılarını yerine sistem programlarını kullanılır.

BIL309 İşletim Sistemleri

2.27

Doç.Dr. Ahmet ZENGİN



Sistem Programları

- Sistem programları program geliştirme ve çalıştırma için ortam sağlar.
 - Bu programların bir kısmı basit bir arayüz ile sistem çağrısında bulunurken diğer kısmı ise çok daha karmaşık işlemler gerçekleştirir.
- **Dosya Yönetimi** – Oluşturma, silme, kopyalama, yeniden adlandırma, yazdırma, listeleme gibi dosya ve dizinler üzerinde gerçekleştirilen işlemlerdir.
- **Durum Bilgisi**
 - Gün, saat, kullanılabilir disk alanı, disk kapasitesi vb. bilgiler.
 - Performans, sistem günlüğü, hata ayıklama bilgileri
 - Genel olarak, bu programlar, çıktıyı biçimler ve terminalde yada diğer çıkış birimlerinde yazdırır
 - Çoğu sistem, geçmiş durum bilgilerini yeniden kullanmak amacıyla kayıt defterinde saklar.

BIL309 İşletim Sistemleri

2.28

Doç.Dr. Ahmet ZENGİN



Sistem Programları (Devam)

- **Dosya işlemleri**
 - Dosya oluşturmak ve oluşturulan dosyalar üzerinde değişiklikler yapmak üzere metin editörleri mevcuttur.
 - Dosya içeriğinde arama yapmak ya da içerik üzerinde değişiklikler yapmak için özel komutlar barındırır.
- **Programlama Dili Desteği** - Derleyiciler, yorumlayıcılar, hata ayıklayıcılar ve çeviriciler sağlar.
- **Program yükleme ve çalıştırma** – Tam yükleyiciler, bağlantı editörleri, mutlak yükleyiciler, yüksek seviyeli diller ve makine dili için derleyiciler...
- **İletişim** - Bilgisayarlar, kullanıcılar ve işlemler arasında sanal bağlantılar kuran bir mekanizma sağlar.
 - Kullanıcıların birbirlerine mesaj göndermesi, web sayfalarına ulaşması, e-mail ve dosya gönderip alması gibi işlemlerine izin verir.



BIL309 İşletim Sistemleri

2.29

Doç.Dr. Ahmet ZENGİN



İşletim Sistemi Tasarımı ve Uygulaması

- İşletim sistemi tasarımı ve uygulaması kolay olmamakla birlikte geçerliliğini koruyan başarılı tasarımlar mevcuttur.
- İşletim sistemlerinin iç yapıları büyük farklılıklar gösterebilir.
- Bu farklılıkların başında, işletim sisteminin tasarlanma amacındaki ve istenilen özelliklerdeki farklılıklar gelir.
- İşletim sistemi tasarımında seçilen donanım önemli bir etkiye sahiptir.
- Sistemden Beklenenler ve Kullanıcının Beklentileri
 - Kullanıcının beklentileri– işletim sisteminin, güvenilir, hızlı, güvenli, öğrenilmesi kolay, kullanımı kolay ve rahat olması.
 - Sistemden Beklenenler–Tasarımı kolay, uygulanabilir, sürdürülebilir ayrıca esnek, güvenilir, hatasız ve kaynakları verimli kullanan bir yapıda olması.



BIL309 İşletim Sistemleri

2.30

Doç.Dr. Ahmet ZENGİN



İşletim Sistemi Tasarımı ve Uygulaması (Devam)

- Şu iki kavramın birbirinden ayrılması çok önemlidir.
Anlayış/Politika: Ne yapılacaktır?
İşleyiş: Nasıl yapılacaktır?
- İşleyiş bir işin nasıl gerçekleştirileceğini belirtirken; anlayış, ne yapılacağını belirtir.
 - Anlayışın işleyişten ayrılması çok önemli bir ilkedir, bu ilkenin uygulanması ileride anlayış değiştirilmek istendiğinde bize çok büyük esneklik sağlar.



BIL309 İşletim Sistemleri

2.31

Doç.Dr. Ahmet ZENGİN



Basit Yapı

- MS-DOS – En az bellek kullanıp, maksimum fonksiyonelliği sağlamak üzere yazılmıştır.
 - Modüler değildir.
 - Bazı yapılara sahip olmasına karşın arayüzleri ve fonksiyonları ayrı(modüler) değildir.



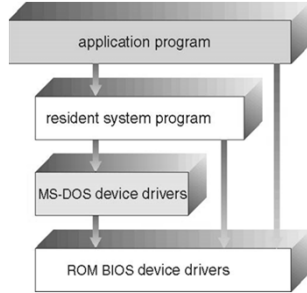
BIL309 İşletim Sistemleri

2.32

Doç.Dr. Ahmet ZENGİN



MS-DOS Katman Yapısı



BIL309 İşletim Sistemleri

2.33

Doç.Dr. Ahmet ZENGİN



Katmanlı Yaklaşım

- İşletim sistemleri katmanlara bölünmüştür, her katman alt katmanların üzerine inşa edilmiştir. En alt katman (0. katman), donanım; en üst katman (N. katman) ise kullanıcı arayüzüdür.
- Modülerlik ile, seçilen katman yalnızca alt katmanlara ait servis ve fonksiyonları kullanır.

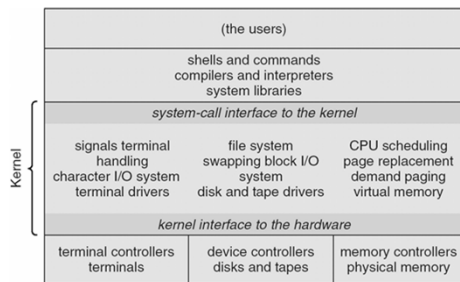
BIL309 İşletim Sistemleri

2.34

Doç.Dr. Ahmet ZENGİN



Geleneksel UNIX Sistem Yapısı



BIL309 İşletim Sistemleri

2.35

Doç.Dr. Ahmet ZENGİN



UNIX

- UNIX – donanım sınırlaması nedeniyle orijinal UNIX kısıtlı bir yapıya sahiptir. UNIX OS'i iki kısma ayırabiliriz:
 - Sistem Programları
 - Çekirdek (Kernel)
 - Sistem çağrıları ile donanım arasındaki her şeyi ifade eder.
 - Bellek yönetimi, kaynak ayrılması, işlemci planlaması ve diğer işletim sistemi görevlerini yapar, ilk katman için çok sayıda fonksiyon yerine getirir.

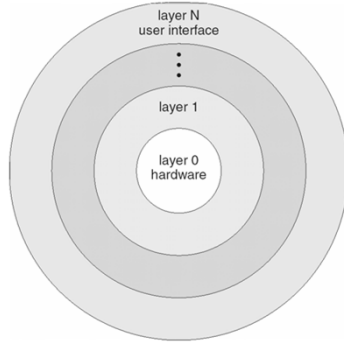
BIL309 İşletim Sistemleri

2.36

Doç.Dr. Ahmet ZENGİN



Katmanlı İşletim Sistemi



BIL309 İşletim Sistemleri

2.37

Doç.Dr. Ahmet ZENGİN



Microkernel Sistem Yapısı

- Kernel'deki 'fazlalıkları' olabildiğince kullanıcı (*user*) alanına taşır.
- Kullanıcı modülleri arasında iletişim, mesaj geçişi yöntemi ile sağlanır.
- Artıları:
 - Mikrokerneli genişletmek kolaydır.
 - İşletim sistemini yeni mimarilere taşıması kolaydır.
 - Kernel modda çalışan kod azaldığı için daha güvenilirdir.
 - Daha güvenli
- Eksileri:
 - Kullanıcı alanı ile kernel alanı arasındaki iletişimden kaynaklanan performans kaybı

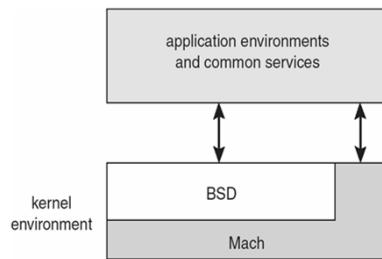
BIL309 İşletim Sistemleri

2.38

Doç.Dr. Ahmet ZENGİN



Mac OS X Yapısı



BIL309 İşletim Sistemleri

2.39

Doç.Dr. Ahmet ZENGİN



Modüler Yapı

- Günümüz işletim sistemleri kernel modülleri uygulamaktadır.
 - Nesne yönelimli yaklaşım hakimdir.
 - Her bir çekirdek bileşeni bağımsızdır.
 - Her modül birbiriyle bilinen arayüzler üzerinden konuşur.
 - Her biri kernel içerisinde ihtiyaç duyulması halinde yüklenebilir.
- Genel olarak, katmanlı yapıya benzer fakat çok daha esnek.

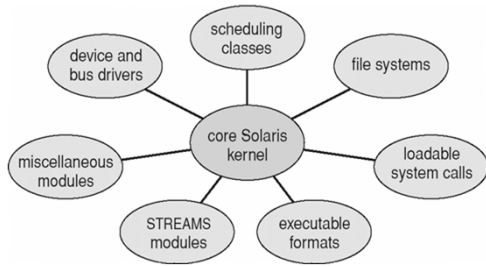
BIL309 İşletim Sistemleri

2.40

Doç.Dr. Ahmet ZENGİN



Solaris Modüler Yaklaşımı



BIL309 İşletim Sistemleri

2.41

Doç.Dr. Ahmet ZENGİN



Sanal Makineler

- Sanal bir makine (**virtual machine**) katmanlı yaklaşımının mantıksal (logical) bir sonucudur.
- Donanım + İşletim Sistemi Çekirdeği = Donanım
- Sanal bir makine alttaki donanıma benzer bir arayüz sağlar
- Ev sahibi işletim sistemi bir illüzyon yapar.
 - Kendi işlemcisi ve sanal belleği olan bir proses oluşturur
- Her bir konuk işletim sistemine donanımın sanal bir kopyası sağlanır.

BIL309 İşletim Sistemleri

2.42

Doç.Dr. Ahmet ZENGİN



Sanal Makineler Tarihi ve Faydaları:

- İlk kez 1972 yılında IBM'in anabilgisayarlarında kullanıldı.
- Tek bir fiziksel donanım üzerinde farklı işletim sistemleri yürütme imkanı sunar.
- Kurulan her işletim sistemini diğerlerinden izole eder.
- Dosya paylaşımına kontrollü olarak izin verilebilir.
- Ağ altyapısı yardımıyla birbirleriyle veya diğer fiziksel sistemlerle haberleşir
- Test ve geliştirme işlemleri için kullanışlıdır.
- Daha düşük kaynak kullanan sistemlerin daha az meşgul sistemler üzerinde çalışma olanağı sağlar
- "Açık Sanal Makine Biçimi", çok sayıda farklı sanal makine çalıştırmanıza izin verir.

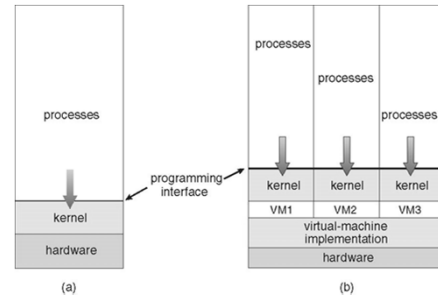
BIL309 İşletim Sistemleri

2.43

Doç.Dr. Ahmet ZENGİN



Sanal Makineler (Devam)



(a) sanal olmayan makine (b) sanal makine

BIL309 İşletim Sistemleri

2.44

Doç.Dr. Ahmet ZENGİN



Para-virtualization

- Donanıma benzer ama tipkısının aynısı olmayan konuk işletim sistemleri
- Donanım üzerinde çalıştırabilmek için konuk işletim sistemi de modifiye edilmelidir.
- Konuk bir işletim sistemi olabilir, ya da "taşıyıcı - container" içinde çalışan Solaris 10 uygulamaları da olabilir.



BIL309 İşletim Sistemleri

2.45

Doç.Dr. Ahmet ZENGİN



Sanallaştırma

- Sanallaştırma zordur—temel işletim sisteminin bir yedeğinin alınması gerekir.
 - Genellikle, kullanıcı modunda çalışır. Sanal kullanıcı modu ve sanal çekirdek modu oluşturulur.
- Gerçek işletim sisteminden daha yavaş çalışma problemi olabilir.
- Donanım desteği gerekir.
 - Daha fazla destek -> daha iyi sanallaştırma
 - i.e. AMD "host" ve "guest" modları sağlar.



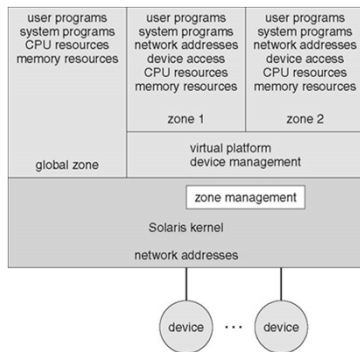
BIL309 İşletim Sistemleri

2.46

Doç.Dr. Ahmet ZENGİN



Solaris 10 with Two Containers



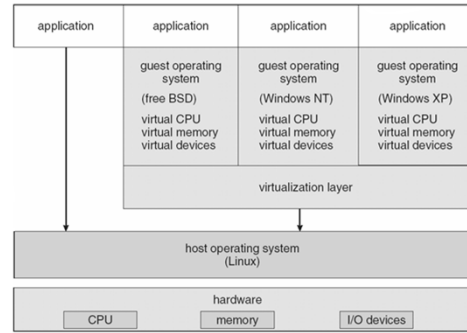
BIL309 İşletim Sistemleri

2.47

Doç.Dr. Ahmet ZENGİN



VMware Mimarisi



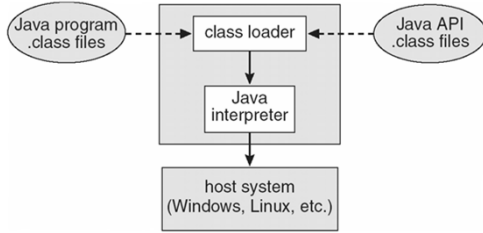
BIL309 İşletim Sistemleri

2.48

Doç.Dr. Ahmet ZENGİN



Java Sanal Makinesi



BIL309 İşletim Sistemleri

2.49

Doç.Dr. Ahmet ZENGİN



İşletim sisteminde Hata Ayıklama

- **Hata yakalama - Debugging** hataları ve böcekleri(bug) bulma ve ayıklama işlemidir.
- İşletim sistemleri hata bilgilerini içeren **log dosyaları** oluşturur.
- Bir uygulamanın hatası prosesin bellek görüntüsünü tutan bir dosya oluşturabilir (**core dump file**). capturing memory of the process
- İşletim Sistemi hatası çekirdek bellek görüntüsünü taşıyan bir dosya üretebilir (**crash dump file**)
- Çöküşlerden sonra, sistem performansı optimize edilir.
- Kernighan's Kuralı: "Hata ayıklama kod yazmaktan iki kat zordur. Bu yüzden, kodu zekice tasarlayıp yazabilirsin, ama hata ayıklamak için yeteri kadar akıllı olmayabilirsin"
- FreeBSD, Mac OS X, Solaris içindeki DTrace aracı, kod çalışırken veriyi takip ederek hataları yakalar

BIL309 İşletim Sistemleri

2.50

Doç.Dr. Ahmet ZENGİN



Solaris 10 dtrace Sistem Çağrısı Gönderiyor

```

# ./all.d 'pgrep xclock' XEventsQueued
dtrace: script './all.d' matched 52377 probes
CPU FUNCTION
0 -> XEventsQueued U
0 -> XEventsQueued U
0 -> _XltTransBytesReadable U
0 <- _XltTransBytesReadable U
0 -> _XltTransSocketBytesReadable U
0 <- _XltTransSocketBytesReadable U
0 -> ioctl U
0 -> ioctl K
0 -> getf K
0 -> set_active_fd K
0 <- set_active_fd K
0 <- getf K
0 -> get_udatamodel K
0 <- get_udatamodel K
...
0 -> releasef K
0 -> clear_active_fd K
0 <- clear_active_fd K
0 -> cv_broadcast K
0 <- cv_broadcast K
0 <- releasef K
0 <- ioctl K
0 <- ioctl U
0 <- _XEventsQueued U
0 <- XEventsQueued U
  
```

BIL309 İşletim Sistemleri

2.51

Doç.Dr. Ahmet ZENGİN



Operating System Generation

- İşletim sistemleri her donanım üzerinde çalışabilecek biçimde tasarlanmıştır; sistem, her bilgisayarda çalışacak biçimde konfigüre edilmelidir.
- Bu amaçla SYSGEN programı, donanım sistemine ilişkin bilgi edinir.
- **Bootling** – bilgisayarın kernel yüklenerek başlatılması işlemidir.
- **Bootstrap program** – Rom bellekte saklanan bir kod parçasıdır ki bu kod parçası kernelin yerini belirleyip, belleğe yükler ve kerneli başlatır

BIL309 İşletim Sistemleri

2.52

Doç.Dr. Ahmet ZENGİN



Sistem Önyükeme

- İşletim sistemi, donanımın üzerinde hazır olmalıdır ki, donanım onu başlatabilsin.
 - **bootstrap loader**(küçük bir kod parçası), kerneli bulur, belleğe yükler ve çalıştırır.
 - Bazen iki aşamada gerçekleşir sabit konumdaki **boot block**, **bootstrap loader**'i yükler ve bootstrap çalışır.
 - Sisteme güç verildiğinde çalışmaya belli bir bellek konumundan başlanır.
 - Firmware, ilk önyükeme kodlarını tutmak için kullanılır.



Bölüm 2 Sonu

