

The logo for Progresswiz, featuring the word "Progresswiz" in a blue, sans-serif font. The letters are slightly shadowed, giving it a 3D effect. The logo is positioned in the upper right corner of the slide.

Progresswiz

Understanding OpSys Perf Metrics

PAUL KOUFALIS
PRESIDENT
PROGRESSWIZ INFORMATIQUE



Why Are You Here?

- Too much *conflicting* information on the Internet
 - If it's on the Internet, it must be true!
- Some of this stuff is counter-intuitive
- What other reasons?

Paul Koufalis? Who is Paul Koufalis?

- Progress DBA and UNIX sysadmin for almost 20 years
- Wide range of clients
 - Small 10 person offices
 - Mfg/distributors with 1000's of users
- Each have their own set of challenges

Menu du jour

- CPU
- Memory
- Disk I/O
- Network (very briefly)
- Emphasis on AIX and Linux
 - 90% of my client base
- Solaris, HPUX and even Windows: same concepts
 - I haven't touched a Solaris box in years...



Common Tools

- AIX & Linux: nmon
 - Get rid of that topas garbage!
- HPUX: glance
- Solaris: Also glance? GlancePlus? nmon?
 - Sorry not sure
- Windows
 - PerfMon, wmic, PowerShell, Process Explorer (SysInternals Tools)

About Windows

- Concepts are *mostly* the same
- Just the tools are different
 - WMIC is a perfect example
- A note on performance:
 - I *used* to be of the opinion that Win2K8 64 bit was close to Linux
 - Recent customer issues => not sure anymore

Progresswiz ■ We'll see...

Windows wmic

- All kinds of cool stuff you can do

```
C:\Users\Paul>wmic process where "commandline like '%toto%'" get processid,commandline
CommandLine                                     ProcessId
_mprosrv toto -S 2500                             5840
```

- Four categories of CPU use
 - User
 - Sys
 - Wait
 - Idle

CPU

- User
 - This is real application work
 - Should be bulk of CPU use
- Sys
 - Kernel stuff like context switches
- Wait I/O
 - See next slide
- Idle
 - Doing nothing

CPU – Wait I/O

- Wait I/O is a kind of idle time – no useful CPU work is being done
 - Process is waiting for disk I/O to complete
- This is NOT necessarily a bad thing
- This does NOT necessarily point to a disk bottleneck

Interpreting CPU Use

- Regular load should be 70-75-80%
 - NOT 100% use all the time!
 - That way you can handle occasional peaks
- At 75-80%:
 - 60% User + 10% Sys + 10% Wait I/O is nice
 - This is NOT a target
 - Just a general feel-good spread of CPU use
- Current large AIX customer (>1000 users)
 - Peak = 85% user + 12% sys + 1-2% WIO

CPU – Mais ce n'est pas tout!

- Four additional metrics to watch
 - Context switches
 - Run queue
 - Load average
 - Forks (*NEW* for 2013!)
 - This was never on level 1 radar until recently
 - UNIX SILENT more costly in 11.x on AIX

Context Switches

- A process only gets to use the CPU for a short period of time
 - Has to take it's stuff with it when it leaves
 - AND bring it back too!
- This is a context switch
- Need CPU cycles to save and load context
- vmstat “cs” column



Run Queue

- Processes that are running or ready to run
- AIX & Linux have one run queue per processor
 - Actually one queue per priority per cpu...
 - Processes tend to stay on the same processor
 - Queues periodically rebalanced
- Blocked (waiting for I/O) processes in blocked queue
- See “r” and “b” columns in vmstat

vmstat – cpu info

```
$ vmstat 5 5
procs  ---- mem-swap-  --io--   ----sys--  ----cpu----
  r   b  swpd  ...  si  so  bi  bo    in    cs  us  sy  id  wa
31  0    0    0  0  0 126 424   255   238 12  0 85  3
30  0    0    0  0  0  0  8 3902   288 95  2  3  0
16  0    0    0  0  0  0 11 3918   300 96  2  3  0
```

Load Average

- Generally the average number of runnable processes over the past x minutes
 - Reported over past 1 min, 5 min and 15 min

```
$ uptime  
01:38:13 up 3:58, 3 users, load average: 15.54,  
10.51, 8.00
```

- Count 1.00 units per CPU for 100%
 - 5.00 is worrisome on 2 CPU system
 - 5.00 is ok on 8 CPU system

“nice” level

- START priority on Windows
 - CAREFUL: Default priority VERY LOW in Task Scheduler
- Play with this at your own risk! (I don't)
- Higher priority = lower nice level: -20 to +19
- Higher priorities get
 - More **AND** longer CPU time splices
- CPU schedulers automatically adjust nice
 - Otherwise low priority tasks may never get CPU time

CPU Alerts

- High CPU and high relative sys% or wio%
- Run queue > 4X #CPU
 - If higher = contention = problem?
- CS are a symptom, not a cause
- Load avg
 - 1 and 5 min just show workload peaks
 - If 15 min consistently high then problem

OpenEdge Example

- Using 20-30 rapid-reader processes
 - FOR EACH ...NO-LOCK
- Four different tables with a few million records each
- -B 100 (yes – one hundred)
- AWS m3.xlarge
 - 4 cores + 15 Gb RAM
 - 1000 IOPS provisioned storage

OpenEdge Example cont...

- Notice no I/O wait
 - DB blocks are in the FS cache
- Now...let's stir up the hornet's nest:

```
sync ; echo 3 > /proc/sys/vm/drop_caches
```
- This empties the FS cache
 - Linux only
 - Other OS: umount/mount FS
 - Or large FS operation (backup)
 - Win: Dynamic Cache Service (older Win)
 - MaxSystemCacheMBytes (I never tried it)
- Forces processes to go back to disk

OpenEdge Example cont...

- Repeat with only 3-4 processes
 - So as not to saturate CPU

Memory

- Ahh...misunderstood memory
- First thing's first:

A modern operating system will always try to use 100% of available memory

- If processes don't want it, the FS cache will take it

Two Kinds of Memory

- Real, physical memory
 - The real RAM installed in the box
- Swap space/paging space
 - Disk space formatted to act as memory in case you run out of physical
- Together = Virtual Memory

Real vs. Swap

- Swap space = credit card debt
 - Sometimes you need it for a few cycles
 - Really you shouldn't touch it EVER
 - Very expensive (29.9% APR)
- Only spend money (memory) you have!

Computational vs. non-computational

- AIX terminology
- Computational memory pages hold working data
- Non-computational memory is files loaded from disk
 - 100 ChUI users do not each have their own copy of \$DLC/bin/_progres in memory

Swapping vs. Paging

- *Swapping* officially means ejecting an entire process to disk
 - The area on disk is called the swap space or paging file
- *Paging* refers to ejecting selected 4K memory pages

Swapping vs. Paging Today

- Often used interchangeably
- *Generally:*
 - Swapping refers to page in/out to the swap space/paging file
 - Paging refers to page in/out to the file system

Bored Yet?

- Simple recommendation: no swapping/paging to swap/paging space
 - “si” and “so” columns in vmstat
 - AIX: watch fr/sr
 - If very active there could be bottleneck at FS cache
- This is *not* the same as paging to file system
 - Kinda important...

Let's See it in Action

- 30 rapid reader _progres
- -B 100
- 4 CPU + 15 Gb RAM AWS m3.xlarge

Stats

Memory Stats

	RAM	High	Low	Swap	Page Size=4 K
Total MB	15360.1	0.0	15360.1	500.0	
Free MB	10673.0	0.0	10673.0	500.0	
Free Percent	69.5%	0.0%	69.5%	100.0%	
		Cached=	4294.4	Active=	3883.9
Buffers=	6.1	Swapcached=	0.0	Inactive =	468.0
Dirty =	0.2	Writeback =	0.0	Mapped =	14.8
Slab =	18.1	Commit_AS =	157.1	PageTables=	4.7

Virtual-Memory

nr_dirty =	39	pgpgin	=	1256	High	Nrml	DMA
nr_writeback=	0	pgpgout	=	0	alloc	0	0 754
nr_unstable =	0	pgpswpin	=	0	refill	0	0 0
nr_table_pgs=	1206	pgpswpout	=	0	steal	0	0 0

Stats (cont...)

Kernel Stats

```
RunQueue          31   Load Average CPU use since boot time
ContextSwitch    405.5  1 mins 20.81 Uptime D=0 Hr=17 Min=20
Forks            0.0   5 mins 11.68 Idle   D=0 Hr=16 Min=24
Interrupts       3928.3 15 mins  7.19 Average CPU use= 5.43%%
```

```
DiskName Busy  Read WriteKB|0      |25      |50      |75      |100|
sda1      2%   624.2   0.0|R
Totals Read-MB/s=0.6  Writes-MB/s=0.0 Transfers/sec=21.9
```

Stats after drop_cache

Memory Stats

	RAM	High	Low	Swap	Page Size=4	KB
Total MB	15360.1	0.0	15360.1	500.0		
Free MB	14777.8	0.0	14777.8	500.0		
Free Percent	96.2%	0.0%	96.2%	100.0%		
		Cached=	204.8	Active=		139.3
Buffers=	0.5	Swapcached=	0.0	Inactive =		117.1
Dirty =	0.0	Writeback =	0.0	Mapped =		14.8
Slab =	8.8	Commit_AS =	157.2	PageTables=		4.7

Virtual-Memory

nr_dirty =	0	pgpgin =	31060	High	Nrml	DMA
nr_writeback=	0	pgpgout =	0	alloc	0	0 8187
nr_unstable =	0	pgpswpin =	0	refill	0	0 0
nr_table_pgs=	1203	pgpswpout =	0	steal	0	0 0

Stats after drop_cache

```
RunQueue            31  Load Average CPU use since boot time
ContextSwitch    953.4  1 mins  16.77 Uptime Ds=0 Hr=17 Min=26
Forks            0.0  5 mins  14.93 Idle   Ds=0 Hr=16 Min=25
Interrupts      4497.3  15 mins 10.21 Average CPU use=  5.83%%
Disk I/O —
DiskName Busy  Read WriteMB|0    |25    |50    |75
100|
sda1      46%   30.1   0.0|RRRRRRRRRRRRRRRRRRRRRRRRRRRR
Totals  Read-MB/s=30.1  Writes-MB/s=0.0  Transfers/sec=873.1
```

Take Action

- Older AIX (pre 5.3)
 - Minperm 5% Maxperm 10%
 - Tell FS cache to use no more than 10%
- Modern AIX
 - Set `Iru_file_repage = 0`
 - Default in 6.1 +
 - Tells VMM to ALWAYS steal FS cache first
- Linux favours FS pages by default
- Windows (pre 2008R2)
 - Dynamic Cache Service

Take Action

- Don't worry about “free” memory
- FS cache over 10% physical RAM is “free”
 - Processes can easily steal it back
- Linux

```
$ free -m
```

	total	used	free	shared	buffers	cached
Mem:	15360	4390	10969	0	5	4036
-/+ buffers/cache:		348	15012			
Swap:	499	0	4991			

- 15 Gb really free even though 4.4 Gb “used”
- AIX: $(\text{numperm}\% - 10) = \text{“freeable” FS cache \%}$

ps lies

```
$ ps aux
USER          PID %CPU %MEM    VSZ   RSS  COMMAND
root          9753  1.0   7.5 1213036 1186432  _mprosrv
root          9759 16.6   2.8 1224036 454452  _progres
root          9760 19.1   3.0 1224036 474504  _progres
root          9761 16.1   2.5 1224032 395552  _progres
root          9762 12.0   2.0 1224040 318968  _progres
root          9765 12.0   2.0 1224032 318996  _progres
...
```

- 30 _progres using 1.2 Gb each (Virtual set)
 - Or 450 Mg each (Resident set)
- Free tells me that only 1 Gb is really being used

Windows

```
PS> get-wmiobject -class
win32_PerfRawData_PerfProc_Process -filter "name like
'%prowin32%'" | select name,workingset,workingsetprivate
name      workingset  workingsetprivate
prowin32  18661376    5996544
```

```
C:\>wmic process where "commandline like '%toto%' and not
commandline like '%wmic%'" get processid, commandline,
virtualsize, workingsetsize, privatepagecount
CmdLine  PrivatePageCount  PID  VirtualSize  WorkingSetSize
_mprosrv...      2256896  5840    87519232    22159360
prowin32...      9232384  5140   132603904    18661376
```

Super Geek: pmap (Linux) or svmon (AIX)

```
$ pmap -d 9787 | grep anon
```

Address	Kbytes	Mode	Offset	Device	Mapping
000000df3000	1616	rwX--	000000df3000	000:000	[anon]
00319ad53000	20	rwX--	00319ad53000	000:000	[anon]
00319b417000	16	rwX--	00319b417000	000:000	[anon]
00319d00a000	184	rwX--	00319d00a000	000:000	[anon]
2b5d530ed000	4	rwX--	2b5d530ed000	000:000	[anon]
2b5d530f6000	8	rwX--	2b5d530f6000	000:000	[anon]
2b5d533e6000	84	rwX--	2b5d533e6000	000:000	[anon]
Ffffffffffff60000	8192	-----	0000000000000	000:000	[anon]

- About 10 Mg used by single _progres PID 9787

Pmap for _mprosrv

```
$ pmap -d 9753 | grep -v lib | grep -v ld | grep -v _mprosrv
9753:  _mprosrv atm -pf server.pf
Address      Kbytes Mode  Offset          Device  Mapping
00000089a000    1252 rwx-- 00000089a000 000:000 [ anon ]
00319ad53000     20 rwx-- 00319ad53000 000:000 [ anon ]
00319b417000     16 rwx-- 00319b417000 000:000 [ anon ]
00319d00a000    184 rwx-- 00319d00a000 000:000 [ anon ]
2ae0a73fa000     4  rwx-- 2ae0a73fa000 000:000 [ anon ]
2ae0a7403000    16  rwx-- 2ae0a7403000 000:000 [ anon ]
2ae0a7612000 1193180 rwxS- 000000000000 000:008 [ shmid=0x78001 ]
7fff0367f000    196 rwx-- 7fff0367f000 000:000 [ stack ]
fffffffffff60000 8192 ---- 000000000000 000:000 [ anon ]
```

- Where to start...
 - SAN vs NAS vs local disks?
 - RAID levels?
 - SAN & controller cache?
 - SSD vs spinning disks?

- Sorry – we're going to stick to O.S. metrics

How Fast Can it Go?

■ Read test:

- cp to /dev/null (many in parallel)

```
$ iostat 5 25
```

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
sda1	895.80	53764.80	86.40	268824	432
sda1	985.80	66204.80	6.40	331024	32
sda1	993.60	61308.80	4.80	306544	24
sda1	995.60	65577.60	6.40	327888	32
sda1	1001.20	57969.60	6.40	289848	32
sda1	1000.60	58176.00	6.40	290880	32

- I have provisioned 1000 IOPS from AWS

How Fast Can it Go?

- Write test:
 - Prostrct create (multiple in //)

```
$ iostat 5 25
```

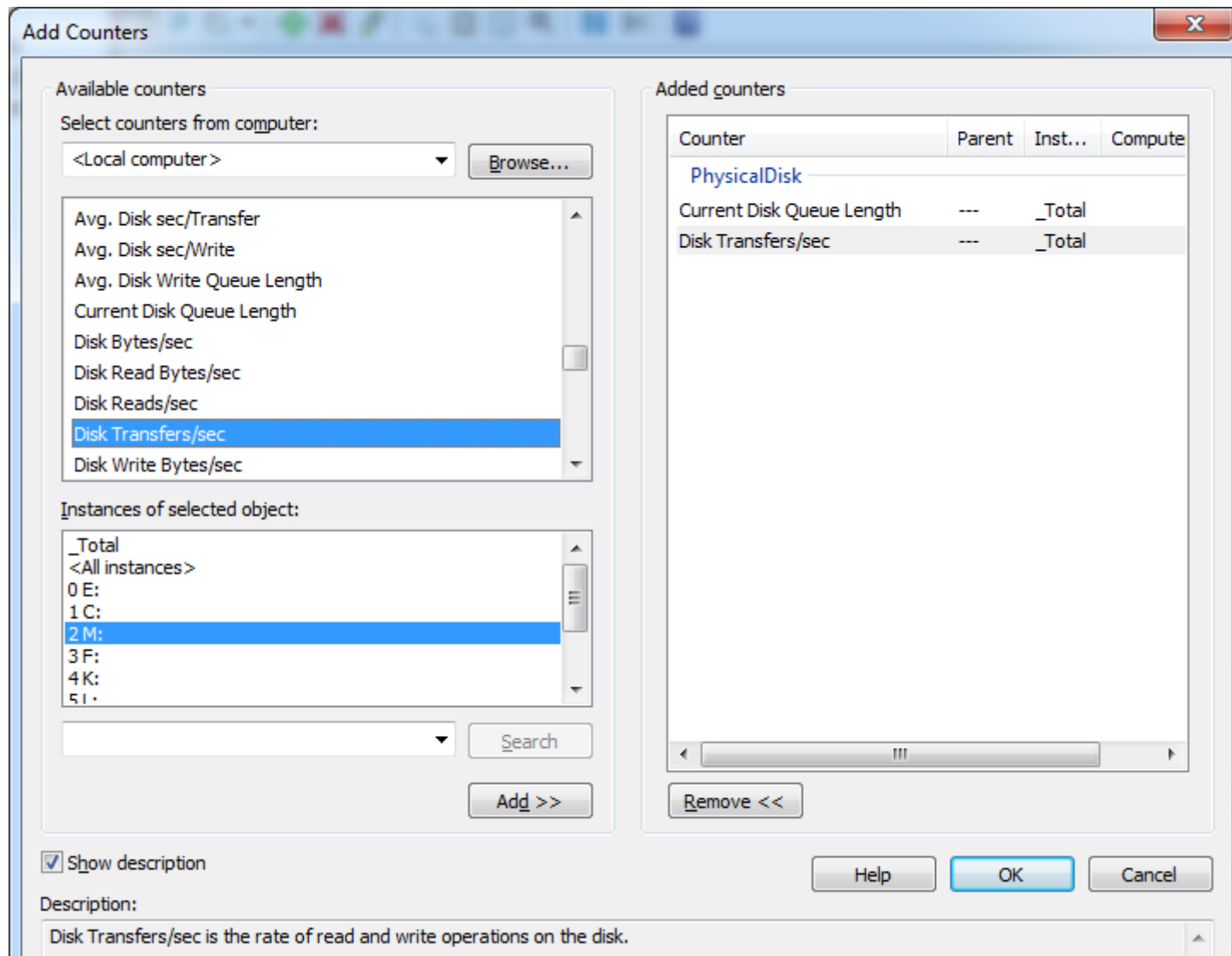
Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
sda1	989.00	9.60	36126.40	48	180632
sda1	990.60	0.00	41275.20	0	206376
sda1	987.60	0.00	40595.20	0	202976
sda1	924.00	0.00	38126.40	0	190632
sda1	1010.60	0.00	41507.20	0	207536

What's Important?

```
$ iostat 5 25
Device:      tps Blk_read/s Blk_wrtn/s Blk_read Blk_wrtn
sda1      1010.60      0.00  41507.20      0  207536
```

- For random DB access, it's tps (IOPS)
- This is why everyone prefers RAID 10
 - Provides the most spindles

Windows PerfMon



What's Important?

```
$ sar -d 5 5
04:32:41 DEV      tps   rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz  await  svctm  %util
04:36:13 dev8  1169.34   72253.31   28.86     61.81     6.10     5.22   0.83  96.87
04:36:18 dev8  1056.20   65534.40   40.00     62.09     6.24     5.91   0.95  99.88
```

- Average queue size: # of requests waiting
 - 6 processes were ready to be serviced
- Average wait time (ms): Time to service a request
 - Including time in the queue

What's Important?

```
$ sar -d 5 5
04:32:41  DEV          tps rd_s/s    w_s/s avgrq-sz avgqu-sz await  svctm  %util
04:36:13 dev8  1169.34 72253.31 28.86   61.81    6.10   5.22   0.83  96.87
04:36:18 dev8  1056.20 65534.40 40.00   62.09    6.24   5.91   0.95  99.88
```

- Look at await – svctm = 5.91 – 0.95
 - 4.96 ms spent waiting in the queue
 - 0.95 ms spent actually servicing requests
- The absolute number (5 ms) doesn't look so bad
- In reality, 83% of I/O time was spent waiting
 - Good thing I already know my max tps is about 1000
 - Ding ding ding: bottleneck

Disk I/O Critical for RDBMS

- Check DB params before you blame disk I/O
 - -B: Are you still stuck on 32 bit OpenEdge?
 - BI Cluster Size: How often are you checkpointing?
 - AI/BI block size: 16Kb set it and forget it
 - Two many APW?
 - Old recommendation of 1 per spindle still floating around
- ...and of course: BAD CODE !!!
 - Table scan after table scan...

AIX: vmstat -v

```
77 pending disk I/Os blocked with no pbuf
```

```
0 paging space I/Os blocked with no psbuf
```

```
2228 filesystem I/Os blocked with no fsbuf
```

```
2692 client filesystem I/Os blocked with no fsbuf
```

```
609566 ext pager filesystem I/Os blocked with no fsbuf
```

- “External pager FS I/Os blocked...”
 - Increase j2_dynamicBufferPreallocation
 - Older AIX: j2_nBufferPerPageDevice
 - JFS: numfsbufs

AIX: lvmo

- vmstat: “Pending disk I/Os block with no pbuf”

```
$ lvmo -v mainvg -a
vgname = mainvg
pv_pbuf_count = 512
total_vg_pbufs = 512
max_vg_pbufs = 524288
pervg_blocked_io_count = 119648
pv_min_pbuf = 512
max_vg_pbuf_count = 0
global_blocked_io_count = 119725
```

- Increase pv_pbuf_count
- Rare in 6.1+

See it in Action

- Same 30 rapid readers
- Same `-B 100`
- Before and after `drop_cache`

See it in action

```
$ iostat 5 50
```

```
04:55:59 DEV      tps rd_s/s  wr_s/s avgrq-sz avgqu-sz await svctm %util
04:56:04 dev8  30.56 1461.90  65.08    49.97    0.08   2.66  1.36   4.17
04:56:09 dev8  34.62 1750.77  12.31    50.93    0.08   2.43  1.22   4.21
04:56:14 dev8  35.17 1747.74   7.86    49.92    0.08   2.21  1.09   3.85
```

```
$ sync ; echo 3 > /proc/sys/vm/drop_caches
```

```
04:57:10 dev8  327.68 22616.95  9.04    69.05    0.69   2.12  0.80  26.27
04:57:16 dev8  328.43 22579.50  9.28    68.78    0.68   2.08  0.81  26.75
04:57:21 dev8  313.98 21858.27  7.87    69.64    0.68   2.18  0.82  25.71
```


Network

- No time to give this subject adequate treatment
 - Sorry!
- Iftop is a nifty tool on Linux
 - Like top/topas/nmon/glance
 - Real-time traffic
- Iptraf much cooler

iptraf

IPTraff

TCP Connections (Source Host:Port)

10.141.159.217:22	>	274	61324	-PA-	eth0
24.203.200.100:53152	>	139	5836	--A-	eth0

IPTraff

Proto/Port	Pkts	Bytes	PktsTo	BytesTo	PktsFrom	BytesFrom
TCP/22	103	18340	35	1452	68	16888

IPTraff

Statistics for eth0

	Total Packets	Total Bytes	Incoming Packets	Incoming Bytes	Outgoing Packets	Outgoing Bytes
Total:	65	17914	23	1294	42	16620
IP:	65	17004	23	972	42	16032
TCP:	65	17004	23	972	42	16032
UDP:	0	0	0	0	0	0
ICMP:	0	0	0	0	0	0
Other IP:	0	0	0	0	0	0
Non-IP:	0	0	0	0	0	0

Total rates: Broadcast packets: 0
Broadcast bytes: 0

Incoming rates:

IP checksum errors: 0

Outgoing rates:

Basic Network Stuff

- Check speed and duplex

```
$ ethtool eth0
```

```
Settings for eth0:
```

```
Supported ports: [ TP ]
```

```
Supported link modes:   10baseT/Half 10baseT/Full  
                        100baseT/Half 100baseT/Full  
                        1000baseT/Full
```

```
Supports auto-negotiation: Yes
```

```
Advertised link modes:  10baseT/Half 10baseT/Full  
                        100baseT/Half 100baseT/Full  
                        1000baseT/Full
```

```
Advertised auto-negotiation: Yes
```

```
Speed: 1000Mb/s
```

```
Duplex: Full
```

```
Port: Twisted Pair
```

```
PHYAD: 1
```

```
Transceiver: internal
```

```
Auto-negotiation: on
```

```
Supports Wake-on: g
```

```
Wake-on: d
```

```
Link detected: yes
```

Netstat and Isof

- On Windows, netstat -o shows PID

```
C:\Users\Paul>netstat -o
```

```
Active Connections
```

Proto	Local Address	Foreign Address	State	PID
TCP	127.0.0.1:2002	i5KOUP:54818	ESTABLISHED	2576
TCP	127.0.0.1:5354	i5KOUP:49155	ESTABLISHED	2344
TCP	127.0.0.1:5354	i5KOUP:49157	ESTABLISHED	2344

- On Linux, use Isof

```
$ lsof -P | grep _mprosrv | grep TCP
```

```
_mprosrv 30988 root 43u IPv4 19186897 TCP *:2500 (LISTEN)
```

```
_mprosrv 31006 root 41u IPv4 19186901 TCP *:2700 (LISTEN)
```

Progresswiz Consulting

- Based in Montréal, Québec, Canada
- Providing technical consulting in Progress[®], UNIX, Windows, MFG/PRO and more
- Specialized in
 - performance tuning
 - availability and business continuity planning
 - security of Progress-based systems

Questions?



Credits

Thanks!

A hand-drawn illustration of a smiling face with arms raised, positioned below the word 'Thanks!'. The drawing is simple and cartoonish, with a large smile and a small '©' symbol at the bottom right.