GitLab без интернета

Инструкция для администратора GitLab и разработчиков. Для закрытого контура заранее подготовьте образы для Runner и пакеты NuGet.

Что получится

GitLab CE и GitLab Runner могут работать без доступа в интернет. Всё внешнее нужно один раз принести внутрь, после этого разработчики и CI получают пакеты только из GitLab. Документ описывает, что принести, куда положить и как подключиться. Проверки автоматизаций и отчётов (ci-checks) работают без интернета только с собственным образом, см. шаг 2.

В открытом контуре интернет нужен в трёх местах. У каждого из них есть замена:

Что	Откуда обычно	Что делаем без интернета
Образы GitLab и сервисов	Docker Hub, registry.ascon.ru, gcr.io	Комплект без интернета от поставщика, образы загружаются из папки images
Образы заданий CI	Docker Hub, mcr.microsoft.com, registry.gitlab.com	Тот же комплект: образы заранее лежат на машине с runner. Для проверок ci-checks нужен свой образ с python3 и git, см. шаг 2
Пакеты NuGet и библиотеки DLL	api.nuget.org	Реестр пакетов GitLab, источник для разработчиков и CI

Шаг 1. Образы

Комплект без интернета готовит поставщик. Скриптов сборки в комплекте нет, поэтому собрать его на месте нельзя. Такой комплект узнаётся по второй строке файла IMAGES.txt:

> Комплект для установки без интернета: все образы, в том числе GitLab и заданий CI, лежат в images\.

Если там написано «с интернетом» или второй строки нет (комплект собран старой версией сборки), запросите комплект без интернета у поставщика или перенесите недостающие образы сами, как описано в документе «Установка стенда», раздел «Установка без интернета». Образы заданий CI переносятся так же, их список описан ниже.

В папке images лежат образы GitLab, Runner и служебный образ Runner gitlab-runner-helper, а также образы заданий: .NET SDK, Alpine и Python. Образ GitLab CLI нужен только устаревшим копиям шаблона CI с заданием release, актуальный шаблон его не использует. Перенесите папку на сервер и установите GitLab по инструкции:

> powershell -ExecutionPolicy Bypass -File .\install.ps1 -Roles gitlab -NoData
>
> **Адрес GitLab.** При -NoData ключ -GitLabHost не применяется, а в .env.example стоит GITLAB_HOST_IP=localhost: адрес GitLab получится http://localhost:8080. Перед установкой задайте в .env GITLAB_HOST_IP=<адрес сервера> и GITLAB_ROOT_PASSWORD (если

> оставить заглушку из .env.example, при установке на одну машину пароль root будет заменён случайным и записан в .env).
>
> На сервере нужно хранилище образов containerd, иначе docker load теряет digest и compose не находит образы. Как его включить, описано в документе «Установка стенда», раздел «Установка без интернета».

Если у вас свой GitLab или комплект собран с интернетом, перенесите образы заданий сами. Точные версии указаны в блоке variables (SDK_IMAGE) и в строках image: файла .gitlab-ci.yml в репозиториях модулей, этот же шаблон выдаёт админка на странице модуля. Ещё runner нужен служебный образ gitlab-runner-helper той же версии, что и сам runner. На машине с интернетом сохраните образ, а на машине с runner загрузите его:

> docker save -o sdk.tar "mcr.microsoft.com/dotnet/sdk:<версия>@sha256:\<digest>"
>
> docker load -i sdk.tar

Шаг 2. Настройка GitLab и Runner

Параметры compose отключают отправку статистики использования и аватары Gravatar, но они действуют для новой базы GitLab: в базе, восстановленной из копии, значения берутся из копии, проверьте их в Admin / Settings. Остальное настраивается вручную:

1. **Проверка версий.** В разделе Admin / Settings / Metrics and profiling / Usage statistics снимите галочку Enable version check.
2. **Runner берёт образы локально.** В config.toml должно быть pull_policy = ["if-not-present"]. При регистрации нового runner укажите --docker-pull-policy if-not-present.
3. **DNS.** Строку dns = ["8.8.8.8", "1.1.1.1"] в config.toml, если она есть, замените на DNS вашей сети или удалите. Без неё job берёт настройки Docker.
4. **Адрес админки.** В группах модулей задайте переменную ADMIN_URL, если админка находится на другой машине. Админка проверяет токен задания обращением к GitLab из своего контейнера: адрес GitLab в настройках группы модулей должен быть доступен оттуда.
5. **Образ для проверок ci-checks.** Задания проверок автоматизаций и отчётов запускаются в образе python:3.12.14-alpine, в котором нет git: по умолчанию задание ставит его командой apk add, то есть обращается в интернет. В закрытой сети соберите образ с python3 и git, загрузите его на runner и задайте переменную CHECKS_IMAGE: тогда apk add не вызывается, без этого проверки не пройдут. Скрипты проверок задание берёт из проекта CHECKS_PROJECT (по умолчанию pilot-ci/checks) ревизией CHECKS_REF (по умолчанию main), подробнее в gitlab/ci-checks/README.md.
6. **Регистрация и чужая копия данных.** Установка с ролью gitlab снимает «Sign-up enabled» сама, и при чистой установке, и после восстановления из копии; если её шаг GitLab после установки не прошёл, снимите флаг вручную: Admin / Settings / General / Sign-up restrictions. Параметр хранится в базе, после восстановления из копии проверьте его ещё раз. Если GitLab развёрнут из чужой копии, её секреты общие для всех получателей: смените пароль root и пароли учёток из копии, удалите config/ssh_host_* и выполните gitlab-ctl reconfigure, пересоздайте runner, отзовите и создайте заново deploy-токены и OAuth-приложения, задайте новые ключи DEPLOY_KEY_TEST, _STAGE, _PROD (подробнее в gitlab/README.md).

7. **Переменные группы и защита веток.** На машине с ролью gitlab установка сама задаёт в группах модулей DEPLOY_KEY_TEST (Masked), DEPLOY_KEY_STAGE и DEPLOY_KEY_PROD (Masked и Protected) и защищает ветки main, master, release/* и dev в репозиториях модулей; вручную это делают, если шаг GitLab после установки пропущен или не удался. Подробнее в документе «Установка стенда», раздел 3.

Файл config.toml лежит в <DATA_ROOT>\gitlab\runner-config. Runner перечитывает его сам, перезапуск не нужен.

Шаг 3. Как положить пакеты в GitLab

Проект для пакетов

1. Пакеты кладутся в реестр проекта pilot-clientsdk в группе клиентских модулей (client-modules/pilot-clientsdk): в него же публикует пакет Pilot.ClientSdk CI проекта (задания nuget_*). Запомните его числовой Project ID, он указан в настройках проекта (Settings / General).
2. Создайте токен для загрузки: Deploy Token проекта со scope write_package_registry, либо Personal Access Token ответственного со scope api.
3. Откройте доступ заданиям CI: в проекте pilot-clientsdk раздел Settings / CI/CD / Job token permissions, добавьте группы с модулями. Заданиям нужно разрешить и читать пакеты, и публиковать их: задания nuget_* отправляют пакет с CI_JOB_TOKEN.

Собрать пакеты на машине с интернетом

Команда dotnet restore с ключом --packages складывает в папку все пакеты решения вместе с зависимостями. Выполните её для каждого репозитория модулей: клона из GitLab или из папки modules\sources комплекта. Папка offline-packages общая. В примере клоны лежат в D:\repos:

> git clone C:\PilotProAdmin\Install\modules\sources\Pilot.ClientSdk.bundle D:\repos\Pilot.ClientSdk
>
> git clone C:\PilotProAdmin\Install\modules\sources\Reports.bundle D:\repos\Reports
>
> dotnet restore D:\repos\Pilot.ClientSdk\Pilot.ClientSdk.sln --packages D:\offline-packages -p:EnableWindowsTargeting=true
>
> Reports ссылается на пакет Pilot.ClientSdk, которого нет в nuget.org, а источник cicd в nuget.config репозитория указан образцом (gitlab.example.local): обычный dotnet restore падает с NU1301, и процедура выше этот пакет не создаёт. Соберите пакет из исходников в локальную папку внутри репозитория Reports (версию возьмите из PackageReference в Reports.csproj):
>
> dotnet pack D:\repos\Pilot.ClientSdk\Pilot.ClientSdk.csproj -c Release -p:Version=2.0.0 -o D:\repos\Reports\local-pkg
>
> Создайте файл D:\repos\Reports\nuget.local.config по образцу nuget.config репозитория (раздел packageSourceMapping оставьте) и укажите источник cicd = local-pkg: путь считается от папки файла. Затем соберите пакеты Reports:
>
> dotnet restore D:\repos\Reports\Reports.csproj --packages D:\offline-packages -p:EnableWindowsTargeting=true -p:RestoreConfigFile=D:\repos\Reports\nuget.local.config

Сборка в CI идёт на Linux и может запросить другие пакеты, чем Windows. Поэтому повторите то же самое в образе SDK, из которого работает CI, с тем же конфигом (папка local-pkg лежит внутри репозитория и видна в контейнере):

> docker run --rm -v D:\offline-packages:/out -v D:\repos:/src -w /src "mcr.microsoft.com/dotnet/sdk:<версия>@sha256:\<digest>" `
>
> dotnet restore Reports/Reports.csproj --packages /out -p:EnableWindowsTargeting=true -p:RestoreConfigFile=Reports/nuget.local.config

Образ SDK один для клиентских и серверных модулей (SDK 10.0.404). Для клиентских модулей на net48 набор пакетов от SDK 10 не изменился. Серверные модули (net8.0, netstandard2.0) собираются так же: dotnet restore в этом образе с тем же ключом --packages. SDK 10 при сборке net8.0 берёт из nuget.org пакеты Microsoft.NETCore.App.Ref, Microsoft.AspNetCore.App.Ref (если проект их использует) и Microsoft.NETCore.App.Host.linux-x64 версии 8.0.x, около 50 МБ вместе: с SDK 8 они лежали внутри образа и не скачивались. Команда restore кладёт их в ту же папку, и дальше они загружаются в реестр вместе с остальными пакетами; сборку серверного модуля без интернета проверьте на стенде.

Загрузить в реестр

Перенесите папку на машину в закрытой сети. Рядом с пакетами, в D:\offline-packages, создайте файл nuget.config:

> <?xml version="1.0" encoding="utf-8"?>
>
> <configuration>
>
> <packageSources>
>
> <clear />
>
> <add key="upload" value="http://<gitlab>:8080/api/v4/projects/<ID>/packages/nuget/index.json" allowInsecureConnections="true" />
>
> </packageSources>
>
> </configuration>

Атрибут allowInsecureConnections нужен, пока GitLab работает по HTTP. Запишите токен и загрузите файлы .nupkg, команды выполняются из этой папки:

> cd D:\offline-packages
>
> dotnet nuget update source upload --username <логин или имя токена> --password <токен> --configfile nuget.config
>
> Get-ChildItem . -Recurse -Filter *.nupkg | ForEach-Object {
>
> dotnet nuget push $_.FullName --source upload --skip-duplicate
>
> }

Ключ --skip-duplicate позволяет запускать загрузку повторно: уже загруженные версии пропускаются. Результат проверьте в проекте: раздел Deploy / Package registry.

Как положить отдельную библиотеку DLL

Библиотеку без пакета лучше завернуть в NuGet: версия фиксируется, а подключение работает так же, как у остальных пакетов. Создайте файл Vendor.Library.csproj рядом с библиотекой:

> <Project Sdk="Microsoft.NET.Sdk">
>
> <PropertyGroup>
>
> <TargetFramework>net48</TargetFramework>
>
> <PackageId>Vendor.Library</PackageId>
>
> <Version>1.0.0</Version>
>
> <IncludeBuildOutput>false</IncludeBuildOutput>
>
> <EnableWindowsTargeting>true</EnableWindowsTargeting>
>
> <NoWarn>$(NoWarn);NU5128</NoWarn>
>
> </PropertyGroup>
>
> <ItemGroup>
>
> <None Include="Vendor.Library.dll" Pack="true" PackagePath="lib/net48/" />
>
> </ItemGroup>
>
> </Project>

Соберите пакет и загрузите его той же командой:

> dotnet pack Vendor.Library.csproj -c Release -o D:\offline-packages
>
> dotnet nuget push D:\offline-packages\Vendor.Library.1.0.0.nupkg --source upload --skip-duplicate

Для архива или инструмента, который не подключается как пакет, используйте Generic-реестр:

> curl.exe --header "PRIVATE-TOKEN: <токен>" --upload-file tool.zip `
>
> "http://<gitlab>:8080/api/v4/projects/<ID>/packages/generic/tool/1.0.0/tool.zip"
>
> **Сборки клиента Pilot.** Модулю Reports нужны библиотеки клиента Pilot из папки lib/pilot. Они уже лежат в его репозитории, поэтому никуда перекладывать их не нужно.

Шаг 4. Как разработчику подключиться

Что нужно получить

- **Учётная запись GitLab** с доступом к группе модулей и проекту pilot-clientsdk.

- **Личный токен.** В разделе User Settings / Access Tokens создайте токен со scope read_api и сроком действия по правилам ИБ.
- **Адрес GitLab** и Project ID проекта pilot-clientsdk, их сообщает администратор.

Подключить источник один раз

> dotnet nuget add source "http://<gitlab>:8080/api/v4/projects/<ID>/packages/nuget/index.json" `
>
> --name cicd --username <логин GitLab> --password <токен>

На Windows токен шифруется средствами DPAPI и хранится в %AppData%\NuGet\NuGet.Config. Ключ --store-password-in-clear-text не используйте. Имя источника cicd должно совпасть с именем в nuget.config репозитория: по нему NuGet находит токен.

Поправить nuget.config в репозитории

В репозиториях модулей nuget.config содержит два источника: nuget.org и cicd с адресом-образцом (gitlab.example.local и номер проекта вместо вашего). В закрытой сети nuget.org недоступен, и восстановление пакетов падает с ошибкой NU1301. Оставьте один источник, впишите в него свой адрес и Project ID проекта client-modules/pilot-clientsdk и загружайте через него все пакеты. Раздел packageSourceMapping приведите в соответствие: все пакеты берутся из источника cicd.

```
<?xml version="1.0" encoding="utf-8"?>

<configuration>

<packageSources>

<clear />

<add key="cicd" value="http://<gitlab>:8080/api/v4/projects/<ID>/packages/nuget/index.json"
allowInsecureConnections="true" />

</packageSources>

<packageSourceMapping>

<packageSource key="cicd">

<package pattern="*" />

</packageSource>

</packageSourceMapping>

</configuration>
```

Вместо проекта можно указать группу: /api/v4/groups/<ID>/-/packages/nuget/index.json. Тогда видны пакеты всех её проектов, но загружать через такой адрес нельзя, только читать.

Получить код и пакеты

> git clone http://<gitlab>:8080/<группа>/<модуль>.git

> cd <модуль>
>
> dotnet restore
>
> dotnet build

Для git по SSH адрес такой: ssh://git@<gitlab>:2224/<группа>/<модуль>.git. По HTTP вместо пароля используйте токен со scope read_repository и write_repository.

Как это работает в CI

Менять ничего не нужно, если nuget.config указывает на тот же хост, по которому GitLab называет сам себя. Шаблон CI передаёт токен задания CI_JOB_TOKEN только источникам с /api/v4/, хост которых равен CI_SERVER_HOST (без порта): источник на другом хосте, а также на IP-адресе при том, что GitLab называет себя именем, токена не получит, и его пакеты не восстановятся (401). Нужно также, чтобы проект модуля был в списке Job token permissions проекта pilot-clientsdk, см. шаг 3. При установке из копии данных с ключом -GitLabHost (--gitlab-host) установщик на машине с ролью gitlab сам заменяет старый адрес GitLab на новый в nuget.config репозиториев групп модулей и отчётов.

Шаг 5. Как добавить новый пакет

1. Разработчик сообщает, какой пакет и какая версия нужны. Если ИБ требует проверку, она проходит на этом этапе.
2. Ответственный на машине с интернетом обновляет ссылку на пакет в решении, снова выполняет dotnet restore --packages и переносит папку.
3. Ответственный выполняет dotnet nuget push. Уже загруженные версии будут пропущены.
4. Разработчик получает пакет командой dotnet restore.

Права на загрузку стоит оставить одному ответственному: тогда в закрытый контур попадает только проверенное.

Проверка

Закройте машине разработчика и серверу доступ в интернет, затем проверьте цепочку целиком:

Что сделать	Что должно получиться
dotnet nuget locals all --clear, затем dotnet restore у разработчика	Пакеты восстанавливаются без ошибок. Команда очищает весь локальный кеш NuGet
Запустить pipeline модуля	Задания проходят, проверки ci-checks идут в образе CHECKS_IMAGE, в логе нет Pulling docker image для внешних реестров
docker images на машине с runner	Есть образы SDK, Alpine, Python или CHECKS_IMAGE и gitlab-runner-helper

Что сделать	Что должно получиться
Открыть страницы GitLab и посмотреть логи межсетевого экрана	Обращений GitLab в интернет нет

Типовые ошибки

Ошибка	Причина	Что делать
NU1301: не удалось загрузить индекс	Источник недоступен, чаще всего nuget.org остался в nuget.config	Оставить только источник GitLab
NU1101: пакет не найден	Пакета или его зависимости нет в реестре	Собрать и загрузить его, см. шаг 3
NU1302: HTTP-источник	SDK запрещает HTTP без явного разрешения	Добавить allowInsecureConnections="true" или перейти на HTTPS
401 Unauthorized при restore	Токен истёк или не имеет нужного scope	Создать новый токен и выполнить dotnet nuget update source
401 или 403 только в CI	Проекта модуля нет в Job token permissions либо источник в nuget.config на другом хосте, чем CI_SERVER_HOST	Добавить проект модуля в настройках проекта pilot-clientsdk и писать в nuget.config то же имя хоста, по которому GitLab называет сам себя
pull access denied или connection refused в job	Образа нет на машине с runner	Загрузить образ docker load, проверить pull_policy
manifest unknown при запуске compose	На сервере обычное хранилище образов, digest потерян	Включить containerd и загрузить образы заново