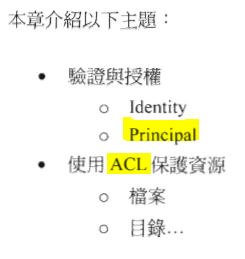
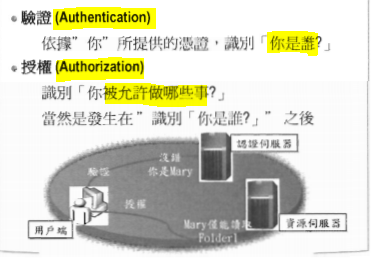
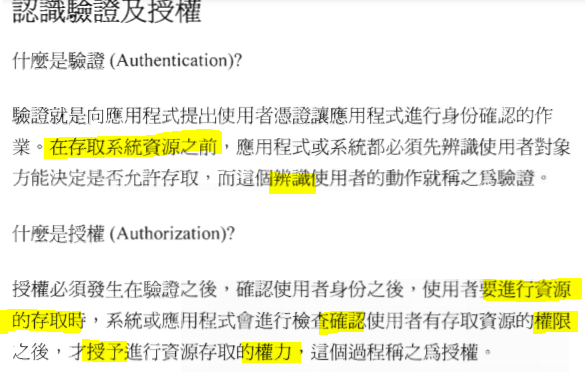
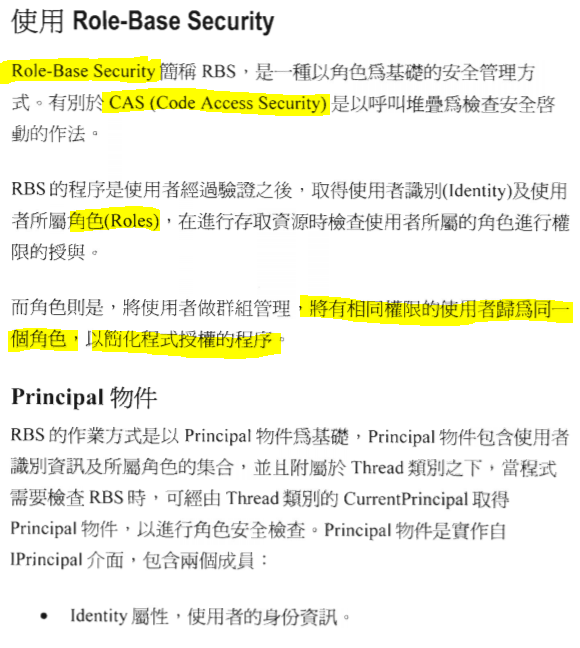
P 505



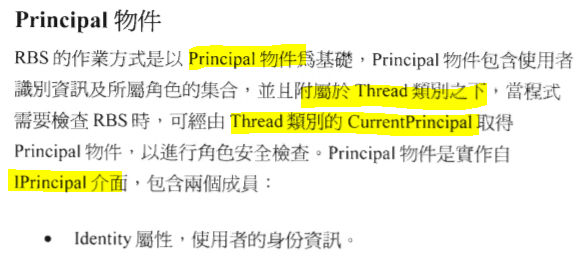
什麼是 Authentication﹑Authorization﹑RBS 和 Roles

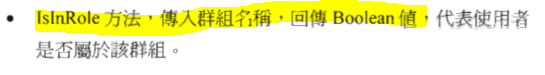


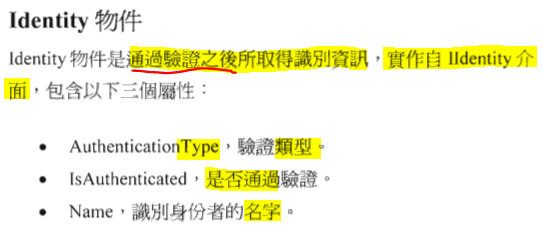




P511 Principal & Identity



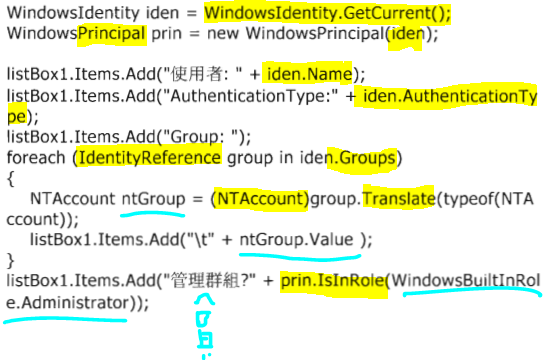




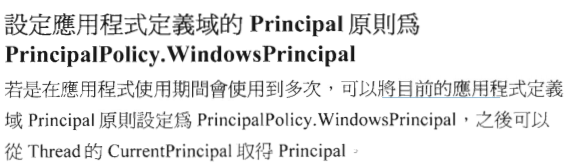
P 513 WindowsIdentity 與 WindowsPrinicpal

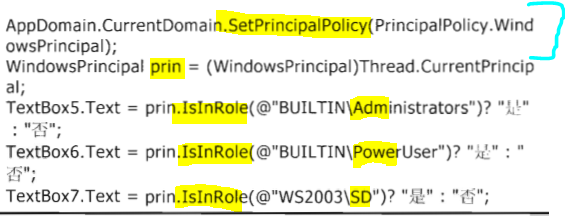


單次 principal 用法︰

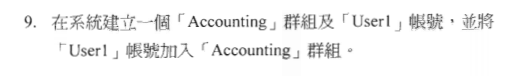


多次 principal 用法︰



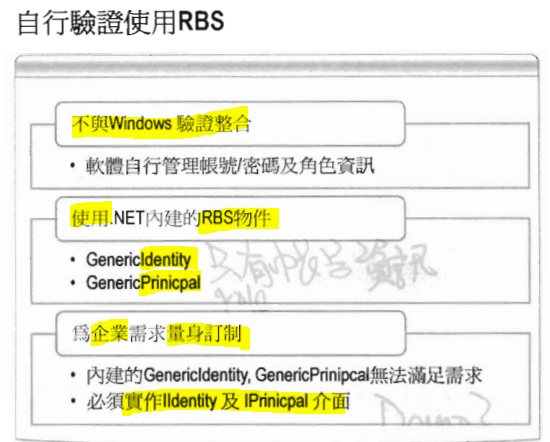


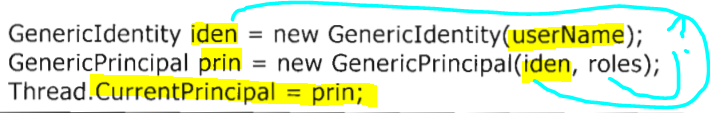
P 516 實作︰ note3\_ex3.RBS



我發現要配合 PDF2 之中的會員機制才可以完成，而且之前提到的都沒有運用到。所以我決定加入前兩頁提到的實作…

P 519 自訂 RBS



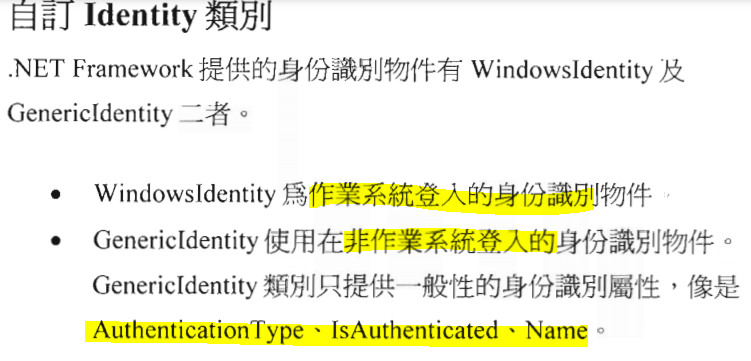


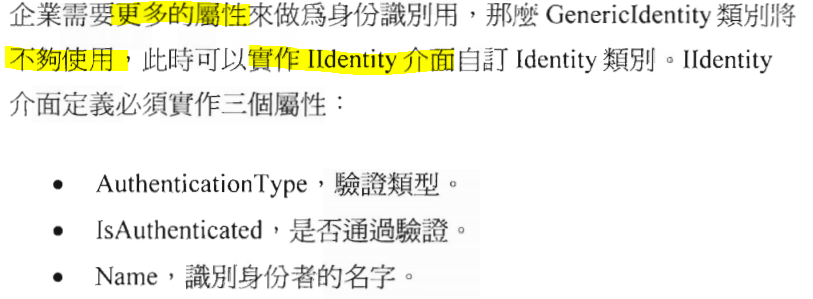
Generic 就可以容納所有 ( Windows 或 自訂的 ) Identity 和 Pricipal

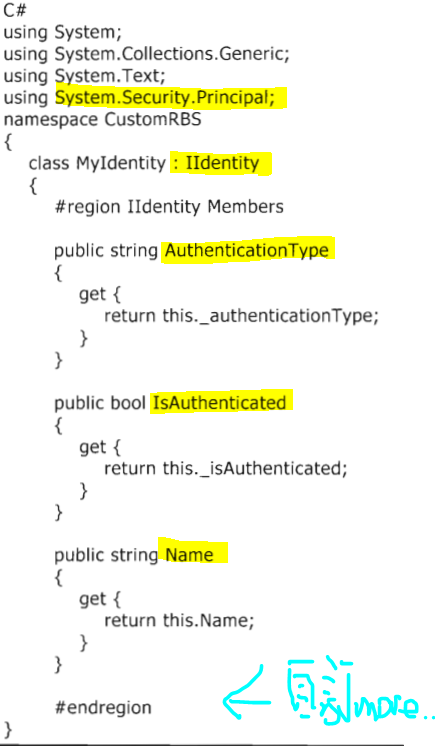
如果沒有特別的物件設定，只是想要「創造」一個新的「原有物件」，用這個就夠了。

如果要增加特別的物件設定，就要從介面開始繼承︰下頁

P 520 自訂 Identity 類別

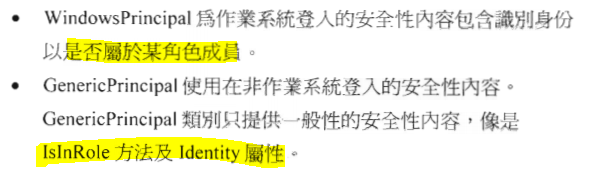




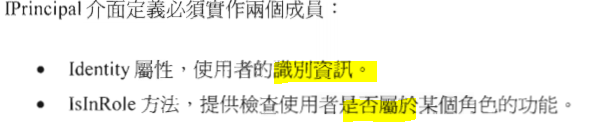


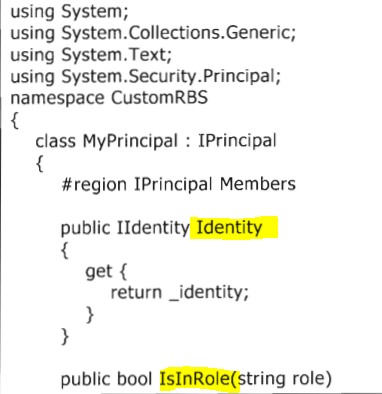
P 521 自訂 Principle 類別

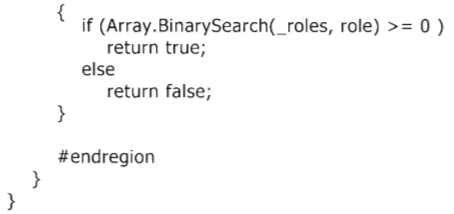
原有︰



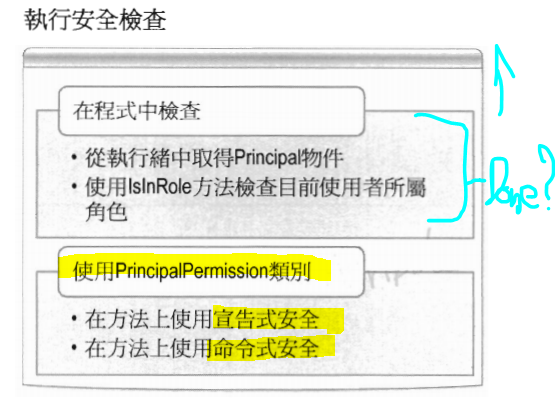
想要更多︰ 實作 IPrincipal

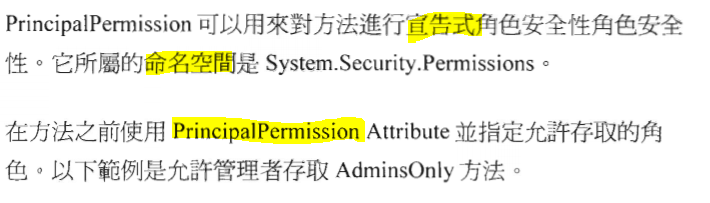


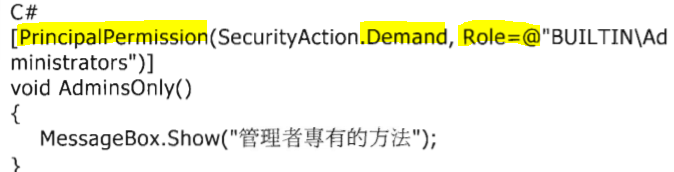


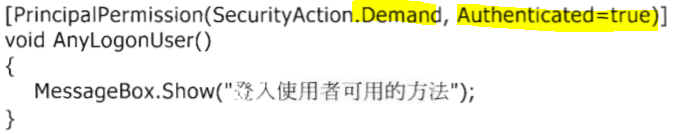


P 524 執行安全檢查PrincipalPermission

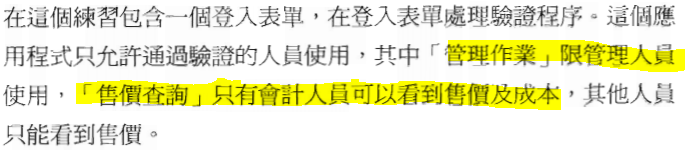




只有管理者可以使用這個方法

所有人都可以用這個方法

P 526 實作 note3\_ex3.Generic\_ID\_Principal\_RBS



protected void Page\_Load(object sender, EventArgs e)

{

GenericIdentity iden = new GenericIdentity("Kate");

//string[] roles = new string[] { "Administrators" }; // <-- 切換身份，以對比效果

string[] roles = new string[] { "NOT\_Administrators" };

Thread.CurrentPrincipal = new GenericPrincipal(iden, roles);

Label1.Text = "登入者 : " + Thread.CurrentPrincipal.Identity.Name;

Label2.Text = "身份驗証成功？ : " + Thread.CurrentPrincipal.Identity.IsAuthenticated;

}

protected void Button1\_Click(object sender, EventArgs e)

{

try

{

AdminsOnly();

Label3.Text = "成功使用";

}

catch

{

Label3.Text = "無法使用";

}

}

[PrincipalPermission(SecurityAction.Demand, Role = "Administrators")]

void AdminsOnly(){ }

protected void Button2\_Click(object sender, EventArgs e)

{

if (Thread.CurrentPrincipal.IsInRole("Administrators"))

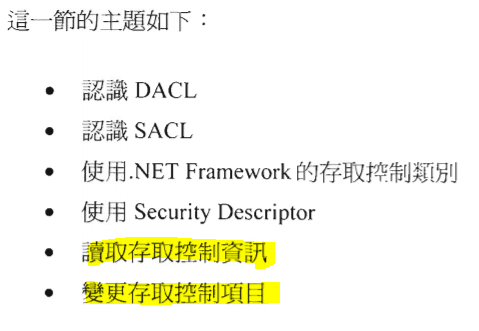
Label3.Text = "售價﹑成本";

else

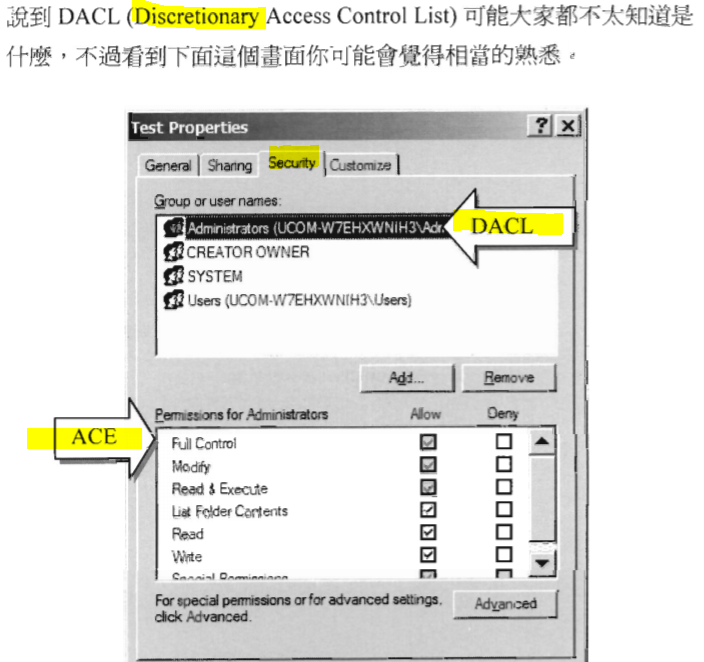
Label3.Text = "售價";

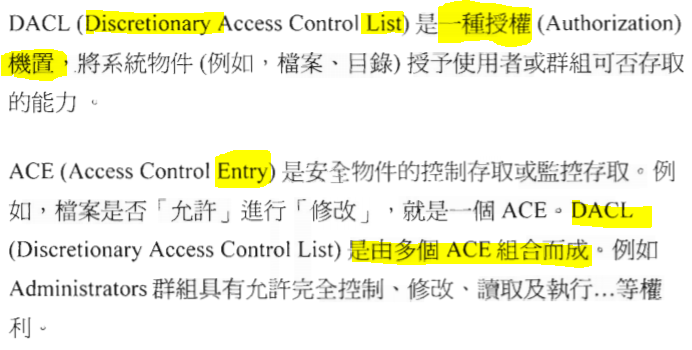
}

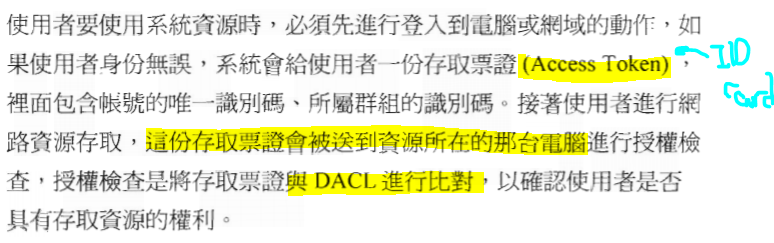
P 531 ACL



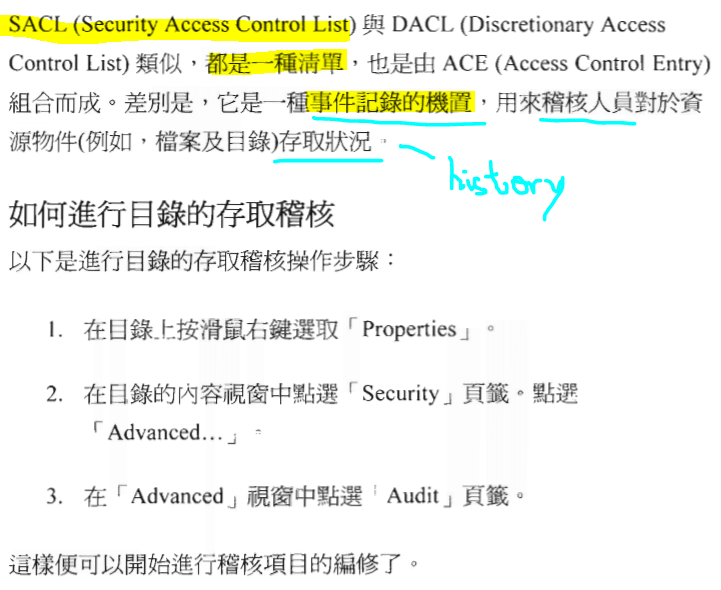
DACL

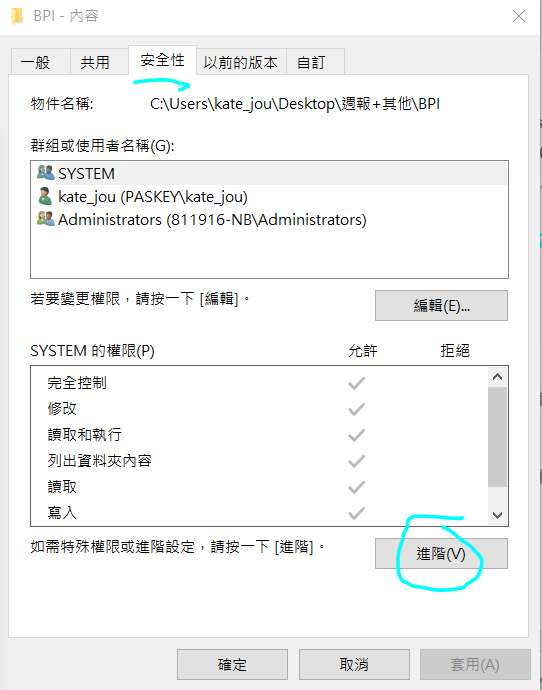




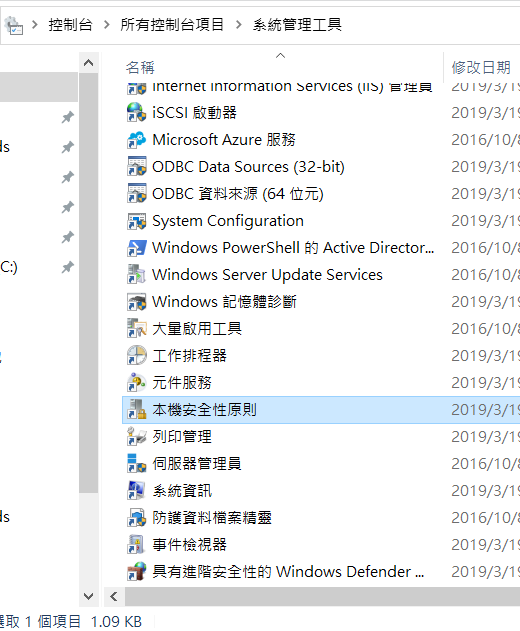
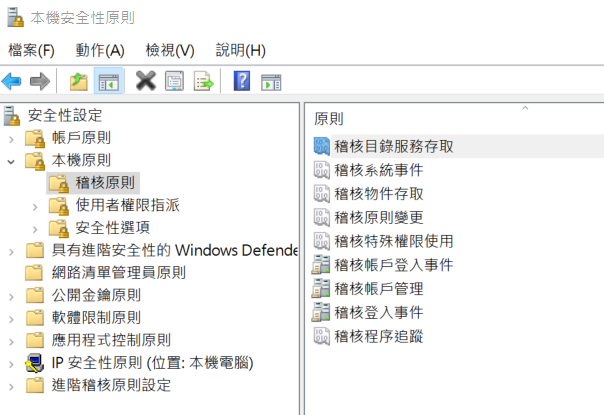
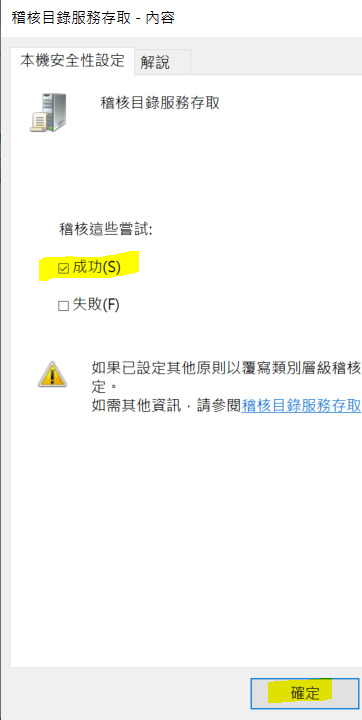


SACL



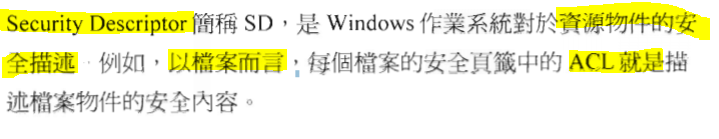
 

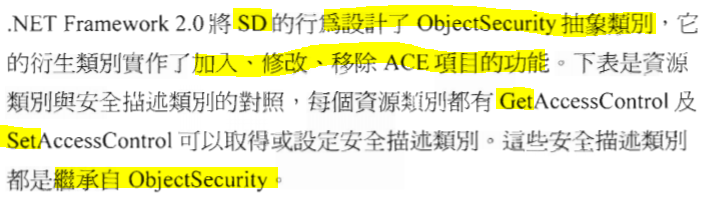


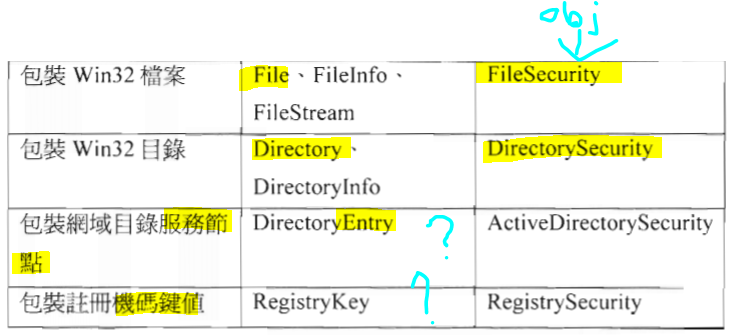


呃，我不啟用了，除非這個是非常機密的電腦，誰開過什麼資料夾都要時時刻刻紀錄，否則應該用不著？

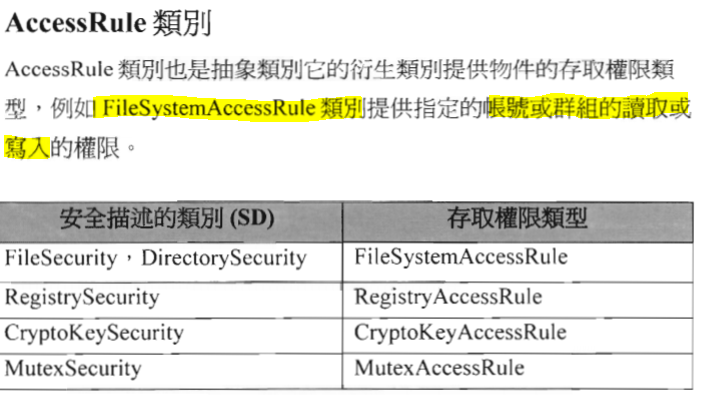
P 537 存取控制類別 ObjectSecurity﹑AccessRule





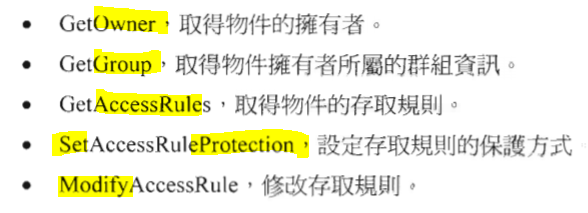


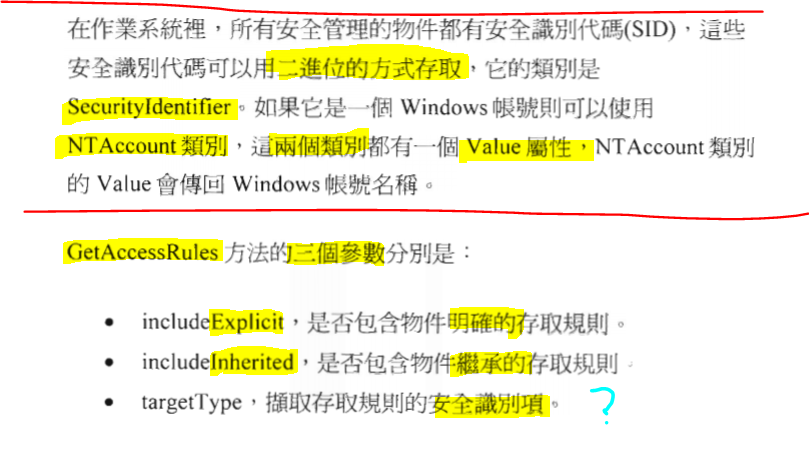
下面還有兩個我看不懂就算了…

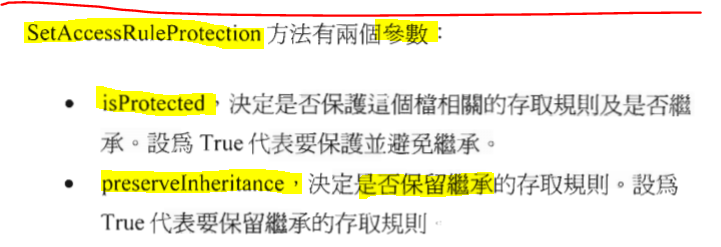


P 539 存取SD 資訊

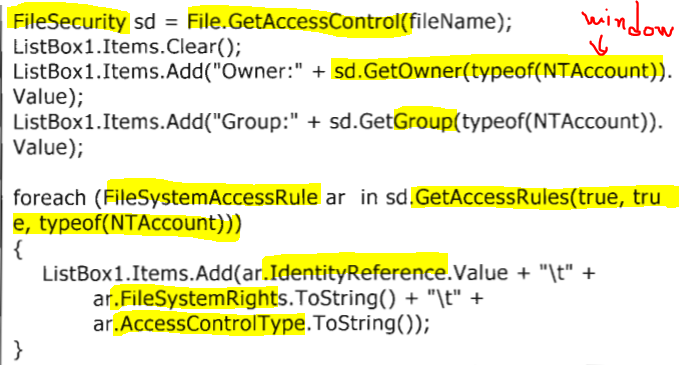
ObjectSecurity :

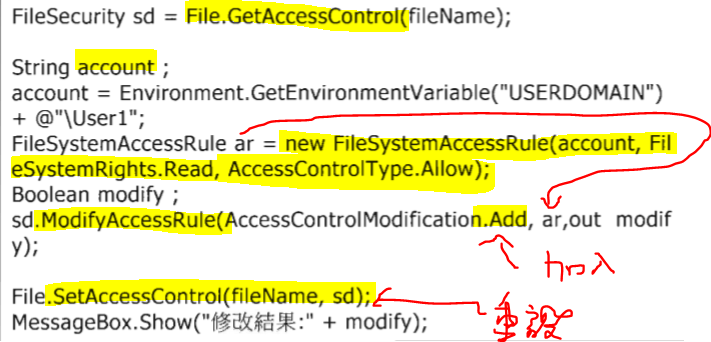




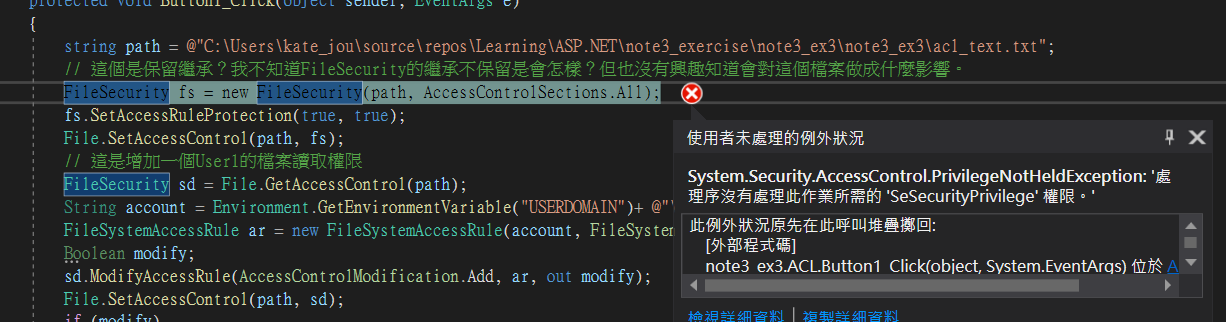


使用例子︰

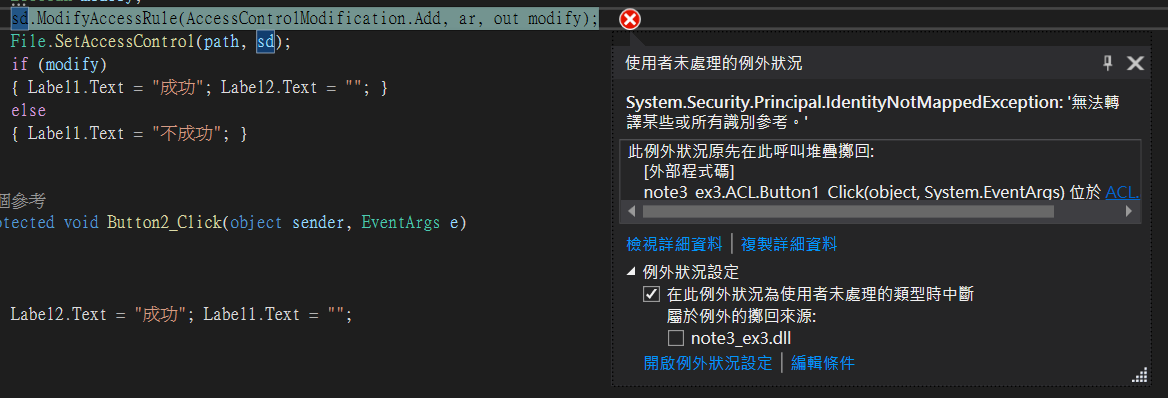




P 543 實作︰ note3\_ex3.ACL



試用 管理員權限去跑…



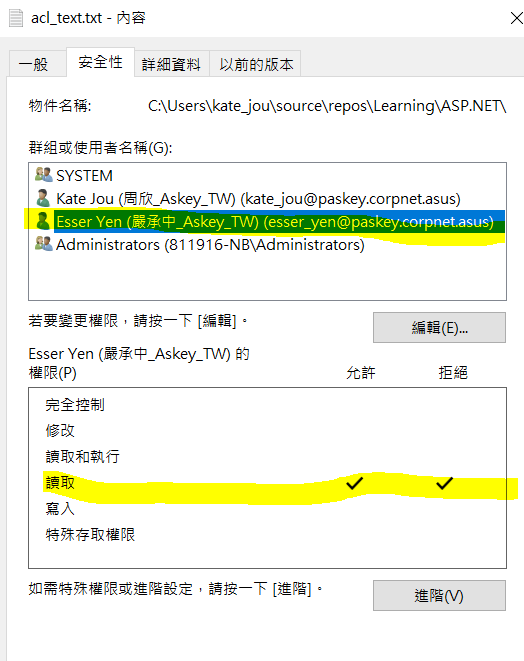
string account = Environment.GetEnvironmentVariable("USERDOMAIN")+ @"\esser\_yen";// <-- 不能用不存用這個Domain之中的人的名字

原本我隨打個User1 但是 accout 會 = PASKEY\User1

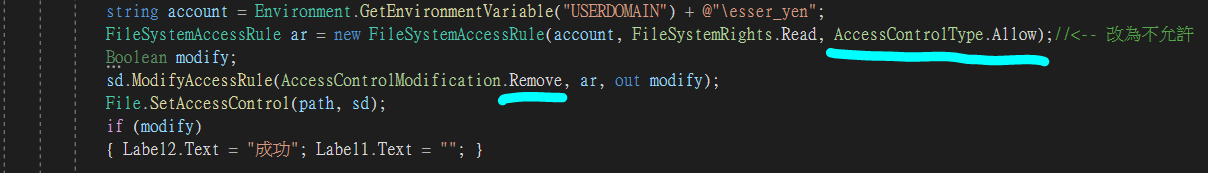
結果找不到這個人，於是有上面的錯誤。

現在我使用真人︰PASKEY\esser\_yen

那就通過了。



呃…增加了讀取權限的「允許」和「拒絕」所以同時存在是什麼鬼，到底算哪邊？



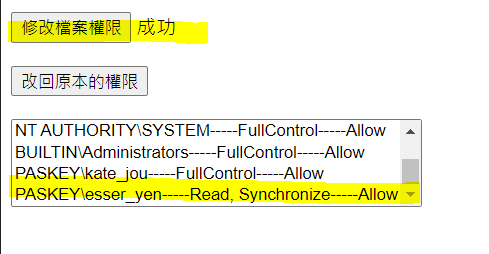
如果是要「收回」允許讀取的權利，不是將AccessControlType改為Deny

而是 AccessControlModification 由 Add 改為 Remove

還有，保不保留繼承這一點，我完全看不出有什麼差別？

fs.SetAccessRuleProtection(true, true);

fs.SetAccessRuleProtection(false, false); // <-- 改為不繼承



我在練習中，加入讀取權限的實作。

P 545 完結