**Problem description**

* Need a way of communicating between two switch NOS process exchanging IPC messages related to ACL/NAT/IPSEC/Routing/Tunnel and many more features.
* We also need a way of vendor neutral way.
* This made us look at SAI which is providing that vendor neutrality.
* SAI has headers and thrift framework that can be used as communication mechanism to use between 2 process in client/server fashion.
* Our NOS is well versed with gRPC way of working and encoding based on proto-buffers.
* This made us think of generating a .proto file from existing SAI headers and use it as communication between systems and data-serialization.

A diagram of a network

Description automatically generated

**Methodology to reuse Thrift framework and generate grpc**

Adding new args/Files:

* Adding a new Argument in SAI/Makefile
  + A new argument will be added into the SAI/Makefile to take from user if he is interested in generating Thrift/protobuff based RPC.
  + The same will be passed to GEN\_SAIRPC\_OPTS.
* Adding a new target in meta/makefile
  + GEN\_SAIRPC\_OPTS options will be passed to meta/Makefile
  + Here, further a new target will be made for grpc.
  + The grpc target will be only be built if the argument given by the user matches grpc.
  + Sample code:

*rpc sai.thrift sai\_rpc\_server.cpp sai\_adapter.py: xml $(XMLDEPS) gensairpc.pl*

*perl -Irpc gensairpc.pl $(GEN\_SAIRPC\_OPTS)*

*grpc sai.proto : xml $(XMLDEPS) gensaigrpc.pl*

*perl -Irpc gensaigrpc.pl $(GEN\_SAIRPC\_OPTS)*

* Adding a new gensairpc.pl
  + A new gensaigrpc.pl which is mostly similar to gensairpc.pl.
  + It will have grpc based changes.
* Adding a new template file for generating proto (sai.grpc.tt)
  + A new sai.grpc.tt will be added to generate the .proto.

Use of existing Files:

* Intend to use the existing perl module (\*.pm) files.
* Use existing pearl files, .c files, .sh files.

End output:

* Intend to generate .proto as of now.
* The user can take the proto file and generate his client/server code in his own language and integrate in his NOS.

Right now, we don't have any GPB notes/HLD for SAI. We will try to comeup with a draft in few weeks if community is OK to go-ahead with this layer on top of SAI.  
At high level, the thought process is to use the same xml generated (while compiling for thrift rpc) and use that xml to generate a .proto, server and client libraries for all supported features in SAI.

We are thinking about:

* when we "make sai",
  + .proto files will be generated along with gpb server and client APIs.
  + libsai.so will be built containing the server-code, sai-adapter.
  + clinet lib.so will also be build.

Highlevel changes anticipated: