Contents

1 Synopsis

Git's partial clone allows cloning repositories without downloading all objects (blobs, trees, ...). These objects are fetched on demand from the remote when needed. However, when a user needs metadata about these remote objects (size, type, hash, ...), Git has no efficient way of doing this without downloading all the object content.

The server side support for `object-info` protocol was implemented by Calvin Wan in 2021 8. Eric Ju built the client-side `remote-object-info` for `cat-file --batch-command`.

This project finishes Eric Ju's work on `remote-object-info` for `git cat-file --batch-command` 1, resolves the pending feedback from Junio Hamano 2 and Jeff King 3 4 5, and extends support for `%(objecttype)`.

Expected project size: 350 hours (Medium)

2 About Me and Contact

Name: Pablo Sabater Jiménez (he/him)

Age: 19

Education: Currently on my second Computer Science year at University of Murcia, Spain

Location: Murcia, Spain (CET, UTC+2)

Languages: C (solid), shell(bash) (good)

Tools: git(proficient)

I've checked that I'm eligible for GSoC 2026.

Email: pabloosabaterr@gmail.com

GitHub: https://github.com/pabloosabaterr

3 Availability

My classes end the first week of May. From then until September I won't have any classes which leaves me free to fully focus on the project. I can dedicate 8+ hours each day, and for sure 40 hours a week.

4 Relevant Projects

- 16 bit CPU emulator. Good example of C programming.

  cpu: https://github.com/pabloosabaterr/CPU16

- Compiler. Good example of working on bigger projects.

  compiler: https://github.com/pabloosabaterr/Orn

5 Pre-GSoC Work

5.1 Introduction

[GSoC] Introduction Pablo Sabater

Description: A mailing list thread where I introduced myself to the git community.

5.2 Microproject

[GSoC PATCH v4] t9200: replace test -f/-d with modern path helpers

Status: Merged to `next` on 2026-03-12 at `8500bdf172`. Will merge to `master`.

Description: Replaces `test -f` with helper `test_path_is_file`, which makes debugging failing tests easier with better reporting. As suggested as microproject.

5.3 Draft Proposal

[GSoC] Proposal: Complete and extend remote-object-info for git cat-file

Description: Proposal draft thread.

5.4 Other Contributions

[GSoC PATCH v2] test-lib: print escape sequence names

Status: merged to `next` on 2026-03-13 at f545ea5a9c.

Description: In failed expected/actual checks printing, the escape sequences were shown as their octal code. This patch fixes that to print the actual escape sequence name, adds tests, and updates the expected output.

[GSoC PATCH] t9200: handle missing CVS with skip_all

Status: Merged to `next` on 2026-03-12 at `8500bdf172`. Will merge to `master`.

Description: Wraps CVS setup in a skip_all for clearer failure reporting and moves Git initialization into its own test_expect_success.

[GSoC PATCH v6 0/3] graph: add –graph-lane-limit option

Status: WIP.

Description: Adds `--graph-lane-limit` option to `--graph` to limit the number of horizontal lanes that will be shown. Helps readability for projects with many branches.

[GSoC PATCH 0/3] receive-pack: fix HEAD check for updateInstead

Status: pending review.

Description: Fix updateInstead HEAD check that only looked for the bare repo context instead of the worktree HEAD, which rejected the pushes even with the wt clean.

5.5 Code Reviews

Re: [PATCH] gc: add git maintenance list command

Description: Code review for a patch sent about simplifying a duplicated code.

Re: [PATCH] t2107: modernize path existence check

Re: [GSoC][PATCH] t2000: modernize path checks to use helper functions

Re: [PATCH] t5315: use test_path_is_file for loose-object check

Description: Reviews to newcomers on their microprojects patches.

[GSoC] Re: [PATCH v11 8/8] cat-file: add remote-object-info to batch-command

Description: While testing Eric's v11 I've found and reported a new bug. On `remote-object-info` when it's preceded by a local query, `data->type` isn't being cleared. Causing it to return the wrong type.

I have also studied the documentation provided and Eric Ju's work from v0 to v11 including all the feedback he got up to March 2025, the feedback he got from Junio Hamano and Jeff King, taking notes about what's left to be done and what else I can contribute to the already proposed project. That's how I've identified everything that I will address on the Problem, Solution and Timeline sections.

I built Eric Ju's v11 and tested the bugs reported to his patch 5, I've confirmed the segfault and the `die()`, and found a new one:

- When a local `info` runs before `remote-object-info` sharing the same format string, `data->type` isn't being cleared. A blob queried remotely after a local commit, `data->type` for blob becomes 'commit' with no error. I reported it on the mailing list 6.

I attempted to test rebasing Eric Ju's v11 to master and got conflicts on 4 out of the 8 commits:

- `d04cf85ece` t1006: split test utility functions into new "lib-cat-file.sh".
  - `t/t1006-cat-file.sh`
- `d918f720d8` fetch-pack: refactor packet writing.
  - `fetch-pack.c`
- `2daf9ed803` transport: add client support for object-info.
  - `Makefile`
- `c3ba4afaf6` cat-file: add remote-object-info to batch-command.
  - `object-file.c`, `object-store-ll.h` (deleted).

I'm being active on the mailing list, learning the Git flow of work and learning from the feedback I've received from the maintainers on my patches and reviewing others.

Following the project guidelines, I haven't done anything on the project that could step on other candidates' work before being accepted, and instead I'm focusing on understanding the project and its needs, and independent patches that will make the Git project more familiar and understandable to me.

6 The Problem

Eric Ju's work remains unmerged after v11 because of these issues:

- The format validation uses `strstr()` which only checks for `%(objectsize)`. This causes two different errors:
  - Atoms that `expand_atom()` recognizes but the remote doesn't (`objecttype`,`deltabase`, ...), `expand_atom()` returns 1, but when accessing `data->type` it only contains garbage, causing segfault, as Jeff King noted 3.

  - Unknown atoms by `expand_atom()`, returns 0, calling `strbuf_expand_bad_format` on `expand_format()`, which calls `die()`, as Jeff King found 3. Both cases block the command, including local `info` queries if the same format string is shared. Unsupported remote placeholders should return an empty string, matching how `for-each-ref` returns empty for known, but inapplicable atoms like `%(tagger)` on non-tags 4 5.

- When local and remote queries are mixed, `data->type` is not being cleared between commands. `remote-object-info` returns the wrong type data from a previous local query 6.

- Style and code issues marked by Junio Hamano 2 and Jeff King 3 5 are still undone.
  - comment style.
  - `#define` formatting.
  - line length.
  - misleading error messages.
  - missing `count > MAX_ALLOWED_OBJ_LIMIT` check at `split_cmdline()`.
  - if/else invert at `get_remote_info()`.

- `%(objecttype)` is not yet supported on either client or server side.

7 The Solution

There are two main goals:

7.1 Goal 1: Rebase and finish Eric's work

Starting from where Eric Ju left off, I will rebase it on top of the current `master` branch and address the feedback left to do:

- Fix style in comments, `#define` formatting and line length.
- Fix misleading error message in the overflow check.
- Add missing `count > MAX_ALLOWED_OBJ_LIMIT` check after `split_cmdline()`.
- Invert if/else on `get_remote_info()` to keep the small block first (the error one) as Junio suggested.

7.1.0.1 Replace `strstr()` format validation with allow_list in `expand_atom()` `strstr()` isn't enough to fully validate the placeholders, it only searches for `%(objectsize)` and unsupported placeholders cause segfaults. Jeff King noted 4 that the fix was to refactor the validation with an allow_list in `expand_atom()` or `expand_format()`. The best option is to place the validation at `expand_atom()`, but why `expand_atom()` ?

- There are two cases, first, inside `expand_atom()` before returning (segfault) and second, calls `die()` when `expand_atom()` returns 0. Placing the `allow_list` at the top of `expand_atom()` prevents both errors, on remote mode, append nothing to `sb` and return 1, accessing `data->type` won't cause segfault and prevents `expand_format()` from reaching `die()`. As extra safety, initializing `data->type` to `OBJ_BAD` and check for `NULL` from `type_name()` makes it that even without `allow_list`, uninitialized data doesn't cause a segfault. At Goal 1, only `%(objectname)` and `%(objectsize)` will be in the allow_list. Goal 2 will bring `%(objecttype)` support.

7.2 Goal 2: Adding `%(objecttype)`

Following what Calvin Wan did in 2021 8 for `%(objectsize)`, v2 protocol needs to be extended on the server side to support the new `%(objecttype)` placeholder:

- extend `object_info_advertise()` at `serve.c`
- add .type to `requested_info` struct at `serve.c`
- support `type` in `cap_object_info()` at `protocol-caps.c`
- look for type at `send_info()` at `protocol-caps.c`

Following object-info protocol docs 7 it should look like:

```
attrs = "size" SP "type"
obj-type = "blob" | "tree" | "commit" | "tag"
obj-info = obj-id SP obj-size SP obj-type
```

```
info = PKT-LINE(attrs LF)
       *PKT-LINE(obj-info LF)
```

`%(objecttype)` needs to be added to the `allow_list`. Client side needs to learn to ask for `%(objecttype)` from remote, parse what has been received and fill `expand_data` with the actual type. This makes it return the object type instead of the empty string returned while it was unsupported.

Default format evolves to `%(objectname) %(objecttype) %(objectsize)`. Test and document new placeholder support and server side extension.

7.3 Backward Compatibility

There are four possible scenarios to happen between client and server:

1. **The server doesn't know type (new client but old server)**:

   After receiving the server capabilities, the client doesn't see `type` being advertised. When the user format string has `%(objecttype)`, `expand_atom()` checks the `allow_list`, finds that type was not fetched. Appends an empty string to the output buffer and returns 1. The user will see an empty field where `type` should be, no errors nor warnings. In Eric Ju's v11, this would crash, as described in The Problem section, the `allow_list` from Goal 1 is what fixes this, following `for-each-ref` behaviour for known but inapplicable atoms as Jeff King suggested 4 5.

2. **The server knows type but the client doesn't (new server but old client)**:

   The server advertises `type`, but the client doesn't know `type` and following `gitprotocol-v2.adoc`, "Clients must ignore all unknown keys", it silently ignores the `type` and only asks for the known (`size`). The server returns only what was requested, user will see the output for `size` but not for `type`. This doesn't need any new code, the v2 protocol already behaves like this.

3. **Both know type (new client and new server)**:

   The server advertises `type`, the client requests `type` and receives the type data. `expand_atom()` finds `type` in the `allow_list`, fills `data->type` and then the user will see the object type in the output. This is Goal 2.

4. **Both know type but protocol middleware doesn't (new client, new server but old middleware)**:

   This becomes case 1 or 2 depending on what side is being affected by the middleware. If the middleware removes `type` from the server advertised capabilities, the client never sees it and treats the server as it was old server, it becomes case 1 (empty string). If the middleware removes `type` from the client request, the server will only see `size` being requested and only returns size data, it becomes case 2.

7.4 Performance Considerations

To get an object type, we have to look only at the header, to get the size `oid_object_info()` at `object-file.c` is being called which already returns the object type in the same call. Sending the string with the type will only be, worst case scenario 6 bytes for the "commit" string.

8 Timeline

I've designed this to work with enough time so final work can be shorter than what's said here

May 1-24: Community Bonding

- Keep working on my ongoing patches and new ones.
- Talk and meet with mentor that I'm assigned with, to get feedback about my proposal, how I will report my progress apart from the code submitted and possible blogs, and tips and tricks to work better at Git.
- Confirm with mentor that the `allow_list` approach is still the best option.
- Draft commits structure.
- Setup a blog to keep track about how GSoC at Git is going.

Week 1-2: (May 26 - June 8)

- First of all will be rebasing Eric Ju's v11.
- Start Goal 1 fixes.
- Fix style and code issues.

Week 3-4: (June 9 - June 22)

- Start with Goal 1 implementations (allow_list approach).

Week 5-6: (June 23 - July 6):

- Goal 1 should be polished or close to the final form.
- Send patch series for Goal 1.
- Start Goal 2.
- Prepare the midterm report.

Midterm evaluation (July 7 - 11) as specified on GSoC timeline docs

- Goal 1 submitted.

Week 7-8: (July 14 - July 27)

- Start with server side v2 protocol extension (`%(objecttype)`).

Week 9-10: (July 28 - August 10)

- Add `%(objecttype)` to the `allow_list` from Goal 1.
- Client side extension.
- End to end tests and documentation.
- Default format becomes `%(objectname) %(objecttype) %(objectsize)`.
- Send patch series.

Week 11-12: (August 11 - August 24)

- Goal 2 should be close to be done.
- Polish everything, all tests pass, good test coverage, no style/comment issues.
- Final documentation review.
- Prepare for final evaluation.

Final evaluation (August 18-24) as specified on GSoC timeline docs

8.1 Additional objectives

If there is enough time, or for future work after the project. I've some ideas on how this could evolve:

8.2 More placeholders support

I've checked that Eric's v11 patch only supports `%(objectsize)` on server side, but on the client side there are other placeholders that can be added too. With the `allow_list` and having Goal 2 implemented, adding more placeholders becomes trivial.

- `%(objectsize:disk)`: Returns the size on the disk (compressed or as a delta) instead of returning the uncompressed size that `%(objectsize)` does. To do this, the server would need to send what's the actual size on disk data.
- `%(deltabase)`: Returns the delta base object OID. non delta objects return zero OID as it does on local.

8.3 Returning missing blobs from a tree ordered

In a partial clone, someone might want to know what blobs are missing inside a concrete tree and order them before fetching them. The idea is to build on top of `remote-object-info` and what's been built in Goal 1 and Goal 2: Given a tree hash, return the missing blobs (inside that tree) ordered by an orderable atom (size, name, type, ...).

This looks similar to Stolee's work on `git-backfill` 9, the key difference is that `git-backfill` fetches the missing objects from a path/object, while this would only query the metadata of the missing blobs without fetching them and ordered by a given atom.

Thanks for reading my proposal and considering my application. I'm very excited about this opportunity,

Pablo

[1]: https://lore.kernel.org/git/20250221190451.12536-1-eric.peijian@gmail.com/ "Eric Ju's v11 patch"

[2]: https://lore.kernel.org/git/xmqqo6yr3wc4.fsf@gitster.g/ "Junio Hamano feedback"

[3]: https://lore.kernel.org/git/20250224234720.GC729825@coredump.intra.peff.net/ "Jeff King feedback"

[4]: https://lore.kernel.org/git/20250313060250.GH94015@coredump.intra.peff.net/ "options for strstr() by Jeff King"

[5]: https://lore.kernel.org/git/20250324033922.GB690093@coredump.intra.peff.net/ "Jeff King follow-up"

[6]: https://lore.kernel.org/git/20260312214154.89120-1-pabloosabaterr@gmail.com/ "data->type not being cleared bug"

[7]: https://github.com/git/git/blob/master/Documentation/gitprotocol-v2.adoc#object-info "object-info protocol docs"

[8]: https://lore.kernel.org/git/20220728230210.2952731-1-calvinwan@google.com/#t "Calvin Wan's patch series"

[9]: https://lore.kernel.org/git/pull.2070.git.1773707361.gitgitgadget@gmail.com/ "git-backfill extension from Stolee"