

Linux Performansını Arttırmak

S.Çağlar Onur
<cağlar@pardus.org.tr>

- Performans neden önemlidir?
- Performansı arttırmak:
 - Hızlı çekirdek
 - Hızlı sistem açılışı
 - Hızlı uygulama açılışı
- Performanslı kod yazmak
- Performans araçları

Performans Neden Önemlidir?

- Herkesin süper, hızlı, son model bilgisayarını yok!
- Uzun süre açılmayan her uygulama/sistem kullanıcının canını sıkıyor!
- Aynı iş daha az kaynak ile yapılabilir!
- Herkesin çok zamanı yok!

Performansı Arttırmak



Hızlı Çekirdek

- Çekirdek mesajlarının konsola basılması zaman alıyor,
- Hele framebuffer kullanılıyorsa,
- Çekirdeğe açılırken sessiz ol demek;
 - `root=dev/hda1 quiet`
- - **Sistem açılış zamanını %30 ile %50 arası hızlandırıyor!**

Hızlı Çekirdek

- Her açılışta çekirdek gecikme döngüsünü hesaplar
- ~25 jiffy (1/saat hızı) sürer
- **dmesg | grep Bogomips**
- **Calibrating delay using timer specific routine.. 3464.74 Bogomips (lpj=1732372)**
-
- **Çekirdeğe lpj değerini vermek bu süreyi kazandırır**

Hızlı Çekirdek

- IDE disklerin tanınması ~5sn. sürer,
- Bu sürenin önemli bir kısmı msleep(50) ler ile geçer (drivers/ide/ide-probe.c),
- Bu süre ufak bir çekirdek yaması ile;
 - **~3 sn. kısaltılabilir**

- Farklı ihtiyaçlar için farklı I/O scheduler (elevator);
 - noop: Basit FIFO, her I/O için minimum CPU kullanımı
 - deadline: 5 adet I/O queue arasında round robin
 - anticipatory: deadline + kontrol edilen gecikme (6 ms.)
 - **cfq**: Process başına I/O queue, mevcut tüm I/O bantgenişliğini eşit dağıtır.

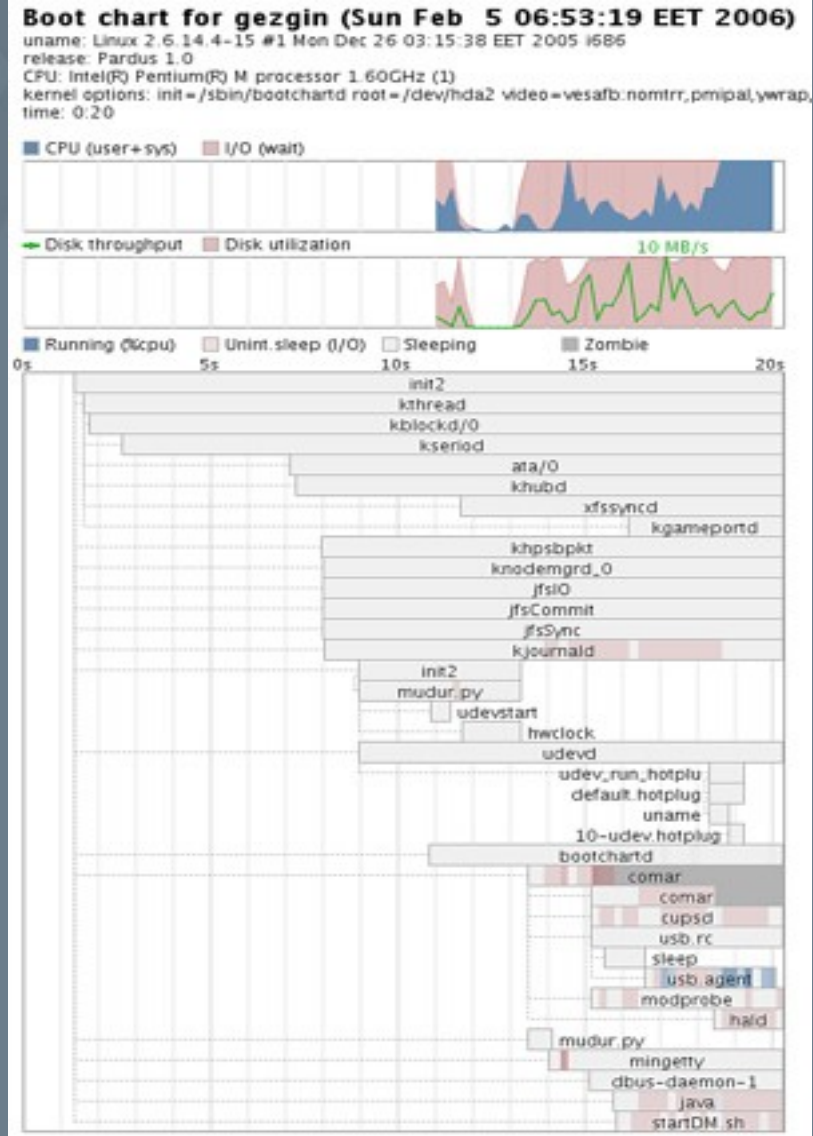
Hızlı Sistem Açılışı



Hızlı Sistem Açılışı

- Sistem açılışları yavaş,
- En büyük sebebi ihtiyaç duyulmayan geriye uyumluluk için yazılan kötü kod,
- Bir diğer sebebi yazılan kötü kod,
- Basit bir süreci karışıklaştırma sorunu,
- Kabuk kullanımı, paralel çalıştıramama, sıraya koyamama

Hızlı Sistem Açılışı



Hızlı Uygulama Açılışı



Hızlı Uygulama Açılışı

- Kullanılabilirlik testlerine göre kullanıcı uygulamayı çalıştırdıktan sonra $\sim 0.3 - 0.5$ sn arasında pencereyi görmek istiyor
- Uygulama açılma zamanının bir kısmı linker (ld.so) tarafından harcanıyor
- Çoğunlukla çalışma zamanı link etme işlemi en uzun süren süreç

Hızlı Uygulama Açılışı

- Örnek bir KD uygulamasında;
 - **LD**: 0.65 sn.
 - **Qt**: 0.13 sn.
 - **KDE**: 0.11 sn.
 - **KWrite**: 0.48 sn.
 - Pencerenin gelmesi toplam 1.37 sn. sürüyor
 - (Sayılar güncel olmayabilir)

Hızlı Uygulama Açılışı

- Sorunun büyük kısmı ld'de;
 - Symbol arama süreci uzun sürüyor (özellikle C++ uygulamalarında)
 - Relocation (ikili dosyanın içindeki çağrılarının dışardan çözülen kütüphane çağrılarını gösterme süreci) işlemi bellek tüketici, zaman alıcı
 - Lazy Binding (gerektiğinde çöz) C uygulamaları için mükemmele yakın çalışırken C++ için yavaş

Hızlı Uygulama Açılışı

- Prelink
 - Tüm sistemi tara ve tüm uygulama/kütüphanelere belirlenmiş adres alanları ver
 - Her kütüphane/uygulama değiştiğinde bu değerler yeniden hesaplanmalı
- Preload
 - Uygulamanın ihtiyaç duyacağı kütüphaneleri uygulama açılmadan çok önce belleğe yükle

Hızlı Uygulama Açılışı

- -Bdirect
 - Sembol arama sürecini yok et/azalt!
- -dynsort
 - Dinamik link bölümünü sırala, link işlemi!
sırasında önbellek kullanımını azalt
- -hashvals
 - Önceden hesaplanmış hashleri sakla,
ihtiyaç olunca buraya bak!
 - (Micheal Meeks)

Performanslı Kod Yazmak



- Kod yazma sırasında performans;
 - C kodları için likely, unlikely, inline kullanımı,
 - C++ için vtablelara dikkat!,
 - Class Base: 10 virtual fonksiyona sahip
 - Class Derive (Base): 1 virtual fonksiyonu override ediyor
 - base = 10 fonk. = 10 relocation
 - derive = 20 fonk. = 20 relocation
 - liboil ve benzeri çözümler
 - BoehmGC

- Derleme sırasında performans;
 - Static linkleme
 - Shared linkleme
 - GCC optimizasyonları (mtune, -O2 | -Os v.s)
 - Binutils ile -as-needed
 - GCC 4.x ile visibility
 - Gereksiz sembolleri strip etmek

Performanslı Kod Yazmak

- Derlenmeyen diller için performans;
 - Python!!!
 - JIT derleyicileri (Python ve Pysco)
 - Az karışıklık, az kod, yüksek soyutlama
 - GC (çöp toplayıcılar)
 - Optimize edilmiş veya derlenmiş hallerinin çalıştırılması (pyc, pyo)

Performans Araçları



- Basit araçlar;
 - ps
 - Çalışan süreçlerle ilgili bilgi verir
 - top
 - Çalışan süreçlerle ilgili gerçek zamanlı bilgi verir
 - vmstat
 - Bellek, sayfalama, I/O ve CPU aktivitesi ile ilgili bilgi verir
 - ...

Performans Araçları

- Uygulama özel araçları;
 - strace
 - ltrace
 - ld.so
 - oprofile
 - valgrind / cachegrind

strace

- Çekirdeğe yapılan sistem çağrılarını gösterir
- `$ strace -c ls`
- `Ağ.desktop build-from-tarballs.sh eclipse.desktop Home.desktop sandbox
svn sysinfo.desktop trash.desktop`
- | <code>% time</code> | <code>seconds</code> | <code>usecs/call</code> | <code>calls</code> | <code>errors</code> | <code>syscall</code> |
|---------------------|----------------------|-------------------------|--------------------|---------------------|----------------------|
| 52.73 | 0.000029 | 1 | 48 | 21 | <code>open</code> |
| 47.27 | 0.000026 | 1 | 41 | | <code>mmap2</code> |
| ... | | | | | |
| 0.00 | 0.000000 | 0 | 12 | | <code>read</code> |
| 100.00 | 0.000055 | | 201 | 30 | <code>total</code> |

ltrace

- strace ile aynı konseptte sahiptir ama kütüphane çağrılarını gösterir
- `$ ltrace -c ls`
- `Ağ.desktop build-from-tarballs.sh eclipse.desktop Home.desktop sandbox
svn sysinfo.desktop trash.desktop`
- | <code>% time</code> | <code>seconds</code> | <code>usecs/call</code> | <code>calls</code> | <code>function</code> |
|---------------------|----------------------|-------------------------|--------------------|-----------------------|
| 23.40 | 0.002014 | 2014 | 1 | qsort |
| 19.33 | 0.001664 | 1664 | 1 | setlocale |
| ... | | | | |
| 0.17 | 0.000015 | 15 | 1 | textdomain |
| 100.00 | 0.008608 | | 313 | total |

- link süreçlerini gösterebilir!
- env LD_DEBUG=statistics konqueror
- 6062: runtime linker statistics:
- 6062: total startup time in dynamic loader: 65974640 clock cycles
- 6062: time needed for relocation: 61836996 clock cycles (93.7%)
- 6062: number of relocations: 24547
- 6062: number of relocations from cache: 63115
- 6062: number of relative relocations: 37780
- 6062: time needed to load objects: 3813029 clock cycles (5.7%)
- 6062: runtime linker statistics:
- 6062: final number of relocations: 37804
- 6062: final number of relocations from cache: 75994

- LD_DEBUG=all tüm sembol ismi çözme sürecini gösterir (LD_DEBUG_OUTPUT)
- Örneğin konqueror için bu işlemi dosyaya yazdırınca;
-
- **env LD_DEBUG=all LD_DEBUG_OUTPUT=konqi konqueror**
- **\$ ll -h konqi.6065**
- **-rw-r--r-- 1 caglar users 33M May 11 11:54 konqi.6065**

OProfile

- Sisteminizin profilcisi,
- Düşük performans farkı ile çalışır,
- İşlemcilerden gelen özel performans olaylarını(performance counter events) dinler,
- Ayrıntılı rapor, özet, istatistik bilgi ve grafik üretebilir,
- Grafik arabirimi ile kolayca kullanılabilir

OProfile

- `# opcontrol --vmlinux=/usr/src/linux/arch/i386/boot/compressed/vmlinux`
- `# opcontrol --start`
- Using default event: `CPU_CLK_UNHALTED:100000:0:1:1`
- Using 2.6+ OProfile kernel interface.
- Reading module info.
- Using log file `/var/lib/oprofile/oprofiled.log`
- Daemon started.
- Profiler running.
-

OProfile

- `zangetsu ~ # oprofile`
- CPU: PIII, speed 800 MHz (estimated)
- Counted CPU_CLK_UNHALTED events (clocks processor is not halted) with a unit mask of 0x00 (No unit mask) count 100000
- CPU_CLK_UNHALT...|
- samples| %|
- -----
- 710691 30.3535 vmlinux
- 283530 12.1095 libfb.so
- 196727 8.4022 libc-2.3.5.so
- 125992 5.3811 libqt-mt.so.3.3.5
- 102543 4.3796 Xorg
- 85589 3.6555 mono
-

Valgrind

- Uygulamanın bellek kullanımını, kaçakları (memory leak) gösterir/bulunmasına yardımcı olur
- Çok gelişmiş bir uygulamadır
- Cachegrind KDE arayüzü Kcachegrind ile uygulama profilleme işlemi grafik arabirim ile yapılabilir, ayrıntılı rapor, çizelge, çizim elde edilebilir

Diğer Performans Arttırmaları

- Suspend to Disk/RAM
 - Hızlı açılış
- Cpufreq
 - Az pil tüketimi
- kexec
 - Hızlı reboot
- Hızlı dosya sistemi
 - Hızlı I/O

Kaynaklar

- Sunum:
 - Fosdem 2005, Embedded Linux Optimization sunumu,
- Kitap:
 - Optimizing Linux Performance, Philip G. Ezolt
- Döküman:
 - KDE Init, Dirk Muller
- Blog:
 - Optimized Code, Robert Love

Linux Performansını Arttırmak

Teşekkürler!

...Sorular...