Go Module Problems

Problem Description

1. **Problem 1**: I expect go module to discover this while "go get github.com/notional-labs/tinyport@8fb5769c543a0702a400a516e8d0116afa9a41ad". But in reality, it returns:

```
https://sum.golang.org/lookup/github.com/notional-labs/tinyport@v0.18.1-
0.20220307035957-8fb5769c543a:
410 Gone
```

That code makes me super confused cause this revision: "github.com/notional-labs/tinyport@8fb5769c543a0702a400a516e8d0116afa9a41ad" is public. It is supposed to be found.

--> package is found and downloaded after discovery 3. However, new problem arises.

2. **Problem 2**:

```
go: finding module for package github.com/notional-
labs/tinyport/tinyport/pkg/openapiconsole
go: finding module for package github.com/notional-
labs/tinyport/tinyport/pkg/cosmoscmd
github.com/notional-labs/temp/app imports
    github.com/notional-labs/tinyport/tinyport/pkg/cosmoscmd: module
github.com/notional-labs/tinyport@latest found (v0.19.4), but does not contain package
github.com/notional-labs/tinyport/tinyport/pkg/cosmoscmd
github.com/notional-labs/temp/app imports
    github.com/notional-labs/tinyport/tinyport/pkg/openapiconsole: module
github.com/notional-labs/tinyport@latest found (v0.19.4), but does not contain package
github.com/notional-labs/tinyport/tinyport/pkg/openapiconsole
```

It is so strange to see that starport folders and files are in tinyport package, weird?

--> missing package is found after discovery 6. However, new problem arises.

3. **Problem 3**:

```
go: github.com/notional-labs/tinyport@v1.1.0 used for two different module paths
(github.com/notional-labs/tinyport and github.com/tendermint/starport)
exit status 1
```

Go proxy realizes that these two are different module paths leading to exit status.

-->Problem is solved after conclusion 1

Discovery (I am batman :>)

1. sum.golang.org - an auditable checksum database which will be used by the go command to authenticate modules. The go command can use this in many situations to detect misbehavior by proxies or origin servers.

- It could be that: after downloading module through go proxy, it goes to sum.golang.org for verifying the module that has just been downloaded. This is where shit fucks up.

2. proxy.golang.org - A module proxy is an HTTP server that can respond to GET requests for paths specified below (module querying)

    - when I "curl https://proxy.golang.org/github.com/notional-labs/tinyport/@latest", the response still returns. It means that go module proxy can still find my module. But why no revision v0.18.1-0.20220307035957-8fb5769c543a ?

3. Further reading shows me that proxy.golang.org will index packages by (Semantic Verion) [https://semver.org/].

    - I create a release: "v0.19.4-tinyport.1" that reflects this revision 8fb5769c543a0702a400a516e8d0116afa9a41ad and proxy.golang.org has been able to identify it.
    - Probably will work by now

4. Discovery 4: after a version A is released and go proxy catches that version source code. Even if I change version A later, the code remains the same in go proxy.

    - the only way to update that persists to go proxy is to create a new version.

5. Discovery 5: why "v0.19.4" is considered later than "v0.19.5-tinyport.1"?

    - that is because "v0.19.5-tinyport.1" is considered a pre-release version. By specification 11.4, a pre-release version has lower precedence than a normal version. Therefore, "v0.19.4" is considered later than "v0.19.5-tinyport.1".
    - if this mechanism applies, I should have deleted all starport tags on tinyport repo. But, those tags will still be recognized in go proxy server. F**k.

6. Discovery 6: you cannot delete version in go proxy :((. Once a version is recognized, it stays there forever unless one files a request to remove those version. One can only retract it

    - the only way for go proxy server to recognize tinyport is to create a newer **MAJOR** version than starport (v0.19.4) --> v1.1.0

Conclusion

1. For problem 3, if go proxy server realizes that these two module paths are different, we can directly insert "github.com/notional-labs/tinyport v1.1.0" into require() in go.mod