CPE-03:不得實作未經使用者同意，使行動應用 App 可擅自修改使用者資料的行為，包括在使用者無確認情況下刪除或修改使用者連絡人資料、通話記錄、簡訊資料和多媒體簡訊資料的行為。

安全程式碼範例(iOS：Objective-C)

iOS 不允許 App 存取通話紀錄、簡訊資料與多媒體簡訊，但允許經使用者授權後，取得手機聯絡人相關資料，至於修改、刪除聯絡人資料，也建議須使用者確認。
針對[檢查授權、新增、修改、刪除聯絡人資料]提供相關程式碼如下。

檢查授權

```
// Check the authorization status of our application for Address Book
-(void)checkAddressBookAccess
{
    switch (ABAddressBookGetAuthorizationStatus())
    {
        // Update our UI if the user has granted access to their Contacts
        case  kABAuthorizationStatusAuthorized:
            [self accessGrantedForAddressBook];
            break;
            // Prompt the user for access to Contacts if there is no
definitive answer
        case  kABAuthorizationStatusNotDetermined :
            [self requestAddressBookAccess];
            break;
            // Display a message if the user has denied or restricted
access to Contacts
        case  kABAuthorizationStatusDenied:
        case  kABAuthorizationStatusRestricted:
        {
            UIAlertView *alert = [[UIAlertView alloc]
initWithTitle:@"Privacy Warning"

message:@"Permission was not granted for Contacts."
                                                              delegate:nil
                                                     cancelButtonTitle:@"OK"
                                                     otherButtonTitles:nil];
            [alert show];
        }
            break;
        default:
            break;
    }
}
```

新增聯絡人

```
   ABNewPersonViewController *picker = [[ABNewPersonViewController alloc]
init];
    picker.newPersonViewDelegate = self;
    UINavigationController *navigation = [[UINavigationController alloc]
initWithRootViewController:picker];
    [self presentViewController:navigation animated:YES completion:nil];
```

修改(顯示+修改) 聯絡人

```
    // Search for the person named "Appleseed" in the address book
    NSArray *people = (NSArray
*)CFBridgingRelease(ABAddressBookCopyPeopleWithName(self.addressBook,
CFSTR("Appleseed")));
```

```
    // Display "Appleseed" information if found in the address book
    if ((people != nil) && [people count])
    {
      ABRecordRef person = (  bridge ABRecordRef)[people objectAtIndex:0];
      ABPersonViewController *picker = [[ABPersonViewController alloc]
init];
      picker.personViewDelegate = self;
      picker.displayedPerson = person;
      // Allow users to edit the person’s information
      picker.allowsEditing = YES;
      [self.navigationController pushViewController:picker animated:YES];
    }
```

刪除聯絡人

```
        CFErrorRef *error = NULL;
        ABAddressBookRemoveRecord(self.addressBook, (ABRecordRef)person,
error);
        if(error != NULL)
        {
            //if any error happen
            UIAlertView    *alert =[[UIAlertView alloc]
initWithTitle:@"Error" message:@"Deleting Contact" delegate:self
cancelButtonTitle:@"Cancel" otherButtonTitles:@"OK",nil];
            [alert show];
        }
        ABAddressBookSave(self.addressBook, NULL);
```

CPF-08:需注意記憶體暫存的敏感性資料，如固定金鑰與密碼的安全性，當不需要時或於一定合理期間應強制清除或使用於一定期間就失效的可變量金鑰替代。

安全程式碼範例(iOS：Objective-C)

例如 extern unsigned char * CC_SHA256(const void *data, CC_LONG len, unsigned char *md); 中的參數 data 來源，除了標準的資料輸入以外，還可加入長度、固定的 Salt 與私密的計算。例如由 byte array 而來的 hash，或者計算後的 XOR。

```
unsigned char obfuscatedSecretKey[] = { 0x33, 0xAD, 0xBE, 0xEF, 0xDE, 0xAD,
0xBE, 0xEF, 0xDE, 0xAD, 0xBE, 0xEF, 0xDE, 0xAD, 0xBE, 0xEF };
int hash = [Server ComputeHash:obfuscatedSecretKey];
//One-at-a-Time Hash
+(int) ComputeHash: (unsigned char*) data;
{
   const int p = 16777619;
   int hash = (int)2166136261;
   int len = (int)strlen((char*)data);
   for (int i = 0; i < len; i++)
       hash = (hash ^ data[i]) * p;
   hash += hash << 13;
   hash ^= hash >> 7;
   hash += hash << 3;
   hash ^= hash >> 17;
   hash += hash << 5;
   return hash;
}
   // XOR the class name against the obfuscated key, to form the real key.
   unsigned char actualSecretKey[sizeof(obfuscatedSecretKey)];
   for (int i=0; i<sizeof(obfuscatedSecretKey); i++) {
       actualSecretKey[i] = obfuscatedSecretKey[i] ^ obfuscatedSecretKey[i];
   }
```

CPF-17:行動應用 App 有提供標準的設定參數檔函數，但由於駭客取得安裝檔之後，非常容易就可以修改這些標準的參數檔，於是為了加強安全層級，應用程式的設定參數，建議放在安全的地方，例如編譯至程式碼中，或者進行加密。

不安全程式碼範例(iOS：Objective-C)

Application Setting Read Example
將設定置於 plist 設定檔，容易被破解後修改

```
•       //取得 plist 檔案路徑
•       NSString *filePath = [[NSBundle mainBundle] pathForResource:@"secrets"
    ofType:@"plist"];
•       NSFileManager *fileManager = [NSFileManager defaultManager];
•       [self AppendLog:[NSString stringWithFormat:@"filePath=%@", filePath]];
•       //判斷 plist 檔案存在才讀取
•       if ([fileManager fileExistsAtPath: filePath]) {
•           NSMutableDictionary *data = [[NSMutableDictionary alloc]
    initWithContentsOfFile:filePath];
•           NSString *hash_password = [data objectForKey:@"hash_password"]; //從
    data 中取值
•           NSString *setting_lifes = [data objectForKey:@"setting_lifes"]; //從
    data 中取值
•       } else{
•           [self AppendLog:@"file not exists"];
    }
```

安全程式碼範例(iOS：Objective-C)

protect application parameter : example code

- 將設定或參數置於程式碼，或進行隱藏

```
//Stored as String in program, still not so safe to cracker
NSString *APIKey = @"abcdef123456";
// Stored as binary array to prevent strings attack
 // Stored using SHA(class name) XOR secret
 unsigned char obfuscatedSecretKey[] = { 0x3e, 0xcc, 0x9d, 0xca, 0x2f, 0x8a,
0xf2, 0xc1, 0x57, 0x01, 0xc2, 0x2e, 0x44, 0xea, 0x0d, 0xf5, 0x60, 0x09, 0x99,
0xe7, 0xb1, 0x13, 0x2a, 0x6f, 0x2c, 0xa2, 0xfe, 0x12, 0xbb, 0x58, 0x90, 0x85 };
```

CPL-01:行動應用 App 應設計並實作適當身分認證機制，並依使用者身分授權，以防止敏感資料被非授權人員存取。

安全程式碼範例(iOS：Swift)-參考 CPF-13 安全範例：使用 OAuth2。

```
通常會使用第三方帳號認證，茲提供 OAuth2 範例，以取得第三方的 Token，當作認證 ID
OAuth 2 應用範例

1.註冊並宣告服務
let googleConfig = GoogleConfig(
clientId: "YOUR GOOGLE CLIENT ID",                                  // [1] Define a
Google configuration
scopes:["https://www.googleapis.com/auth/drive"])                   // [2] Specify
scope

let gdModule = AccountManager.addGoogleAccount(googleConfig)        // [3] Add it to
AccountManager
self.http.authzModule = gdModule                                    // [4] Inject
the AuthzModule
                                                                    // into the HTTP
layer object

let multipartData = MultiPartData(data: self.snapshot(),            // [5] Define
multi-part
          name: "image",
          filename: "incognito photo",
          mimeType: "image/jpg")
let multipartArray =  ["file": multipartData]

self.http.POST("https://www.googleapis.com/upload/drive/v2/files",    // [6]
Upload image
               parameters: multipartArray,
       completionHandler: {(response, error) in
  if (error != nil) {
    self.presentAlert("Error", message: error!.localizedDescription)
  } else {
    self.presentAlert("Success", message: "Successfully uploaded!")
  }
})
```

```
2.註冊 App 應用連結
<key>CFBundleURLTypes</key>
<array>
    <dict>
        <key>CFBundleURLSchemes</key>
        <array>
            <string>com.raywenderlich.Incognito</string>
        </array>
    </dict>
</array>
```

```
3.接收 App 應用連結
func application(application: UIApplication,
  openURL url: NSURL,
  sourceApplication: String?,
  annotation: AnyObject?) -> Bool {
    let notification = NSNotification(name: AGAppLaunchedWithURLNotification,
      object:nil,
      userInfo:[UIApplicationLaunchOptionsURLKey:url])
    NSNotificationCenter.defaultCenter().postNotification(notification)
    return true
}
```

CPL-04:避免在行動應用 App 中直接使用設備 ID 作為使用者身分追蹤識別作為唯一因素，建議以帳號及密碼作為主要因素，設備 ID 可作為多因素認證的其他因素。

安全程式碼範例(iOS：Objective-C)

Device.UUID 範例碼

```
[NSString stringWithFormat:@"UUID=%@", [[[UIDevice currentDevice]
identifierForVendor] UUIDString]];
```

CPM-03:行動應用 App 連線使用交談識別碼，應實作具備逾時失效(Session time-out)機制。

安全程式碼範例(iOS：Objective-C)

- *iOS: local-session-timeout code 實作*
- *1.啟動時間計算*

```
    [myApp setLoginDate:[NSDate date]];
- (void) setLoginDate: (NSDate *)indate
{
    loginDate = indate;
}
```

- *2.檢查是否在安全時間內*

```
[NSString stringWithFormat:@"isValidDate=%d",[myApp
isLoginDateValid],
- (BOOL) isLoginDateValid
{
    if (!loginDate)
        return FALSE;
    //valid for 60 seconds
    NSDate *max date = [loginDate dateByAddingTimeInterval:60];
    return
        (([loginDate compare:[NSDate date]] == NSOrderedAscending)
&&
        ([max date compare:[NSDate date]] ==
NSOrderedDescending));
}
```

CPM-05:行動應用程式應使用憑證綁定(Certificate Pinning)方式驗證並確保連線之伺服器為行動應用程式開發人員所指定。

安全程式碼範例(iOS：Objective-C)

完整的 SSL Pinning 範例(以 https://www.owasp.org 與 https://gca.nat.gov.tw)
準備：取出 SSL 網站的公鑰

1.由網站獲取公鑰，以利 https ssl ceriticate 認證時比對

```
  ex +'/BEGIN CERTIFICATE/,/END CERTIFICATE/p' <(echo | openssl s client -
showcerts -connect www.owasp.org:443) -scq > file.crt
  ex +'/BEGIN CERTIFICATE/,/END CERTIFICATE/p' <(echo | openssl s client -
showcerts -connect gca.nat.gov.tw:443) -scq > gca.crt
```

2.然後轉成 der

```
  openssl x509 -outform der -in owasp.crt -out owasp2.der
  openssl x509 -outform der -in gca.crt -out gca2.der
```

3.附加為專案內容，以用於比對

範例一：元件 NSURLConnectionDelegate

1.實作 NSURLConnectionDelegate

2.建立 NSURLConnection

```
   urlconnection = [[NSURLConnection alloc] initWithRequest:theRequest
delegate:self];
```

3.監聽、驗證 SSL Pinning

```
- (BOOL)connection:(NSURLConnection *)connection
canAuthenticateAgainstProtectionSpace:(NSURLProtectionSpace *)protectionSpace {
return [protectionSpace.authenticationMethod
isEqualToString:NSURLAuthenticationMethodServerTrust];
}
//可在 didReceiveAuthenticationChallenge 進行憑證驗證
//或者實作 willSendRequestForAuthenticationChallenge 自行更早決定成功/失敗/驗證等
//於 willSendRequestForAuthenticationChallenge 可回傳以下幾種方式
// 1.useCredential:forAuthenticationChallenge:
// 2.continueWithoutCredentialForAuthenticationChallenge:
// 3.cancelAuthenticationChallenge:
// 4.performDefaultHandlingForAuthenticationChallenge:
// 5.rejectProtectionSpaceAndContinueWithChallenge:
- (void)connection:(NSURLConnection *)connection
didReceiveAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge {
   //Enable Pinning
   if ([[[challenge protectionSpace] authenticationMethod] isEqualToString:
NSURLAuthenticationMethodServerTrust]){
       do
       {
           SecTrustRef serverTrust = [[challenge protectionSpace] serverTrust];
           if(nil == serverTrust)
           {
               [self AppendLog:@"serverTrust is nil"];
               break; /* failed */
           }
           OSStatus status = SecTrustEvaluate(serverTrust, NULL);
           if(!(errSecSuccess == status))
           {
               [self AppendLog:@"SecTrustEvaluate status is errSecSuccess"];
               break; /* failed */
           }
           SecCertificateRef serverCertificate =
SecTrustGetCertificateAtIndex(serverTrust, 0);
```

```
          if(nil == serverCertificate)
            {
                [self AppendLog:@"serverCertificate is nil"];
                break; /* failed */
            }
            CFDataRef serverCertificateData =
SecCertificateCopyData(serverCertificate);
            if(nil == serverCertificateData)
            {
                [self AppendLog:@"serverCertificateData is nil"];
                break; /* failed */
            }
            const UInt8* const data = CFDataGetBytePtr(serverCertificateData);
            const CFIndex size = CFDataGetLength(serverCertificateData);
            NSData* cert1 = [NSData dataWithBytes:data length:(NSUInteger)size];
            if(nil == cert1)
            {
                [self AppendLog:@"server cert is nil"];
                break; /* failed */
            }
            NSString *file = [[NSBundle mainBundle] pathForResource:@"domain"
ofType:@"der"];
            NSData* cert2 = [NSData dataWithContentsOfFile:file];
            if(nil == cert1 || nil == cert2)
            {
                [self AppendLog:@"local pinning cert is nil"];
                break; /* failed */
            }
            const BOOL equal = [cert1 isEqualToData:cert2];
            if(!equal){
                [self AppendLog:@"certs not equal"];
                break; /* failed */
            }
            [self AppendLog:@"certs are GOOD"];
            // The only good exit point
            return [[challenge sender] useCredential: [NSURLCredential
credentialForTrust: serverTrust]
                                          forAuthenticationChallenge: challenge];

        } while(0);
    }
    // Bad dog
    return [[challenge sender] cancelAuthenticationChallenge: challenge];
}
```

範例二：元件 NSURLSession

1.建立 NSURLSession，並啟動之 by NSURLSessionDataTask

```
•    NSURLSessionConfiguration *sessionConfig = [NSURLSessionConfiguration
  defaultSessionConfiguration];
•    urlsession = [NSURLSession sessionWithConfiguration:sessionConfig
  delegate:self delegateQueue:nil];
•    [[urlsession dataTaskWithURL:url completionHandler:^(NSData *  Nullable
  data, NSURLResponse *  Nullable response, NSError *  Nullable error) {
•        // response management code
•        if (error)
•        {
•            [self AppendLog:[NSString stringWithFormat:@">> dataTask Error:%@",
  error.description]];
•        }
•        else
•        {
•            [self AppendLog:[NSString stringWithFormat:@">> dataTask got data
  %lu", data.length]];
•        }
•    }] resume];
```

2.驗證 SSL Pinning

```
• -(void)URLSession:(NSURLSession *)session
  didReceiveChallenge:(NSURLAuthenticationChallenge *)challenge
```

```
completionHandler:(void (^)(NSURLSessionAuthChallengeDisposition,
NSURLCredential *  Nullable))completionHandler {
    // Get remote certificate
    SecTrustRef serverTrust = challenge.protectionSpace.serverTrust;
    SecCertificateRef certificate = SecTrustGetCertificateAtIndex(serverTrust,
0);
    // Set SSL policies for domain name check
    NSMutableArray *policies = [NSMutableArray array];
    [policies addObject:(  bridge transfer id)SecPolicyCreateSSL(true,
(  bridge CFStringRef)challenge.protectionSpace.host)];
    SecTrustSetPolicies(serverTrust, (  bridge CFArrayRef)policies);
    // Evaluate server certificate
    SecTrustResultType result;
    SecTrustEvaluate(serverTrust, &result);
    BOOL certificateIsValid = (result == kSecTrustResultUnspecified || result
== kSecTrustResultProceed);
    // Get local and remote cert data
    NSData *remoteCertificateData =
CFBridgingRelease(SecCertificateCopyData(certificate));
    NSString *pathToCert = [[NSBundle mainBundle]pathForResource:@"domain"
ofType:@"der"];
    NSData *localCertificate = [NSData dataWithContentsOfFile:pathToCert];
    // The pinnning check
    if ([remoteCertificateData isEqualToData:localCertificate] &&
certificateIsValid) {
        NSURLCredential *credential = [NSURLCredential
credentialForTrust:serverTrust];
        [self AppendLog:@"URLSession pinning completionHandler
credential(GOOD)"];
        completionHandler(NSURLSessionAuthChallengeUseCredential, credential);
    } else {
        [self AppendLog:@"URLSession pinning completionHandler BAD(NULL)"];
        completionHandler(NSURLSessionAuthChallengeCancelAuthenticationChalleng
e, NULL);
    }
}
```

CPN-07 行動應用 App 應避免使用內嵌瀏覽器元件技術和防止連線劫持。

安全程式碼範例(iOS：Objective-C)

最好的方法是不使用內嵌瀏覽器元件。

- *謹慎使用 UIWebView : stringByEvaluatingJavaScriptFromString*

```
- (NSString *)stringByEvaluatingJavaScriptFromString:(NSString
*)script
```

安全程式碼範例(主機：Server)-參考 Server-06

增加 X-Frame-Options

```
X-Frame-Options: DENY

X-Frame-Options: SAMEORIGIN

X-Frame-Options: ALLOW-FROM https://example.com/
```

DENY
The page cannot be displayed in a frame, regardless of the site attempting to do so.

SAMEORIGIN
The page can only be displayed in a frame on the same origin as the page itself.

ALLOW-FROM *uri*
The page can only be displayed in a frame on the specified origin.

CPO-03:行動應用 App 程式碼應運用人工或工具使之增加複雜度，並輔以限制除錯器使用、反追蹤、二進位剝離等措施，使惡意人士使用逆向工程方法分析程式碼難度增加。

安全程式碼範例(iOS：Objective-C)

提供偵測裝置是否處於 jail-broken 模式原始碼

```
+(BOOL)isJailbroken{
    #if !(TARGET IPHONE SIMULATOR)
    if ([[NSFileManager defaultManager]
fileExistsAtPath:@"/Applications/Cydia.app"]){
        return YES;
    }else if([[NSFileManager defaultManager]
fileExistsAtPath:@"/Library/MobileSubstrate/MobileSubstrate.dylib"]){
        return YES;
    }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/bin/bash"]){
        return YES;
    }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/usr/sbin/sshd"]){
        return YES;
    }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/etc/apt"]){
        return YES;
    }
    if([[UIApplication sharedApplication] canOpenURL:[NSURL
URLWithString:@"cydia://package/com.example.package"]]){
        //Device is jailbroken
        return YES;
    }
    //try write /private
    NSError *error;
    NSString *stringToBeWritten = @"This is a test.";
    [stringToBeWritten writeToFile:@"/private/jailbreak.txt" atomically:YES
encoding:NSUTF8StringEncoding error:&error];
    if(error==nil){
        //Device is jailbroken
        return YES;
    } else {
        [[NSFileManager defaultManager] removeItemAtPath:@"/private/jailbreak.txt"
error:nil];
    }
    #endif
    //All checks have failed. Most probably, the device is not jailbroken
    return NO;
}
```

iOS:提供拒絕 DEBUG 模式原始碼(for release version)

```
//拒絕 DEBUG 模式的執行
#import <dlfcn.h>
#import <sys/types.h>
typedef int (*ptrace ptr t)(int  request, pid t  pid, caddr t  addr, int  data);
#if !defined(PT DENY ATTACH)
#define PT DENY ATTACH 31
#endif
void disable gdb() {
    void* handle = dlopen(0, RTLD GLOBAL | RTLD NOW);
    ptrace ptr t ptrace ptr = dlsym(handle, "ptrace");
    ptrace ptr(PT DENY ATTACH, 0, 0, 0);
    dlclose(handle);
}
int main(int argc, char * argv[]) {
    #if !(DEBUG) // Don't interfere with Xcode debugging sessions.
        disable gdb();
    #endif
    @autoreleasepool {
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([AppDelegate
class]));
    }
```

```
}
iOS 混亂器(obfuscator)說明
iOS 系統內定安裝後，已經實作了 Class rename；而字串的保護需自行實作；而降低可視度，通常也實現在混亂器工具中
例如：PreEmptive Protection for iOS 實現如下的內容
降低可視度
int max(int x, int y) {
    int switchvar = 1942843;
    while (true) {
        switch (switchvar) {
            case 598232:
                return y;
            case 1942843:
                if (x >= y) {
                    switchvar = 8289314;
                } else {
                    switchvar = 598232;
                }
              break;
            case 8289314:
                return x;
        }
    }
}
```

CPQ-01:行動應用 App 應實作驗證使用者輸入字串資料型別及長度之正確性，避免惡意輸入導致應用程式毀損、緩衝區溢位、各種注入攻擊發生。

安全程式碼範例(iOS: Objective-C)

驗證輸入長度、格式Email

如

```
- (BOOL) validEmail:(NSString*) emailString {
    if([emailString length]==0){
        return NO;
    }
    NSString *regExPattern = @"[A-Z0-9a-z. %+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,4}";
    NSRegularExpression *regEx = [[NSRegularExpression alloc]
initWithPattern:regExPattern options:NSRegularExpressionCaseInsensitive
error:nil];
    NSUInteger regExMatches = [regEx numberOfMatchesInString:emailString
options:0 range:NSMakeRange(0, [emailString length])];
    if (regExMatches == 0) {
        return NO;
    } else {
        return YES;
    }
}
```

安全程式碼範例(網頁主機：Server .NET C#)

- 驗證輸入長度、格式

  .NET 使用 RegularExpressionValidator ，使用正規表示式來驗證使用者輸入的格式。

  例如檢查英數字輸入 6 至 10 位。

  ```
  ValidationExpression="[a-zA-Z]{6,10}"
  ```

CPQ-02:行動應用 App 需要驗證使用者輸入、伺服器端傳入及其他裝置輸入之資料，防止因 Buffer overflow/underflow 造成安全性漏洞。

不安全程式碼範例(iOS：Objective-C)

- *strcat,strcpy,strncat,strncpy,sprintf,vsprintf,gets*

函式不會控制資料結尾，除了長度不安全以外，也會造成沒有結尾。

```
int len = (int)self.myInput.text.length;
    char buf[len+1];
    //strncpy
    strncpy(buf,  [self.myInput.text UTF8String] , len);
```

安全程式碼範例(iOS：Objective-C)

- *strlcat,strlcpy,strlcat,strlcpy,snprintf,asprintf,vsnprintf,vasprintf,fgets*

函式會控制資料結尾，除了長度安全以外，也會有結尾。

```
int len = (int)self.myInput.text.length;
    char buf[len+1];
    //strlpy
    strlcpy(buf, [self.myInput.text UTF8String] , len);
```

CPQ-03:行動應用 App 提供使用者輸入值儘量可以參數化(Query parameterization)。

安全程式碼範例

Language - Library	Parameterized Query
Java - Standard	• String custname = request.getParameter("customerName"); • String query = "SELECT account balance FROM user data WHERE user name = ? "; • **PreparedStatement pstmt = connection.prepareStatement(query);** • **pstmt.setString(1, custname);** • ResultSet results = pstmt.executeQuery( );

Java - Hibernate	• //HQL
	• Query safeHQLQuery = session.createQuery("from Inventory where productID=:productid");
	• safeHQLQuery.setParameter("productid", userSuppliedParameter);
	• //Criteria API
	• String userSuppliedParameter = request.getParameter("Product-Description"); // This should REALLY be validated too
	• // perform input validation to detect attacks
	• Inventory inv = (Inventory) session.createCriteria(Inventory.class).add
	• (Restrictions.eq("productDescription", userSuppliedParameter)).uniqueResult();

.NET/C#	• String query = "SELECT account balance FROM user data WHERE user name = ?";
	• try {
	• OleDbCommand command = new OleDbCommand(query, connection);
	• **command.Parameters.Add(new OleDbParameter("customerName", CustomerName Name.Text));**
	• OleDbDataReader reader = command.ExecuteReader();
	• // …
	• } catch (OleDbException se) {
	• // error handling
	• }

ASP.NET	• string sql = "SELECT * FROM Customers WHERE CustomerId = @CustomerId"; • **SqlCommand command = new SqlCommand(sql);** • **command.Parameters.Add(new SqlParameter("@CustomerId", System.Data.SqlDbType.Int));** • command.Parameters["@CustomerId"].Value = 1;
Ruby - ActiveRecord	• **# Create** • Project.create!(:name => 'owasp') • **# Read** • Project.all(:conditions => "name = ?", name) • Project.all(:conditions => { :name => name }) • Project.where("name = :name", :name => name) • **# Update** • project.update attributes(:name => 'owasp') • **# Delete**

Language - Library	Parameterized Query
	• Project.delete(:name => 'name')
Ruby	• insert new user = db.prepare "INSERT INTO users (name, age, gender) VALUES (?, ? ,?)" • insert_new_user.execute 'aizatto', '20', 'male'
PHP - PDO	• $stmt = $dbh->prepare("INSERT INTO REGISTRY (name, value) VALUES (:name, :value)"); • **$stmt->bindParam(':name', $name);** • **$stmt->bindParam(':value', $value);**
Cold Fusion	• <cfquery name = "getFirst" dataSource = "cfsnippets"> • **SELECT * FROM #strDatabasePrefix# courses WHERE intCourseID =** • **<cfqueryparam value = #intCourseID# CFSQLType = "CF SQL INTEGER">** • </cfquery>
Perl - DBI	• my $sql = "INSERT INTO foo (bar, baz) VALUES (?, ?)"; • **my $sth = $dbh->prepare($sql);** • **$sth->execute($bar, $baz);**

CPQ-05:行動應用 App 應實作過濾使用者輸入及伺服器端傳入資料中易導致 SQL injection 之字串。

不安全程式碼範例(iOS：Objective-C)

sqlite 錯誤方式

```
- (IBAction) insertUnsafe : (id) sender
{
    BOOL rst = [self executeSQL:[NSString stringWithFormat:@"INSERT INTO
Setting(SKey,SValue) VALUES('%@','%@')", self.edit key.text,
self.edit value.text]];
}
```

安全程式碼範例(iOS：Objective-C)

資料庫：sqlite 正確方式，即 CPQ-03：Query parameterization

```
- (IBAction) insertSafe : (id) sender
{
    sqlite3 stmt *statement;
    BOOL DONE = TRUE;
    char *sql = "INSERT INTO Setting(SKey,SValue) VALUES(?,?)";

    if (sqlite3 prepare v2(database,sql,-1,&statement,NULL) == SQLITE OK) {
        sqlite3 bind text(statement,1,[self.edit key.text UTF8String],-1,
SQLITE TRANSIENT);
        sqlite3 bind text(statement,2,[self.edit value.text UTF8String],-1,
SQLITE TRANSIENT);
        if (sqlite3 step(statement) != SQLITE DONE) {
            DONE = FALSE;
            NSLog(@"Error");
        }
    }
    else
        DONE = FALSE;
    sqlite3 finalize(statement);
}
```

CPQ-08:行動應用 App 應實作過濾使用者輸入及伺服器端傳入資料中易導致 XML Injection 之字串。

不安全程式碼範例(網頁主機：Server .NET C#)

使用字串加減維護 XML

```
xmlstr = "<username>" + "foo<" + "</username>";
```

安全程式碼範例(網頁主機：Server .NET C#)

使用元件維護 XML

```
XmlNode node = doc.CreateNode("username");
node.Value = "foo<";
```

iOS-01:謹慎使用 Keychain 儲存密碼(Use the Keychain carefully)。

不安全程式碼範例

使用其他屬性的參數，造成可能的漏洞，例如備份檔被其他機器還原。

安全程式碼範例(Objective-C)

使用嚴格參數，以免資料洩漏

```
• // Protect the keychain entry so it's only valid when the device is
  unlocked.
• [dictionary setObject:( bridge
  id)kSecAttrAccessibleWhenUnlockedThisDeviceOnly forKey:( bridge
  id)kSecAttrAccessible];
```

iOS-02：為保護敏感資料不被以截圖方式儲存於檔案系統，行動應用 App 使用 API 設定或編寫程式阻擋敏感資料區快照功能(Snapshots)或以覆蓋方式清除。

安全程式碼範例(Objective-C)

- 提供蓋掉螢幕的範例碼、提供隱藏欄位的建議方法
- 先製作滿版圖片 secure-image.png (320x568 for iPhone)

```
- (void)applicationDidEnterBackground:(UIApplication *)application {
    // Use this method to release shared resources, save user data, invalidate timers, and store enough application state information to restore your application to its current state in case it is terminated later.
    // If your application supports background execution, this method is called instead of applicationWillTerminate: when the user quits.
        if (!self.backgroundImage){
            UIImageView *myBanner = [[UIImageView alloc] initWithImage:[UIImage imageNamed:@"secure-image.png"]];
            self.backgroundImage = myBanner;
        }
        [self.window addSubview:self.backgroundImage];
    // you can hide security field here
    // 隱藏欄位
    // [viewControl.secure field setHidden:TRUE];
}
- (void)applicationWillEnterForeground:(UIApplication *)application {
    // Called as part of the transition from the background to the inactive state; here you can undo many of the changes made on entering the background.
    if (self.backgroundImage)
        [self.backgroundImage removeFromSuperview];
    // you should visi security field here
    // 回復已隱藏欄位
    // [viewControl.secure field setHidden:FALSE];
}
```

iOS-03:啟用自動引用計數(Automatic Reference Counting ,ARC)，避免記憶體物件弱點產生。

不安全程式碼範例 (Objective-C)

不啟用 ARC 時，又沒有完成釋放記憶體。

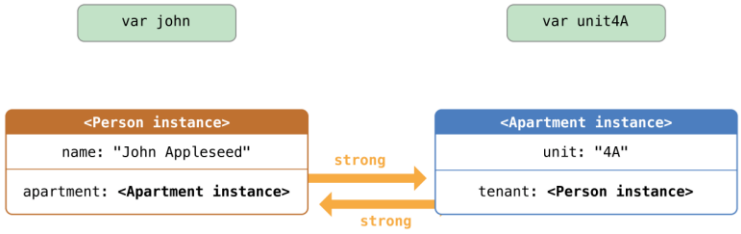
```
@interface myInputStream : NSObject {
    NSInputStream *stream;
}
+ (void)initialize
{
    stream = NSInputStream inputStreamWithFileAtPath:somepath]];
}
- (void)dealloc
{
          //需要手動release
    //[stream release];
    [super dealloc];
}
```

不安全程式碼範例 (Swift)

Swift 循環強型鏈結問題

```
物件宣告
class Person {
    let name: String
    init(name: String) { self.name = name }
    var apartment: Apartment?
     deinit { print("\(name) is being deinitialized") }
}
class Apartment {
    let unit: String
    init(unit: String) { self.unit = unit }
    var tenant: Person?
    deinit { print("Apartment \(unit) is being deinitialized") }
}
參數宣告
var john: Person?
var unit4A: Apartment?
建立實體
john = Person(name: "John Appleseed")
unit4A = Apartment(unit: "4A")
關係指派
john!.apartment = unit4A
unit4A!.tenant = john
釋放實體
john = nil
unit4A = nil
```

強型鏈結而導致 ARC 無法正確運作，記憶體無法被釋放。



安全程式碼範例(Swift)

循環強型鏈結問題，改用弱型鏈結

```
class Apartment {
    let unit: String
    init(unit: String) { self.unit = unit }
    weak var tenant: Person?
    deinit { print("Apartment \(unit) is being deinitialized") }
}
```

iOS-04:啟用 App Transport Security(ATS)設定。

不安全程式碼範例

範例一：全面取消 ATS

```
NSAppTransportSecurity

    NSAllowsArbitraryLoads = YES
```

範例二：某網域降級 TLS

```
NSAppTransportSecurity

    NSExceptionDomains

        "less-secure.example.com"

            NSExceptionRequiresForwardSecrecy = NO

            NSExceptionMinimumTLSVersion = "TLSv1.0"
```

範例三：某網域取消 ATS

```
NSAppTransportSecurity

    NSAllowsArbitraryLoads = NO  // Shown for clarity; this is the
default

    NSExceptionDomains

        "secure-server-i-control.example.com"

        NSExceptionAllowsInsecureHTTPLoads = YES

        NSExceptionRequiresForwardSecrecy = NO

        NSExceptionMinimumTLSVersion = "TLSv1.0"
```

Server-01 與行動應用 App 連接之所有後端服務伺服器(包含網頁、資料庫及中介等)作業系統應有效強化及進行安全設定配置，並持續進行安全性程式修補。

不安全程式碼範例(網站主機：Server)

例如過時 OpenSSL 版本 OpenSSL/1.0.1e 與太多資訊洩漏。

```
telnet ---.---.---.tw 80
Trying 0.0.0.0……..….
Connected to ---.---.---.tw.
Escape character is '^]'.
HEAD / HTTP/1.0
```

```
HTTP/1.1 200 OK
Date: Sat, 06 Aug 2016 01:57:04 GMT
Server: Apache/2.2.29 (Unix) mod_ssl/2.2.29 OpenSSL/1.0.1e-fips DAV/2 PHP/5.3.29
Last-Modified: Thu, 30 Jun 2016 06:22:49 GMT
ETag: "bc16f1-7f8-53678e6296040"
Accept-Ranges: bytes
Content-Length: 2040
Connection: close
Content-Type: text/html


遺失與主機的連線。
```

安全程式碼範例(網站主機：Server)

較少資訊洩漏

```
telnet www.------.com.tw 80
HTTP/1.0 200 OK
Date: Sat, 30 Jul 2016 05:52:56 GMT
P3P: policyref="http://info.------.com/w3c/p3p.xml", ... Cache-Control: max-age=3600,
public
Vary: Accept-Encoding
Content-Type: text/html; charset=UTF-8
Server: ATS
```

安全程式碼範例(網站主機：Server .NET)

啟用 HSTS 機制

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <rewrite>
            <rules>
                <rule name="HTTP to HTTPS redirect" stopProcessing="true">
                    <match url="(.*)" />
                    <conditions>
                        <add input="{HTTPS}" pattern="off" ignoreCase="true" />
                    </conditions>
                    <action type="Redirect" url="https://{HTTP HOST}/{R:1}"
                        redirectType="Permanent" />
                </rule>
            </rules>
            <outboundRules>
                <rule name="Add Strict-Transport-Security when HTTPS" enabled="true">
                    <match serverVariable="RESPONSE Strict Transport Security"
                        pattern=".*" />
                    <conditions>
                        <add input="{HTTPS}" pattern="on" ignoreCase="true" />
                    </conditions>
                    <action type="Rewrite" value="max-age=31536000" />
                </rule>
            </outboundRules>
        </rewrite>
    </system.webServer>
</configuration>
```

Server-06:網頁伺服器需預防網頁掛馬及點擊劫持攻擊。

安全程式碼範例(網頁主機：Server)

增加 X-Frame-Options

```
X-Frame-Options: DENY

X-Frame-Options: SAMEORIGIN

X-Frame-Options: ALLOW-FROM https://example.com/
```

DENY
The page cannot be displayed in a frame, regardless of the site attempting to do so.

SAMEORIGIN
The page can only be displayed in a frame on the same origin as the page itself.

ALLOW-FROM *uri*
The page can only be displayed in a frame on the specified origin.

Server-10:參考 CPG-01~CPG-03 實作 TLS 伺服器端設定。

不安全程式碼範例(主機：Server)

OpenSSL 1.01f(含以前) 版本
2048 位元以下、逾期、失效、根憑證失效、掛失等憑證。

安全程式碼範例(主機：Server)

OpenSSL 1.01g(含之後) 版本
2048 位元(含)以上，合法有效憑證。