

PHP: Secure file transfer in PHP

application available at the link: <https://wiki.ostrowski.net.pl/copy-file/>

Welcome page

- Upon entering index.php, the user sees two buttons: „I'm Sender” and „I'm Receiver”.
- Selecting one of the options switches to the appropriate mode - sender or receiver.

Code generation ("Sender")

- When you click 'I'm Sender', the application draws a friendly code in the form of two members: adjective-animal (e.g. brave-fox, calm-owl).
- This code is not the name of the catalogue itself, but only the seed for further steps.

Directory assignment

- Based on the unique code, a random directory ID is created (e.g. a3f1b5c7d9e2f0a1...), stored in the mappings/<kod>.
- At the same time, a physical directory uploads/<ID>/, where the encrypted files will be stored.

Encryption key derivation

- The AES-256 key for encryption/decryption is derived on the fly from the user code: SHA-256(code) → bits → key (32 bytes).
- This ensures that the key is never stored on disk.

File upload

The sender selects a file and uploads it using a form.

Application:

- Reads the original file from a temporary location.
- Generates a random initialisation vector (IV) of 16 bytes.
- Encrypts the contents of the file with the AES-256-CBC algorithm (using the key and IV).
- Saves to uploads/<ID>/<nazwa_pliku>.enc content: [IV][ciphertext].

Code display

After a successful upload, the sender sees a message with his code (e.g. brave-fox) and can pass it on to the recipient.

Deletion of data after shutdown (cleanup)

In the sender's view, the JavaScript listens for the unload event (closing/refreshing the page).

If the user does not just click „Send” (i.e. really leaves the tab), a light signal is sent (`navigator.sendBeacon`) with the mode parameter=`cleanup&code=<kod>`.

On the server, the handling of this action deletes the `uploads/<ID>/` and the `mappings/` mapping file.`<kod>`.

Receiving a file (Receiver)

In receiver mode, a form appears into which the code received from the sender (e.g. brave-fox) must be entered.

Once approved:

- The application reads the corresponding directory ID from the `mappings/brave-fox` file.
- It browses the contents of `uploads/<ID>/` and lists all `.enc` files.
- Downloads the file.

The recipient clicks „Download” next to the selected encrypted file.

Server:

- Reads `uploads/<ID>/<plik>.enc`.
- Separates the IV (first 16 bytes) from the ciphertext.
- Decrypts the content with the same key derived from the code.
- Sends the decrypted file in response as an attachment.

Reuse and security

- The recipient does not store the code in a session - it must be typed in from scratch each time.
- Directories on the server have random names, not explicitly linked to the code, which prevents the key from being guessed.
- The encryption key never goes to disk - it is always generated in memory.

Thanks to this procedure:

- Security: the key is not written to files, the code does not reveal the directory name.
- Ease of use: simple, readable adjective-animal type code.
- Automatic clean-up: the sender leaves no traces when the session is closed.

Solution code

[index.php](#)

```
<?php
// index.php
// _____
// CONFIGURATION
define('UPLOAD_DIR', __DIR__ . '/uploads/');
define('MAP_DIR', __DIR__ . '/mappings/');
define('IV_LEN', 16); // AES block size
define('DIR_ID_LEN', 16); // bytes for random dir ID

$adjectives = ['lista', 'przymiotników'];
// longer the better but still bruteforcable if you have 100 adjectives
// and animals
//you get 10k possiblities which is not that much, but anyway it is
// better then
//retying 32 bit random codes or something ;- )
$animals = ['lista', 'zwierząt'];

// Ensure dirs exist
foreach ([UPLOAD_DIR, MAP_DIR] as $d) {
    if (!is_dir($d)) mkdir($d, 0700, true);
}

// _____
// HELPERS

function make_code(array $a, array $b): string {
    return $a[array_rand($a)] . '-' . $b[array_rand($b)];
}

function derive_key(string $code): string {
    return hex2bin(hash('sha256', $code)); // 32 bytes
}

function encrypt_file(string $src, string $dst, string $code): bool {
    $key = derive_key($code);
    $iv = random_bytes(IV_LEN);
    $pt = file_get_contents($src);
    $ct = openssl_encrypt($pt, 'AES-256-CBC', $key, OPENSSSL_RAW_DATA,
    $iv);
    return $ct === false
        ? false
        : file_put_contents($dst, $iv . $ct) !== false;
}

function decrypt_file(string $path, string $code) {
    $key = derive_key($code);
```

```
    $data = file_get_contents($path);
    $iv    = substr($data, 0, IV_LEN);
    $ct    = substr($data, IV_LEN);
    return openssl_decrypt($ct, 'AES-256-CBC', $key, OPENSSSL_RAW_DATA,
$iv);
}

function rmdir(string $dir) {
    if (!is_dir($dir)) return;
    foreach (scandir($dir) as $f) {
        if ($f==='.'||$f==='..') continue;
        $p = "$dir/$f";
        is_dir($p) ? rmdir($p) : unlink($p);
    }
    rmdir($dir);
}

function valid_code(string $c): bool {
    return preg_match('/^[a-z]+-[a-z]+$/', $c) === 1;
}

// Lookup directory ID by code
function get_dir_by_code(string $code): ?string {
    $map = MAP_DIR . $code;
    if (!file_exists($map)) return null;
    return trim(file_get_contents($map));
}

// Create mapping code→random dir ID
function create_mapping(string $code): string {
    // generate a random hex ID
    $dirId = bin2hex(random_bytes(DIR_ID_LEN));
    // create upload dir
    mkdir(UPLOAD_DIR . $dirId, 0700, true);
    // write mapping file (only dirId)
    file_put_contents(MAP_DIR . $code, $dirId);
    return $dirId;
}

// Remove mapping + uploads
function cleanup(string $code) {
    $dirId = get_dir_by_code($code);
    if ($dirId) {
        rmdir(UPLOAD_DIR . $dirId);
        unlink(MAP_DIR . $code);
    }
}

// _____
// ROUTING
$mode = $_REQUEST['mode'] ?? null;
```

```
$code = $_REQUEST['code'] ?? null;

// 1) CLEANUP on unload
if ($mode==='cleanup' && valid_code($code)) {
    cleanup($code);
    http_response_code(204);
    exit;
}

// 2) DOWNLOAD (receiver)
if ($mode==='download' && valid_code($code) && isset($_GET['file'])) {
    $dirId = get_dir_by_code($code);
    $file = basename($_GET['file']);
    $path = UPLOAD_DIR . "$dirId/$file";
    if (!$dirId || !file_exists($path)) {
        http_response_code(404);
        exit('Not found.');
```

```

<b>{$name}</b>.<br>Share this code:<br><b>{$code}</b>";
        } else {
            $error = "Encryption failed.";
        }
    }
}

// 4) RECEIVER ENTER CODE
if ($_SERVER['REQUEST_METHOD'] === 'POST'
    && $_POST['mode'] === 'receiver'
    && !empty($_POST['code'])) {
    $code = $_POST['code'];
    if (!valid_code($code)) {
        $error = "Invalid code.";
    }
    // else $code is used for this request only
}

// _____
// PAGE
?><!DOCTYPE html>
<html lang="en"><head><meta charset="UTF-8">
  <title>Secure File Exchange</title>
  <style>
    body { font-family:sans-serif; background:#f4f4f9; margin:0;
padding:20px; }
    .box { background:#fff; padding:20px; border-radius:8px;
        max-width:400px; margin:40px auto; box-shadow:0 2px 6px
rgba(0,0,0,0.1); }
    input,button,a { display:block; width:100%; margin:8px 0; }
    .error{color:#c00} .success{color:#080}
  </style>
</head><body>
  <div class="box">
    <h1>Secure File Exchange</h1>

    <?php if (!$mode): // Landing ?>
      <a href="?mode=sender"><button>I'm Sender</button></a>
      <a href="?mode=receiver"><button>I'm Receiver</button></a>

    <?php elseif ($mode === 'sender'): // Sender UI ?>
      <h2>Sender</h2>
      <?php if (!isset($code)): $code = make_code($adjectives,$animals);
endif; ?>
      <?php if (!empty($error)): ?><p class="error"><?= $error
?></p><?php endif; ?>
      <?php if (!empty($success)): ?><p class="success"><?= $success
?></p><?php endif; ?>

```

```

<p>Your code:<br><b><?=  

  htmlspecialchars($code) ?></b></p>
<form id="uploadForm" method="post" enctype="multipart/form-data">
  <input type="hidden" name="mode" value="sender">
  <input type="hidden" name="code" value="<?=  

htmlspecialchars($code) ?>">
  <input type="file" name="file" required>
  <button type="submit">Upload & Encrypt</button>
</form>
<a href="index.php">← Back</a>

<script>
  let submitting = false;
  document.getElementById('uploadForm')
    .addEventListener('submit', ()=>{ submitting = true; });
  window.addEventListener('unload', ()=>{
    if (!submitting) {
      navigator.sendBeacon('?mode=cleanup&code=<?=  

    $code ?>');
    }
  });
</script>

<?php elseif ($mode==='receiver'): // Receiver UI ?>
<h2>Receiver</h2>
<?php if (empty($code) || !empty($error)): ?>
  <form method="post">
    <input type="hidden" name="mode" value="receiver">
    <input type="text" name="code" placeholder="Enter code"
required>
    <button type="submit">Submit</button>
  </form>
  <?php if (!empty($error)): ?><p class="error"><?=  

$error  

?></p><?php endif; ?>

<?php else:
  $dirId = get_dir_by_code($code);
  $files = $dirId
    ? array_diff(scandir(UPLOAD_DIR . $dirId), ['.', '..'])
    : [];
  if (empty($files)): ?>
    <p>Waiting for sender...</p>
  <?php else: ?>
    <ul>
      <?php foreach ($files as $f): ?>
        <li>
          <?=  

htmlspecialchars(pathinfo($f, PATHINFO_FILENAME)) ?>  

          - <a href="?mode=download&code=<?=  

htmlspecialchars($code) ?>&file=<?=  

urlencode($f) ?>">Download</a>
        </li>
      <?php endforeach; ?>
    </ul>
  <?php endif; ?>

```

```
    <a href="index.php?mode=receiver">← New code</a>
  <?php endif; ?>

  <?php else: // Invalid mode ?>
    <p class="error">Invalid mode.</p>
    <a href="index.php">Back</a>
  <?php endif; ?>

</div>
</body></html>
```