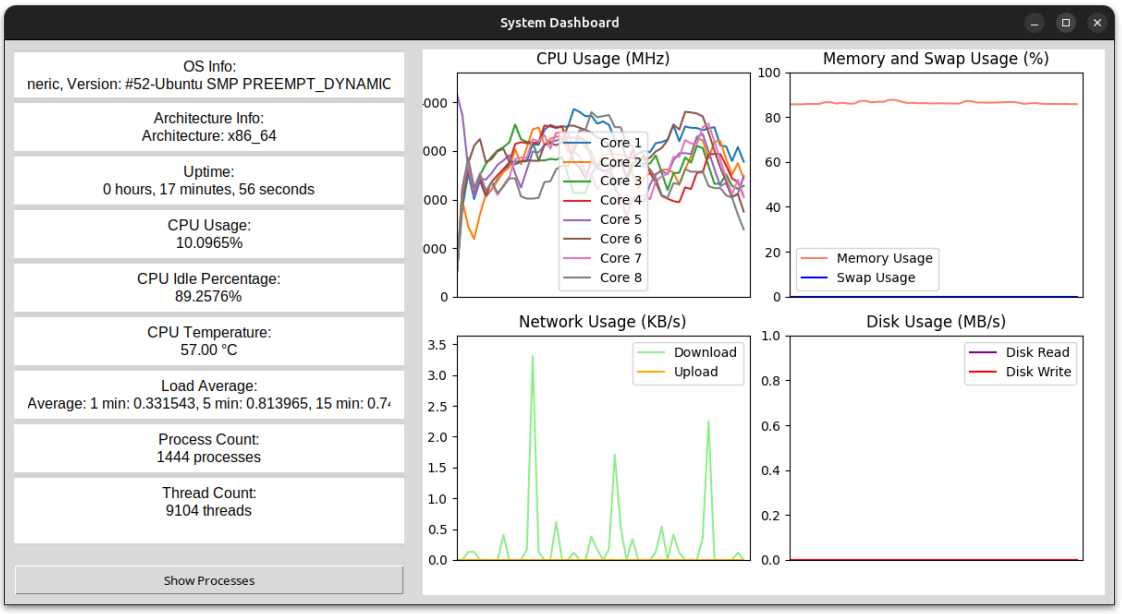
Dashboard

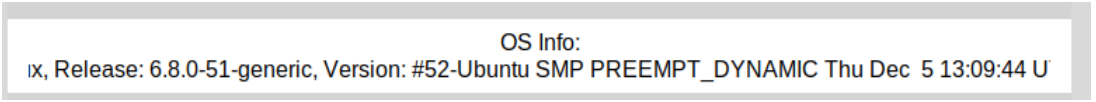
Daniel Zaki Sommer e Victor Filipe Coelho Guimarães (S73).

Todas as funções em questão estão definidas no código em C++ contido no arquivo getSysInfo.cpp, que é o backend do sistema.



A função getOsInfo coleta o nome do sistema operacional, a versão e outras informações usando a chamada de sistema uname. Segue resumo:

1. A função uname é chamada para preencher a estrutura utsname com informações sobre o sistema.
2. Os campos `sysname`, `release` e `version` da estrutura utsname contêm o nome do sistema operacional, a versão e outras informações.
3. Essas informações são formatadas em uma string e retornadas.



A função `getArchitectureInfo` coleta informações sobre a arquitetura do sistema usando a chamada de sistema `uname`. Segue resumo:

1. A função `uname` é chamada para preencher a estrutura `utsname` com informações sobre o sistema.

2. O campo `machine` da estrutura `utsname` contém informações sobre a arquitetura do sistema.
3. Essas informações são formatadas em uma string e retornadas.

```
Architecture Info:
Architecture: x86_64
```

A função `getUptime` coleta o tempo de atividade do sistema usando a chamada de sistema `sysinfo`. Segue resumo:

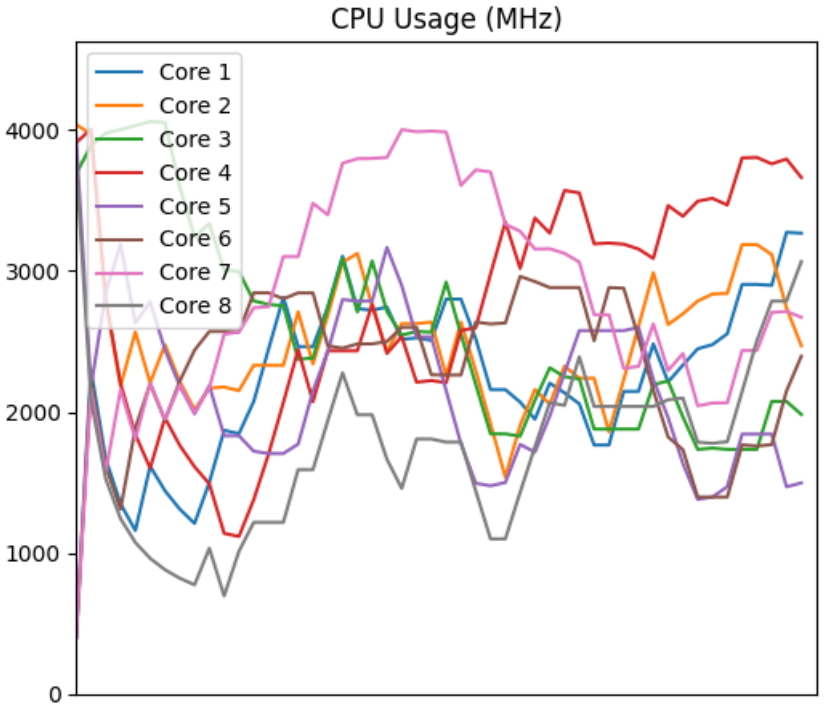
1. A função `sysinfo` é chamada para preencher a estrutura `sysinfo` com informações sobre o sistema.
2. O campo `uptime` da estrutura `sysinfo` contém o tempo de atividade do sistema em segundos.
3. Esse tempo é convertido em horas, minutos e segundos.
4. Essas informações são formatadas em uma string e retornadas.

```
Uptime:
0 hours, 30 minutes, 54 seconds
```

A função `getCpuUsage` coleta o uso da CPU em percentual lendo as estatísticas da CPU a partir do arquivo stat. Segue resumo:

1. O arquivo stat é aberto e a primeira linha, que contém as estatísticas da CPU, é lida.
2. Os valores de tempo de CPU (user, nice, system, idle, iowait, irq, softirq, steal) são extraídos da linha.
3. O tempo total da CPU e o tempo ocioso são calculados.
4. A porcentagem de uso da CPU é calculada como a diferença entre o tempo total e o tempo ocioso, dividido pelo tempo total.
5. Essa informação é formatada em uma string e retornada.

```
CPU Usage:
10.9456%
```



A função `getCpuIdlePercentage` coleta o percentual de tempo ocioso da CPU lendo as estatísticas da CPU a partir do arquivo stat. Segue resumo:

1. O arquivo stat é aberto e a primeira linha, que contém as estatísticas da CPU, é lida.
2. Os valores de tempo de CPU (user, nice, system, idle, iowait, irq, softirq, steal) são extraídos da linha.
3. O tempo total da CPU e o tempo ocioso são calculados.
4. A porcentagem de tempo ocioso da CPU é calculada como o tempo ocioso dividido pelo tempo total, multiplicado por 100.
5. Essa informação é formatada em uma string e retornada.

```
CPU Idle Percentage:
88.3832%
```

A função `getCpuTemperature` coleta a temperatura da CPU lendo o valor do arquivo temp. Segue resumo:

1. O arquivo temp é aberto e o valor da temperatura é lido.
2. A temperatura é lida em miligrados e convertida para graus Celsius.
3. Essa informação é formatada em uma string e retornada.

```
CPU Temperature:
62.00 °C
```

A função `getLoadAverage` coleta a média de carga do sistema usando a chamada de sistema `sysinfo`. Segue resumo:

1. A função `sysinfo` é chamada para preencher a estrutura `sysinfo` com informações sobre o sistema.
2. Os campos `loads` da estrutura `sysinfo` contêm as médias de carga do sistema para os últimos 1, 5 e 15 minutos.
3. Essas médias de carga são convertidas de uma unidade fixa para um valor de ponto flutuante.
4. Essas informações são formatadas em uma string e retornadas.

```
Load Average:
Load Average: 1 min: 1.09375, 5 min: 1.2417, 15 min: 1.01953
```

A função `getProcessCount` coleta o número de processos em execução usando a chamada de sistema `sysinfo`. Segue resumo:

1. A função `sysinfo` é chamada para preencher a estrutura `sysinfo` com informações sobre o sistema.
2. O campo `procs` da estrutura `sysinfo` contém o número de processos em execução.
3. Essa informação é formatada em uma string e retornada.

```
Process Count:
1475 processes
```

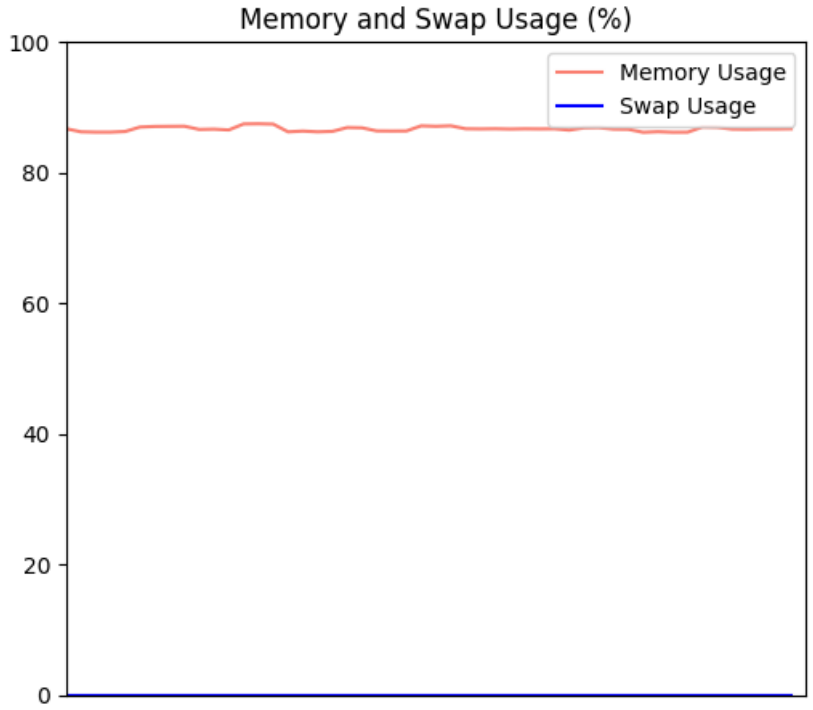
A função `getThreadCount` coleta o número de threads lendo as estatísticas do sistema a partir do arquivo stat. Segue resumo:

1. O arquivo stat é aberto e cada linha é lida.
2. A linha que contém a chave "processes" é encontrada.
3. O valor associado à chave "processes" é extraído.
4. Essa informação é formatada em uma string e retornada.

```
Thread Count:
11485 threads
```

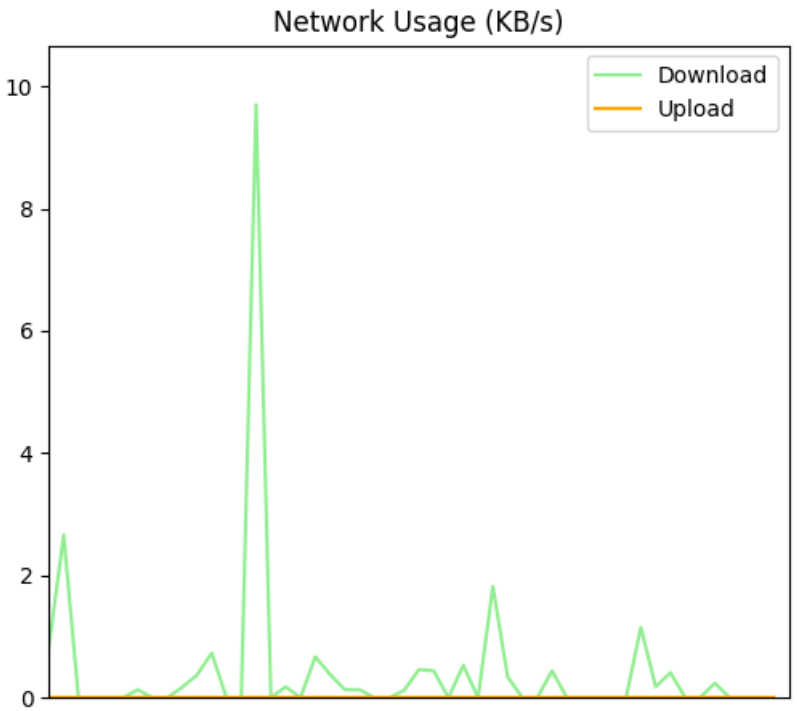
A função `getTotalMemory` coleta a memória total do sistema usando a chamada de sistema `sysinfo`. A função `getFreeMemory` coleta a memória livre do sistema da mesma forma. A função `getSwapUsage` coleta o uso da memória swap. Segue resumo:

1. A função `sysinfo` é chamada para preencher a estrutura `sysinfo` com informações sobre o sistema.
2. Os campos `totalram` e `freeram` da estrutura `sysinfo` contêm a memória total e a memória livre, respectivamente.
3. Os campos `totalswap` e `freeswap` contêm a memória swap total e a memória swap livre, respectivamente.
4. Essas informações são convertidas para megabytes e formatadas em strings.



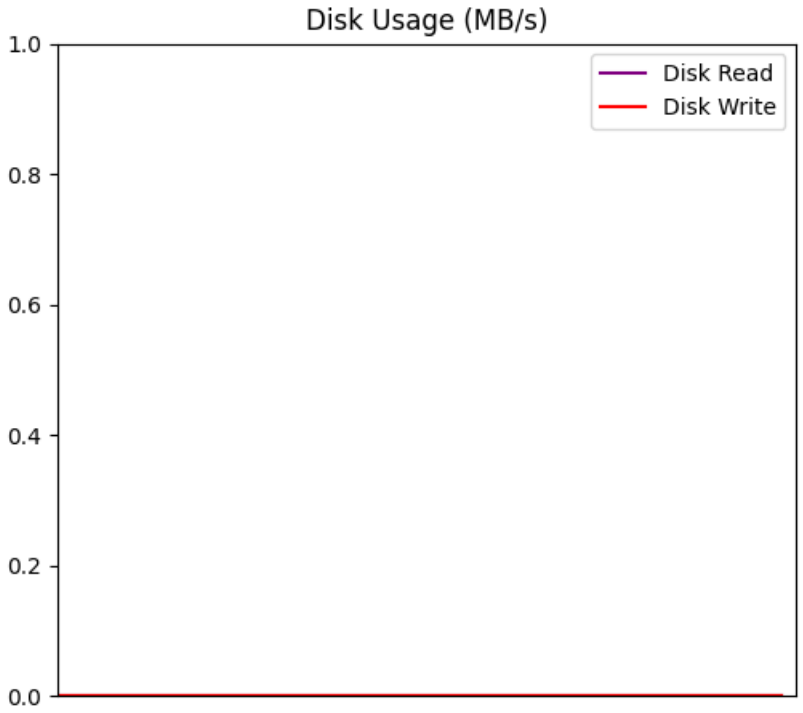
As funções `getNetworkReceiveRate` e `getNetworkTransmitRate` coletam as taxas de recebimento e transmissão de rede lendo as estatísticas de rede a partir do arquivo dev. Segue resumo:

1. O arquivo dev é aberto e cada linha é lida.
2. As linhas que contêm as estatísticas de rede para cada interface são analisadas.
3. Os valores de bytes recebidos e transmitidos são extraídos.
4. A diferença entre os valores atuais e os valores anteriores é calculada.
5. Essa diferença é convertida para kilobytes por segundo.
6. Essas informações são formatadas em strings e retornadas.



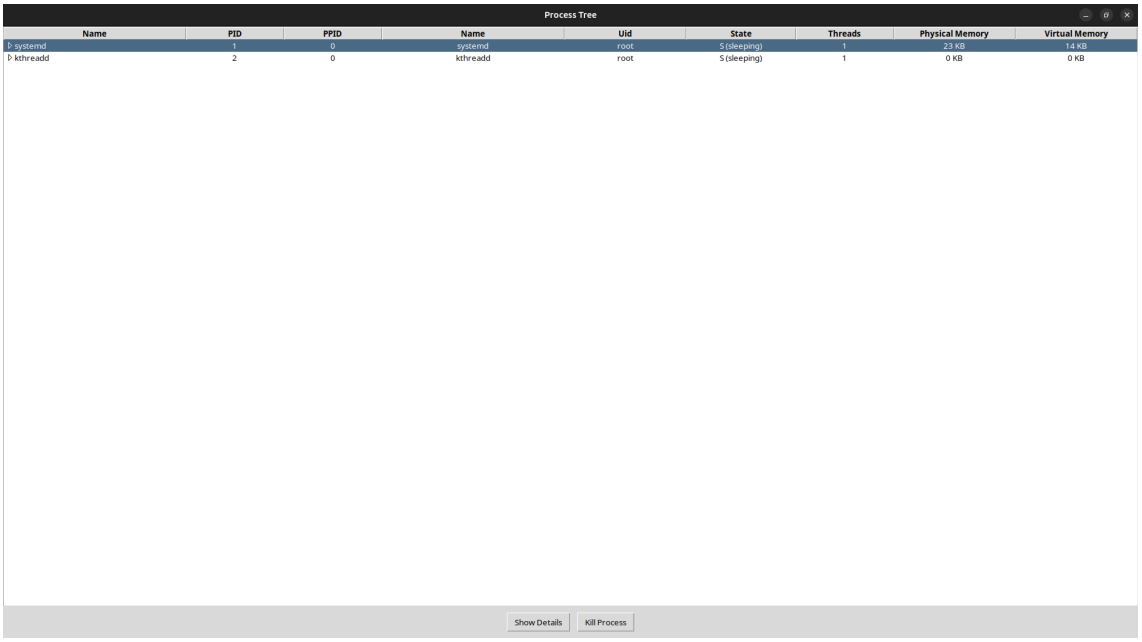
As funções `getDiskRead` e `getDiskWrite` coletam o uso de leitura e escrita do disco lendo as estatísticas do disco a partir do arquivo diskstats. Segue resumo:

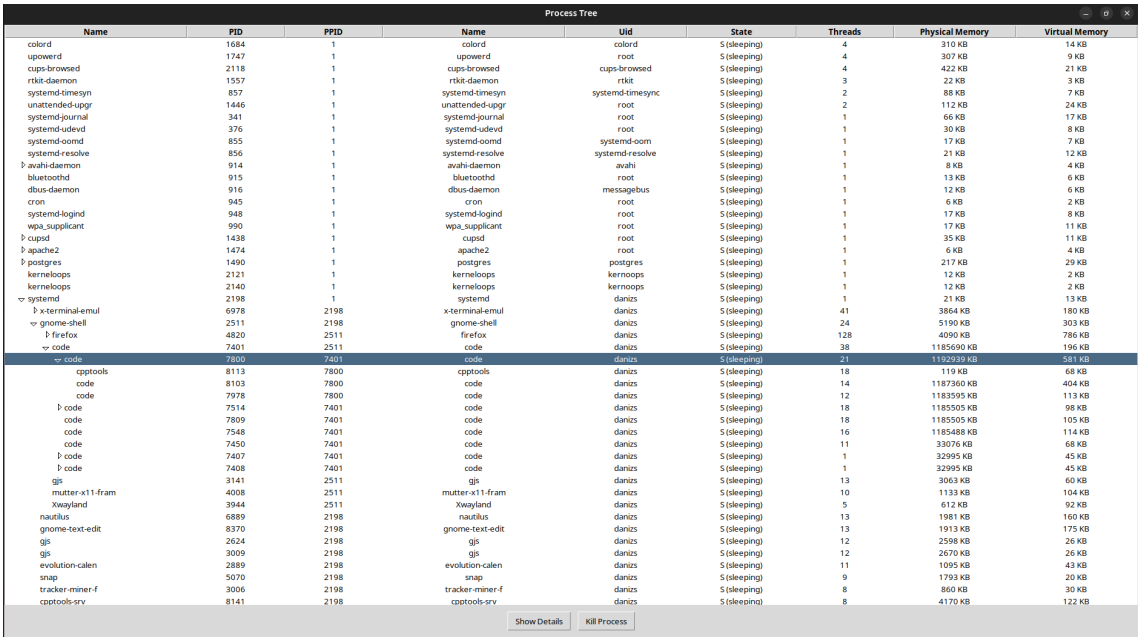
1. O arquivo diskstats é aberto e cada linha é lida.
2. As linhas que correspondem ao dispositivo de disco desejado (por exemplo, `sda` ou `nvme0n1`) são encontradas.
3. Os valores de leitura e escrita de setores são extraídos.
4. A diferença entre os valores atuais e os valores anteriores é calculada.
5. Essa diferença é convertida para megabytes (assumindo que cada setor tem 512 bytes).
6. Essas informações são formatadas em strings e retornadas.



A função `getProcessesInfo` coleta informações detalhadas sobre todos os processos em execução no sistema. Segue resumo:

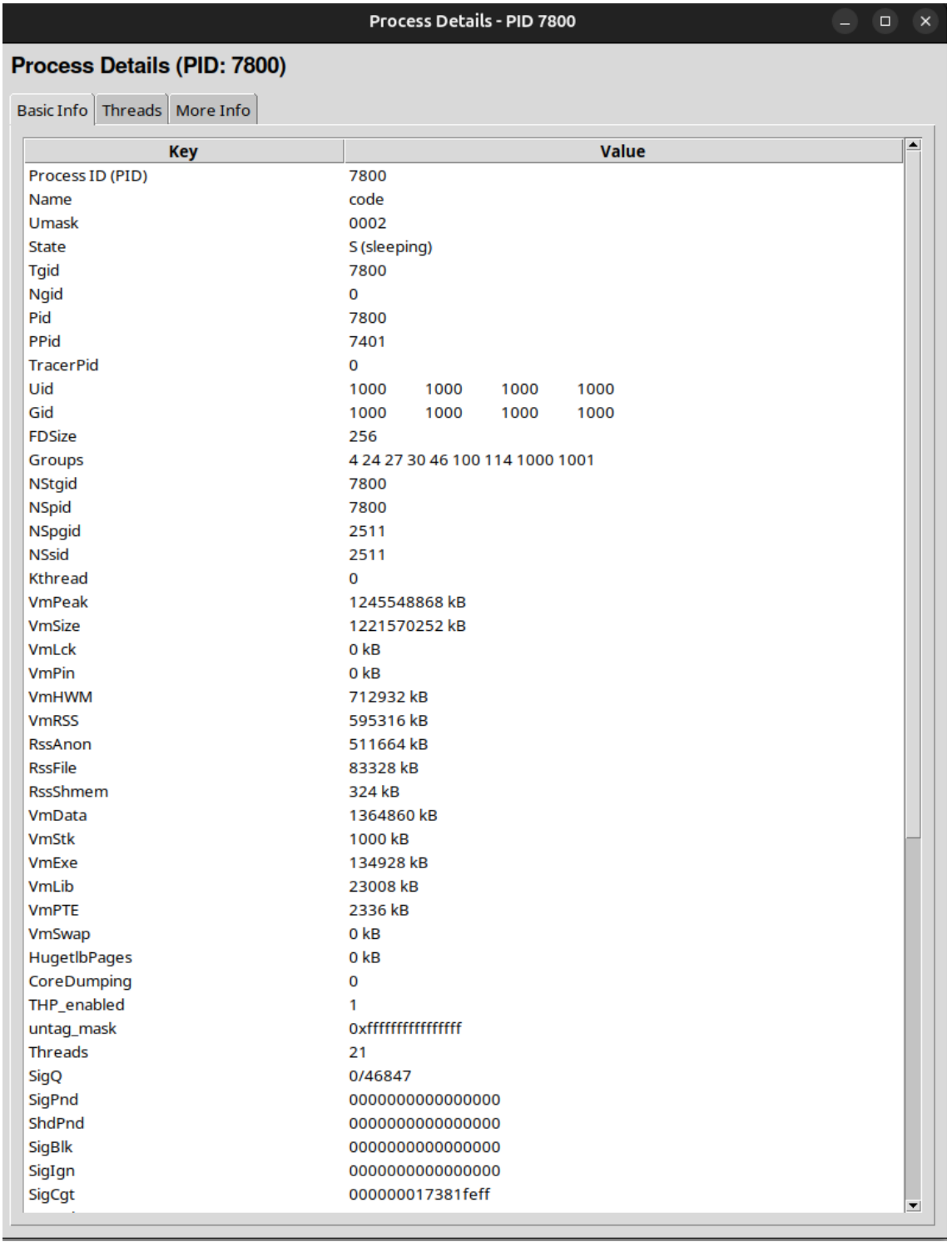
1. O diretório proc é aberto e cada entrada é lida.
2. Para cada entrada que representa um processo (diretório cujo nome é um número), o arquivo `/proc/[pid]/status` é aberto.
3. As informações do processo, como PID, PPID, nome, estado, número de threads, memória virtual e memória física, são extraídas do arquivo `status`.
4. O UID do processo é usado para obter o nome do usuário associado.
5. Essas informações são formatadas em uma string e retornadas.

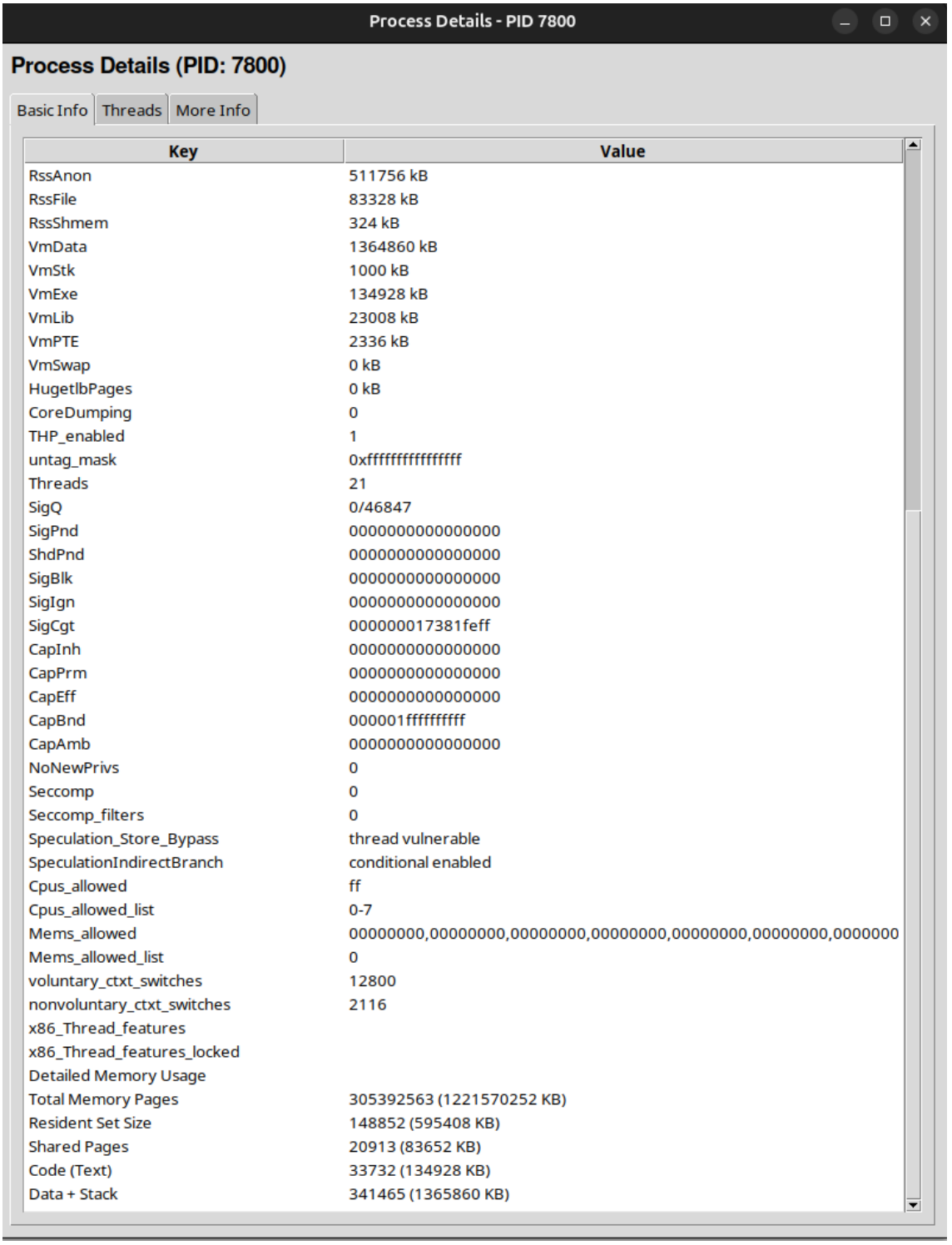




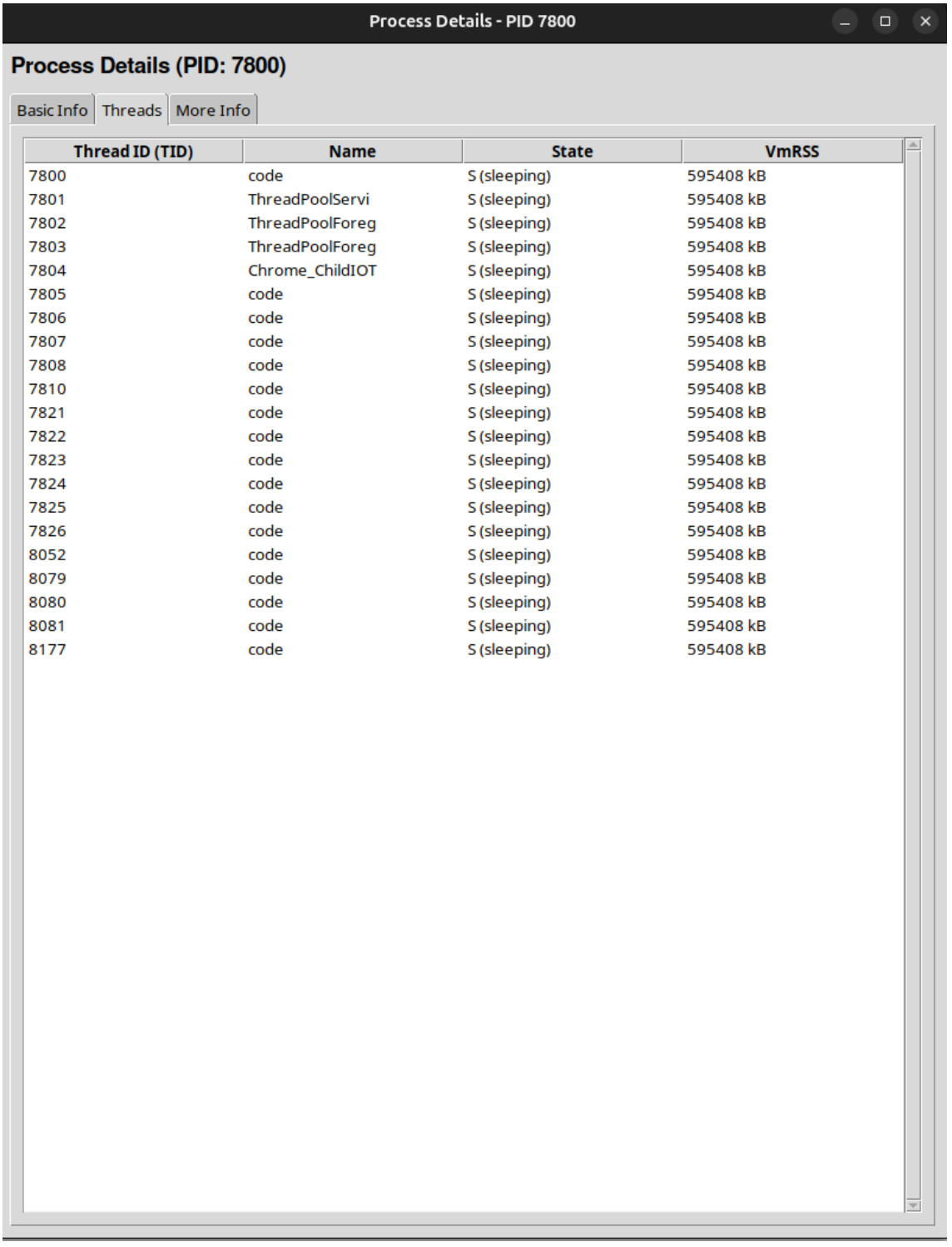
A função `getSpecificProcess` coleta informações detalhadas sobre um processo específico usando o PID. Segue resumo:

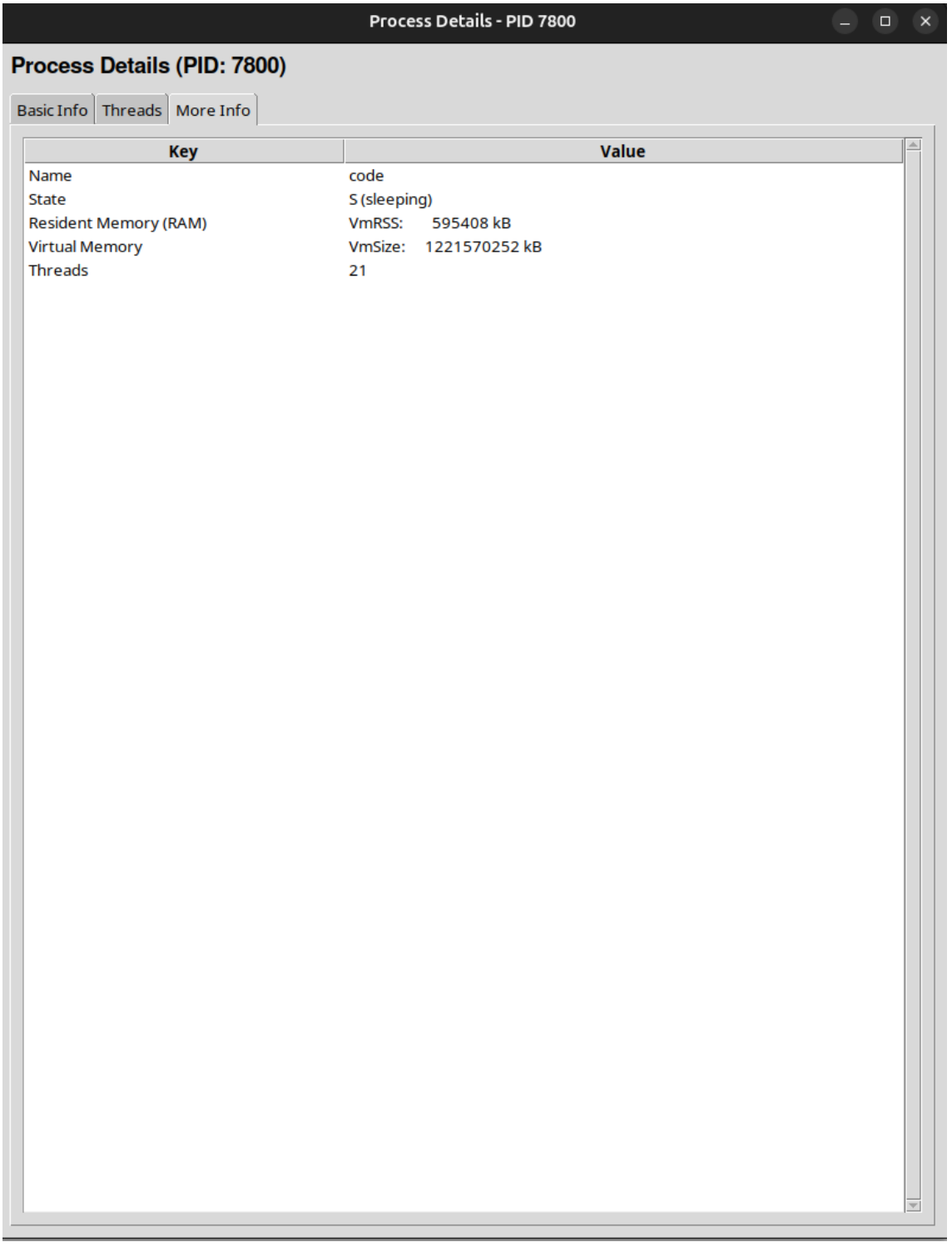
1. O diretório `/proc/[pid]` é aberto e os arquivos `status` e `statm` são lidos.
2. As informações do processo, como nome, estado, memória virtual, memória física e número de threads, são extraídas do arquivo `status`.
3. Informações detalhadas de memória são extraídas do arquivo `statm`.
4. Essas informações são formatadas em uma string e retornadas.


Process Details - PID 7800
Process Details (PID: 7800)
Basic Info
Threads
More Info
Key
Value
Process ID (PID) 7800
Name code
Umask 0002
State S (sleeping)
Tgid 7800
Ngid 0
Pid 7800
PPid 7401
TracerPid 0
Uid 1000 1000 1000 1000
Gid 1000 1000 1000 1000
FDSize 256
Groups 4 24 27 30 46 100 114 1000 1001
NStgid 7800
NSpid 7800
NSpgid 2511
NSsid 2511
Kthread 0
VmPeak 1245548868 kB
VmSize 1221570252 kB
VmLck 0 kB
VmPin 0 kB
VmHWM 712932 kB
VmRSS 595316 kB
RssAnon 511664 kB
RssFile 83328 kB
RssShmem 324 kB
VmData 1364860 kB
VmStk 1000 kB
VmExe 134928 kB
VmLib 23008 kB
VmPTE 2336 kB
VmSwap 0 kB
HugetlbPages 0 kB
CoreDumping 0
THP_enabled 1
untag_mask 0xffffffffffffffff
Threads 21
SigQ 0/46847
SigPnd 0000000000000000
ShdPnd 0000000000000000
SigBlk 0000000000000000
SigIgn 0000000000000000
SigCgt 000000017381feff



Key	Value
RssAnon	511756 kB
RssFile	83328 kB
RssShmem	324 kB
VmData	1364860 kB
VmStk	1000 kB
VmExe	134928 kB
VmLib	23008 kB
VmPTE	2336 kB
VmSwap	0 kB
HugetlbPages	0 kB
CoreDumping	0
THP_enabled	1
untag_mask	0xffffffffffffffff
Threads	21
SigQ	0/46847
SigPnd	0000000000000000
ShdPnd	0000000000000000
SigBlk	0000000000000000
SigIgn	0000000000000000
SigCgt	000000017381feff
CapInh	0000000000000000
CapPrm	0000000000000000
CapEff	0000000000000000
CapBnd	000001ffffffffff
CapAmb	0000000000000000
NoNewPrivs	0
Seccomp	0
Seccomp_filters	0
Speculation_Store_Bypass	thread vulnerable
SpeculationIndirectBranch	conditional enabled
Cpus_allowed	ff
Cpus_allowed_list	0-7
Mems_allowed	00000000,00000000,00000000,00000000,00000000,00000000,0000000
Mems_allowed_list	0
voluntary_ctxt_switches	12800
nonvoluntary_ctxt_switches	2116
x86_Thread_features	
x86_Thread_features_locked	
Detailed Memory Usage	
Total Memory Pages	305392563 (1221570252 KB)
Resident Set Size	148852 (595408 KB)
Shared Pages	20913 (83652 KB)
Code (Text)	33732 (134928 KB)
Data + Stack	341465 (1365860 KB)


Process Details - PID 7800
Process Details (PID: 7800)
Basic Info
Threads
More Info
Thread ID (TID) Name State VmRSS
7800 code S (sleeping) 595408 kB
7801 ThreadPoolServi S (sleeping) 595408 kB
7802 ThreadPoolForeg S (sleeping) 595408 kB
7803 ThreadPoolForeg S (sleeping) 595408 kB
7804 Chrome_ChildIOT S (sleeping) 595408 kB
7805 code S (sleeping) 595408 kB
7806 code S (sleeping) 595408 kB
7807 code S (sleeping) 595408 kB
7808 code S (sleeping) 595408 kB
7810 code S (sleeping) 595408 kB
7821 code S (sleeping) 595408 kB
7822 code S (sleeping) 595408 kB
7823 code S (sleeping) 595408 kB
7824 code S (sleeping) 595408 kB
7825 code S (sleeping) 595408 kB
7826 code S (sleeping) 595408 kB
8052 code S (sleeping) 595408 kB
8079 code S (sleeping) 595408 kB
8080 code S (sleeping) 595408 kB
8081 code S (sleeping) 595408 kB
8177 code S (sleeping) 595408 kB


Process Details - PID 7800
Process Details (PID: 7800)
Basic Info
Threads
More Info
Key
Value
Name
code
State
S (sleeping)
Resident Memory (RAM)
VmRSS: 595408 kB
Virtual Memory
VmSize: 1221570252 kB
Threads
21