**Unoffical Goma Setup Guide** <https://kubala.github.io/>

Notes: The examples shows linux command, so doesn’t work in every cases... For Setting up Goma server first check instructions below.

**Goma client compile**

Follow instructions on <https://chromium.googlesource.com/infra/goma/client/>

Notes on compiling for Windows:

1. Install depot\_tools <http://commondatastorage.googleapis.com/chrome-infra-docs/flat/depot_tools/docs/html/depot_tools_tutorial.html#_setting_up>Set env vars (I don’t if it’s really necessary):
   * DEPOT\_TOOLS\_WIN\_TOOLCHAIN=0
   * vs2019\_install=”C:\Program Files (x86)\Microsoft Visual Studio\2019\<Community|Professional>”
   * or vs2017\_install=”C:\Program Files (x86)\Microsoft Visual Studio\2017\ < Community|Professional >”
2. Before building with: gn gen --args='is\_debug=false use\_lld=true' out/Release
   * Install/set Python2 (otherwise fails with Python3 “basestring not supported”)
   * Modify path vcvarsall.bat in file client\build\toolchain\win\setup\_toolchain.py (fails otherwise)
3. (Opt) Unit test must be executed in enclosing folder, not as written in instructions
4. Autoninja + gomacc: doesn’t substitutes compiler commands (cl.exe -> gomacc cl) in case of ninja builds, only with gn.

Hint: cmake -S <src> -B <build-dir> -G Ninja –DCMAKE\_C\_COMPILER\_LAUNCHER=<points to gomacc executable> –DCMAKE\_CXX\_COMPILER\_LAUNCHER=<points to gomacc executable> fixes at project generation

**Goma server compile**

Notes:

1. Install Go lang first
2. Follow instructions on <https://chromium.googlesource.com/infra/goma/server/>, docker for chromium is obsolete I think
3. client\_auth login uses OAuth2
4. Generate a service account json: <https://cloud.google.com/docs/authentication/production#cloud-console>and pass as arg to remoteexec\_proxy.
5. client\_auth login display license agreement which states some info, files, code will be forwarded to servers. **Must investigate** which data will be forwarded. We’ve tried first with 3rdparty source code, not ours.

**Setting up RBE/Buildgrid**

The [kubala](https://kubala.github.io/) site’s description is very accurate, but bit outdated.

Notes:

1. For building Buildgrid on Windows with Python 3.9 you have to call env\Scripts\python -m pip install grpcio after env\Scripts\python -m pip install --upgrade setuptools pip wheel otherwise is fails miserably. It’s searchin for an older version and is doesn’t work.
2. “I*n Goma server repo, edit cmd/remoteexec\_proxy/main.go and apply following patch*” part is not mentioned on <https://chromium.googlesource.com/infra/goma/server/>. I don’t know if it matters or not.

**Test findings**

We had established connection between the Buildgrid and Goma server, and also between Buildgrid bots and goma client. Compilation seemingly ran through gomacc. We could see tasks run in proxy\_compiler monitor (localhost:8088), but they have always fallen back to local compilation. Haven’t got any cache hit.

**Other helpful stuff**

These are my scripts, these may help I hope and not mislead😊

Looks interesting to do it with gn + clang, but we haven’t tried it:

<https://llvm.org/devmtg/2018-10/slides/Weber-FunnerLLVMDevelopment.pdf>