

PRÉ-REQUISITOS · IMPLANTAÇÃO

Pré-requisitos: Implantação da plataforma GitLab™ em Kubernetes

Documento enviado após a escolha do serviço PS-IMP-02: o que o cliente provisiona, as credenciais que entrega à Pointer, as informações a enviar e o checklist de prontidão.

CÓDIGO

PR-IMP-02

TIPO

Pré-requisitos

VERSÃO

1.2

DATA

1 de outubro de 2026

O3S Consultoria e Tecnologia da Informação Ltda.
contato@pointertech.digital · +55 (61) 99874-4648 · pointertech.digital

GITLAB is a trademark of GitLab Inc. in the United States and other countries and regions.



GitLab
Select
Partner

PRÉ-REQUISITOS · PR-IMP-02 · OFERTA PS-IMP-02

Pré-requisitos: Implantação da plataforma GitLab em Kubernetes

Este documento é enviado depois que o cliente escolhe o serviço **PS-IMP-02 — Implantação da plataforma GitLab em Kubernetes**. É dirigido às equipes de infraestrutura, plataforma Kubernetes, banco de dados, rede e segurança do cliente e reúne:

1. a infraestrutura que o cliente provisiona: cluster, serviços externos e, no Cloud Native Hybrid, VMs de backend; ou, quando o pipeline cria o Cloud Native Hybrid na AWS (EKS e VMs de backend), a conta, a identidade e as cotas que o cliente fornece;
2. as credenciais que o cliente entrega à Pointer e a forma de entrega;
3. o formulário com as informações não secretas do ambiente;
4. o checklist de prontidão, com o que é conferido automaticamente pelo pipeline e o que é conferido na reunião de prontidão.

As especificações seguem as arquiteturas de referência GitLab Cloud Native e Cloud Native Hybrid (documentação oficial consultada em 29/09/2026) e a matriz de versões do GitLab Helm chart. O prazo de devolução do formulário e do checklist é definido na reunião de kick-off. A instalação só começa depois de uma verificação prévia (preflight) sem erros contra o cluster e os serviços externos reais.

Os recursos opcionais (GitLab Operator, Container Registry, frota de runners do cliente, agentes GitLab para Kubernetes e Grafana gerenciado, este sob consulta) têm seções próprias e só se aplicam quando contratados.

Conteúdo

Infraestrutura que o cliente provisiona

Cloud Native Hybrid criado pelo pipeline na AWS

GitLab Operator e atualização sem parada

Frota de runners do cliente e agentes GitLab para Kubernetes

Monitoramento: Prometheus e Grafana gerenciado (sob consulta)

Versões de Kubernetes

Capacidade do cluster: Cloud Native

Capacidade do cluster e VMs: Cloud Native Hybrid

Conectividade de rede

Entrega das credenciais

Como a Pointer usa essas informações

Critérios de aceite

Credenciais que o cliente entrega à Pointer

Informações a enviar

Checklist de prontidão

1. Infraestrutura que o cliente provisiona

Item	Especificação	Obrigatório
Cluster Kubernetes	Cluster existente, administrado pelo cliente, em versão suportada pelo GitLab Helm chart (tabela <i>Versões de Kubernetes</i>), ou, no Cloud Native Hybrid na AWS, EKS criado pelo pipeline (seção <i>Cloud Native Hybrid criado pelo pipeline na AWS</i>). Distribuições com guia oficial do chart: AKS, EKS, GKE, OpenShift e OKE; RKE2, K3s ou outra distribuição conforme são aceitas com alerta. Segundo a documentação GitLab, o chart foi projetado para cluster com pelo menos 8 vCPU e 30 GB de memória, e rede, storage classes e autenticação do Kubernetes ficam fora do escopo do suporte da GitLab.	Sim
Nós	Nós prontos e agendáveis, com arquitetura amd64 ou arm64 (arm64: alerta; outra: erro). Imagens FIPS só existem para x86-64 (documentação do chart). A documentação GitLab orienta evitar instâncias <i>burstable</i> (desempenho inconsistente).	Sim
Grupos de nós e rótulos	Com rótulos (padrão), os nós levam o rótulo workload com os valores webservice , sidekiq , support e, na Cloud Native, gitaly , e o preflight soma a capacidade alocável de cada grupo. Sem rótulos, o preflight compara a soma do cluster com a soma dos grupos. Na Cloud Native, um nó para cada pod de Gitaly (antiafinidade obrigatória; 3 pods nas arquiteturas S a XL). A documentação GitLab aceita Webservice, Sidekiq e suporte em um grupo único dimensionado para os três e exige nós dedicados para o Gitaly.	Sim
Capacidade alocável	Conforme as tabelas por arquitetura. Por grupo de nós: abaixo de 90% do mínimo (réplicas mínimas × request por pod), erro; abaixo do necessário para as réplicas máximas, alerta. Os totais cobrem só os componentes GitLab; os processos de sistema do Kubernetes exigem recursos adicionais (documentação GitLab).	Sim

Item	Especificação	Obrigatório
StorageClass	Cloud Native: StorageClass existente para os volumes persistentes do Gitaly (o GET não cria StorageClass em cluster próprio), com tamanho por pod definido no formulário (padrão 50 GiB; mínimo 10 GiB; abaixo de 50 GiB, alerta). Preflight: StorageClass inexistente, erro; sem expansão de volume (<code>allowVolumeExpansion</code>), alerta; cluster sem StorageClass padrão, alerta. Documentação do chart: SSD e <code>reclaimPolicy: Retain</code> para a StorageClass padrão; alguns provedores, como o EKS, não têm StorageClass padrão; mudar o armazenamento depois da primeira implantação exige edição manual de objetos do Kubernetes.	Condicional: Cloud Native
Namespace e políticas	Namespace dedicado (padrão <code>gitlab</code> ; <code>default</code> gera alerta), criado pelo pipeline se não existir; release Helm sempre com o nome <code>gitlab</code> (release existente: alerta, e a execução vira atualização). Cotas (ResourceQuota, LimitRange) e políticas (Pod Security, OPA, Kyverno) do cluster são levantadas no discovery.	Sim
LoadBalancer	Provedor de Services do tipo LoadBalancer para o Envoy Gateway, padrão do chart desde a versão GitLab 19.0: balanceador da nuvem ou, em data center, soluções como MetalLB ou ServiceLB; IP fixo por anotações do Service. Em RKE2, K3s, OpenShift ou outra distribuição, o preflight emite alerta para confirmar esse provedor. Opção NGINX Ingress: IP externo obrigatório; segundo a documentação do chart, está deprecated e disponível até ser removido na versão 20.0.	Sim
DNS	Registros do host da URL externa, de <code>registry.<host></code> (ou do nome do Container Registry) e de <code>kas.<host></code> apontando para o endereço do LoadBalancer do Gateway, como exige a documentação do GitLab Helm chart; um registro wildcard <code>*.<host></code> cobre registry e kas. A verificação pós-implantação confere o host da URL externa e <code>kas.<host></code> . GitLab Pages: registro DNS wildcard do domínio do Pages. O pipeline não cria registros DNS, inclusive no EKS criado por ele; na troca do Helm chart pelo GitLab Operator, o endereço do LoadBalancer muda e os registros são atualizados.	Sim
Porta SSH do Git	Padrão 22, no listener SSH do Envoy Gateway; outra porta pode ser informada no formulário.	Não
Certificado TLS	Certificado do host externo e chave privada em PEM, com os intermediários da cadeia; SAN obrigatório, cobrindo os hostnames usados, inclusive <code>kas.<host></code> (com o KAS habilitado, padrão, a validação emite alerta para conferir) e o nome do Container Registry, quando habilitado. É publicado como Secret usado pelo Gateway nos listeners da instância GitLab, do registry, do Pages e do KAS. Alternativa: Let's Encrypt pelo cert-manager, com e-mail de registro. GitLab Pages com HTTPS: certificado wildcard do domínio do Pages.	Condicional: certificado do cliente (padrão)

Item	Especificação	Obrigatório
PostgreSQL externo	PostgreSQL 17 para GitLab 19.x (documentação GitLab: versão mínima e máxima 17.x, sempre na última minor), alcançável a partir dos pods. <code>max_locks_per_transaction</code> de pelo menos 128 (exigência do pipeline: com o valor 64, a criação do schema falha com <i>out of shared memory</i>; 128 é o padrão do Linux package). Extensões exigidas pela sonda: <code>pg_trgm</code>, <code>btree_gist</code>, <code>amcheck</code>, <code>plpgsql</code> e <code>pg_stat_statements</code> (a documentação GitLab lista as quatro primeiras como obrigatórias). Com usuário administrador, o GET cria usuário, banco e extensões; sem ele (alerta), o usuário da aplicação, o banco (padrão <code>gitlabhq_production</code>) e as extensões precisam existir antes. Parâmetros obrigatórios da documentação GitLab para banco externo, conferidos pela sonda com alerta quando fora do exigido: <code>work_mem</code> ≥ 8 MB, <code>maintenance_work_mem</code> ≥ 64 MB, <code>max_connections</code> ≥ 400, <code>shared_buffers</code> ≥ 2 GB e <code>statement_timeout</code> entre 15.000 e 60.000 ms; com balanceamento de carga do banco, <code>hot_standby_feedback</code> = <code>on</code> (não conferido pela sonda). Serviços gerenciados: instalar extensões exige <code>rds_superuser</code> (Amazon RDS), <code>azure_pg_admin</code> (Azure Database for PostgreSQL; no Flexible Server, com allow-list das extensões) ou <code>cloudsqlsuperuser</code> (Cloud SQL); Amazon Aurora e Google AlloyDB são incompatíveis.	Condicional: obrigatório na Cloud Native; no Hybrid, alternativa às VMs de PostgreSQL
Redis externo	Redis 7.0 ou superior (abaixo de 7.0: erro; abaixo de 7.2: alerta), instância standalone com ou sem alta disponibilidade (Redis Cluster e variantes serverless não são suportados), com eviction policy configurável, autenticação por senha e TLS opcional; alcançável a partir dos pods (a sonda testa conexão, AUTH e PING de dentro do cluster). A arquitetura Cloud Native especifica duas instâncias, cache e persistente (tabela por arquitetura); o formulário do pipeline recebe um único endereço de Redis. Serviços citados pela documentação GitLab: Google Memorystore e Amazon ElastiCache for Valkey 7.2; na Azure, Redis ou Valkey autogerenciado em VM.	Condicional: obrigatório na Cloud Native; no Hybrid, alternativa às VMs de Redis
Object storage e buckets	Serviço com API S3-compatível (endpoint, região, endereçamento <i>path style</i> quando o serviço exigir e credenciais de acesso) e I3 buckets criados antes, com os nomes exatos e o prefixo definido no formulário (padrão <code>gitlab</code>): <code><prefixo>-artifacts</code>, <code><prefixo>-dependency-proxy</code>, <code><prefixo>-lfs</code>, <code><prefixo>-packages</code>, <code><prefixo>-terraform-state</code>, <code><prefixo>-uploads</code>, <code><prefixo>-pages</code>, <code><prefixo>-mr-diffs</code>, <code><prefixo>-ci-secure-files</code>, <code><prefixo>-backups</code>, <code><prefixo>-agent-plan-content</code>, <code><prefixo>-ci-catalog-bundles</code> e <code><prefixo>-tmp</code>, e, com Container Registry, <code><prefixo>-registry</code>. A sonda confere cada bucket de dentro do cluster (inexistente ou inacessível: erro). A documentação do chart exige um bucket separado por tipo; caso contrário, a restauração de backup falha. No Cloud Native Hybrid criado pelo pipeline na AWS, os buckets são criados pelo pipeline (padrão).	Sim

Item	Especificação	Obrigatório
VMs de backend	VMs conforme a tabela <i>Cloud Native Hybrid</i> , com os mesmos requisitos das VMs da implantação em servidores Linux (PR-IMP-01): sistema operacional da matriz (suportados Ubuntu 22.04 e 24.04, Debian 12 e 13, Red Hat Enterprise Linux 9 e 10 e Amazon Linux 2023), instalação limpa, usuário com sudo sem senha acessível por SSH a partir do runner, <code>python3</code> , NTP, volume de <code>/var/opt/gitlab</code> e, a partir de 3k, latência menor que 5 ms entre os nós. Com PostgreSQL externo, não há VMs de PostgreSQL, PgBouncer, Consul e PostgreSQL do Praefect (nem balanceador interno, quando também não há Praefect); com Redis externo, não há VMs de Redis.	Condicional: Cloud Native Hybrid
Conta de serviço e kubeconfig	Conta de serviço dedicada ao engajamento com permissão <code>cluster-admin</code> (o GET cria namespace, CRDs do Gateway API e recursos de escopo de cluster), conferida pelo preflight como <code>kubectl auth can-i '*' '*' --all-namespaces</code> . Kubeconfig com a CA do cluster e o contexto informado no formulário (padrão: contexto atual), gerado com o administrador do cluster e revogado no encerramento. No EKS criado pelo pipeline, dispensado: o kubeconfig é gerado a partir do estado do Terraform, com a identidade AWS do ambiente.	Condicional: cluster existente
Saída de rede dos nós do cluster	registry.gitlab.com (imagens Cloud Native GitLab, inclusive a imagem toolbox da versão alvo usada pela sonda) e docker.io (Envoy Gateway), ou registry espelho configurado no cluster. A documentação do GitLab.com lista também <code>.storage.googleapis.com</code> e <code>cdn.registry.gitlab-static.net</code> para download de imagens de registry.gitlab.com. Com o Prometheus instalado pelo pipeline (padrão), também quay.io, registry.k8s.io e docker.io (imagens do kube-prometheus-stack).	Sim
VM do runner	Linux com Docker Engine, 2 vCPU, 4 GB de memória e 40 GB de disco (a imagem do pipeline ocupa cerca de 6 GB). Somente conexões de saída, nenhuma porta de entrada; acesso à API do cluster (normalmente 443 ou 6443), à URL externa e, no Hybrid, SSH às VMs de backend. Com proxy corporativo, o proxy é configurado no serviço do runner.	Sim
Serviços do cliente	LDAP ou Active Directory, SMTP e cluster de busca externo, quando habilitados, alcançáveis a partir dos pods nas portas informadas no formulário. Na Cloud Native, a busca avançada só funciona com cluster externo. A documentação do chart registra que o GKE bloqueia a porta 25.	Condicional: recurso habilitado
Licença	Assinatura GitLab Premium ou GitLab Ultimate: activation code (ativação online) ou arquivo de licença (offline). O GET exige Premium ou superior; a edição Community Edition é recusada pela validação, e a ausência de licença gera alerta.	Sim

2. Cloud Native Hybrid criado pelo pipeline na AWS

Item	Especificação	Obrigatório
Conta e identidade	Usuário ou role IAM dedicado ao engajamento, com access key cadastrada como variável protegida com escopo do ambiente, com permissão para criar os recursos do módulo AWS do GitLab Environment Toolkit (VMs, rede, grupos de segurança, roles IAM, buckets e o cluster EKS), leitura da chave KMS gerenciada <code>alias/aws/s3</code> e acesso ao cluster criado (a mesma identidade gera o token de acesso ao EKS nos jobs de implantação). O módulo EKS do GET sempre cria o provedor OpenID Connect do IAM do cluster: a política da identidade e as políticas da organização (SCP do AWS Organizations) precisam permitir as ações do IAM sobre provedores OpenID Connect (<code>iam:CreateOpenIDConnectProvider</code>, <code>iam:GetOpenIDConnectProvider</code>, <code>iam>DeleteOpenIDConnectProvider</code>, <code>iam:ListOpenIDConnectProviders</code>, <code>iam:UpdateOpenIDConnectProviderThumbprint</code>, <code>iam:AddClientIDToOpenIDConnectProvider</code>, <code>iam:RemoveClientIDFromOpenIDConnectProvider</code>, <code>iam:TagOpenIDConnectProvider</code>, <code>iam:UntagOpenIDConnectProvider</code> e <code>iam:ListOpenIDConnectProviderTags</code>). Para aumentar um pool depois da criação, também <code>eks:ListNodegroups</code>, <code>eks:DescribeNodegroup</code> e <code>eks:UpdateNodegroupConfig</code>. Revogada no encerramento.	Condicional: Hybrid criado pelo pipeline
Região e cotas	Região definida no formulário, com cotas da conta suficientes para as VMs de backend, os nós dos pools do EKS, os endereços IP públicos (com rede nova e subnets privadas, um NAT gateway com um Elastic IP para cada par de subnets pública e privada) e os balanceadores.	Condicional: Hybrid criado pelo pipeline
Rede	Rede nova (CIDR /24 ou maior) ou VPC e subnets existentes; CIDRs com acesso às VMs; com subnets privadas, grupo de segurança de acesso para o runner do engajamento. O endpoint da API do EKS é privado: o runner do engajamento fica na rede criada ou com rota até ela.	Condicional: Hybrid criado pelo pipeline
Cluster EKS	Versão da matriz do GitLab Helm chart (tabela <i>Versões de Kubernetes</i>); pools Webservice, Sidekiq e suporte com tipo e quantidade do formulário ou padrão do pipeline (c7i.2xlarge no Webservice; m7i.xlarge no Sidekiq e no suporte; quantidade calculada pelos totais da arquitetura, abaixo deles com alerta); ARNs IAM de administradores do cliente com acesso ao cluster (opcional). Aumento de pool por merge request.	Condicional: Hybrid criado pelo pipeline
Buckets, tags e estado	Buckets criados pelo pipeline (padrão) ou existentes; tags de custo e governança do cliente em todos os recursos; estado no GitLab-managed Terraform state do projeto do engajamento, com transferência ao cliente definida no encerramento; destruição só por decisão registrada em merge request.	Condicional: Hybrid criado pelo pipeline

Item	Especificação	Obrigatório
DNS e certificados	Continuam com o cliente: registros de <code>gitlab</code> , <code>registry</code> e <code>kas</code> para o LoadBalancer do Envoy Gateway e certificado com esses nomes.	Condicional: Hybrid criado pelo pipeline

- Sob consulta: AKS e GKE criados pelo pipeline (clusters existentes nesses provedores entram como cluster existente) e Cloud Native (S a XL) com cluster criado pelo pipeline.
- O Terraform roda em runners hospedados do GitLab.com (padrão; usa só a API da AWS) ou em runner indicado pelo cliente.

3. GitLab Operator e atualização sem parada

- GitLab Operator: GitLab 17.6.0 ou superior; o GET instala cert-manager, Envoy Gateway, o GitLab Operator e o recurso GitLab com os mesmos values do Helm chart, com alerta da validação (suporte Beta no GET e limitações conhecidas do Operator na documentação GitLab); a reexecução das migrations pelo pipeline não se aplica ao Operator. GitLab Operator em OpenShift: sob consulta.
- Troca de uma instalação pelo Helm chart para o GitLab Operator: backup sob demanda e cópia dos segredos antes, janela de manutenção (a instância fica fora do ar até o DNS apontar para o balanceador novo do Gateway) e atualização dos registros de `gitlab`, `registry` e `kas`.
- Atualização sem parada: Cloud Native Hybrid com backends em alta disponibilidade (Patroni e Consul, 3k ou mais) ou com PostgreSQL externo, e 2 ou mais réplicas de Webservice e Sidekiq; uma versão minor por vez. No Hybrid 2k com PostgreSQL em VM e na Cloud Native, a atualização usa janela de manutenção.

4. Frota de runners do cliente e agentes GitLab para Kubernetes

Item	Especificação	Obrigatório
Frota em Kubernetes	Chart oficial gitlab-runner no namespace informado (padrão <code>gitlab-runner</code>), em cluster existente (kubeconfig com permissão para criar namespace, Secrets e o release Helm) ou no EKS criado pelo pipeline (acesso lido do estado do Terraform). A frota também pode usar VMs com executor Docker e docker-autoscaler na AWS, com os requisitos do documento PR-IMP-01. Executor shell e docker-autoscaler no Azure: sob consulta.	Condicional: frota contratada
Token de API da frota	Token de acesso com escopos <code>create_runner</code> e <code>manage_runner</code> ; de administrador da instância para runners de instância ou de Owner do grupo para runners de grupo.	Condicional: frota contratada

Item	Especificação	Obrigatório
Governança da frota	Por runner: escopo e alvo, tags, execução sem tag, proteção, timeout máximo, concorrência, imagem padrão, modo privilegiado (alerta), requests de CPU e memória dos pods de job e versão do chart. Métricas pelo chart, quando habilitadas.	Condicional: frota contratada
KAS na instância	Habilitado por padrão em <code>kas.<host></code> , pelo mesmo LoadBalancer da instância GitLab, com o certificado do cliente cobrindo esse nome.	Condicional: agentes contratados
Token de API dos agentes	Token de acesso com escopo <code>api</code> de usuário Maintainer dos projetos de configuração dos agentes; o grupo desses projetos precisa existir.	Condicional: agentes contratados
Acesso a cada cluster com agente	Kubeconfig com permissão para criar o namespace do agente, o Secret do token e o release Helm, inclusive as permissões de escopo de cluster que o chart cria para o agente (padrão do chart: cluster-admin, com alerta da validação; ClusterRole do cliente opcional); no EKS criado pelo pipeline, acesso lido do estado do Terraform.	Condicional: agentes contratados
Rede dos agentes	Agente com saída para <code>kas.<host></code> na 443 (WebSocket seguro); nós do cluster com acesso ao registry da imagem do agente; runner do engajamento com acesso ao API server do cluster e a charts.gitlab.io.	Condicional: agentes contratados
CA da instância	Com autoridade certificadora não pública, a cadeia da AC da URL externa, para os runners e o agente confiarem na instância.	Condicional: AC não pública

5. Monitoramento: Prometheus e Grafana gerenciado (sob consulta)

Item	Especificação	Obrigatório
Prometheus (kube-prometheus-stack)	Instalado pelo GET no namespace <code>monitoring</code> (padrão; pode ser desligado quando o cliente tem monitoramento próprio), com volume persistente de 100 GB (padrão) na StorageClass informada, na do cluster ou, no EKS criado pelo pipeline, numa StorageClass gp3 criada pela implantação. O runner baixa o chart de prometheus-community.github.io e os nós baixam as imagens de quay.io, registry.k8s.io e docker.io (ou do registry espelho). A verificação pós-implantação consulta os alvos ativos pelo proxy de serviços do API server.	Não (padrão: habilitado)
Coleta das VMs de backend (Hybrid)	Do Prometheus no cluster para as VMs de backend: TCP 9100 (node exporter), 9187 (PostgreSQL), 9188 (PgBouncer), 9121 (Redis), 9236 (Gitaly), 9652 (Praefect) e 1936 (HAProxy interno); com Consul em VM, agente Consul no cluster com acesso ao Consul das VMs em TCP 8300 e TCP e UDP 8301.	Condicional: Hybrid com Prometheus

Item	Especificação	Obrigatório
Grafana gerenciado pela Pointer (sob consulta)	Subchart do kube-prometheus-stack com HTTPS nativo, Service LoadBalancer (anotações do formulário) e registro DNS do host do Grafana para ele; certificado e chave desse host; volume persistente (padrão 10 GB). Login pela instância GitLab: aplicação OAuth criada na instância pela Pointer (redirect <code>https://<host do Grafana>/login/gitlab</code> ; escopos <code>openid</code> , <code>email</code> e <code>profile</code>) e grupos com acesso e com papel de administrador ou editor; o Grafana chama a URL externa da instância e, com AC não pública, recebe a cadeia da AC.	Condicional: Grafana contratado

6. Versões de Kubernetes

Kubernetes	Situação no GitLab Helm chart	Versão GitLab mínima	Resultado no preflight
1.37	Suportada	19.5	Aceita; erro com versão GitLab abaixo de 19.5
1.36	Suportada	19.3	Aceita; erro com versão GitLab abaixo de 19.3
1.35	Suportada	18.9	Aceita
1.34	Deprecated	18.6	Alerta
1.33 ou anterior	Não suportada	—	Erro
Acima de 1.37	Fora da matriz do pipeline	—	Alerta

- Fonte: documentação do GitLab Helm chart (tabela de versões de Kubernetes). O chart suporta três versões minor por vez e passa a suportar uma nova versão três meses após o lançamento.
- O GET escolhe a versão do chart correspondente à versão GitLab do engajamento: por exemplo, GitLab 19.4.1 corresponde ao chart 10.4.1. Com GitLab 19.4.x, Kubernetes 1.37 gera erro.
- kubectl e Helm usados pelo pipeline vêm da imagem do GET; o cliente não instala essas ferramentas. O preflight emite alerta quando o cluster fica a mais de uma versão minor do kubectl da imagem (1.35).

7. Capacidade do cluster: Cloud Native

Item	S	M	L	XL
Carga de referência	até 100 RPS	até 200 RPS	até 500 RPS	até 1.000 RPS
Web service: pods (mín.–máx.); por pod 4 vCPU / 5 GB de request, 7 GB de limite	6–9	14–21	28–42	56–84

Item	S	M	L	XL
Grupo webservice : alocável mínimo / para réplicas máximas	24 vCPU · 30 GB / 36 vCPU · 45 GB	56 vCPU · 70 GB / 84 vCPU · 105 GB	112 vCPU · 140 GB / 168 vCPU · 210 GB	224 vCPU · 280 GB / 336 vCPU · 420 GB
Sidekiq: pods (mín.-máx.); por pod 0,9 vCPU / 2 GB de request, 4 GB de limite	8–12	16–24	32–48	64–96
Grupo sidekiq : alocável mínimo / para réplicas máximas	7,2 vCPU · 16 GB / 10,8 vCPU · 24 GB	14,4 vCPU · 32 GB / 21,6 vCPU · 48 GB	28,8 vCPU · 64 GB / 43,2 vCPU · 96 GB	57,6 vCPU · 128 GB / 86,4 vCPU · 192 GB
Gitaly: pods × request (igual ao limite)	3 × 7 vCPU / 30 GB	3 × 15 vCPU / 62 GB	3 × 31 vCPU / 126 GB	3 × 63 vCPU / 254 GB
Grupo gitaly : 3 nós dedicados, alocável total	21 vCPU · 90 GB	45 vCPU · 186 GB	93 vCPU · 378 GB	189 vCPU · 762 GB
Grupo support (serviços de suporte)	12 vCPU · 48 GB	12 vCPU · 48 GB	12 vCPU · 48 GB	24 vCPU · 96 GB
Total alocável mínimo no cluster (aritmética)	64,2 vCPU · 184 GB	127,4 vCPU · 336 GB	245,8 vCPU · 630 GB	494,6 vCPU · 1.266 GB
PostgreSQL externo	8 vCPU / 32 GB	16 vCPU / 64 GB	32 vCPU / 128 GB	64 vCPU / 256 GB
Redis cache externo	2 vCPU / 8 GB	2 vCPU / 8 GB	2 vCPU / 16 GB	2 vCPU / 16 GB
Redis persistente externo	2 vCPU / 8 GB	2 vCPU / 8 GB	2 vCPU / 16 GB	2 vCPU / 16 GB
Object storage	S3-compatível, 13 buckets	S3-compatível, 13 buckets	S3-compatível, 13 buckets	S3-compatível, 13 buckets

- Fonte: arquitetura de referência GitLab Cloud Native (documentação oficial, consultada em 29/09/2026). Alocável mínimo = réplicas mínimas × request por pod; para réplicas máximas = réplicas máximas × request por pod; totais por aritmética.
- Segundo a documentação GitLab, os requests e limites dos pods consideram a capacidade do nó menos a reserva para processos do Kubernetes (2 GB de memória e 1 vCPU por nó), e o mínimo de pods fica em torno de 2/3 do máximo.
- Volume de cada pod de Gitaly: tamanho definido no formulário (padrão 50 GiB), dimensionado pelo volume de repositórios. A página Cloud Native da documentação GitLab não especifica tamanho, classe ou IOPS desses volumes.

8. Capacidade do cluster e VMs: Cloud Native Hybrid

Item	2k	3k	5k	10k	25k	50k
Carga de referência	até 40 RPS / 2.000 usuários	até 60 RPS / 3.000 usuários	até 100 RPS / 5.000 usuários	até 200 RPS / 10.000 usuários	até 500 RPS / 25.000 usuários	até 1.000 RPS / 50.000 usuários
Grupo webservice (request total)	12 vCPU · 15 GB	16 vCPU · 20 GB	36 vCPU · 45 GB	80 vCPU · 100 GB	140 vCPU · 175 GB	308 vCPU · 385 GB
Grupo sidekiq (request total)	3,6 vCPU · 8 GB	7,2 vCPU · 16 GB	7,2 vCPU · 16 GB	12,6 vCPU · 28 GB	12,6 vCPU · 28 GB	12,6 vCPU · 28 GB

Item	2k	3k	5k	10k	25k	50k
Grupo support	4 vCPU · 15 GB	4 vCPU · 15 GB	4 vCPU · 15 GB	8 vCPU · 30 GB	8 vCPU · 30 GB	8 vCPU · 30 GB
Total no cluster (aritmética)	19,6 vCPU · 38 GB	27,2 vCPU · 51 GB	47,2 vCPU · 76 GB	100,6 vCPU · 158 GB	160,6 vCPU · 233 GB	328,6 vCPU · 443 GB
VM Consul	—	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB
VM PostgreSQL	1 × 2 vCPU / 7,5 GB	3 × 2 vCPU / 7,5 GB	3 × 4 vCPU / 15 GB	3 × 8 vCPU / 30 GB	3 × 16 vCPU / 60 GB	3 × 32 vCPU / 120 GB
VM PgBouncer	—	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB
VM balanceador interno (HAProxy)	—	1 × 4 vCPU / 3,6 GB	1 × 4 vCPU / 3,6 GB	1 × 4 vCPU / 3,6 GB	1 × 8 vCPU / 7,2 GB	1 × 16 vCPU / 14,4 GB
VM Redis (com Sentinel a partir de 3k)	1 × 1 vCPU / 3,75 GB	3 × 2 vCPU / 7,5 GB	3 × 2 vCPU / 7,5 GB	—	—	—
VM Redis cache	—	—	—	3 × 4 vCPU / 15 GB	3 × 4 vCPU / 15 GB	3 × 4 vCPU / 15 GB
VM Redis persistente	—	—	—	3 × 4 vCPU / 15 GB	3 × 4 vCPU / 15 GB	3 × 4 vCPU / 15 GB
VM Gitaly	1 × 4 vCPU / 15 GB	3 × 4 vCPU / 15 GB	3 × 8 vCPU / 30 GB	3 × 16 vCPU / 60 GB	3 × 32 vCPU / 120 GB	3 × 64 vCPU / 240 GB
VM Praefect (somente Gitaly Cluster)	—	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 2 vCPU / 1,8 GB	3 × 4 vCPU / 3,6 GB	3 × 4 vCPU / 3,6 GB
VM PostgreSQL do Praefect (somente Gitaly Cluster)	—	1 × 2 vCPU / 1,8 GB	1 × 2 vCPU / 1,8 GB	1 × 2 vCPU / 1,8 GB	1 × 2 vCPU / 1,8 GB	1 × 2 vCPU / 1,8 GB
Total de VMs com Gitaly Cluster	3 VMs · 7 vCPU · 26,25 GB	20 VMs · 48 vCPU · 111,6 GB	20 VMs · 66 vCPU · 179,1 GB	23 VMs · 120 vCPU · 381,6 GB	23 VMs · 202 vCPU · 660,6 GB	23 VMs · 354 vCPU · 1.207,8 GB
Total de VMs com Gitaly Sharded	3 VMs · 7 vCPU · 26,25 GB	16 VMs · 40 vCPU · 104,4 GB	16 VMs · 58 vCPU · 171,9 GB	19 VMs · 112 vCPU · 374,4 GB	19 VMs · 188 vCPU · 648 GB	19 VMs · 340 vCPU · 1.195,2 GB

- Fonte: seção Cloud Native Hybrid das arquiteturas de referência GitLab 2k a 50k (documentação oficial, consultada em 29/09/2026). Grupos de nós: totais de request dos componentes GitLab; os processos de sistema do Kubernetes exigem recursos adicionais. Totais por aritmética.
- No Hybrid, balanceador externo, GitLab Rails, Sidekiq e monitoramento não ficam em VM. Gitaly Cluster ou Sharded só a partir de 3k.
- Com PostgreSQL externo, não há VMs de PostgreSQL, PgBouncer, Consul e PostgreSQL do Praefect; segundo a documentação GitLab, o PgBouncer empacotado só funciona com o PostgreSQL empacotado. Com Redis externo, não há VMs de Redis.
- Segundo a documentação GitLab, o Praefect exige PostgreSQL de terceiros para alta disponibilidade; com PostgreSQL de terceiros, o banco do Praefect pode ficar no mesmo servidor do banco principal.
- Busca avançada no modo de nós: VMs OpenSearch adicionais (mínimo 1; abaixo de 3, alerta).

9. Conectividade de rede

Origem	Destino	Porta	Finalidade
Usuários e runners de CI/CD	IP do LoadBalancer (URL externa)	443 (HTTPS)	Acesso web, API e Git por HTTPS
Usuários	IP do LoadBalancer	22 (padrão)	SSH do Git
VM do runner	gitlab.com, registry.gitlab.com, *.gitlab-static.net, charts.gitlab.io, docker.io	443 (HTTPS)	Projeto do engajamento, imagens do pipeline, GitLab Helm chart e CRDs do Envoy Gateway baixados pelo GET
VM do runner	*.storage.googleapis.com	443 (HTTPS)	Download de imagens de registry.gitlab.com, conforme a documentação do GitLab.com
VM do runner	API do cluster	Normalmente 443 ou 6443	kubectl e Helm
VM do runner	URL externa da instância	443 (HTTPS)	Health check do GET e verificação pós-implantação
VM do runner	VMs de backend (Hybrid)	SSH (padrão 22)	Coleta de fatos e execução do GET
Nós do cluster	registry.gitlab.com e docker.io, ou registry espelho	443 (HTTPS)	Imagens Cloud Native GitLab e Envoy Gateway
Pods	PostgreSQL externo	5432 (padrão)	Banco de dados principal
Pods	Redis externo	6379 (padrão)	Redis da instância
Pods	Object storage	Porta do endpoint	Objetos e backups
Pods	VMs de backend (Hybrid)	Portas padrão do Linux package: PostgreSQL 5432, PgBouncer 6432, Redis 6379, Redis Sentinel 26379, Gitaly 8075 (9999 com TLS), Praefect 2305 (3305 com TLS)	Acesso aos componentes com estado; PgBouncer e Praefect pelo balanceador interno

Origem	Destino	Porta	Finalidade
VMs de backend (Hybrid)	Outras VMs de backend	Consul 8300, 8301 (TCP e UDP), 8500, 8600; Patroni 8008; demais da tabela oficial	Comunicação entre os componentes em VM
Todas as VMs de backend (Hybrid)	packages.gitlab.com	443 (HTTPS)	Linux package e chave GPG do repositório oficial
Todas as VMs de backend (Hybrid)	Repositórios de pacotes do sistema operacional	Conforme o repositório	Pacotes instalados pelo GET
Pods	LDAP, SMTP e busca externa	Portas informadas no formulário	Autenticação, e-mail e busca avançada
Clusters com agente GitLab para Kubernetes	kas. (LoadBalancer do Gateway)	443 (WebSocket seguro)	Conexão do agente ao KAS
Prometheus no cluster (Hybrid)	VMs de backend	9100, 9187, 9188, 9121, 9236, 9652 e 1936; com Consul em VM, 8300 e 8301 (TCP e UDP)	Coleta de métricas dos backends em VM
Runner do provisionamento (runners hospedados do GitLab.com, padrão, ou runner indicado pelo cliente)	API da AWS; API do projeto do engajamento no GitLab.com (estado do Terraform); registro dos providers do Terraform	443 (HTTPS)	Plano, aplicação e destruição do Cloud Native Hybrid criado pelo pipeline
VM do runner	API privada do EKS criado pelo pipeline; API da AWS (token de acesso ao EKS)	443 (HTTPS)	Implantação e rotinas no EKS criado pelo pipeline
VM do runner	prometheus-community.github.io	443 (HTTPS)	Chart do kube-prometheus-stack

- Portas das VMs de backend conforme a tabela oficial de portas padrão do Linux package (reproduzida no documento PR-IMP-01); a tabela não informa direção nem protocolo, exceto Consul 8301 (TCP e UDP). As regras entre pods e VMs são levantadas no discovery com a equipe de rede do cliente.
- A documentação do GitLab.com lista, para liberação em firewall, gitlab.com, .gitlab.com, gitlab-static.net, .gitlab.io e .gitlab.net, e, para download de imagens de registry.gitlab.com, .storage.googleapis.com e .cdn.registry.gitlab-static.net.
- O runner não recebe conexões: nenhuma porta de entrada é aberta na VM do runner. Proxy corporativo: HTTPS_PROXY e NO_PROXY no serviço do runner e no executor Docker.

10. Entrega das credenciais

As credenciais listadas neste documento são cadastradas como **variáveis de CI/CD protegidas** no projeto do engajamento, com escopo restrito ao ambiente (por exemplo `producao`), e do tipo arquivo para kubeconfig, chaves, certificados e arquivo de licença. Os valores **nunca são enviados por e-mail, chat, issue ou arquivo**; um valor exposto em qualquer desses canais é revogado e substituído. A forma de cadastro de cada valor é combinada no kick-off. No encerramento do engajamento, as variáveis são revogadas e o kubeconfig da conta de serviço é revogado com o administrador do cluster.

Restrição da plataforma GitLab para o mascaramento de variáveis (documentação oficial de variáveis de CI/CD): só pode ser mascarado um valor em uma única linha, sem espaços e com 8 ou mais caracteres, e a opção de ocultar o valor (*hidden*) só pode ser definida na criação da variável. Conteúdos de várias linhas (kubeconfig, chave SSH, certificados, chaves PEM, arquivo de licença) não atendem a essa condição: para eles, a proteção é a variável protegida, com escopo do ambiente, do tipo arquivo. Senhas e tokens em uma linha devem atender à condição para serem mascarados.

O pipeline recusa valores que contenham a sequência `#f` e valida, antes de cada execução, se todas as variáveis exigidas existem e se as do tipo arquivo foram cadastradas como arquivo. O kubeconfig é gravado em arquivo temporário com permissão restrita durante o job. Os arquivos gerados pelo pipeline nunca contêm credenciais: as senhas entram nos Secrets do chart somente durante a execução, no runner da rede do cliente, e o as-built registra apenas os nomes das variáveis.

A credencial da AWS (Cloud Native Hybrid criado pelo pipeline) e os tokens de API da frota de runners e dos agentes seguem a mesma regra: variáveis protegidas com escopo do ambiente (ou do ambiente do componente da frota e dos agentes), revogadas no encerramento. O estado do Terraform usa a credencial do próprio job no projeto do engajamento, sem variável adicional.

11. Como a Pointer usa essas informações

As informações do formulário se tornam a **configuração do engajamento**, versionada no repositório do projeto do engajamento e alterada somente por merge request revisado por outro consultor. Credenciais não entram nesse repositório.

A cada alteração, o pipeline **valida a configuração automaticamente**, em runner hospedado, sem credenciais e sem acesso à rede do cliente: arquitetura, distribuição e seção do cluster, StorageClass, ingress, serviços externos, dimensionamento dos pods, VMs de backend no Hybrid e recursos opcionais. A mesma etapa gera os values do GitLab Helm chart, as tarefas que criam os Secrets e a lista exata das credenciais exigidas.

Antes da instalação, a **verificação prévia (preflight)** roda contra o cluster real, a partir do runner na rede do cliente: versão do Kubernetes, permissão, StorageClass, nós e capacidade por grupo de nós. Para testar os serviços externos do ponto de vista dos pods, o preflight cria no namespace um Job temporário, com a imagem toolbox da versão GitLab alvo e sem token de conta de serviço, que testa PostgreSQL, Redis e os buckets; o Job, o Secret e o ConfigMap da sonda são removidos ao final, e o namespace é criado se ainda não existir. No Hybrid, as VMs de backend passam pela mesma verificação da implantação Linux. O relatório com erros e alertas fica anexado ao pipeline.

A implantação repete essa verificação e não começa se houver qualquer erro; alertas aceitos são registrados no registro de decisões do engajamento. Cada implantação exige aprovação de uma pessoa diferente de quem a executa, e só uma execução por ambiente acontece de cada vez. Em falha, o pipeline grava um diagnóstico somente leitura do cluster.

Quando o pipeline cria o Cloud Native Hybrid na AWS, a mesma configuração gera o módulo Terraform em runner hospedado, sem credenciais; o plano, com o resumo das ações, é revisado antes da aplicação, que exige a mesma aprovação. Os endereços das VMs e o acesso ao EKS são lidos do estado do Terraform por todos os jobs seguintes.

12. Critérios de aceite

- Todas as checagens obrigatórias da verificação pós-implantação com resultado OK.
- Acesso pela URL externa com certificado válido; clone por HTTPS e por SSH na porta do Git definida.
- Login LDAP e envio de e-mail de teste validados com o cliente, quando habilitados.
- CronJob de backup executado ao menos uma vez (ou backup sob demanda), com o backup presente no bucket de backups.
- Runners da frota online na instância e agentes GitLab para Kubernetes conectados, quando contratados.

13. Credenciais que o cliente entrega à Pointer

Nenhum valor secreto é enviado por e-mail, chat, issue ou arquivo. Cada credencial é cadastrada como variável de CI/CD protegida no projeto do engajamento, com escopo do ambiente, e revogada no encerramento.

Credencial	Finalidade	Permissões	Formato e validade	Obrigatório
Kubeconfig da conta de serviço dedicada	Acesso do runner à API do cluster para a verificação prévia, a instalação pelo Helm chart, a verificação, o diagnóstico e as rotinas de dia 2.	Conta de serviço dedicada ao engajamento com cluster-admin (<code>kubectl auth can-i '*' '*' --all-namespaces = yes</code>), com a CA do cluster; gerada com o administrador do cluster.	Arquivo kubeconfig Até o encerramento; revogado com o administrador do cluster e removido do projeto do engajamento.	Condicional: cluster existente (dispensado no EKS criado pelo pipeline)
Chave SSH do usuário de implantação	Acesso do runner às VMs de backend para a verificação prévia, a execução do GET e a verificação; no Hybrid criado pelo pipeline, a chave pública é gravada nas VMs pelo pipeline.	Usuário presente em todas as VMs de backend, com sudo sem senha; a chave pública correspondente autorizada nesse usuário.	Chave privada OpenSSH (arquivo) Até o encerramento do engajamento, quando a variável é revogada.	Condicional: Cloud Native Hybrid
Certificado TLS e chave privada do host externo	Entrada HTTPS da instância pelo Gateway.	Certificado com SAN do host externo e dos hostnames adicionais usados.	Dois arquivos PEM: certificado com os intermediários da cadeia e chave privada Validade do certificado emitido; a variável é revogada no encerramento.	Condicional: certificado do cliente (padrão; dispensado com Let's Encrypt)
Activation code ou arquivo de licença	Ativação da assinatura GitLab Premium ou GitLab Ultimate na instância.	Assinatura ativa no portal de clientes da GitLab (Customers Portal).	Activation code (texto em uma linha) ou arquivo <code>.gitlab-license</code> Vigência da assinatura; a variável é revogada no encerramento.	Sim

Credencial	Finalidade	Permissões	Formato e validade	Obrigatório
Access key e secret key do object storage	Gravação e leitura dos objetos e dos backups nos buckets do ambiente; usadas também pela sonda.	Acesso aos 13 buckets listados neste documento.	Dois valores de texto em uma linha Continuam em uso pela instância entregue; a variável do projeto do engajamento é revogada no encerramento.	Sim
Senha do usuário da aplicação no PostgreSQL externo	Conexão da instância ao banco principal; usada também pela sonda.	Usuário da aplicação (padrão <code>gitlab</code>) com acesso ao banco da instância.	Texto em uma linha; para mascaramento, sem espaços e com 8 ou mais caracteres; sem a sequência <code>#</code> Continua em uso pela instância entregue; a variável é revogada no encerramento.	Condicional: PostgreSQL externo (obrigatório na Cloud Native)
Usuário administrador e senha do PostgreSQL externo	Preparo do banco pelo GET: criação do usuário da aplicação, do banco e das extensões.	Superusuário ou papel equivalente do serviço gerenciado (<code>rds_superuser</code> , <code>azure_pg_admin</code> ou <code>cloudsqlsuperuser</code>).	Nome do usuário no formulário; senha em texto em uma linha Até o encerramento, quando a variável é revogada.	Condicional: preparo do banco pelo GET (sem ele, usuário, banco e extensões são criados antes pelo cliente)

Credencial	Finalidade	Permissões	Formato e validade	Obrigatório
Senha do Redis externo	Autenticação da instância no Redis; usada também pela sonda.	Senha de AUTH da instância Redis.	Texto em uma linha; para mascaramento, sem espaços e com 8 ou mais caracteres; sem a sequência <code>#{</code> Continua em uso pela instância entregue; a variável é revogada no encerramento.	Condicional: Redis externo com autenticação (obrigatório na Cloud Native)
Certificado wildcard e chave privada do GitLab Pages	HTTPS dos sites publicados pelo GitLab Pages.	Certificado wildcard do domínio do Pages.	Dois arquivos PEM: certificado e chave privada Validade do certificado emitido; a variável é revogada no encerramento.	Condicional: GitLab Pages com HTTPS
Senha da conta de bind LDAP	Consulta ao diretório para autenticação dos usuários.	Conta de bind (DN) informada no formulário.	Texto em uma linha Continua em uso pela instância entregue; a variável é revogada no encerramento.	Condicional: LDAP com conta de bind
Senha do usuário SMTP	Envio de e-mails da instância.	Usuário SMTP informado no formulário, autorizado a enviar pelo remetente definido.	Texto em uma linha Continua em uso pela instância entregue; a variável é revogada no encerramento.	Condicional: SMTP com autenticação

Credencial	Finalidade	Permissões	Formato e validade	Obrigatório
Senha do cluster de busca externo	Conexão da busca avançada ao cluster Elasticsearch ou OpenSearch do cliente.	Usuário informado no formulário.	Texto em uma linha Continua em uso pela instância entregue; a variável é revogada no encerramento.	Condicional: busca avançada externa com usuário
Senha inicial do usuário root	Primeiro acesso administrativo à instância. Definida no kick-off (cliente ou Pointer) e cadastrada como variável protegida.	Administrador da instância GitLab.	Texto em uma linha; para mascaramento, sem espaços e com 8 ou mais caracteres; sem a sequência #f Trocada na entrega (pendência registrada no as-built); a variável é revogada no encerramento.	Sim
Token do Gitaly	Autenticação entre os componentes e o Gitaly; o GET cria o Secret correspondente no Kubernetes em todas as arquiteturas. Definido no kick-off (cliente ou Pointer) e cadastrado como variável protegida.	Uso interno entre os componentes da instância.	Texto em uma linha; para mascaramento, sem espaços e com 8 ou mais caracteres; sem a sequência #f Continua em uso pela instância entregue; a variável é revogada no encerramento.	Sim

Credencial	Finalidade	Permissões	Formato e validade	Obrigatório
Senhas internas dos componentes em VM (Cloud Native Hybrid)	Senha do usuário gitlab no PostgreSQL e senha do Redis quando ficam em VM; senhas do Patroni, do usuário do Consul e do PgBouncer (3k ou mais sem PostgreSQL externo); tokens externo e interno do Praefect e senha do banco do Praefect (3k ou mais com Gitaly Cluster). Definidas no kick-off (cliente ou Pointer) e cadastradas como variáveis protegidas.	Uso interno entre os componentes da instância.	Texto em uma linha; para mascaramento, sem espaços e com 8 ou mais caracteres; sem a sequência <code>#f</code> Continuam em uso pela instância entregue; as variáveis são revogadas no encerramento.	Condicional: Cloud Native Hybrid com componentes em VM
Credencial da conta AWS (<code>AWS_ACCESS_KEY_ID</code> e <code>AWS_SECRET_ACCESS_KEY</code>)	Plano, aplicação e destruição do Cloud Native Hybrid criado pelo pipeline e acesso ao EKS nos jobs de implantação e dos componentes da frota e dos agentes que usam esse cluster.	Identidade IAM dedicada ao engajamento, com as permissões da seção <i>Cloud Native Hybrid criado pelo pipeline na AWS</i> e acesso ao cluster.	Dois valores de texto em uma linha Até o encerramento; revogada pelo cliente.	Condicional: Hybrid criado pelo pipeline
Cadeia da autoridade certificadora da URL externa	Confiança dos nós com Linux package do Hybrid, dos runners da frota, dos agentes e do Grafana (sob consulta) na URL externa, quando o certificado não é de autoridade certificadora pública.	Cadeia pública da AC emissora (não é segredo); cadastrada como variável do tipo arquivo com escopo do ambiente.	Arquivo PEM Validade da AC; a variável é revogada no encerramento.	Condicional: autoridade certificadora não pública
Token de API da frota de runners (<code>PS_RUNNER_TOKEN_API</code>)	Criação, consulta, recriação e remoção dos runners da frota na instância.	Escopos <code>create_runner</code> e <code>manage_runner</code> ; usuário administrador (runners de instância) ou Owner do grupo (runners de grupo).	Texto em uma linha Até o encerramento; revogado pelo cliente.	Condicional: frota de runners
Token de API dos agentes (<code>PS_AGENTES_TOKEN_API</code>)	Projeto de configuração, config.yaml, registro, tokens e consulta das conexões dos agentes.	Escopo <code>api</code> ; usuário Maintainer dos projetos de configuração dos agentes.	Texto em uma linha Até o encerramento; revogado pelo cliente.	Condicional: agentes GitLab para Kubernetes

Credencial	Finalidade	Permissões	Formato e validade	Obrigatório
Kubeconfig dos clusters da frota e dos agentes	Instalação do chart do GitLab Runner e do agente GitLab para Kubernetes nos clusters do cliente que não foram criados pelo pipeline.	Conta de serviço dedicada, com permissão para criar namespace, Secrets e o release Helm (seção <i>Frota de runners do cliente e agentes GitLab para Kubernetes</i>).	Arquivo kubeconfig Até o encerramento; revogado com o administrador do cluster.	Condicional: frota em Kubernetes ou agentes em cluster existente
Certificado TLS e chave privada do host do Grafana (sob consulta)	HTTPS do Grafana gerenciado pela Pointer.	Certificado com SAN do host do Grafana.	Dois arquivos PEM: certificado e chave privada Validade do certificado emitido; a variável é revogada no encerramento.	Condicional: Grafana gerenciado (sob consulta)

14. Informações a enviar

Informações sem sigilo, devolvidas pelo cliente antes da reunião de prontidão. Os campos marcados como obrigatórios são conferidos pela validação automática do pipeline.

Identificação e contatos

Informação	Exemplo	Obrigatório	Observação
Denominação oficial do cliente	Órgão Exemplo	Sim	
Sigla do cliente	exemplo	Sim	Minúsculas, números e hífen; até 40 caracteres.
Número do contrato, pedido ou ordem de serviço	Contrato 12/2026	Não	
Nome do ambiente	producao	Sim	Minúsculas, números e hífen. Cada ambiente tem configuração própria.
Patrocinador	Nome, cargo, e-mail	Sim	Decisões de escopo, prioridade e aceite.
Ponto focal técnico	Nome, e-mail, telefone	Sim	
Administrador do cluster Kubernetes	Nome e e-mail	Sim	Gera e revoga o kubeconfig da conta de serviço.
Contatos de banco de dados, rede e segurança	Nome e e-mail por área	Sim	

Informação	Exemplo	Obrigatório	Observação
Aprovador das execuções em produção	Nome e e-mail	Sim	No projeto do engajamento, cada implantação é aprovada por pessoa diferente de quem executa.
Janela de manutenção	Sábados, 22h às 2h	Sim	

Plataforma GitLab

Informação	Exemplo	Obrigatório	Observação
Versão GitLab	19.4.1	Sim	Versão exata X.Y.Z; compatível com a versão do Kubernetes (tabela de versões). O GET escolhe o chart correspondente.
Edição	Enterprise Edition	Não	Enterprise Edition (padrão) ou FIPS (imagens FIPS só para x86-64). Community Edition não é aceita.
URL externa	https://gitlab.cliente.gov.br	Sim	https:// seguido do host, sem caminho.
Forma de licenciamento	Activation code	Não	Activation code (padrão; ativação online) ou arquivo de licença (offline).
Porta SSH do Git	22	Não	Padrão 22 (listener SSH do Envoy Gateway).

Dimensionamento e arquitetura

Informação	Exemplo	Obrigatório	Observação
Número de usuários atual e projeção de 1 a 3 anos	4.000 hoje; 6.000 em 3 anos	Sim	
Carga medida (RPS)	Relatório do GitLab Performance Tool ou análise de logs	Condicional: instância GitLab existente	
Uso intenso de monorepos, CI/CD, pacotes ou registry	Monorepos de uso moderado	Sim	Critério de escolha entre S, M, L e XL na Cloud Native.
Topologia	Cloud Native	Sim	Cloud Native (tudo no cluster, serviços externos) ou Cloud Native Hybrid (componentes com estado em VMs ou serviços externos).
Forma da infraestrutura	Hybrid criado pelo pipeline na AWS	Sim	Cluster existente ou, no Cloud Native Hybrid, EKS e VMs de backend criados pelo pipeline na AWS (AKS e GKE: sob consulta).

Informação	Exemplo	Obrigatório	Observação
Forma de instalação	GitLab Helm chart	Não	GitLab Helm chart (padrão) ou GitLab Operator (GitLab 17.6.0 ou superior; em OpenShift, sob consulta).
Arquitetura de referência	M	Sim	Cloud Native: S, M, L ou XL. Hybrid: 2k, 3k, 5k, 10k, 25k ou 50k.
Modo do Gitaly	Gitaly Cluster	Condicional: Hybrid 3k ou mais	Gitaly Cluster com Praefect (padrão) ou Gitaly Sharded. Na Cloud Native, o Gitaly é sempre sharded.
Prefixo de nomes	gitlab	Não	Prefixo dos buckets e, no Hybrid, dos nomes de servidor; padrão gitlab.
Ajuste de dimensionamento dos pods	Webservice: mínimo 6, máximo 9	Não	Réplicas e recursos de Webservice, Sidekiq, Gitaly e suporte; valores abaixo da arquitetura geram alerta.

Cluster Kubernetes

Informação	Exemplo	Obrigatório	Observação
Distribuição	AKS	Sim	AKS, EKS, GKE, OpenShift, OKE, RKE2, K3s ou outra conforme.
Versão do Kubernetes	1.35	Sim	Conforme a tabela de versões.
Cluster dedicado ou compartilhado	Compartilhado	Sim	
Namespace	gitlab	Não	Padrão gitlab; criado pelo pipeline se não existir.
Contexto do kubeconfig	cliente-prd	Não	Padrão: contexto atual do kubeconfig.
Cotas e políticas do namespace e do cluster	ResourceQuota por namespace; Kyverno	Sim	ResourceQuota, LimitRange, Pod Security, OPA ou Kyverno.
Nós rotulados por carga	Sim	Não	Padrão: sim (rótulo workload = webservice, sidekiq, support e, na Cloud Native, gitaly).
Nós e capacidade alocável por grupo	webservice: 3 nós de 16 vCPU / 64 GB	Sim	Conforme as tabelas de capacidade.
StorageClass do Gitaly	managed-csi	Condicional: Cloud Native	StorageClass existente; tipo de disco, IOPS, expansão de volume e política de backup ou snapshot.
Tamanho do volume de cada pod de Gitaly (GiB)	200	Não	Cloud Native; padrão 50; mínimo 10.
Registry espelho	harbor.cliente.gov.br	Condicional: nós sem acesso a registry.gitlab.com e docker.io	Configurado no cluster pelo cliente.

Cloud Native Hybrid criado pelo pipeline (AWS)

Informação	Exemplo	Obrigatório	Observação
Conta AWS e região	Conta 1234...; sa-east-1	Condicional: Hybrid criado pelo pipeline	
Rede	Rede nova 10.45.0.0/16, 2 subnets públicas e 2 privadas	Condicional: Hybrid criado pelo pipeline	Rede nova (CIDR) ou VPC e subnets existentes.
CIDRs com acesso às VMs e grupos de segurança de acesso	10.0.0.0/8; sg-0abc...	Condicional: Hybrid criado pelo pipeline	Rede interna e a do runner do engajamento.
Versão do EKS	1.35	Condicional: Hybrid criado pelo pipeline	Conforme a tabela de versões.
Pools do EKS (tipo e quantidade)	Webservice: c7i.2xlarge × 3	Não	Padrão do pipeline pelos totais da arquitetura.
ARNs IAM com acesso administrativo ao cluster	arn:aws:iam::1234...:role/admins	Não	
Tipos de instância das VMs de backend e disco	Gitaly: m7i.xlarge; 100 GB	Não	Padrão: menor tipo da lista do pipeline que atende a arquitetura; disco padrão 100 GB.
Tags de custo e governança	centro-de-custo: TI-123	Não	Aplicadas a todos os recursos criados.
Destino da infraestrutura no encerramento	Entregue ao cliente	Condicional: Hybrid criado pelo pipeline	Entregue ao cliente (transferência do estado do Terraform) ou destruída por decisão registrada em merge request.

Exposição, DNS e TLS

Informação	Exemplo	Obrigatório	Observação
Ingress	Envoy Gateway	Não	Envoy Gateway (padrão) ou NGINX Ingress (deprecated; exige IP externo).
Provedor de LoadBalancer	MetaLB	Sim	Balancedador da nuvem, MetaLB, ServiceLB ou outro.
IP externo ou anotações do LoadBalancer	metallb.universe.tf/loadBalancerIPs: 10.0.0.100	Condicional: IP fixo ou NGINX Ingress	Com NGINX Ingress, IPv4 obrigatório; com Envoy Gateway, IP fixo por anotação.

Informação	Exemplo	Obrigatório	Observação
Registros DNS	gitlab.cliente.gov.br, registry.gitlab.cliente.gov.br e kas.gitlab.cliente.gov.br → 10.0.0.100	Sim	Host da URL externa, registry e kas (ou wildcard do host), para o LoadBalancer do Gateway.
Origem do certificado	Certificado do cliente	Não	Certificado do cliente (padrão) ou Let's Encrypt pelo cert-manager.
Autoridade certificadora, SANs e validade	AC interna; gitlab.cliente.gov.br; até 10/2027	Condicional: certificado do cliente	SAN obrigatório.
E-mail de registro do Let's Encrypt	infra@cliente.gov.br	Condicional: Let's Encrypt	

PostgreSQL externo

Informação	Exemplo	Obrigatório	Observação
Serviço	Azure Database for PostgreSQL Flexible Server	Condicional: PostgreSQL externo	Serviço gerenciado ou servidor próprio; Amazon Aurora e Google AlloyDB são incompatíveis.
Endereço	pg.cliente.gov.br	Condicional: PostgreSQL externo	Alcançável a partir dos pods.
Porta	5432	Não	Padrão 5432.
Usuário da aplicação	gitlab	Não	Padrão gitlab.
Banco	gitlabhq_production	Não	Padrão gitlabhq_production.
Usuário administrador	postgres	Não	Para o preparo do banco pelo GET; sem ele, usuário, banco e extensões são criados antes.
Versão e parâmetros	17.6; max_connections 400; max_locks_per_transaction 128	Condicional: PostgreSQL externo	Conforme a tabela de infraestrutura.
Alta disponibilidade e backup do banco	Réplica standby; backup diário do provedor	Condicional: PostgreSQL externo	

Redis externo

Informação	Exemplo	Obrigatório	Observação
Serviço	Amazon ElastiCache for Valkey	Condicional: Redis externo	Instância standalone; Redis Cluster e variantes serverless não são suportados.
Endereço	redis.cliente.gov.br	Condicional: Redis externo	Alcançável a partir dos pods; o pipeline recebe um único endereço.

Informação	Exemplo	Obrigatório	Observação
Porta	6379	Não	Padrão 6379.
TLS	Não	Não	Padrão: sem TLS.
Autenticação por senha	Sim	Não	Padrão: sim; sem senha, somente em ambiente de teste (alerta).
Versão e alta disponibilidade	7.2 com réplica	Condicional: Redis externo	

VMs de backend (Cloud Native Hybrid)

Informação	Exemplo	Obrigatório	Observação
Endereço de cada VM, por papel	Gitaly: 10.0.5.1, 10.0.5.2, 10.0.5.3	Condicional: Hybrid	Todos os papéis em VM da arquitetura; um endereço por VM.
Sistema operacional e versão de cada VM	Ubuntu 24.04	Condicional: Hybrid	Matriz de sistemas operacionais do documento PR-IMP-01.
Usuário SSH de implantação	ubuntu	Condicional: Hybrid	Mesmo usuário em todas as VMs, com sudo sem senha.
Porta SSH das VMs	22	Não	Padrão 22.
Servidor NTP interno	ntp.cliente.gov.br	Condicional: VMs sem acesso a pool.ntp.org	Usado nas verificações de horário do Gitaly e do Praefect.

Rede e runner

Informação	Exemplo	Obrigatório	Observação
VM do runner	10.0.0.50, Ubuntu 24.04, Docker Engine instalado	Sim	
Endereço e porta da API do cluster	https://10.0.0.10:6443	Sim	Alcançável a partir da VM do runner.
Proxy de saída	http://proxy.cliente.gov.br:3128; exceções: *.cliente.gov.br	Condicional: saída por proxy	

Object storage

Informação	Exemplo	Obrigatório	Observação
Serviço S3-compatível	Amazon S3	Sim	
Endpoint	https://s3.sa-east-1.amazonaws.com	Sim	URL do serviço.
Região	sa-east-1	Sim	Valor aceito pelo serviço.
Endereçamento path style	Não	Não	O pipeline usa path style quando o campo não é informado.

Informação	Exemplo	Obrigatório	Observação
Prefixo dos buckets	gitlab	Não	Padrão: prefixo de nomes.
Confirmação dos 13 buckets criados	Criados em 15/10/2026	Sim	Nomes exatos conforme a tabela de infraestrutura.

Recursos opcionais

Informação	Exemplo	Obrigatório	Observação
GitLab Pages	Sim; https://pages.cliente.gov.br	Não	URL raiz com DNS wildcard; com HTTPS, certificado wildcard.
Busca avançada	Cluster externo: https://opensearch.cliente.gov.br:9200 ; usuário gitlab	Não	Na Cloud Native, somente cluster externo; no Hybrid, também VMs OpenSearch instaladas pelo GET.
LDAP / Active Directory	ldap.cliente.gov.br; 636; simple_tls; OU=Usuarios,DC=cliente,DC=gov,DC=br; sAMAccountName	Não	Mesmos campos do documento PR-IMP-01: rótulo, servidor, porta, criptografia, verificação de certificado, Active Directory, base DN, atributo de login, conta de bind, filtro, base de grupos e grupo de administradores.
SMTP	smtp.cliente.gov.br; 587; starttls; gitlab@cliente.gov.br	Não	Servidor, porta, TLS, usuário, domínio, remetente e reply-to.
Container Registry	https://registry.gitlab.cliente.gov.br	Não	Subdomínio coberto pelo certificado; storage no object storage (bucket do registry).
Escopos da busca global	Código, commits e wiki ligados	Não	Código, commits e wiki exigem busca avançada.
Servidor do agente GitLab para Kubernetes (KAS)	Habilitado	Não	Padrão: habilitado, em kas..
Prometheus: habilitado, retenção, disco e StorageClass	Sim; 30 dias; 100 GB; gp3	Não	Padrão: habilitado, 100 GB; desligar quando o cliente tem monitoramento próprio.

Informação	Exemplo	Obrigatório	Observação
Grafana gerenciado (sob consulta): host, anotações do LoadBalancer e grupos	https://grafana.cliente.gov.br; infra; infra/sre	Condicional: Grafana contratado	Host com DNS para o LoadBalancer do Grafana e certificado próprio.

Frota de runners e agentes

Informação	Exemplo	Obrigatório	Observação
Runners da frota	k8s-geral: Kubernetes, namespace gitlab-runner; escopo instância; tags k8s	Condicional: frota contratada	Por runner: nome, modalidade, escopo e alvo, tags, execução sem tag, proteção, timeout máximo, concorrência, imagem padrão, modo privilegiado, requests de CPU e memória e versão do chart.
Agentes	producao: projeto plataforma/kubernetes-agentes; contexto cliente-prd; grupo apps	Condicional: agentes contratados	Por agente: nome, projeto de configuração, cluster e contexto, namespace, ClusterRole (opcional), grupos e projetos com acesso de CI/CD e modo de acesso.

Backup e operação

Informação	Exemplo	Obrigatório	Observação
Backup agendado	Sim	Não	Padrão: habilitado (CronJob do toolbox para o bucket de backups).
Horário do backup	Diário às 02:00	Não	Formato cron de 5 campos; agenda não diária gera alerta.
Retenção	7	Não	Padrão 7; convertida em quantidade máxima de backups.
RTO e RPO	RT0 8 h; RPO 24 h	Sim	
Política de atualização de versão	Trimestral, em janela de manutenção	Sim	Considerar a matriz de versões do Kubernetes.
Monitoramento corporativo existente	Prometheus do cluster	Não	
Requisitos regulatórios de dados	Logs de pipeline e artefatos somente em território nacional	Sim	

15. Checklist de prontidão

Itens conferidos antes da execução. "Preflight automático" indica verificação feita pelo pipeline contra o ambiente real; os demais são conferidos na reunião de prontidão ou pelo cliente.

☐ Item	Verificado por
☐ Todas as credenciais exigidas cadastradas como variáveis protegidas do ambiente, com o tipo correto (arquivo para kubeconfig, chaves, certificados e licença). Falta ou tipo incorreto: erro.	Preflight automático
☐ Cluster acessível com o kubeconfig entregue (erro) e permissão cluster-admin da conta de serviço (erro).	Preflight automático
☐ Versão do Kubernetes na matriz do chart e compatível com a versão GitLab (fora da matriz ou versão GitLab abaixo do mínimo: erro; 1.34 ou versão acima de 1.37: alerta).	Preflight automático
☐ StorageClass informada existente (erro), com expansão de volume (alerta) e StorageClass padrão no cluster (alerta).	Preflight automático
☐ Nós prontos e agendáveis, amd64 ou arm64 (nenhum nó pronto ou outra arquitetura: erro; arm64: alerta).	Preflight automático
☐ Nós rotulados por carga (grupo sem nó rotulado: erro) e capacidade alocável por grupo: abaixo de 90% do mínimo, erro; abaixo do necessário para as réplicas máximas, alerta.	Preflight automático
☐ Cloud Native: pelo menos um nó para cada pod de Gitaly (erro).	Preflight automático
☐ Release Helm <code>gitlab</code> inexistente no namespace (existente: alerta de atualização).	Preflight automático
☐ Imagem toolbox da versão alvo baixada pelos nós de registry.gitlab.com ou do registry espelho: sem ela, a sonda não conclui (erro).	Preflight automático
☐ PostgreSQL alcançável a partir dos pods, versão 17 para GitLab 19.x, <code>max_locks_per_transaction</code> ≥ 128 e extensões exigidas disponíveis (erro).	Preflight automático
☐ Redis alcançável a partir dos pods, com AUTH e PING (erro), versão 7.0 ou superior (erro) e 7.2 ou superior (alerta).	Preflight automático
☐ Os 13 buckets existentes e acessíveis com as credenciais entregues (erro).	Preflight automático
☐ Hybrid: VMs de backend aprovadas na mesma verificação da implantação Linux (SSH, sistema operacional, CPU, memória, disco, NTP).	Preflight automático
☐ RKE2, K3s, OpenShift ou outra distribuição: provedor de LoadBalancer confirmado para o Envoy Gateway (o preflight emite alerta).	Reunião de prontidão
☐ Parâmetros obrigatórios do PostgreSQL externo (<code>work_mem</code> , <code>maintenance_work_mem</code> , <code>max_connections</code> , <code>shared_buffers</code> , <code>statement_timeout</code>) dentro do exigido pela documentação GitLab (fora dele: alerta).	Preflight automático
☐ <code>hot_standby_feedback = on</code> no PostgreSQL externo, quando há balanceamento de carga do banco.	Reunião de prontidão
☐ Nós com saída para docker.io (Envoy Gateway) ou registry espelho configurado.	Reunião de prontidão
☐ Cotas e políticas do namespace e do cluster compatíveis com os pods do chart e com o Job temporário da sonda.	Reunião de prontidão
☐ Registros DNS da URL externa apontando para o IP do LoadBalancer; DNS wildcard do Pages, quando habilitado.	Cliente
☐ Certificado TLS com SAN, cadeia de intermediários e chave correspondente; certificado wildcard do Pages, quando habilitado.	Reunião de prontidão
☐ VM do runner com Docker Engine, 2 vCPU, 4 GB e 40 GB; runner registrado no projeto do engajamento, online e com acesso à API do cluster e à URL externa.	Reunião de prontidão

☐	Item	Verificado por
☐	Saída do runner liberada para gitlab.com, registry.gitlab.com, <i>.gitlab-static.net</i> , <i>.storage.googleapis.com</i> , <i>charts.gitlab.io</i> e <i>docker.io</i> .	Reunião de prontidão
☐	Hybrid: latência entre as VMs de backend menor que 5 ms (3k ou mais) e regras de firewall entre pods e VMs liberadas.	Cliente
☐	LDAP, SMTP e cluster de busca externo alcançáveis a partir dos pods, com as contas informadas.	Reunião de prontidão
☐	Assinatura GitLab Premium ou GitLab Ultimate ativa, com activation code ou arquivo de licença disponível.	Cliente
☐	Hybrid criado pelo pipeline: credencial AWS cadastrada no ambiente, permissões da identidade (inclusive provedores OpenID Connect do IAM) e políticas da organização conferidas, cotas da região suficientes e plano do Terraform revisado antes da aplicação.	Reunião de prontidão
☐	Hybrid criado pelo pipeline: runner do engajamento com rota para a rede criada e para o endpoint privado da API do EKS.	Reunião de prontidão
☐	Registros DNS de gitlab, registry e kas apontando para o LoadBalancer do Gateway e certificado com esses nomes (KAS sem SAN: alerta da validação).	Cliente
☐	Prometheus: saída do runner para prometheus-community.github.io e dos nós para quay.io, registry.k8s.io e docker.io (ou registry espelho); no Hybrid, portas de coleta das VMs de backend liberadas a partir do cluster.	Reunião de prontidão
☐	Frota de runners e agentes: tokens de API com os escopos exigidos, acesso aos clusters, grupo dos projetos de configuração existente e configuração validada pelo pipeline.	Reunião de prontidão
☐	Administrador do cluster, aprovador das execuções, janela de manutenção e contatos definidos.	Reunião de prontidão

O3S Consultoria e Tecnologia da Informação Ltda. · contato@pointertech.digital · +55 (61) 99874-4648 ·
pointertech.digital

PR-IMP-02 · versão 1.2 · 1 de outubro de 2026. GITLAB is a trademark of GitLab Inc. in the United States and other countries and regions. A Pointer é parceira GitLab Select.