7.6 - The PHP-CLI API

- The PHP-CLI API provides a binding to the TET C API
- All the functions provided by the API are referenced from a single PHP module `phptet.php`
- This module must be included at the start of the test set code

```
include “phptet.php”
```

Interface to the user-written code

- The PHP-CLI binding uses the TET dynamic interfaces, so unlike the Shell and Perl APIs you do not need to setup an iclist for the TCM to communicate with the code

  - Thereby simplifying the writing of test code
Defining a PHP-CLI Test Set

- The test set is contained in a single file consisting of an optional startup and cleanup function and a series of test functions.

Defining the startup and cleanup functions

- At the start of the test set are the optional startup and cleanup functions:
  
  ```php
  function startup()
  {
    tet_infoline("Test startup");
  }

  function cleanup()
  {
    tet_infoline("Test cleanup");
  }
  ```
Defining the test methods

- The test functions are defined in sequence
  
  ```
  test1()
  {
    tet_infoline("Calling test 1")
  }
  
  test2()
  {
    tet_infoline("Calling test 2")
  }
  ...
  ```

Defining the test list array

- The test list is defined as an associative array

  ```
  $testlist = array(1 => "test1", "test2", "test3");
  ```
Registering the test functions

- To run the tests you must register the test functions

  \texttt{phptet\_init(}$\texttt{testlist, \texttt{“startup”, \texttt{“cleanup”);}$

- If there is no startup or cleanup

  \texttt{phptet\_init(}$\texttt{testlist, TET\_NULLFP, TET\_NULLFP);}$

PHP-CLI API Features - Using the TET API

- Uses the same TET C API names

  \begin{verbatim}
  $myvar = tet\_getvar("MY\_CONFIG\_VARIABLE");
  tet\_infoline("output text goes here");
  tet\_result(TET\_PASS);
  tet\_delete(1);
  \end{verbatim}
PHP-CLI API Features - TET Globals

- TET Globals must be declared before being referenced in a test function

```php
function test1() {
    global $tet_thistest;
    global $tet_pname;

    tet_infoline("This test is " . $tet_thistest);
    tet_infoline("This testset is " . $tet_pname);
}
```

PHP-CLI API Features - Handling Errors

- Errors are handled through return values

```php
test1() {
    $var = tet_getvar("MY_VAR");
    if ($var == "") {
        tet_infoline("Parameter MY_VAR is missing");
        return;
    }
}
```
Note

- The PHP-CLI TET API wrapper is generated using the SWIG toolkit that has some limitations which mean that not all the TET C API features are presently available in PhpTET API, e.g. varargs (tet_printf())

Building the phptet Module

- Need to have installed a supported version of TET and PHP
- Set TET_ROOT in the environment
- Check the settings of parameters in the Makefile
  - PHP_DIR - Location of the PHP include directory
- Build the phptet module using the make utility
Building the phptet Module

$ ls
Makefile   php.ini   phptet.php phptet_dyn.php
Makefile.swig php_phptet.h phptet.ppt phptet_profile
README     phptet.i   phptet_demo phptet_wrap.c

$make
/usr/bin/gcc -I/usr/local/include/php -I/usr/local/include/php/main \
   -I/usr/local/include/php/Zend -I/usr/local/include/php/TSRM \
   -c phptet_wrap.c
/usr/bin/gcc -shared -fPIC -o php_tet.so phptet_wrap.o \
   /user2/tet/lib/tet3/tcm.o /user2/tet/lib/tet3/libapi.a

Setup for using the phptet module

- Edit the php.ini file
- Set include_path to the location of the phptet modules (phptet.php and phptet_dyn.php)
- Set extension_dir to the location of the shared library (php_tet.so)
- See the README for more detailed instructions
Using the PHP-CLI TET Module

- Include the `phptet.php` module
- Define the startup and cleanup functions (optional)
- Define the test functions
- Define the test list array

Register the test list, startup and cleanup methods with `phptet_init()`
- Invoke `tcc` to run the test cases
Example PHP-CLI test case

```php
#!/usr/bin/env php
<?
include "phptet.php";

function test1()
{
    $var = tet_getvar("PHPTET_TC1_VAR");
    if ($var == "")
    {
        tet_infoline("Failed to get a value for PHPTET_TC1_VAR");
        tet_result(TET_UNRESOLVED);
        return;
    }
    tet_infoline("PHPTET_TC1_VAR is set to " . $var);
    tet_result(TET_PASS);
}

function test2()
{
    tet_result(TET_UNTESTED);
}

$testlist = array(1 => "test1", "test2");
phptet_init($testlist, TET_NULLFP, TET_NULLFP);
?>
```

The phptet_demo test suite

- phptet_demo is an example test suite

```
```

Mar-05 TETware training course 7-82

Mar-05 TETware training course 7-83
Invoke tcc to run the phptet_demo test suite

- Ensure that you environment is set, see phptet_profile
- Assuming phptet_demo is installed under TET_ROOT, change to the directory and invoke tcc:
  - tcc -p -e
- Otherwise set TET_SUITE_ROOT to where the phptet_demo directory is and invoke tcc (See next slide)

Running the phptet_demo test suite

[phptet_demo] ls
results tests tet_code tet_scen tetexec.cfg
[phptet_demo] export TET_SUITE_ROOT=`pwd`
[phptet_demo] tcc -p -e .
tcc: journal file is /home/neil/work/phptet/phptet_demo/results/0001e/journal
11:58:08 Execute /tests/tc1
11:58:09 Execute /tests/tc2
11:58:10 Execute /tests/tc3
[phptet_demo]