Poiché questa implementazione di `Dispose` non è `synchronized`, è possibile che `Cleanup` sia chiamato da un primo thread e poi un secondo thread prima che `_myObj` sia impostato a `null`. In base a ciò che accade quando viene eseguito il codice di cleanup, si può trattare di un problea di sicurezza o meno.
Un problema importante con l'implementazione di Dispose non sincronizzata comporta un reale problema nella gestione di risorse. La deallocazione impropria può causare una gestione dell'utilizzo errata, che spesso conduce a vulnerabilità di sicurezza.
In alcune applicazioni, potrebbe essere possibile che altri thread accedano ai membri della classe prima che i loro costruttori di classe siano completamente eseguiti. È necessario esaminare tutti i costruttori di classe per assicurarsi che non ci siano problemi di protezione e sincronizzare i thread, se necessario.

7.6.10.10Serializzazione e deserializzazione

Poiché la serializzazione può consentire ad altri moduli di visualizzare o modificare i dati di istanza dell'oggetto che altrimenti sarebbero inaccessibili, è necessaria una autorizzazione speciale per la serializzazione del codice. Per default questa autorizzazione non viene fornita al codice scaricato da internet o intranet; solo al codice sul computer locale è concessa questa autorizzazione.
Normalmente, tutti i campi dell'istanza di un oggetto vengono serializzati, il che significa che vengono serializzati anche i dati. È possibile che il codice possa interpretare il formato per determinare i valori dei dati, indipendentemente dall'accessibilità dei singoli membri. Analogamente, la deserializzazione estrae i dati dalla rappresentazione serializzata e imposta lo stato dell'oggetto direttamente, di nuovo indipendentemente dalle regole di accessibilità.
Qualsiasi oggetto che potrebbe contenere dati sensibili alla sicurezza dovrebbe essere reso non serializable. Se trettamente necessario adottare questa tecnica, occorre comunque creare campi specifici non serializzabili, quelli che - per esempio - contengano dati sensibili. Se questo non può essere fatto, tenere presente che questi dati saranno esposti a qualsiasi modulo che abbia l'autorizzazione a serializzare/deserializzare. In tal caso assicurarsi che nessun modulo non attendibile possa ottenere tale autorizzazione.
L'interfaccia `ISerializable` è destinata esclusivamente all'infrastruttura di serializzazione. Se si fornisce una serializzazione personalizzata implementando `ISerializable`, assicurarsi di adottare le seguenti precauzioni:
Il metodo `GetObjectData` dovrebbe essere protetto in modo esplicito o richiedendo l'autorizzazione SecurityPermission con SerializationFormatter specificata. Occorre anche assicurarsi che non venga rilasciata alcuna informazione sensibile con l'output del metodo.
Esempio:

```
[SecurityPermissionAttribute(SecurityAction.Demand,SerializationFormatter = true)]
public override void GetObjectData(SerializationInfo info,
StreamingContext context)
{
}
```

Il costruttore speciale utilizzato per la serializzazione dovrebbe inoltre eseguire una convalida completa degli input e dovrebbe essere dichiarata private o protected per proteggere la classe da un uso improprio e abusivo.

7.7 ASP

ASP (Active Server Page) identifica non un linguaggio di programmazione, ma una tecnologia Microsoft, per la creazione di pagine web dinamiche attraverso linguaggi di script come VBScript e Microsoft JScript. ASP sfrutta non solo la connettività del web server ma, si può interfacciare (attraverso oggetti COM) con tutte le risorse disponibili sul server e, in maniea trasparente, sfruttare tecnologie diverse.
Vengono di seguito analizzate le principali vulnerabilità e relative contromisure da adottare.

7.7.1 Cross-site scripting (XSS)

Come riconoscerla