PHP output buffering analysis in c code level

whoami

- Name: 김승현 (sqrtrev)
- Age: 24
- E-mail: sqrtrev@gmail.com
- https://vuln.live
- Super Guesser 팀장
- 고려대학교 재학
- 정보보호대학원 학석사 연계과정

career

- HITCON CTF 2020 우승
- Dragon CTF 2020 우승
- Balsn CTF 2020 우승
- DEFCON 29 Finalist
- Line ctf 2021 3위
- zer0pts ctf 2021 우승

Review: JustCTF 2020 – Baby CSP

```php
if(isset($_GET['flag'])){
 if(isAdmin()){
    header('X-Content-Type-Options: nosniff');
    header('X-Frame-Options: DENY');
    header('Content-type: text/html; charset=UTF-8');
    echo $flag;
    die();
 }
 else{
     echo "You are not an admin!";
     die();
 }
}

for($i=0; $i<10; $i++){
    if(isset($_GET['alg'])){
        $_nonce = hash($_GET['alg'], $nonce);
        if($_nonce){
            $nonce = $_nonce;
            continue;
        }
    }
    $nonce = md5($nonce);
}

if(isset($_GET['user']) && strlen($_GET['user']) <= 23) {
    header("content-security-policy: default-src 'none'; style-src 'nonce-$nonce'; sc
    echo <<<EOT
        <script nonce='$nonce'>
            setInterval(
                ()=>user.style.color=Math.random()<0.3?'red':'black'
            ,100);
        </script>
        <center><h1> Hello <span id='user'>{$_GET['user']}</span>!!</h1>
        <p>Click <a href="?flag">here</a> to get a flag!</p>
EOT;
}else{
    show_source(__FILE__);
}
```

- user를 받으면 CSP를 헤더에 포함시킴
- nonce를 hash할 알고리즘을 입력받음
- flag를 GET으로 받을 경우 admin인치 체크하고 flag 출력

Review: JustCTF 2020 – Baby CSP

```php
5   if(isset($_GET['flag'])){
6    if(isAdmin()){
7       header('X-Content-Type-Options: nosniff');
8       header('X-Frame-Options: DENY');
9       header('Content-type: text/html; charset=UTF-8');
10      echo $flag;
11      die();
12   }
13   else{
14       echo "You are not an admin!";
15       die();
16   }
17  }
18
19  for($i=0; $i<10; $i++){
20      if(isset($_GET['alg'])){
21          $_nonce = hash($_GET['alg'], $nonce);
22          if($_nonce){
23              $nonce = $_nonce;
24              continue;
25          }
26      }
27      $nonce = md5($nonce);
28  }
29
30  if(isset($_GET['user']) && strlen($_GET['user']) <= 23) {
31      header("content-security-policy: default-src 'none'; style-src 'nonce-$nonce'; sc
32      echo <<<EOT
33          <script nonce='$nonce'>
34              setInterval(
35                  ()=>user.style.color=Math.random()<0.3?'red':'black'
36              ,100);
37          </script>
38          <center><h1> Hello <span id='user'>{$_GET['user']}</span>!!</h1>
39          <p>Click <a href="?flag">here</a> to get a flag!</p>
40  EOT;
41  }else{
42      show_source(__FILE__);
```

Warning: hash(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in **/home/sqrtrev/www/static/just.php** on line **23**

Warning: hash(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in **/home/sqrtrev/www/static/just.php** on line **23**

Warning: hash(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in **/home/sqrtrev/www/static/just.php** on line **23**

Warning: hash(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in **/home/sqrtrev/www/static/just.php** on line **23**

Warning: Cannot modify header information - headers already sent by (output started at /home/sqrtrev/www/static/just.php:23) in **/home/sqrtrev/www/static/just.php** on line **34**

Hello !!

- Warning으로 output buffer size인 4096을 채움
- Output buffer가 가득 차서 output이 먼저 sent
- 그 이후 header() 함수를 호출하므로 CSP가 적용 X
- User에서 XSS 트리거

Before analysis

Output buffer가 가득 차면 Response를 먼저 보낸다?

분석 전 추측(Guessing)

- php.ini에 명시되어 있는 output_buffer 만큼 response를 위한 공간 할당한다?
- Socket위에서 통신을 할 것이니 PHP가 php.ini에 명시된 output buffer 크기만큼만 send를 반복한다?
- response가 임시 파일로 저장된다고 가정하면, PHP에서 보내줄 때 php.ini에서 명시된 output buffer 만큼만 읽어서 전송한다?
- Output buffer 사이즈에 도달하면 PHP는 일단 현재 버퍼를 전송한다?

Author(terjanq)'s comment

Order matters

Normally, in PHP, when you return any body data before `header()` is called, the call will be ignored because the response was already sent to the user and headers must be sent first. In the application there was no explicit data returned before calling `header("content-security-policy: ...");` but because warnings were displayed first, they went into the response buffer before the header had a chance to get there in time.

PHP is known for **buffering the response to 4096** bytes by default, so by providing enough data inside warnings, the response will be sent before the CSP header, causing the header to be ignored. Hence, it is possible to execute our SVG payload.

There is also another limit for the size of the warning (1kb if I recall correctly) so it is needed to force around 4 warnings 1000 characters each.

- 순서 문제
- header()가 호출 전에 response 될 경우 header() 무시
- CSP가 추가 되기전에 다른 명시적 data 리턴이 없음
- 다만, warning은 response 버퍼에 먼저 담김
- 따라서 header 만나기 전에 버퍼가 에러로 가득차서 header()가 실행될 기회를 잃음
- 가득 찬 버퍼는 Client에게 보내짐

Before analysis

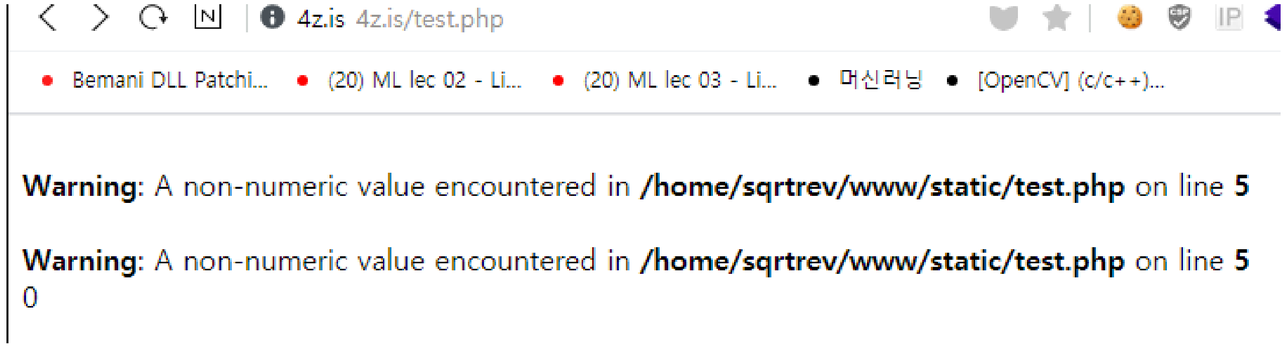
분석 전 간단한 테스트

테스트 1

```php
<?php
ini_set('display_errors',true);

echo 'sqrt'+'rev';

header("test: 123");
?>
```



▼ Response Headers view source

Connection: Upgrade, Keep-Alive
Content-Encoding: gzip
Content-Length: 128
Content-Type: text/html; charset=UTF-8
Date: Mon, 21 Jun 2021 06:48:24 GMT
Keep-Alive: timeout=5, max=100
Server: gws
test: 123
Upgrade: h2
Vary: Accept-Encoding

Warning이 존재한다. 다만, header() 함수는 정상 실행

Before analysis

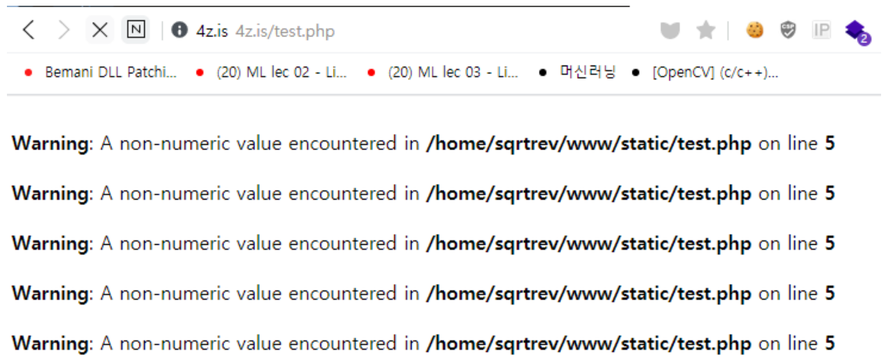
분석 전 간단한 테스트

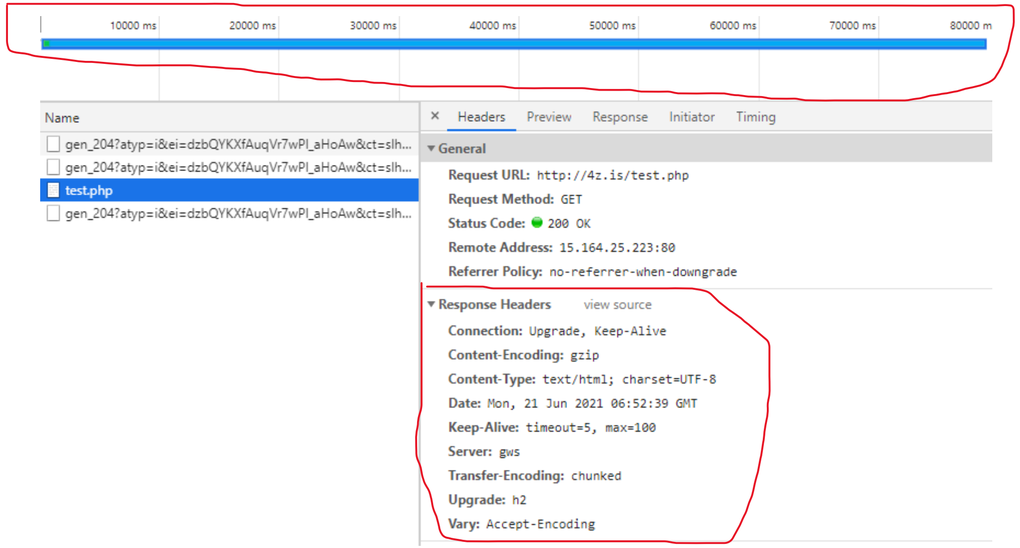
테스트 2

```php
<?php
ini_set('display_errors',true);

for($i = 0; $i < 0xffff; $i++)
        $tmp = 'sqrt'+'rev';

header("test: 123");
?>
```





상당히 의미 있는 결과가 나왔다.

- 페이지 로드 시, 헤더에는 test: 123이 존재하지 않았다.
- 페이지 통신바에서 test.php와 통신이 수 초 동안 진행했다.
- Wireshark를 이용한 패킷 캡쳐 시, header는 평문으로,
  content는 일부가 gzip으로 압축되어 보내졌다.
- 이후 나머지 부분이 특정 사이즈의 gzip으로 여러 번으로 나눠져 보내졌다
- 최초 TCP 연결 시, header를 제외한 gzip 데이터는 0x1faa(8106) 만큼 보내짐
- 이후 gzip 데이터는 0x1fa0(8096) 만큼 보내짐

Analysis

분석 대상: PHP-7.4.13

```
PHP_INI_BEGIN()
    PHP_INI_ENTRY_EX("highlight.comment",       HL_COMMENT_COLOR,   PHP_INI_ALL,    NULL,          php_ini_color_displayer_cb)
    PHP_INI_ENTRY_EX("highlight.default",       HL_DEFAULT_COLOR,   PHP_INI_ALL,    NULL,          php_ini_color_displayer_cb)
    PHP_INI_ENTRY_EX("highlight.html",          HL_HTML_COLOR,      PHP_INI_ALL,    NULL,          php_ini_color_displayer_cb)
    PHP_INI_ENTRY_EX("highlight.keyword",       HL_KEYWORD_COLOR,   PHP_INI_ALL,    NULL,          php_ini_color_displayer_cb)
    PHP_INI_ENTRY_EX("highlight.string",        HL_STRING_COLOR,    PHP_INI_ALL,    NULL,          php_ini_color_displayer_cb)

    STD_PHP_INI_ENTRY_EX("display_errors",      "1",        PHP_INI_ALL,        OnUpdateDisplayErrors,  display_errors,         php_core_globals,   core_globals, display_errors_mode)
    STD_PHP_INI_BOOLEAN("display_startup_errors",   "0",    PHP_INI_ALL,        OnUpdateBool,           display_startup_errors, php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("enable_dl",            "1",        PHP_INI_SYSTEM,     OnUpdateBool,           enable_dl,              php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("expose_php",           "1",        PHP_INI_SYSTEM,     OnUpdateBool,           expose_php,             php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("docref_root",            "",         PHP_INI_ALL,        OnUpdateString,         docref_root,            php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("docref_ext",             "",         PHP_INI_ALL,        OnUpdateString,         docref_ext,             php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("html_errors",          "1",        PHP_INI_ALL,        OnUpdateBool,           html_errors,            php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("xmlrpc_errors",        "0",        PHP_INI_SYSTEM,     OnUpdateBool,           xmlrpc_errors,          php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("xmlrpc_error_number",    "0",        PHP_INI_ALL,        OnUpdateLong,           xmlrpc_error_number,    php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("max_input_time",         "-1",   PHP_INI_SYSTEM|PHP_INI_PERDIR,      OnUpdateLong,           max_input_time, php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("ignore_user_abort",    "0",        PHP_INI_ALL,        OnUpdateBool,           ignore_user_abort,      php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("implicit_flush",       "0",        PHP_INI_ALL,        OnUpdateBool,           implicit_flush,         php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("log_errors",           "0",        PHP_INI_ALL,        OnUpdateBool,           log_errors,             php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("log_errors_max_len",  "1024",        PHP_INI_ALL,        OnUpdateLong,           log_errors_max_len,     php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("ignore_repeated_errors",   "0",    PHP_INI_ALL,        OnUpdateBool,           ignore_repeated_errors, php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("ignore_repeated_source",   "0",    PHP_INI_ALL,        OnUpdateBool,           ignore_repeated_source, php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("report_memleaks",      "1",        PHP_INI_ALL,        OnUpdateBool,           report_memleaks,        php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("report_zend_debug",    "1",        PHP_INI_ALL,        OnUpdateBool,           report_zend_debug,      php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("output_buffering",       "0",        PHP_INI_PERDIR|PHP_INI_SYSTEM,  OnUpdateLong,   output_buffering,      php_core_globals,   core_globals)
    STD_PHP_INI_ENTRY("output_handler",         NULL,       PHP_INI_PERDIR|PHP_INI_SYSTEM,  OnUpdateString, output_handler,     php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("register_argc_argv",   "1",        PHP_INI_PERDIR|PHP_INI_SYSTEM,  OnUpdateBool,   register_argc_argv,    php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("auto_globals_jit",     "1",        PHP_INI_PERDIR|PHP_INI_SYSTEM,  OnUpdateBool,   auto_globals_jit,   php_core_globals,   core_globals)
    STD_PHP_INI_BOOLEAN("short_open_tag",   DEFAULT_SHORT_OPEN_TAG, PHP_INI_SYSTEM|PHP_INI_PERDIR,      OnUpdateBool,           short_tags,             zend_compiler_globals,  compiler_
    STD_PHP_INI_BOOLEAN("track_errors",         "0",        PHP_INI_ALL,        OnUpdateBool,           track_errors,           php_core_globals,   core_globals)
```

STD_PHP_INI_ENTRY로 output_buffering을 설정해줌

Analysis

```c
int php_request_startup(void)
{
    int retval = SUCCESS;

    zend_interned_strings_activate();

#ifdef HAVE_DTRACE
    DTRACE_REQUEST_STARTUP(SAFE_FILENAME(SG(request_info).path_translated),
#endif /* HAVE_DTRACE */

#ifdef PHP_WIN32
# if defined(ZTS)
    _configthreadlocale(_ENABLE_PER_THREAD_LOCALE);
# endif
    PG(com_initialized) = 0;
#endif

#if PHP_SIGCHILD
    signal(SIGCHLD, sigchld_handler);
#endif

    zend_try {
        PG(in_error_log) = 0;
        PG(during_request_startup) = 1;

        php_output_activate();

        /* initialize global variables */
        PG(modules_activated) = 0;
        PG(header_is_being_sent) = 0;
        PG(connection_status) = PHP_CONNECTION_NORMAL;
        PG(in_user_include) = 0;

        zend_activate();
        sapi_activate();

#ifdef ZEND_SIGNALS
        zend_signal_activate();
#endif
```

```c
#endif

        if (PG(max_input_time) == -1) {
            zend_set_timeout(EG(timeout_seconds), 1);
        } else {
            zend_set_timeout(PG(max_input_time), 1);
        }

        /* Disable realpath cache if an open_basedir is set */
        if (PG(open_basedir) && *PG(open_basedir)) {
            CWDG(realpath_cache_size_limit) = 0;
        }

        if (PG(expose_php)) {
            sapi_add_header(SAPI_PHP_VERSION_HEADER, sizeof(SAPI_PHP_VERSION_HEADER)-1, 1);
        }

        if (PG(output_handler) && PG(output_handler)[0]) {
            zval oh;

            ZVAL_STRING(&oh, PG(output_handler));
            php_output_start_user(&oh, 0, PHP_OUTPUT_HANDLER_STDFLAGS);
            zval_ptr_dtor(&oh);
        } else if (PG(output_buffering)) {
            php_output_start_user(NULL, PG(output_buffering) > 1 ? PG(output_buffering) : 0, PHP_OUTPUT_HANDLER_STDFLAGS);
        } else if (PG(implicit_flush)) {
            php_output_set_implicit_flush(1);
        }

        /* We turn this off in php_execute_script() */
        /* PG(during_request_startup) = 0; */

        php_hash_environment();
        zend_activate_modules();
        PG(modules_activated)=1;
    } zend_catch {
        retval = FAILURE;
    } zend_end_try();

    SG(sapi_started) = 1;

    return retval;
}
```

- main/main.c
- 최초 실행 시 php_output_start_user 실행

Analysis

```c
PHPAPI int php_output_start_user(zval *output_handler, size_t chunk_size, int flags)
{
    php_output_handler *handler;

    if (output_handler) {
        handler = php_output_handler_create_user(output_handler, chunk_size, flags);
    } else {
        handler = php_output_handler_create_internal(ZEND_STRL(php_output_default_handler_name), php_output_handler_default_func, chunk_size, flags);
    }
    if (SUCCESS == php_output_handler_start(handler)) {
        return SUCCESS;
    }
    php_output_handler_free(&handler);
    return FAILURE;
}
```

- main/output.c
- php_output_start_user(NULL, PG(output_buffering), PHP_OUTPUT_HANDLER_STDFLAGS)
- chunk_size는 output_buffering 사이즈
- php_output_handler_create_internal 함수를 호출

```c
PHPAPI php_output_handler *php_output_handler_create_internal(const char *name, size_t name_len, php_output_handler_context_func_t output_handler, size_t chunk_size, int flags)
{
    php_output_handler *handler;
    zend_string *str = zend_string_init(name, name_len, 0);

    handler = php_output_handler_init(str, chunk_size, (flags & ~0xf) | PHP_OUTPUT_HANDLER_INTERNAL);
    handler->func.internal = output_handler;
    zend_string_release_ex(str, 0);

    return handler;
}
```

- main/output.c
- php_output_handler_init에 zend_string_init한 결과값과 chunk_size, flag를 전달

Analysis

```c
static inline php_output_handler *php_output_handler_init(zend_string *name, size_t chunk_size, int flags)
{
    php_output_handler *handler;

    handler = ecalloc(1, sizeof(php_output_handler));
    handler->name = zend_string_copy(name);
    handler->size = chunk_size;
    handler->flags = flags;
    handler->buffer.size = PHP_OUTPUT_HANDLER_INITBUF_SIZE(chunk_size);
    handler->buffer.data = emalloc(handler->buffer.size);

    return handler;
}
```

```c
#define PHP_OUTPUT_HANDLER_INITBUF_SIZE(s) \
( ((s) > 1) ? \
    (s) + PHP_OUTPUT_HANDLER_ALIGNTO_SIZE - ((s) % (PHP_OUTPUT_HANDLER_ALIGNTO_SIZE)) : \
    PHP_OUTPUT_HANDLER_DEFAULT_SIZE \
)
#define PHP_OUTPUT_HANDLER_ALIGNTO_SIZE     0x1000
#define PHP_OUTPUT_HANDLER_DEFAULT_SIZE     0x4000
```

```
>>> 4096 + 0x1000 - (4096 % 0x1000)
8192
>>> hex(8192)
'0x2000'
>>>
```

- main/output.c 및 main/php_output.h
- buffer.size = PHP_OUTPUT_HANDLER_INITBUF_SIZE(buffer_size);
- PHP_OUTPUT_HANDLER_INITBUF_SIZE는 php_output.h에 매크로로 정의
- buffer.size = 8192가 된다 (default setting, output_buffer = 4096)
- buffer.data = emallc(buffer.size)

Analysis

```c
static inline int php_output_handler_append(php_output_handler *handler, const php_output_buffer *buf)
{
    if (buf->used) {
        OG(flags) |= PHP_OUTPUT_WRITTEN;
        /* store it away */
        if ((handler->buffer.size - handler->buffer.used) <= buf->used) {
            size_t grow_int = PHP_OUTPUT_HANDLER_INITBUF_SIZE(handler->size);
            size_t grow_buf = PHP_OUTPUT_HANDLER_INITBUF_SIZE(buf->used - (handler->buffer.size - handler->buffer.used));
            size_t grow_max = MAX(grow_int, grow_buf);

            handler->buffer.data = erealloc(handler->buffer.data, handler->buffer.size + grow_max);
            handler->buffer.size += grow_max;
        }
        memcpy(handler->buffer.data + handler->buffer.used, buf->data, buf->used);
        handler->buffer.used += buf->used;

        /* chunked buffering */
        if (handler->size && (handler->buffer.used >= handler->size)) {
            /* store away errors and/or any intermediate output */
            return OG(running) ? 1 : 0;
        }
    }
    return 1;
}
```

- main/output.c
- php_output_handler_append = output 내용을 추가해주는 함수
- buffer의 size가 부족하다 생각되는 경우, 추가 공간 할당
- buffer.data에 새로운 buf->data만큼 추가해줌
- 에러나 즉시 리턴해야 할 output이 있을 경우 일단 추가함

Analysis

```
SAPI_API int sapi_header_op(sapi_header_op_enum op, void *arg)
{
    sapi_header_struct sapi_header;
    char *colon_offset;
    char *header_line;
    size_t header_line_len;
    int http_response_code;

    if (SG(headers_sent) && !SG(request_info).no_headers) {
        const char *output_start_filename = php_output_get_start_filename();
        int output_start_lineno = php_output_get_start_lineno();

        if (output_start_filename) {
            sapi_module.sapi_error(E_WARNING, "Cannot modify header information - headers already sent by (output started at %s:%d)",
                output_start_filename, output_start_lineno);
        } else {
            sapi_module.sapi_error(E_WARNING, "Cannot modify header information - headers already sent");
        }
        return FAILURE;
    }
```

- Cannot modify header information - headers already sent by
- main/SAPI.c

Analysis

```
329    #define php_error zend_error
```

- main/php.h

```
ZEND_API ZEND_COLD void zend_error(int type, const char *format, ...) {
    const char *filename;
    uint32_t lineno;
    va_list args;

    get_filename_lineno(type, &filename, &lineno);
    va_start(args, format);
    zend_error_va_list(type, filename, lineno, format, args);
    va_end(args);
}
```

- Zend/zend.c
- zend_error_va_list 호출

Analysis

```c
static ZEND_COLD void zend_error_va_list(
        int type, const char *error_filename, uint32_t error_lineno,
        const char *format, va_list args)
{
    va_list usr_copy;
    zval params[5];
    zval retval;
    zval orig_user_error_handler;
    zend_bool in_compilation;
    zend_class_entry *saved_class_entry;
    zend_stack loop_var_stack;
    zend_stack delayed_oplines_stack;
    zend_array *symbol_table;
    zend_class_entry *orig_fake_scope;

    /* Report about uncaught exception in case of fatal errors */
    if (EG(exception)) {
        zend_execute_data *ex;
        const zend_op *opline;

        switch (type) {
            case E_CORE_ERROR:
            case E_ERROR:
            case E_RECOVERABLE_ERROR:
            case E_PARSE:
            case E_COMPILE_ERROR:
            case E_USER_ERROR:
                ex = EG(current_execute_data);
                opline = NULL;
                while (ex && (!ex->func || !ZEND_USER_CODE(ex->func->type))) {
                    ex = ex->prev_execute_data;
                }
                if (ex && ex->opline->opcode == ZEND_HANDLE_EXCEPTION &&
                    EG(opline_before_exception)) {
                    opline = EG(opline_before_exception);
                }
                zend_exception_error(EG(exception), E_WARNING);
                EG(exception) = NULL;
                if (opline) {
                    ex->opline = opline;
                }
                break;
            default:
                break;
        }
    }

#ifdef HAVE_DTRACE
```

```c
#ifdef HAVE_DTRACE
    if (DTRACE_ERROR_ENABLED()) {
        char *dtrace_error_buffer;
        zend_vspprintf(&dtrace_error_buffer, 0, format, args);
        DTRACE_ERROR(dtrace_error_buffer, (char *)error_filename, error_lineno);
        efree(dtrace_error_buffer);
    }
#endif /* HAVE_DTRACE */

    /* if we don't have a user defined error handler */
    if (Z_TYPE(EG(user_error_handler)) == IS_UNDEF
        || !(EG(user_error_handler_error_reporting) & type)
        || EG(error_handling) != EH_NORMAL) {
        zend_error_cb(type, error_filename, error_lineno, format, args);
    } else switch (type) {
        case E_ERROR:
        case E_PARSE:
        case E_CORE_ERROR:
        case E_CORE_WARNING:
        case E_COMPILE_ERROR:
        case E_COMPILE_WARNING:
            /* The error may not be safe to handle in user-space */
            zend_error_cb(type, error_filename, error_lineno, format, args);
            break;
        default:
            /* Handle the error in user space */
            va_copy(usr_copy, args);
            ZVAL_STR(&params[1], zend_vstrpprintf(0, format, usr_copy));
            va_end(usr_copy);

            ZVAL_LONG(&params[0], type);

            if (error_filename) {
                ZVAL_STRING(&params[2], error_filename);
            } else {
                ZVAL_NULL(&params[2]);
            }

            ZVAL_LONG(&params[3], error_lineno);

            symbol_table = zend_rebuild_symbol_table();

            /* during shutdown the symbol table table can be still null */
            if (!symbol_table) {
                ZVAL_NULL(&params[4]);
            } else {
```

- Zend/zend.c
- Uncaught exception 확인
- User defined error handler 확인

Analysis

```
        ZVAL_ARR(&params[4], zend_array_dup(symbol_table));
    }

    ZVAL_COPY_VALUE(&orig_user_error_handler, &EG(user_error_handler));
    ZVAL_UNDEF(&EG(user_error_handler));

    /* User error handler may include() additinal PHP files.
     * If an error was generated during comilation PHP will compile
     * such scripts recursively, but some CG() variables may be
     * inconsistent. */

    in_compilation = CG(in_compilation);
    if (in_compilation) {
        saved_class_entry = CG(active_class_entry);
        CG(active_class_entry) = NULL;
        SAVE_STACK(loop_var_stack);
        SAVE_STACK(delayed_oplines_stack);
        CG(in_compilation) = 0;
    }

    orig_fake_scope = EG(fake_scope);
    EG(fake_scope) = NULL;

    if (call_user_function(CG(function_table), NULL, &orig_user_error_handler, &retval, 5, params) == SUCCESS) {
        if (Z_TYPE(retval) != IS_UNDEF) {
            if (Z_TYPE(retval) == IS_FALSE) {
                zend_error_cb(type, error_filename, error_lineno, format, args);
            }
            zval_ptr_dtor(&retval);
        }
    } else if (!EG(exception)) {
        /* The user error handler failed, use built-in error handler */
        zend_error_cb(type, error_filename, error_lineno, format, args);
    }

    EG(fake_scope) = orig_fake_scope;

    if (in_compilation) {
        CG(active_class_entry) = saved_class_entry;
        RESTORE_STACK(loop_var_stack);
        RESTORE_STACK(delayed_oplines_stack);
        CG(in_compilation) = 1;
    }

    zval_ptr_dtor(&params[4]);
```

```
            zval_ptr_dtor(&params[2]);
            zval_ptr_dtor(&params[1]);

            if (Z_TYPE(EG(user_error_handler)) == IS_UNDEF) {
                ZVAL_COPY_VALUE(&EG(user_error_handler), &orig_user_error_handler);
            } else {
                zval_ptr_dtor(&orig_user_error_handler);
            }
            break;
    }

    if (type == E_PARSE) {
        /* eval() errors do not affect exit_status */
        if (!(EG(current_execute_data) &&
            EG(current_execute_data)->func &&
            ZEND_USER_CODE(EG(current_execute_data)->func->type) &&
            EG(current_execute_data)->opline->opcode == ZEND_INCLUDE_OR_EVAL &&
            EG(current_execute_data)->opline->extended_value == ZEND_EVAL)) {
            EG(exit_status) = 255;
        }
    }
}
```

- Zend/zend.c
- zend_error_cb 호출 (User error handler fail)

Analysis

```c
int zend_startup(zend_utility_functions *utility_functions) /* {{{ */
{
#ifdef ZTS
    zend_compiler_globals *compiler_globals;
    zend_executor_globals *executor_globals;
    extern ZEND_API ts_rsrc_id ini_scanner_globals_id;
    extern ZEND_API ts_rsrc_id language_scanner_globals_id;
#else
    extern zend_ini_scanner_globals ini_scanner_globals;
    extern zend_php_scanner_globals language_scanner_globals;
#endif

    zend_cpu_startup();

#ifdef ZEND_WIN32
    php_win32_cp_set_by_id(65001);
#endif

    start_memory_manager();

    virtual_cwd_startup(); /* Could use shutdown to free the main cwd but i

#if defined(__FreeBSD__) || defined(__DragonFly__)
    /* FreeBSD and DragonFly floating point precision fix */
    fpsetmask(0);
#endif

    zend_startup_strtod();
    zend_startup_extensions_mechanism();

    /* Set up utility functions and values */
    zend_error_cb = utility_functions->error_function;
```

- Zend/zend.c
- zend_error_cb = utility_functions->error_function

Analysis

```
zuf.error_function = php_error_cb;
zuf.printf_function = php_printf;
zuf.write_function = php_output_write;
zuf.fopen_function = php_fopen_wrapper_for_zend;
zuf.message_handler = php_message_handler_for_zend;
zuf.get_configuration_directive = php_get_configuration_directive_for_zend;
zuf.ticks_function = php_run_ticks;
zuf.on_timeout = php_on_timeout;
zuf.stream_open_function = php_stream_open_for_zend;
zuf.printf_to_smart_string_function = php_printf_to_smart_string;
zuf.printf_to_smart_str_function = php_printf_to_smart_str;
zuf.getenv_function = sapi_getenv;
zuf.resolve_path_function = php_resolve_path_for_zend;
zend_startup(&zuf);
setlocale(LC_CTYPE, "");
zend_update_current_locale();
```

- main/main.c | function php_module_startup
- zend_startup(&zuf)
- zuf.error_function = php_error_cb

```
        if (!module_initialized || PG(log_errors)) {
            char *log_buffer;
#ifdef PHP_WIN32
            if (type == E_CORE_ERROR || type == E_CORE_WARNING) {
                syslog(LOG_ALERT, "PHP %s: %s (%s)", error_type_str, buffer, GetCommandLine());
            }
#endif
            spprintf(&log_buffer, 0, "PHP %s:  %s in %s on line %" PRIu32, error_type_str, buffer, error_filename, error_lineno);
            php_log_err_with_severity(log_buffer, syslog_type_int);
            efree(log_buffer);
        }

        if (PG(display_errors) && ((module_initialized && !PG(during_request_startup)) || (PG(display_startup_errors)))) {
            if (PG(xmlrpc_errors)) {
                php_printf("<?xml version=\"1.0\"?><methodResponse><fault><value><struct><member><name>faultCode</name><value><int
            } else {
                char *prepend_string = INI_STR("error_prepend_string");
```

- main/main.c | function php_error_cb
- php_log_error_with_severity(log_buffer, syslog_type_int)

Analysis

```c
PHPAPI ZEND_COLD void php_log_err_with_severity(char *log_message, int syslog_type_int)
{
    int fd = -1;
    time_t error_time;

    if (PG(in_error_log)) {
        /* prevent recursive invocation */
        return;
    }
    PG(in_error_log) = 1;

    /* Try to use the specified logging location. */
    if (PG(error_log) != NULL) {
#ifdef HAVE_SYSLOG_H
        if (!strcmp(PG(error_log), "syslog")) {
            php_syslog(syslog_type_int, "%s", log_message);
            PG(in_error_log) = 0;
            return;
        }
#endif
        fd = VCWD_OPEN_MODE(PG(error_log), O_CREAT | O_APPEND | O_WRONLY, 0644);
        if (fd != -1) {
            char *tmp;
            size_t len;
            zend_string *error_time_str;

            time(&error_time);
#ifdef ZTS
            if (!php_during_module_startup()) {
                error_time_str = php_format_date("d-M-Y H:i:s e", 13, error_time, 1);
            } else {
                error_time_str = php_format_date("d-M-Y H:i:s e", 13, error_time, 0);
            }
#else
            error_time_str = php_format_date("d-M-Y H:i:s e", 13, error_time, 1);
#endif
            len = spprintf(&tmp, 0, "[%s] %s%s", ZSTR_VAL(error_time_str), log_message, PHP_EOL);
#ifdef PHP_WIN32
            php_flock(fd, 2);
            /* XXX should eventually write in a loop if len > UINT_MAX */
            php_ignore_value(write(fd, tmp, (unsigned)len));
#else
            php_ignore_value(write(fd, tmp, len));
#endif
```

```c
#endif
            efree(tmp);
            zend_string_free(error_time_str);
            close(fd);
            PG(in_error_log) = 0;
            return;
        }
    }

    /* Otherwise fall back to the default logging location, if we have one */

    if (sapi_module.log_message) {
        sapi_module.log_message(log_message, syslog_type_int);
    }
    PG(in_error_log) = 0;
}
/* }}} */
```

- main/main.c
- sapi_module.log_message 호출

Analysis

```c
SAPI_API void sapi_startup(sapi_module_struct *sf)
{
    sf->ini_entries = NULL;
    sapi_module = *sf;

#ifdef ZTS
    ts_allocate_fast_id(&sapi_globals_id, &sapi_globals_offset, sizeof(sapi_globals_struct), (ts_allocate_ctor) sapi_globals_ctor, (ts_allocate_dtor) sapi_globals_dtor);
# ifdef PHP_WIN32
    _configthreadlocale(_ENABLE_PER_THREAD_LOCALE);
# endif
#else
    sapi_globals_ctor(&sapi_globals);
#endif

#ifdef PHP_WIN32
    tsrm_win32_startup();
#endif

    reentrancy_startup();
}
```

- sapi/SAPI.c
- sapi_module = *sf 정의

Analysis

```c
static int
php_apache_server_startup(apr_pool_t *pconf, apr_pool_t *plog, apr_pool_t *ptemp, server_rec *s)
{
    void *data = NULL;
    const char *userdata_key = "apache2hook_post_config";

    /* Apache will load, unload and then reload a DSO module. This
     * prevents us from starting PHP until the second load. */
    apr_pool_userdata_get(&data, userdata_key, s->process->pool);
    if (data == NULL) {
        /* We must use set() here and *not* setn(), otherwise the
         * static string pointed to by userdata_key will be mapped
         * to a different location when the DSO is reloaded and the
         * pointers won't match, causing get() to return NULL when
         * we expected it to return non-NULL. */
        apr_pool_userdata_set((const void *)1, userdata_key, apr_pool_cleanup_null, s->process->pool);
        return OK;
    }

    /* Set up our overridden path. */
    if (apache2_php_ini_path_override) {
        apache2_sapi_module.php_ini_path_override = apache2_php_ini_path_override;
    }
#ifdef ZTS
    php_tsrm_startup();
# ifdef PHP_WIN32
    ZEND_TSRMLS_CACHE_UPDATE();
# endif
#endif

    zend_signal_startup();

    sapi_startup(&apache2_sapi_module);
    if (apache2_sapi_module.startup(&apache2_sapi_module) != SUCCESS) {
        return DONE;
    }
    apr_pool_cleanup_register(pconf, NULL, php_apache_server_shutdown, apr_pool_cleanup_null);
    php_apache_add_version(pconf);

    return OK;
}
```

- sapi/apache2handler/sapi_apache2.c
- sapi_startup(&apache2_sapi_module)
- 즉, apache2_sapi_module이 sapi_module이 됨

Analysis

```
static sapi_module_struct apache2_sapi_module = {
    "apache2handler",
    "Apache 2.0 Handler",

    php_apache2_startup,                /* startup */
    php_module_shutdown_wrapper,            /* shutdown */

    NULL,                       /* activate */
    NULL,                       /* deactivate */

    php_apache_sapi_ub_write,           /* unbuffered write */
    php_apache_sapi_flush,              /* flush */
    php_apache_sapi_get_stat,           /* get uid */
    php_apache_sapi_getenv,             /* getenv */

    php_error,                  /* error handler */

    php_apache_sapi_header_handler,         /* header handler */
    php_apache_sapi_send_headers,           /* send headers handler */
    NULL,                       /* send header handler */

    php_apache_sapi_read_post,          /* read POST data */
    php_apache_sapi_read_cookies,           /* read Cookies */

    php_apache_sapi_register_variables,
    php_apache_sapi_log_message,            /* Log message */
    php_apache_sapi_get_request_time,       /* Request Time */
    NULL,                       /* Child Terminate */

    STANDARD_SAPI_MODULE_PROPERTIES
};
```

- sapi/apache2handler/sapi_apache2.c
- apache2_sapi_module 정의
- apache2_sapi_module.log_message
  = php_apache_sapi_log_message

Analysis

```c
static void php_apache_sapi_log_message(char *msg, int syslog_type_int)
{
    php_struct *ctx;
    int aplog_type = APLOG_ERR;

    ctx = SG(server_context);

    switch (syslog_type_int) {
#if LOG_EMERG != LOG_CRIT
        case LOG_EMERG:
            aplog_type = APLOG_EMERG;
            break;
#endif
#if LOG_ALERT != LOG_CRIT
        case LOG_ALERT:
            aplog_type = APLOG_ALERT;
            break;
#endif
        case LOG_CRIT:
            aplog_type = APLOG_CRIT;
            break;
        case LOG_ERR:
            aplog_type = APLOG_ERR;
            break;
        case LOG_WARNING:
            aplog_type = APLOG_WARNING;
            break;
        case LOG_NOTICE:
            aplog_type = APLOG_NOTICE;
            break;
#if LOG_INFO != LOG_NOTICE
        case LOG_INFO:
            aplog_type = APLOG_INFO;
            break;
#endif
#if LOG_NOTICE != LOG_DEBUG
        case LOG_DEBUG:
            aplog_type = APLOG_DEBUG;
            break;
#endif
    }

    if (ctx == NULL) { /* we haven't initialized our ctx yet, oh well */
        ap_log_error(APLOG_MARK, APLOG_ERR | APLOG_STARTUP, 0, NULL, "%s", msg);
    } else {
        ap_log_rerror(APLOG_MARK, aplog_type, 0, ctx->r, "%s", msg);
    }
}
```

- sapi/apache2handler/sapi_apache2.c
- ap_log_error 혹은 ap_log_rerror 호출
- 위 두 함수는 apache2의 api 함수
- ap_log_error document
- ap_log_rerror document

Question?

https://github.com/sqrtrev/analysis-php_output_buffering