Otimizar o BeeGFS no cenário HPC com RHEL 8, rede InfiniBand e armazenamento em discos rotacionais (HDDs), vamos detalhar as configurações para cada componente do BeeGFS, focando em maximizar a performance do sistema de arquivos paralelo.
Lembre-se que a "melhor" configuração pode variar com a carga de trabalho específica. Estas são recomendações para um ponto de partida robusto, visando bom throughput sequencial e eficiência com HDDs.

⓪ Configurações de BIOS/UEFI e Sistema Operacional Base (RHEL 8)

Estas configurações são fundamentais e aplicam-se a todos os nós (clientes, servidores de metadados e servidores de armazenamento BeeGFS).

- **BIOS/UEFI:**
  - **Perfil de Performance:** "Máximo Desempenho" ou "HPC".
  - **Economia de Energia:** Desative C-states (ou intel_idle.max_cstate=0 processor.max_cstate=0 no boot), P-states (ou use o governor performance), SpeedStep.
  - **NUMA:** Habilitado; "Node Interleaving" desabilitado.
  - **Hyper-Threading (SMT):** Teste. Para I/O intensivo, desligar pode ser benéfico.
  - **Slots PCIe:** HCAs InfiniBand em slots de alta largura de banda (x16).
  - **Virtualization:** Desative (VT-x, AMD-V) se não estiver em uso.
- **RHEL 8 OS (Todos os Nós):**
  - **tuned Profile:** Use um perfil base como throughput-performance ou network-latency e personalize-o, ou crie um perfil HPC específico como detalhado anteriormente. Pontos chave para incluir/verificar no perfil tuned para BeeGFS:
    - CPU governor: performance.
    - Transparent Huge Pages: madvise ou never (teste qual funciona melhor para o BeeGFS e suas aplicações).
    - VM settings: vm.dirty_background_ratio=5, vm.dirty_ratio=15 (ou _bytes), vm.swappiness=1, vm.zone_reclaim_mode=0.
  - **Parâmetros de Boot do Kernel (/etc/default/grub):**
    GRUB_CMDLINE_LINUX_DEFAULT="... transparent_hugepage=madvise intel_idle.max_cstate=0 processor.max_cstate=0 idle=poll elevator=mq-deadline"
    - idle=poll: Menor latência (maior consumo de energia).
    - elevator=mq-deadline: Define mq-deadline como padrão. Crucial para os HDDs nos servidores de armazenamento.
    - Aplique com grub2-mkconfig e reinicie.
  - **Limites de Recursos (/etc/security/limits.d/99-hpc.conf):**
    * soft memlock unlimited
    * hard memlock unlimited
    * soft nofile 1048576

* hard nofile 1048576
* soft stack unlimited
* hard stack unlimited
memlock ilimitado é **essencial** para RDMA (InfiniBand).

- **SELinux e Firewall:** Para máxima performance em ambiente seguro:
  - SELinux: SELINUX=disabled em /etc/selinux/config (requer reboot).
  - Firewall: sudo systemctl disable --now firewalld. *(Avalie os riscos de segurança antes de desabilitar)*.

---

① Configurações de Rede InfiniBand (Todos os Nós BeeGFS)

- **Drivers OFED/RDMA:** Instale a versão estável mais recente do OFED do fabricante da HCA ou os pacotes rdma-core do RHEL 8.
  Bash
  sudo dnf install rdma-core libibverbs-utils librdmacm-utils perftest

- **OpenSM:** Garanta que um Subnet Manager esteja ativo na malha InfiniBand.
- **IPoIB:** Configure se necessário para gerenciamento ou outras aplicações, com CONNECTED_MODE=yes e MTU adequado (ex: 4092).
- **Afinidade de IRQ para HCA InfiniBand:**
  - Desabilite irqbalance ou configure-o para ignorar a HCA.
  - Use scripts para distribuir as IRQs da HCA entre os núcleos da CPU do mesmo nó NUMA da HCA.
- **BeeGFS e RDMA:** O BeeGFS suporta RDMA (via verbs API do InfiniBand) diretamente. Esta é a configuração preferida para InfiniBand.
  - Nos arquivos de configuração do BeeGFS (beegfs-meta.conf, beegfs-storage.conf, beegfs-client.conf), especifique as interfaces InfiniBand corretas.
  - Parâmetro chave: connRDMAEnabled=true (geralmente é o padrão se o hardware RDMA for detectado).

---

② Configurações do Cliente BeeGFS (beegfs-client)

- **Arquivo de Configuração (/etc/beegfs/beegfs-client.conf):**
  - sysMgmtdHost: Endereço do seu servidor beegfs-mgmtd.
  - connInterfacesFile: (Recomendado) Caminho para um arquivo que lista as interfaces InfiniBand a serem usadas (ex: ib0).
    - Exemplo de /etc/beegfs/connInterfacesClient.conf: ib0
  - connNetFilterFile: Alternativa ao connInterfacesFile para especificar sub-redes.
  - connRDMAEnabled = true
  - connMaxInternodeNum: Aumente se tiver muitos servidores de armazenamento/metadados (ex: 800, padrão pode ser baixo para clusters

grandes).
    - tuneNumWorkers: Número de threads de trabalho do cliente. Um bom ponto de partida é 8 por nó NUMA que o cliente usa, ou igual ao número de núcleos se não houver certeza. Teste. (Padrão pode ser 16).
    - tuneUseBufferedIO = false (Padrão: false, usa Direct I/O)
        - **Teste este parâmetro!** Para HDDs e certos workloads, tuneUseBufferedIO = true *pode* ser melhor, pois permite que o cache de página do Linux e o readahead do kernel ajudem a agrupar I/Os. Para grandes I/Os sequenciais, Direct I/O (false) é geralmente preferido.
    - Se tuneUseBufferedIO = true:
        - tunePreferredMOUNTReadAheadKB: (Ex: 1024 ou 2048)
        - tunePreferredMOUNTWriteBehindKB: (Ex: 1024 ou 2048)
    - sysMaxCache χρήσηInKB: Tamanho do cache de metadados do cliente. (Padrão 524288 - 512MB). Aumente se tiver muita RAM e lidar com muitos arquivos/diretórios.
    - logLevel: Reduza para 2 ou 3 em produção após a configuração estar estável.
- **Opções de Montagem (Exemplo em /etc/fstab ou comando mount):**
  # Usando arquivo de configuração para todas as opções
  beegfs_ondemand /mnt/beegfs beegfs
  cfgFile=/etc/beegfs/beegfs-client.conf,_netdev,rw,noatime,nodiratime 0 0

  # Ou especificando na linha de montagem (menos comum se já no cfgFile)
  # beegfs_mgmtd_ip_ou_hostname:/ /mnt/beegfs beegfs \
  #
  cfgFile=/etc/beegfs/beegfs-client.conf,connRDMAEnabled=true,connInterfacesFile=/etc/beegfs/connInterfacesClient.conf,_netdev,rw,noatime,nodiratime 0 0

    - _netdev: Garante que a rede esteja ativa.
    - noatime, nodiratime: Reduz I/O de metadados.

---

③ Configurações do Servidor de Metadados BeeGFS (beegfs-meta)

- **Arquivo de Configuração (/etc/beegfs/beegfs-meta.conf):**
    - sysMgmtdHost: Endereço do servidor beegfs-mgmtd.
    - storeMetaDirectory: **CRÍTICO!** Idealmente, este diretório deve estar em SSDs/NVMe rápidos para baixa latência de metadados. Se precisar usar HDDs, use um RAID 1 ou RAID 10 de HDDs rápidos e configure o agendador de I/O para mq-deadline para esses discos.
    - connInterfacesFile: (Recomendado) Caminho para arquivo com interfaces InfiniBand.
    - connRDMAEnabled = true
    - connMaxInternodeNum: Ajuste conforme necessário.

- tuneNumWorkers: Similar ao cliente, ajuste baseado nos núcleos do servidor de metadados. (Padrão pode ser 16).
- logLevel: 2 ou 3 em produção.

---

④ Configurações do Servidor de Armazenamento BeeGFS (beegfs-storage)

- **Arquivo de Configuração (/etc/beegfs/beegfs-storage.conf):**
  - sysMgmtdHost: Endereço do servidor beegfs-mgmtd.
  - storeStorageDirectory: Lista dos diretórios de armazenamento (targets), cada um geralmente em um HDD ou LUN RAID separado.
  - connInterfacesFile: (Recomendado) Caminho para arquivo com interfaces InfiniBand.
  - connRDMAEnabled = true
  - connMaxInternodeNum: Ajuste conforme necessário.
  - tuneNumWorkers: Ajuste baseado nos núcleos do servidor de armazenamento e no número de targets que ele gerencia. (Padrão pode ser 16).
  - tuneUseDirectIO = true (Padrão): Geralmente recomendado para servidores de armazenamento para evitar double caching.
  - tuneBindToNumaNode: Se os servidores de armazenamento forem NUMA, configure para o(s) nó(s) NUMA que têm os discos e as HCAs InfiniBand para otimizar a localidade.
  - logLevel: 2 ou 3 em produção.
- **Otimizações de Sistema para HDDs nos Servidores de Armazenamento:**
  - **Agendador de I/O: ABSOLUTAMENTE CRÍTICO!** Para os HDDs que hospedam os storeStorageDirectory:
    - Use mq-deadline ou kyber. Configure via udev rules para persistência ou parâmetro de boot do kernel nesses servidores.
      - Exemplo udev rule (/etc/udev/rules.d/60-scheduler.rules): ACTION=="add|change", KERNEL=="sd[a-z]", ATTR{queue/scheduler}="mq-deadline"

  - **Read-Ahead do Dispositivo de Bloco:** Ajuste o read-ahead para os HDDs.
    Bash
    sudo blockdev --setra <valor_em_setores_de_512_bytes> /dev/sdX
    # Exemplo: para 2MB de readahead (2048KB): 2048 * 2 = 4096 setores
    sudo blockdev --setra 4096 /dev/sdX
    Faça isso para cada HDD de armazenamento. Pode ser colocado em um script de inicialização.
  - **Sistema de Arquivos nos HDDs (Targets):** Use XFS (comum) formatado com opções para performance, como alinhamento com a geometria do RAID (se houver) e inodes grandes se necessário.

Bash
sudo mkfs.xfs -f -s size=4096 -d su=<stripe_unit_KB>,sw=<stripe_width_N_disks> -i size=512 /dev/sdX
(Ajuste su e sw se estiver usando RAID por baixo do XFS).

- **Opções de Montagem para os Targets XFS (no servidor de armazenamento):**
  - Em /etc/fstab nos servidores de armazenamento:
    /dev/sdXN /path/to/beegfs_targetX xfs defaults,noatime,nodiratime,largeio,inode64 0 0

---

⑤ Servidor de Gerenciamento BeeGFS (beegfs-mgmtd)

- **Arquivo de Configuração (/etc/beegfs/beegfs-mgmtd.conf):**
  - storeMgmtdDirectory: Diretório para dados de gerenciamento. Pode ser em disco local.
  - connInterfacesFile, connRDMAEnabled: Configure se o mgmtd precisar se comunicar via RDMA (geralmente não é gargalo de performance).
  - logLevel: 2 ou 3.
  - Este serviço é geralmente leve em termos de recursos.

---

⑥ Aplicação e MPI

- **MPI:** Use uma biblioteca MPI otimizada para InfiniBand (Open MPI, Intel MPI, MVAPICH2) configurada para ucx PML ou openib BTL.
- **MPI-IO:**
  - Use MPI-IO para acesso paralelo através do cliente BeeGFS.
  - O BeeGFS geralmente se apresenta como um sistema de arquivos POSIX, então o MPI-IO sobre ele deve funcionar bem.
  - Ajuste os "hints" do MPI-IO para otimizar para HDDs: aumente buffers de I/O coletivo, alinhe acessos.

---

⑦ Teste e Monitoramento Contínuo

- Use benchmarks como IOR, mdtest, fio (com múltiplos clientes), e benchmarks de aplicação para testar a performance do BeeGFS.
- **Monitore durante os testes:**
  - **Todos os Nós:** dstat -Tcdnpgymsir --top-cpu --top-bio --top-latency, vmstat, iostat, sar.
  - **Nós BeeGFS:** Use beegfs-ctl --nodetype=client --getentryinfo /mnt/beegfs/somefile para inspecionar o layout.
  - **Servidores de Armazenamento:** Foque no iostat para os HDDs (%util, avgqu-sz, await).

- **Rede InfiniBand:** ibtop (se disponível), ibstat, contadores de performance.
- **Logs do BeeGFS:** Verifique os logs em /var/log/beegfs/ para erros ou warnings.

Pontos Chave Específicos para BeeGFS com HDDs:

1. **Otimização dos HDDs nos Servidores de Armazenamento:** Agendador (mq-deadline), readahead (blockdev), e um sistema de arquivos XFS bem configurado nos targets são absolutamente cruciais.
2. **RDMA:** Garanta que o BeeGFS esteja usando RDMA em todas as comunicações entre cliente, metadados e armazenamento via InfiniBand.
3. **tuneUseBufferedIO no Cliente:** Este é um parâmetro chave para testar. O padrão Direct I/O (false) é frequentemente bom para HPC, mas para HDDs, o I/O bufferizado (true) pode, em alguns casos, ajudar a agrupar operações e se beneficiar do cache de página do Linux. Teste com seu workload.
4. **tuneNumWorkers:** Ajuste este parâmetro nos clientes, servidores de metadados e armazenamento com base nos núcleos disponíveis e na carga.
5. **Diretório de Metadados (storeMetaDirectory):** Se possível, coloque em SSDs, mesmo que os dados estejam em HDDs. Isso melhora significativamente a performance de operações dependentes de metadados.

Lembre-se de aplicar as mudanças gradualmente e testar o impacto de cada uma. A documentação oficial do BeeGFS também é uma fonte valiosa de informações de tuning.
Ok, para o seu cenário específico com BeeGFS no RHEL 8, rede InfiniBand e armazenamento em discos rotacionais (HDDs), vamos detalhar uma configuração focada em máxima performance possível para o sistema de arquivos paralelo.
A estratégia principal será otimizar o throughput para grandes operações sequenciais, já que HDDs sofrem mais com acessos aleatórios e latência. Usaremos RDMA sobre InfiniBand para a comunicação entre os componentes BeeGFS.
Lembre-se: Algumas dessas configurações são para os **nós clientes**, outras para os **servidores BeeGFS (meta, storage, mgmtd)**. Indiquei onde aplicável.

⓪ Configurações de BIOS/UEFI (Em todos os nós: Clientes e Servidores BeeGFS)

- **Perfil de Performance:** "Máximo Desempenho" ou "HPC".
- **Economia de Energia:** Desative C-states (ou intel_idle.max_cstate=0 processor.max_cstate=0), P-states (ou configure para performance via tuned), SpeedStep, etc.
- **NUMA:** Habilitado. "Node Interleaving" (ou "Memory Interleaving") **desabilitado**.
- **Hyper-Threading (SMT):** Teste. Para I/O intensivo com HDDs, **desabilitar** o HT pode ser benéfico.
- **Slots PCIe:** HCAs InfiniBand em slots PCIe com máxima largura de banda (ex: x16).
- **Virtualization (VT-x, AMD-V):** Desative se não estiver usando.

---

① Configurações do RHEL 8 (Aplicáveis a Clientes e Servidores BeeGFS, salvo indicação)

1. tuned Profile Personalizado (Exemplo para Clientes e Servidores)

Crie /etc/tuned/beegfs-hpc-hdd/tuned.conf:

Ini, TOML

```
[main]
summary=Perfil otimizado para BeeGFS HPC com HDDs e InfiniBand
include=throughput-performance # Bom ponto de partida, focado em throughput

[cpu]
governor=performance
# force_latency=cstate.id_no_zero:1 # Considere para latência mínima, mas alto consumo

[vm]
transparent_hugepages=madvise # 'never' pode ser testado, mas 'madvise' é mais seguro
swappiness=1
dirty_background_ratio=5      # Inicia flush para HDDs mais cedo
dirty_ratio=15                # Limite geral para dados sujos, ajustado para HDDs
# Se servidores de armazenamento tiverem muita RAM:
# dirty_background_bytes=536870912 # 512MB
# dirty_bytes=2147483648          # 2GB (Ajuste conforme RAM e testes)
min_free_kbytes=2097152       # Ex: 2GB (Manter mais memória livre)
# Para servidores BeeGFS, especialmente storage e meta:
# vfs_cache_pressure=50 # Favorece manter inodes/dentries em cache

[sysctl]
# --- Kernel e VM ---
vm.zone_reclaim_mode=0
kernel.sched_min_granularity_ns=10000000
kernel.sched_migration_cost_ns=5000000
kernel.sched_autogroup_enabled=0 # Pode ajudar em cargas HPC

# --- Rede (Relevante para IPoIB ou TCP fallback) ---
# net.core.rmem_max = ... (valores altos se TCP for usado extensivamente)
# net.core.wmem_max = ...

# --- ESPECÍFICO PARA SERVIDORES DE ARMAZENAMENTO BEEGFS COM HDDs ---
```

Estes são aplicados diretamente no SO dos servidores de armazenamento,
mas podem ser incluídos aqui se o mesmo perfil for usado e adaptado.
Note: O agendador de I/O para os HDDs é melhor configurado via udev ou boot param.

Ative o perfil (em clientes e servidores):
sudo tuned-adm profile beegfs-hpc-hdd

2. Parâmetros de Boot do Kernel (/etc/default/grub)

Adicione à linha GRUB_CMDLINE_LINUX (em clientes e servidores):
transparent_hugepage=madvise intel_idle.max_cstate=0 processor.max_cstate=0 idle=poll elevator=mq-deadline slab_common.transparent_hugepage=never

- elevator=mq-deadline: **CRUCIAL** como padrão para os HDDs nos servidores de armazenamento. tuned ou udev podem refinar por dispositivo.
- idle=poll: Baixa latência, alto consumo.
- Se transparent_hugepage=never for escolhido em vez de madvise, slab_common.transparent_hugepage=never ajuda.

Aplique o GRUB e reinicie.

3. Limites de Recursos (/etc/security/limits.d/99-hpc.conf) (Clientes e Servidores)

* soft memlock unlimited
* hard memlock unlimited
* soft nofile 1048576
* hard nofile 1048576
* soft stack unlimited
* hard stack unlimited
* soft nproc unlimited # Ou valor alto como 65536
* hard nproc unlimited

memlock é vital para RDMA (InfiniBand).

4. SELinux e Firewall (Clientes e Servidores)

Para máxima performance em ambiente seguro:

- SELinux: SELINUX=disabled em /etc/selinux/config (requer reboot).
- Firewall: sudo systemctl disable --now firewalld. *Avalie os riscos de segurança.*

---

② Configurações de Rede InfiniBand (Clientes e Servidores BeeGFS)

- **Drivers OFED/RDMA:**
  - Instale o OFED do fabricante da HCA ou os pacotes rdma-core, libibverbs-utils,

librdmacm-utils, perftest e **libbeegfs-ib** (para suporte RDMA no BeeGFS).
Bash
sudo dnf install rdma-core libibverbs-utils librdmacm-utils perftest libbeegfs-ib

- **Subnet Manager (OpenSM):** Ativo na malha InfiniBand.
- **IPoIB:** Configure se necessário para gerenciamento. Dê um IP às interfaces IB. BeeGFS RDMA não usa IPoIB para dados, mas precisa de uma interface de rede "up" com IP para descoberta inicial em algumas configurações ou para TCP fallback.
- **Afinidade de IRQ para HCA InfiniBand:** Essencial. Desabilite irqbalance ou configure-o para ignorar a HCA. Use scripts para distribuir IRQs da HCA entre os núcleos da CPU do mesmo nó NUMA da HCA.
- **BeeGFS para usar RDMA:**
  - BeeGFS geralmente detecta e usa RDMA automaticamente se libbeegfs-ib estiver instalado e o hardware/drivers estiverem presentes.
  - Verifique nos logs dos serviços BeeGFS (/var/log/beegfs-*.log) se as conexões estão sendo estabelecidas via RDMA.
  - Nos arquivos de configuração BeeGFS (beegfs-*.conf em /etc/beegfs/), certifique-se de que:
    - connRDMAEnabled = true (geralmente é o padrão).
    - connInterfacesFile (ou connNetFilterFile, dependendo da versão) pode ser usado para especificar quais interfaces de rede devem ser usadas (ex: ib0). Se não especificado, BeeGFS tentará usar todas as disponíveis.

---

③ Configurações Específicas do BeeGFS

A. Em TODOS os Servidores BeeGFS (mgmtd, meta, storage) e Clientes:

- **/etc/beegfs/beegfs-common.conf** (se existir, ou parâmetros equivalentes nos arquivos individuais)
  - connMaxInternodeNum: Padrão é 8. Para clusters maiores ou com muita comunicação, pode ser necessário aumentar (ex: 16, 32). Teste o impacto no consumo de memória.
  - connAuthFile: Configure para segurança se necessário.
  - logLevel: Defina para 2 ou 3 para produção após a estabilização, para reduzir o overhead de logging (0 é quiet, 1 é erros críticos, 2 erros normais, 3 avisos, 4 info, 5 debug).

B. Servidores de Gerenciamento (beegfs-mgmtd)

- **/etc/beegfs/beegfs-mgmtd.conf**:
  - storeMgmtdDirectory: Diretório para dados de gerenciamento (geralmente em disco local rápido, SSD se possível, mas para mgmtd não é tão crítico quanto meta).

    - connInterfacesFile / connNetFilterFile / connRDMAEnabled.

C. Servidores de Metadados (beegfs-meta)

- **/etc/beegfs/beegfs-meta.conf**:
    - storeMetaDirectory: **CRUCIAL**. Idealmente, este diretório deve estar em SSDs/NVMe para performance de metadados. Se *obrigatoriamente* em HDDs (não recomendado para alta performance), o HDD deve usar mq-deadline e ter bom desempenho de acesso aleatório.
    - connInterfacesFile / connNetFilterFile / connRDMAEnabled.
    - tuneNumWorkers: Número de threads worker. Padrão (0) é auto-detect (baseado em núcleos). Pode ser ajustado (ex: 16, 32, 64). Monitore a carga da CPU.
    - tuneBindToNumaNode: Se o servidor meta for NUMA, pode especificar o nó para afinidade.
- **Sistema de Arquivos Local para storeMetaDirectory:** XFS ou ext4 com otimizações para metadados (ex: XFS com logbsize=256k, allocsize=4m).

D. Servidores de Armazenamento (beegfs-storage) - Onde os HDDs estão

- **Configurações do SO nos Servidores de Armazenamento:**
    - **Agendador de I/O:** Para os HDDs que hospedam os storeStorageDirectory: **mq-deadline** ou kyber. Configure via udev rules para persistência:
        - Crie /etc/udev/rules.d/60-hdd-scheduler.rules: ACTION=="add|change", KERNEL=="sd[a-z]", ATTR{queue/rotational}=="1", ATTR{queue/scheduler}="mq-deadline"

        - Recarregue: sudo udevadm control --reload-rules && sudo udevadm trigger
    - **Read-ahead:** Aumente para os HDDs.
        - sudo blockdev --setra 8192 /dev/sdX (para 4MB; teste valores como 2048, 4096, 16384). Pode ser configurado via udev também.
    - **Fila de Requisições (nr_requests):**
        - echo 256 > /sys/block/sdX/queue/nr_requests (ou mais, como 512, 1024; o padrão pode ser 128). Teste.
- **/etc/beegfs/beegfs-storage.conf**:
    - storeStorageDirectory: Caminho para os diretórios de armazenamento nos HDDs.
    - connInterfacesFile / connNetFilterFile / connRDMAEnabled.
    - tuneNumWorkers: Similar ao beegfs-meta. Padrão (0) é auto. Ajuste se necessário.
    - tuneBindToNumaNode.
    - tuneUseDirectIO = true: **Recomendado para storage servers** para bypassar o cache do SO para dados de arquivos, permitindo que o BeeGFS gerencie seu próprio cache e evitando double caching.
    - storeUseChunks = true (padrão).
    - logLevel = 2 (após estabilização).

- **Sistema de Arquivos Local para storeStorageDirectory (nos HDDs):** XFS é comumente usado. Formate com opções para arquivos grandes e bom alinhamento com a geometria do RAID (se houver). Ex:
  Bash
  mkfs.xfs -f -d sunit=128,swidth=1024 -L beegfs_target /dev/sdX # Exemplo para RAID com stripe de 64KB e 8 discos de dados (128*512B = 64KB, 1024*512B = 512KB)
  # Ajuste sunit/swidth conforme seu RAID ou se for disco único. Para disco único, não use swidth.
  Monte com noatime,nodiratime,largeio,inode64.

E. Nós Clientes (beegfs-client)

- **Opções de Montagem:**
  - Geralmente, o BeeGFS é montado via /etc/beegfs/beegfs-mounts.conf e o serviço beegfs-client.
  - Exemplo de linha em /etc/beegfs/beegfs-mounts.conf: /mnt/beegfs /etc/beegfs/beegfs-client.conf
  - Dentro do /etc/beegfs/beegfs-client.conf ou como opções de montagem passadas diretamente (menos comum para BeeGFS):
    - cfgFile=/etc/beegfs/beegfs-client.conf (implícito se usando beegfs-mounts.conf).
    - BeeGFS tentará RDMA por padrão se libbeegfs-ib estiver instalado.
    - tuneUseBufferedIO:
      - false (Direct I/O - padrão): Geralmente bom para grandes I/Os sequenciais e quando a aplicação faz seu próprio buffering ou usa MPI-IO com buffering. Evita double caching.
      - true (Buffered I/O): Pode ajudar se as aplicações fizerem muitos I/Os pequenos ou não alinhados, pois o kernel do cliente pode agregar I/Os e se beneficiar de seu readahead/writebehind. Para HDDs, isso **precisa ser testado cuidadosamente** com sua carga de trabalho. Pode melhorar a performance para certos padrões, mas degradar para outros.
    - Considere noatime, nodiratime (geralmente controlados pelo servidor BeeGFS, mas verifique se há opções no cliente).
- **/etc/beegfs/beegfs-client.conf**:
  - sysMgmtdHost = <hostname_ou_IP_do_servidor_mgmtd_beegfs>
  - connInterfacesFile / connNetFilterFile / connRDMAEnabled.
  - tuneNumWorkers: Padrão (0) é auto.
  - tunePreferredReadChunkSizeKB, tunePreferredWriteChunkSizeKB: Como o cliente agrupa leituras/escritas. Padrões costumam ser bons.
  - connRDMABufSizeKiB, connRDMABufNum: Ajustes para buffers RDMA. Padrões são um bom começo. Aumentar pode consumir mais memória, mas ajudar em redes muito rápidas.
  - logLevel = 1 ou 2 (após estabilização).

---

④ Aplicação e MPI (Nos Nós Clientes)

- **MPI:** Otimizada para InfiniBand (UCX ou openib BTL).
- **MPI-IO:**
  - Use para acesso paralelo.
  - Hints para MPI-IO (ex: para Open MPI ROMIO):
    - romio_cb_write=enable, romio_cb_read=enable (habilita I/O coletivo).
    - cb_buffer_size (aumente, ex: 16777216 para 16MB ou mais por processo).
    - romio_ds_write=disable, romio_ds_read=disable (desabilita data sieving se os acessos forem contíguos).
    - Alinhe os acessos ao tamanho de stripe do BeeGFS (beegfs-ctl --getentryinfo --mount=/mnt/beegfs /mnt/beegfs/algum_arquivo para ver o chunksize e numtargets).

---

⑤ Teste e Monitoramento Contínuo

- **BeeGFS:**
  - beegfs-ctl --listnodes --nodetype=all --details (verificar conexões RDMA).
  - beegfs-ctl --clientstats --interval=5 --mount=/mnt/beegfs (no cliente).
  - beegfs-df (uso do sistema de arquivos).
- **Sistema:** iostat -xNymz 1 (especialmente nos HDDs dos storage servers, observar %util, avgqu-sz, await), vmstat 1, sar, pcp.
- **Rede IB:** ibtop, ib_read_bw, osu_bw, osu_latency.
- **Benchmarks:** IOR, mdtest, fio (simulando seu padrão de I/O), e, mais importante, suas aplicações reais.

Considerações Cruciais para BeeGFS com HDDs:

- **Metadata em SSDs:** Se possível, mesmo que os dados estejam em HDDs, coloque os storeMetaDirectory dos servidores beegfs-meta em SSDs/NVMe. Isso acelera drasticamente as operações de metadados (listar diretórios, abrir arquivos, etc.).
- **Padrão de I/O:** BeeGFS em HDDs se beneficiará enormemente de I/Os grandes e sequenciais. Otimize suas aplicações para isso.
- **Balanceamento:** Certifique-se de que os dados e metadados estejam bem distribuídos entre os servidores e targets.
- **Teste, Teste, Teste:** A configuração "mais performática" é altamente dependente da sua carga de trabalho específica. Use este guia como um ponto de partida robusto e ajuste com base em benchmarking com suas aplicações.

Esta configuração detalhada deve fornecer uma base sólida para alcançar alta performance com BeeGFS em seu ambiente HPC.