SONiC unit test and function test enhancement

Taskin Ucpinar

EdgeCore

Edgecore SONiC Development and Testing Environment

Towards a deployment-ready SONiC

- End-to-end testing in development workflow
- Coverages and Automation
- HW platform validations
- Pre-SI RAS, Performance testing
- Sonic ecosystem integrations

Edgecore is Dedicated to Future of SONiC

AS7816-64X

AS7212-54X

AS6712-32X

AS7712-32X

AS9716-32D

AS7326-56X

AS7312-54X

AS7712-32X

AS7312-54XS

AS4222-28PE

AS5712-54X

AS7312-54XS

AS7716-32X

TO BE ANNOUNCED DURING OCP

AS7116-54X

AS7512-32X

AS7716-32XB

SONiC

Edge-corE NETWORKS

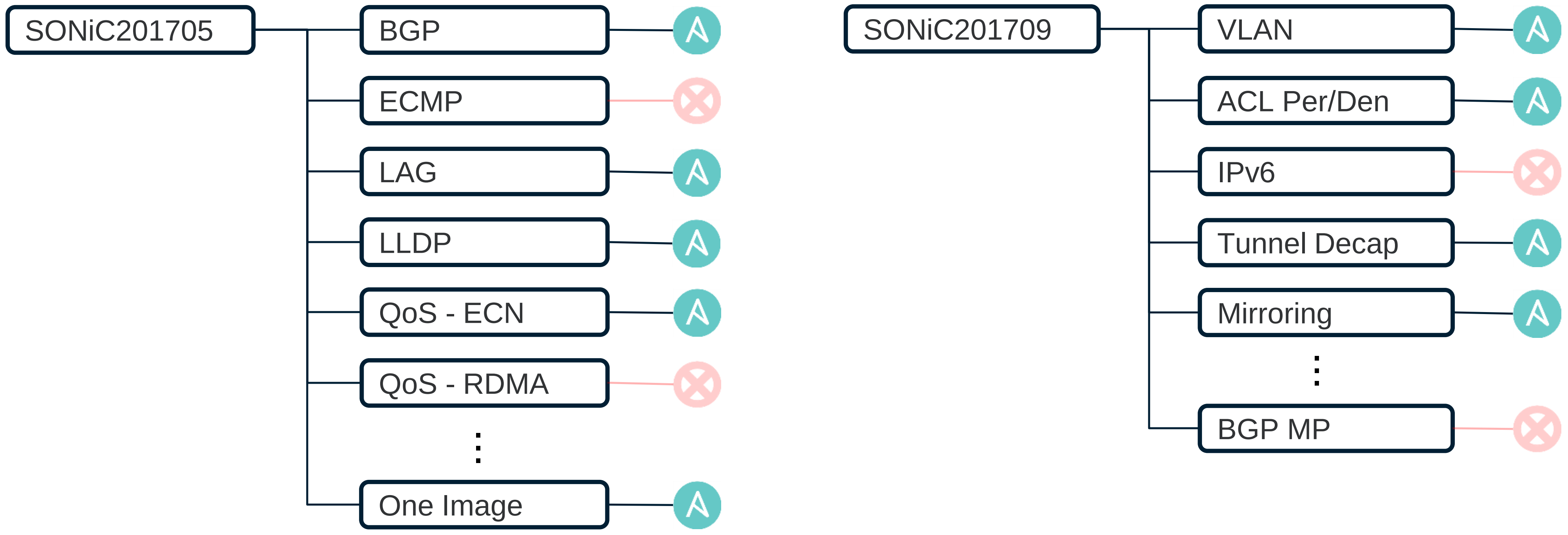
Future of SONiC

- SONiC Feature Set Growth
- Rapid Development Environment
- Ensure Stability/Reliability
- Testing
  - Interoperability
  - Regression
  - CI/CD

Testing - Regressions

- Getting the full picture
- Completing the coverage
- Hardening
- Community Services

Getting The Full Picture



Getting The Full Picture

Release	SAI Version	No of Features	No of Ansible Tests
SONiC201705	0.9.4	18	9
SONiC201709	0.9.4	8	4
SONiC201712	1.0	7	2
SONiC201803	1.2	5	2
SONiC201807	1.3	3	0
SONiC201811	1.3	6	2
SONiC.201903	TBD	16	1
Total		**63**	**20**

Edge-corE NETWORKS

Testing From All Angles

- Test units/components/functions/functionality
- Not only for testing, but for educational purposes
- Independent on features/platforms
- Unit Tests: Typically implemented by

Completing The Picture

100% Coverage

68.3%

- All Features
- Features Missing Tests
- Coverage % Unknown
- Stability Unknown

Automation

- Auto detect and execute
- Regression Testing

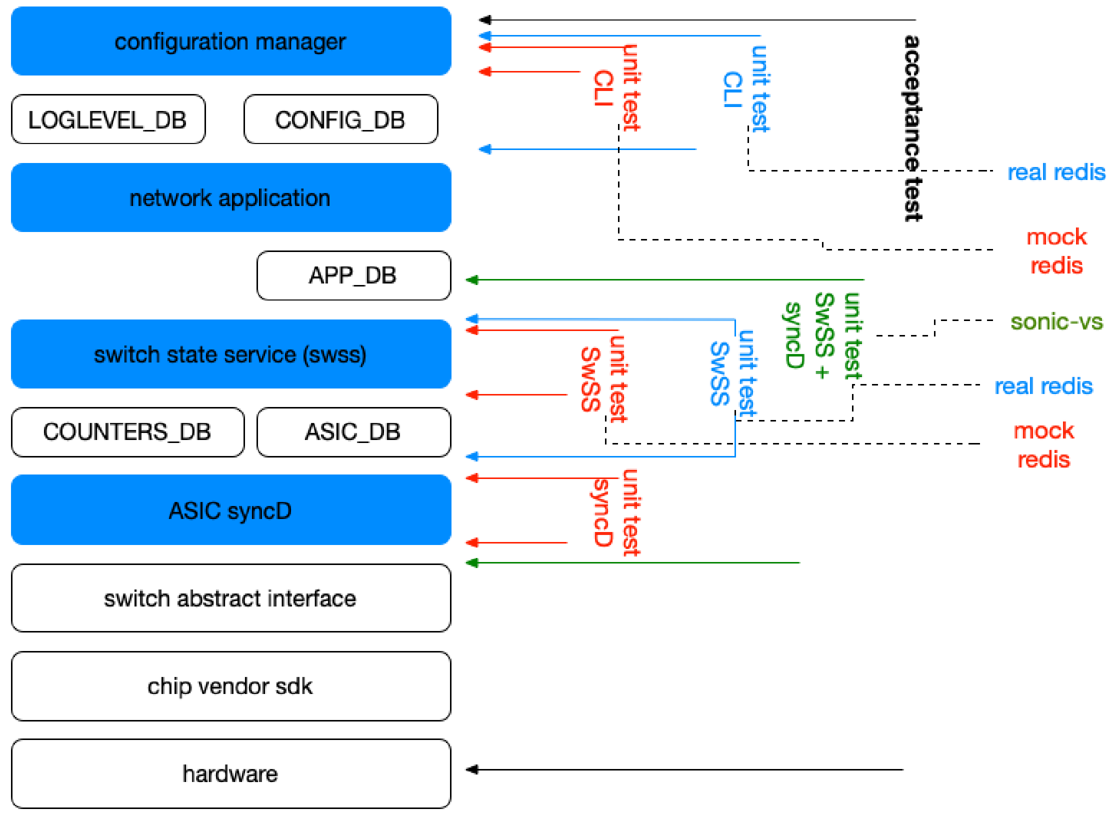
Hardening

- Stability Improvements
- Measure Coverage %
- New tests

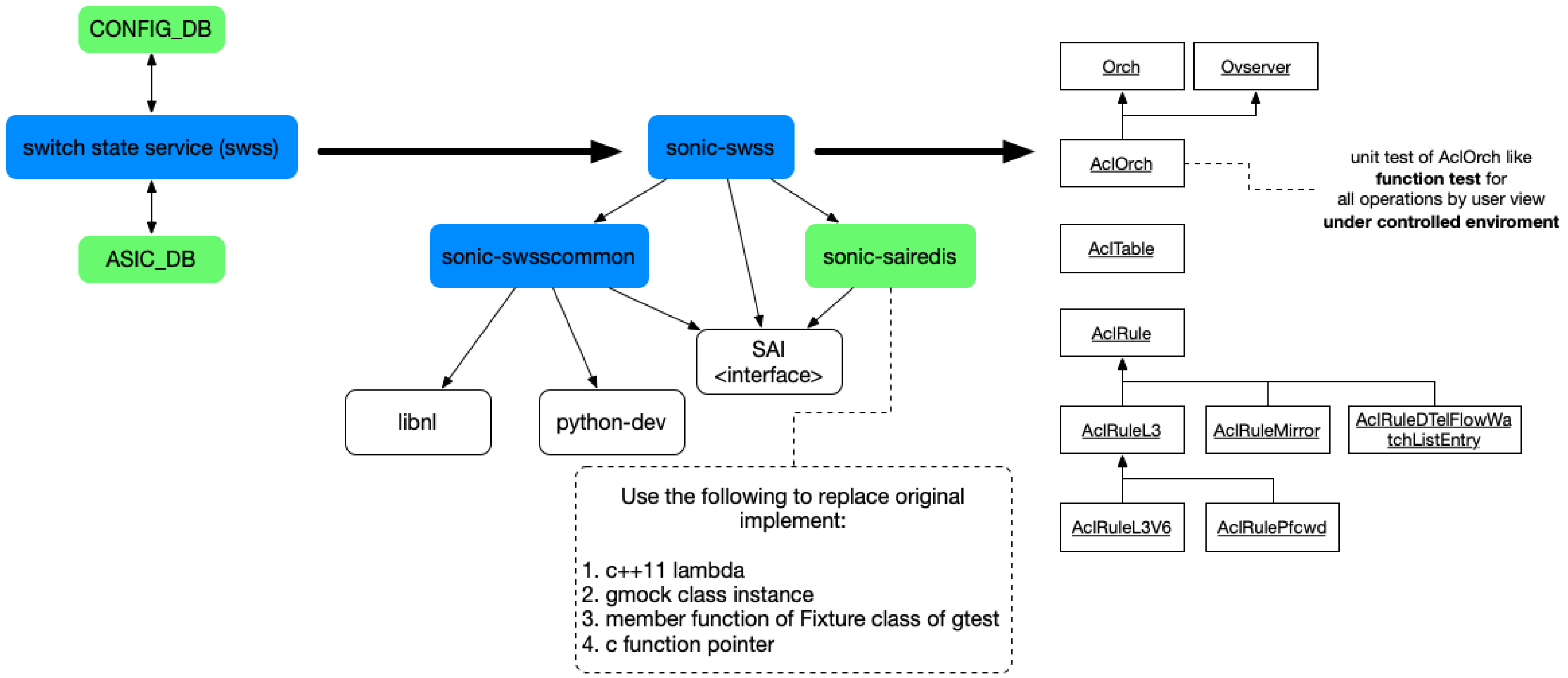
Edgecore Community Labs

- Multiple Community Labs Across Globe
- SONiC Devices and Packet Generators
- Available to Edgecore Partners, Customers, Community

Unit Test Software Architecture



ACL/SwSS Logical View, an Example

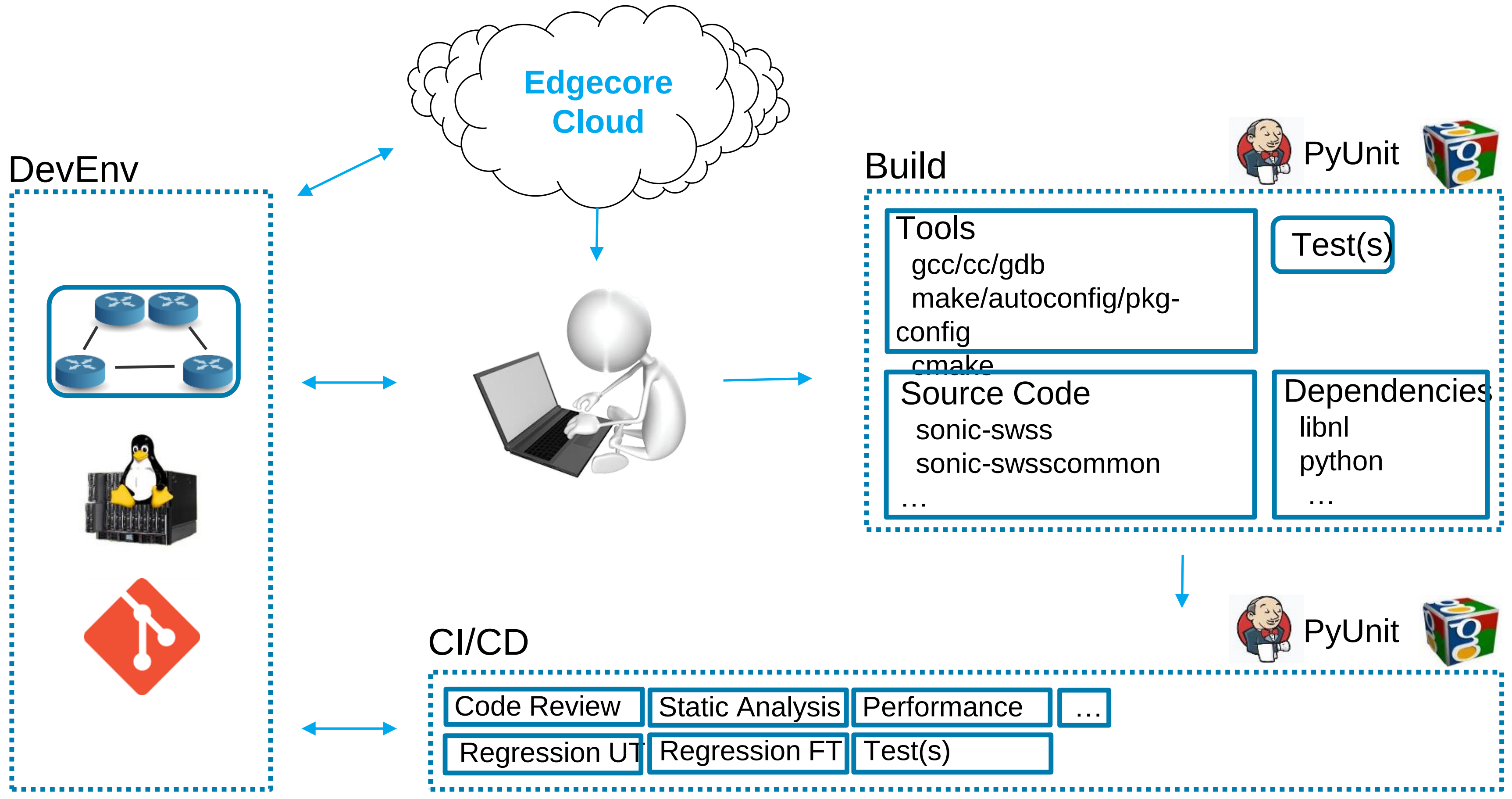


unit test of AclOrch like **function test** for all operations by user view **under controlled enviroment**

Use the following to replace original implement:

1. c++11 lambda
2. gmock class instance
3. member function of Fixture class of gtest
4. c function pointer

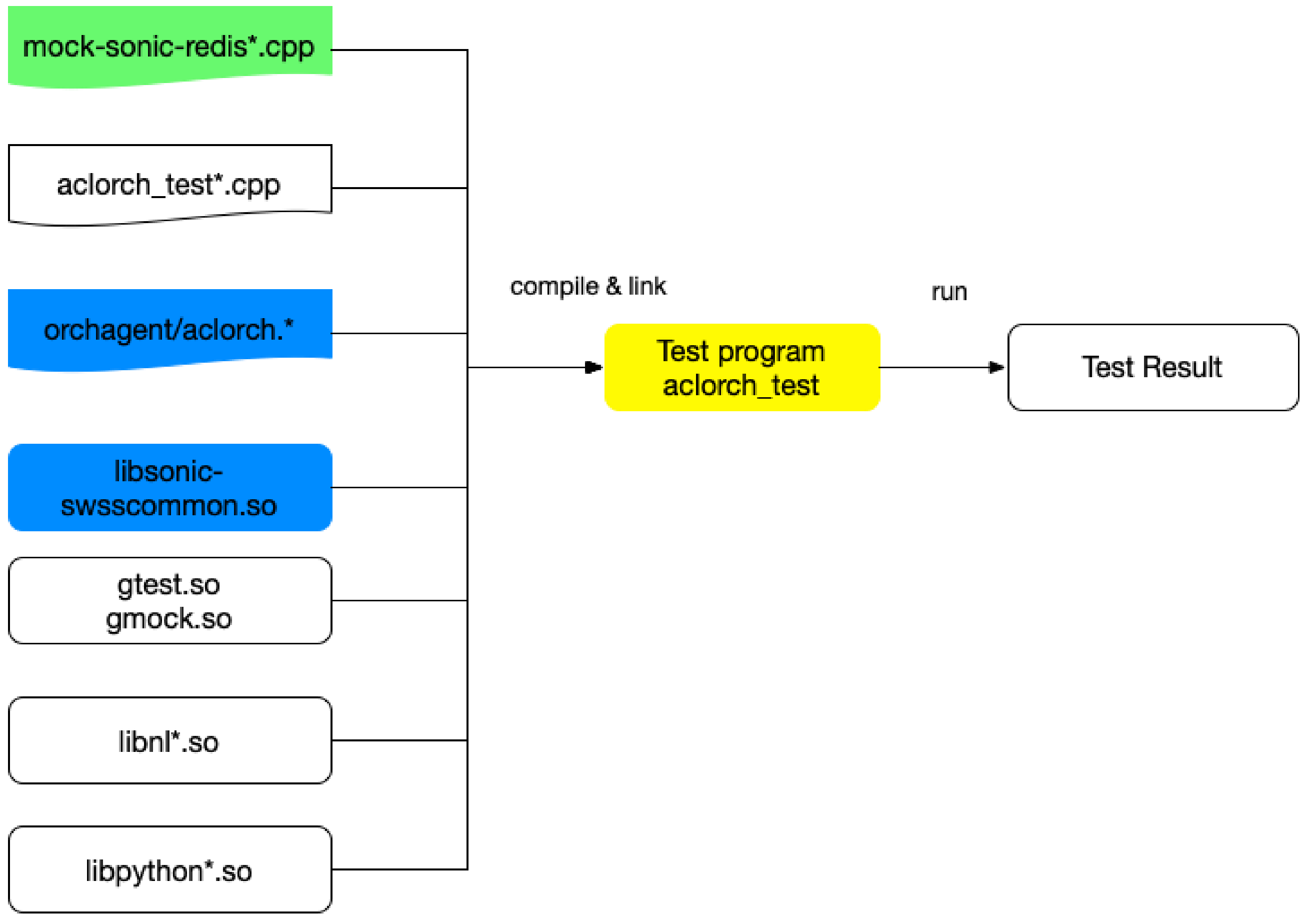
Environment



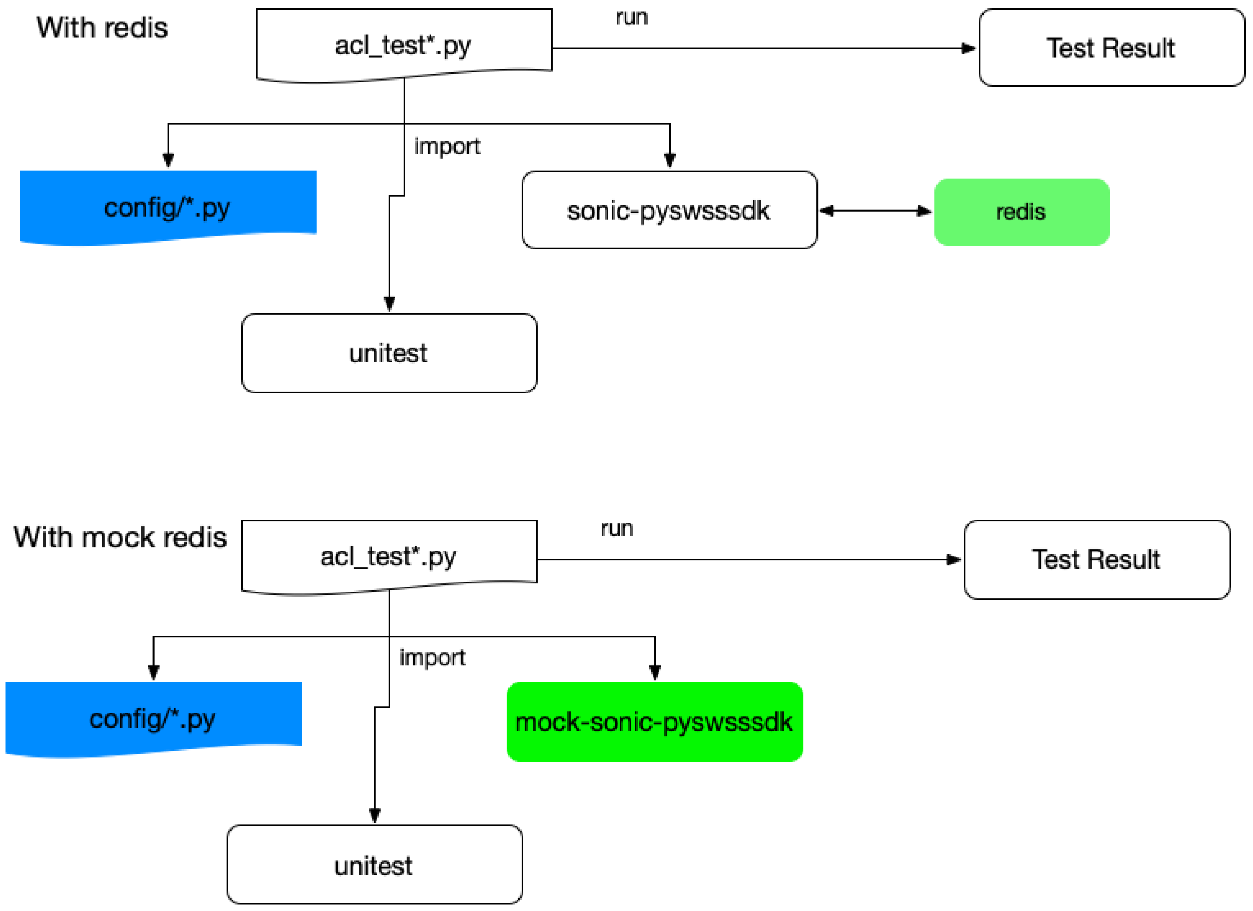
Thank You

Backups

for example


mock-sonic-redis*.cpp
aclorch_test*.cpp
orchagent/aclorch.*
libsonic-swsscommon.so
gtest.so
gmock.so
libnl*.so
libpython*.so
compile & link
Test program
aclorch_test
run
Test Result

ACL/CLI testing



How Gtest Works

Run gtest

- Download and compile gtest into static library
- Create gtest project
- Create a test case – import source code and dependent packages
- Add instrumentation code to source code to be tested
- Run the test

Run gtest for SONiC

- Test behavior before code modification
- Modify code (add code, change code such as bug fixing)
- Test behavior and compare with before

Gtest for SONiC

Our targets

- Create an unit test framework for SONiC contributors as a developing and debugging tool
- Reduce the work that contributors involve to run a test
- Easy management interface

Gtest advantages

- Run the exact same test repeatedly
- Track the context state info when hitting a bug

Constraints of Gtest on SONiC

- Use production language framework
  - Use Python/Go framework for code in languages such as Python/Go
- Typically not cover complicated operations such as send/receive packets, database operations etc.
- Also typically not cover script/shell code

SONiC

- Gtest levels
  - Simulate referenced components or do component crossing test depending on feature/developer requirements
    - Redis, SAI, socket, …
- Gtest performance
  - More closer to real environment, more time to run
- Import source code dependencies
  - Libraries, packages, …
- Test code in container image
  - Container is not required to run gtest, but take effort to run on host directly

ACL Example

- SONiC is composed of components such as each has its own build and unit test code
- ACL
  - Redis (Config) – SWSS/Orchagent – Redis (ASIC) – syncd – SAI
  - Swss-common/sairedis/hiredis reqired; SAI and Redis not necessary
  - 3 levels
    - Mimic database and SAI
    - Use real database and mimic SAI
    - Use real database and virtual switch

STOP
TAKE A BREAK