Advanced Protection of Workflow Sessions with SEWebSession

M. Fonda, S. Moinard, C. Toinard

September 03, 2012

Presentation

- Ph.D Student at ENSI de Bourges (Engineer school) specialised in IT security

Presentation

- Ph.D Student at ENSI de Bourges (Engineer school) specialised in IT security

- Qual'Net : Document Management Systems (Intraqual Doc) and BPM (Intraqual Dynamic)

Motivations

- Control accesses of Web-based applications to the session state.

Motivations

- Control accesses of Web-based applications to the session state.
  - Application on web-based workflow applications.

Motivations

- Control accesses of Web-based applications to the session state.
  - Application on web-based workflow applications.

- Ease the definition of the rules required to protect the session $=>$ Protection language

Protection language

$$allow(Subject, Action, Object)$$

Protection language

$$allow(Subject, Action, Object)$$

- Security context (Subject and Object) : an extensible string representing entity i.e. either the Web application or a data of the session state.

Protection language

$$allow(Subject, Action, Object)$$

- Security context (Subject and Object) : an extensible string representing entity i.e. either the Web application or a data of the session state.
- Action : an unbounded number of elements defining the considered access to the target object.

Protection language

$$allow(Subject, Action, Object)$$

- Security context (Subject and Object) : an extensible string representing entity i.e. either the Web application or a data of the session state.
- Action : an unbounded number of elements defining the considered access to the target object.
- If a rule is not expressed then SEWebSession denies the access (default-deny setup).

Protection language

allow (∗ : *moinard* : *intranet* : *vacationRequest*, *write*, $1 : *supervisor*)

- Subject : session identifier, user identifier, the application domain, the application identifier plus any other additional information.

Protection language

allow (∗ : *moinard* : *intranet* : *vacationRequest*, *write*, $1 : *supervisor*)

- Subject : session identifier, user identifier, the application domain, the application identifier plus any other additional information.
- An action includes a type of access (read/write/create/delete) plus any additional constraint for the action.

Protection language

allow (∗ : *moinard* : *intranet* : *vacationRequest*, *write*, $1 : *supervisor*)

- Subject : session identifier, user identifier, the application domain, the application identifier plus any other additional information.
- An action includes a type of access (read/write/create/delete) plus any additional constraint for the action.
- Object : the session identifier, the data identifier plus any other additional attribute characterising the data.

Protection language

allow (∗ : *moinard* : *intranet* : *vacationRequest*, *write*, $1 : *supervisor*)

- Subject : session identifier, user identifier, the application domain, the application identifier plus any other additional information.
- An action includes a type of access (read/write/create/delete) plus any additional constraint for the action.
- Object : the session identifier, the data identifier plus any other additional attribute characterising the data.

Grants the user Moinard to use the vacationRequest application into the intranet context in order to write the supervisor status.

Implementation

- Implementation over QualNet's Intraqual Dynamic (IIS, ASP.Net and SQL Server).

Implementation

- Implementation over QualNet's Intraqual Dynamic (IIS, ASP.Net and SQL Server).
- The session state can be stored InProc, in a State Server or an SQL Server.

Implementation

- Implementation over QualNet's Intraqual Dynamic (IIS, ASP.Net and SQL Server).
- The session state can be stored InProc, in a State Server or an SQL Server.
- Processing of an HTTP request : ASP.Net provides a serie of events such as BeginRequest, AcquireRequestState, EndRequest, etc.

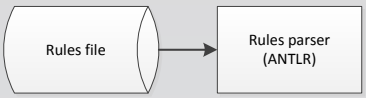
Implementation

- Implementation over QualNet's Intraqual Dynamic (IIS, ASP.Net and SQL Server).
- The session state can be stored InProc, in a State Server or an SQL Server.
- Processing of an HTTP request : ASP.Net provides a serie of events such as BeginRequest, AcquireRequestState, EndRequest, etc.
    - An HttpModule can hook these events.

Implementation

- Implementation over QualNet's Intraqual Dynamic (IIS, ASP.Net and SQL Server).
- The session state can be stored InProc, in a State Server or an SQL Server.
- Processing of an HTTP request : ASP.Net provides a serie of events such as BeginRequest, AcquireRequestState, EndRequest, etc.
  - An HttpModule can hook these events.
  - Each handler can extract information from the request and/or modify the context of the request.

Implementation - HTTPModule



The parsing of the rule file generate an HTTPModule for IIS.

Implementation - HTTPModule

The parsing of the rule file generate an HTTPModule for IIS.

Implementation - HTTPModule

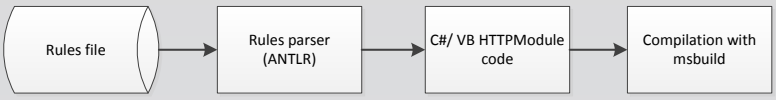
```
1  Private Sub AcquireRequestState(ByVal source As Object, ByVal e As EventArgs)
2
3   'Getting session variables before request execution
4   If fileExtension.Equals(".aspx") Then
5
6     ...
7    idsession = ctx.Session.SessionID
8
9    'rule verification
10   If ctx.Request("supervisor") IsNot Nothing AndAlso
11     CInt(ctx.Session("supervisor")) <> CInt(ctx.Request("supervisor")) AndAlso
12     ctx.Session("login").ToString <> "moinard" Then
13
14    ctx.Response.Clear()
15    ctx.Response.Write("violation detected for session ")
16    ctx.Response.Write( ctx.Session.SessionID & " :")
17    ctx.Response.Write("<br/>login= " & ctx.Session("login").ToString)
18    ctx.Response.Write("<br/>Session uid= " & ctx.Session("supervisor").ToString)
19    ctx.Response.Write("<br/>Requested uid= " & ctx.Request("supervisor").ToString)
20    ctx.Session.Abandon()
21    ctx.Response.end()
22   End If
23
24  End If
25 End Sub
```

Listing 1: Exemple of generated code

Implementation - HTTPModule

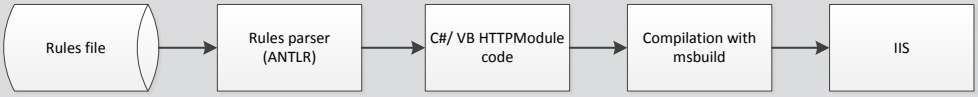
The parsing of the rule file generate an HTTPModule for IIS.

Implementation - HTTPModule



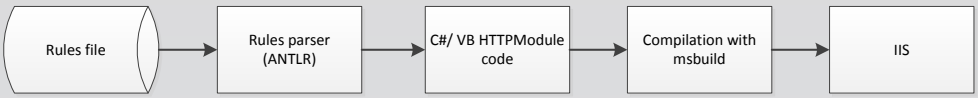
Then the code is compiled as an HTTP Module and added in IIS.

Implementation - HTTPModule



Then the code is compiled as an HTTP Module and added in IIS.

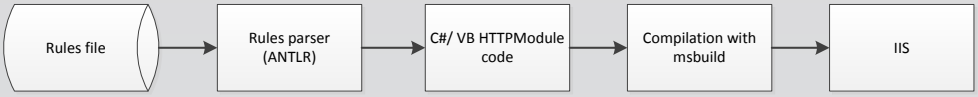
Implementation - HTTPModule



Then the code is compiled as an HTTP Module and added in IIS.

- The module applies the rules by comparing the live session (in IIS memory) and the one in the session server.

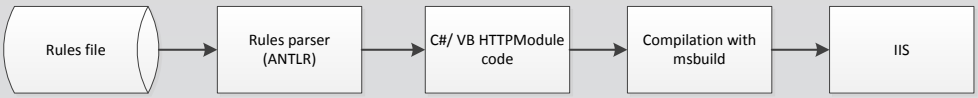
Implementation - HTTPModule



Then the code is compiled as an HTTP Module and added in IIS.

- The module applies the rules by comparing the live session (in IIS memory) and the one in the session server.
- The session server is accessed by HTTP requests (ASP.Net_state service) or SQL queries (SQL Server).

Implementation - HTTPModule



Then the code is compiled as an HTTP Module and added in IIS.

- The module applies the rules by comparing the live session (in IIS memory) and the one in the session server.
- The session server is accessed by HTTP requests (ASP.Net_state service) or SQL queries (SQL Server).
- When the live session satisfies the rules, the Web request continues.

Conclusion

- Protection of the session anywhere it is located.

Conclusion

- Protection of the session anywhere it is located.
- Dedicated language eases formalization of the security requirements.

Conclusion

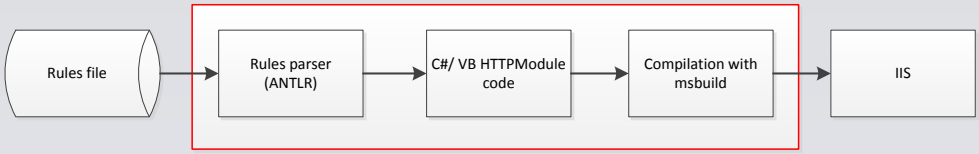
- Protection of the session anywhere it is located.
- Dedicated language eases formalization of the security requirements.
- Unique reference monitor (HTTPModule).

Conclusion

- Protection of the session anywhere it is located.
- Dedicated language eases formalization of the security requirements.
- Unique reference monitor (HTTPModule).
- Future works :

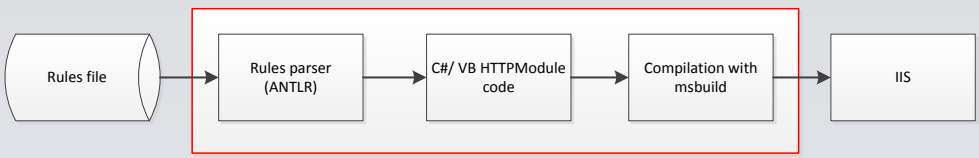
Conclusion

- Protection of the session anywhere it is located.
- Dedicated language eases formalization of the security requirements.
- Unique reference monitor (HTTPModule).
- Future works :
  - Avoid the production of code.



Conclusion

- Protection of the session anywhere it is located.
- Dedicated language eases formalization of the security requirements.
- Unique reference monitor (HTTPModule).
- Future works :
  - Avoid the production of code.



- Use an advanced reference monitor like PIGA to control transitive iinformation flows crossing different Web applications and session state ressources.

Thank you !