第５章　セキュア VBScript / ASP プログラミング

［5-3.］ セッションタイムアウト

ほかのWebアプリケーション動作環境と同様，Internet Infomation Server（IIS）のActive Server Pages（ASP）にもセッション・タイムアウト機能が備わっている。ただし，デフォルトのタイムアウト時間は比較的長いものであり，これを放置しておくことは，セッションの盗用などの不正行為に対して安全であるとはいえない。

● ● ●

WWW セッション

Webアプリケーションでいろいろな処理をさせたい場合，複数のWWWページを組み合わせて会話処理のセッションを組み立てる必要がある。しかし残念なことにHTTPそのものの仕組みだけでは自動的にセッションの維持が行われない。こうした事情から，サーバ側でWebアプリケーションを動かす各種エンジンにはCookieなどを使ったセッションの維持を自動で行う機能が備わっていることが多い。Active Server Pagesのエンジンもそうした機能を持っている。そこで一般的なWebアプリケーション・サーバには「セッション」を管理する機能が何らかの形で提供されている。ちなみにHTTPには，Keep-Aliveと呼ばれる機能があるが，それはHTTPより下位のTCPプロトコル上での接続を維持する機能であり，Webアプリケーションとして求める「セッション」の機能とは別である。

ASP のセッション管理

ASPに備わっているセッション管理機構は次のように動作する。まず最初に，Webブラウザが初めてASPプログラムにアクセスした時，ASPエンジンはセッションIDと呼ばれる，一意かつ事前の予測が困難な値を生成する。生成されたセッションIDは，Webブラウザにクッキーとしても保存させる。次回以降のアクセスにWebブラウザはそのクッキーを送り，ASPエンジンはWebブラウザから受け取ったセッションIDを元にどのセッションの続きであるのかを識別することができる（**図１**）。

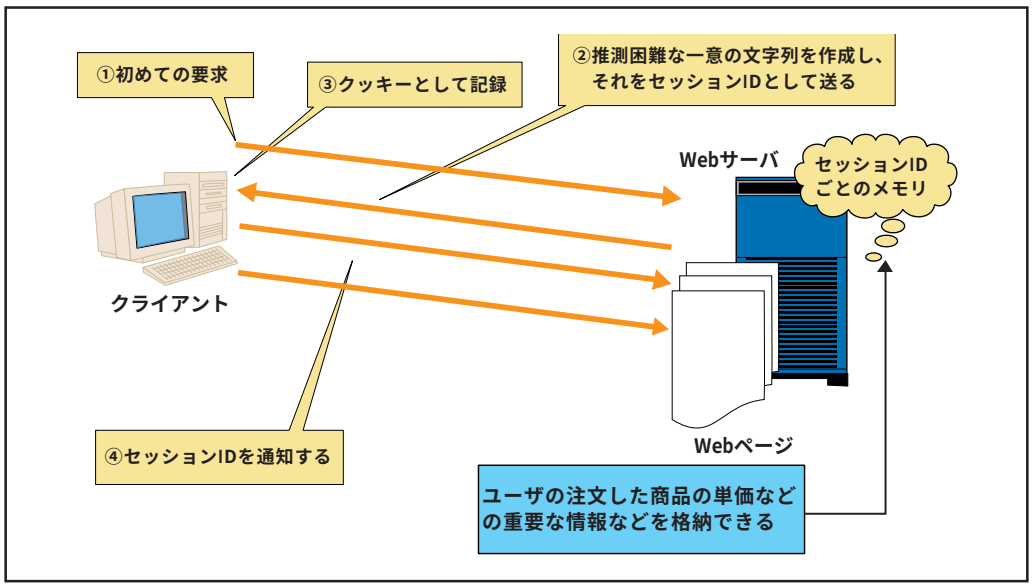
ASPプログラムは，ASPエンジンがセッションIDをもとに設定したセッション環境の下で起動され，ASPプログラムからは他のセッションは見えず，常にひとつのセッション環境が維持されているように見えるのである。

タイムアウト

WWWブラウザから何も通信が来ない間，WWWサーバ側ではユーザが自サイトのページを閲覧中なのか，それまでアクセスしていたWebアプリケーションの操作をやめてしまったかを知ることができない。WWWブラウザからの次の通信を際限なく待ち続けることはシステム資源の面からもセキュリティの観点からも得策

図１　ASP のセッション管理の仕組み

初めてアクセスするユーザに対しては，ASP エンジンは推測困難な ID を生成して，それをセッション ID としてクライアントに送信。二回目以降は，クライアントはその ID をサーバに戻すことでサーバはクライアントの前回のアクセスと関連付けることができる。さらに，セッションIDごとにサーバのメモリ上に変数を格納することができる。



ではないから，ASP のセッションは一定時間でタイムアウトし，うち切られるようになっている。

一定時間経過後もWWWブラウザから再びアクセスがない場合，該当セッションのセッションIDは無効となり，そのセッションのために使われていたサーバ上の資源は解放される。

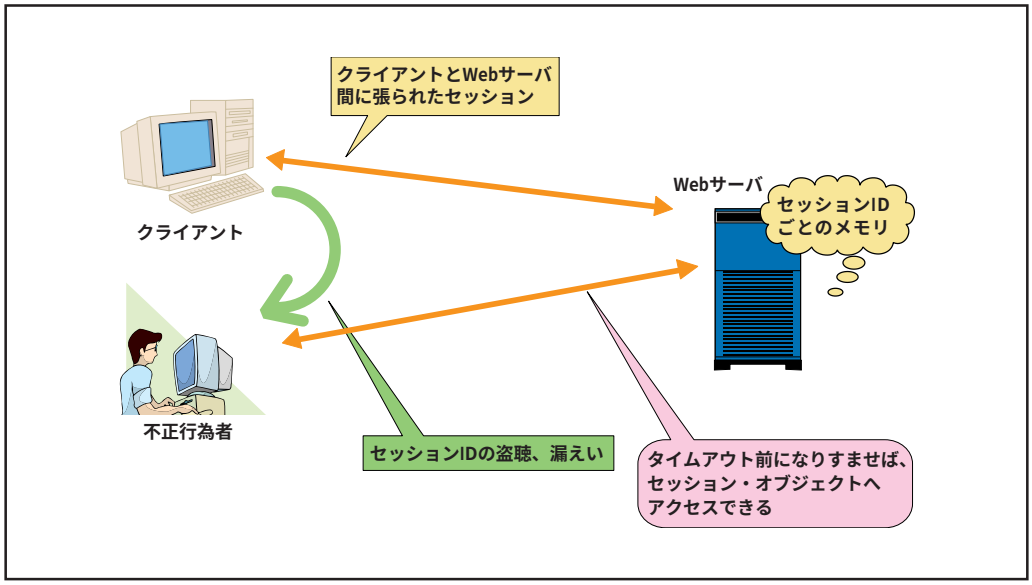
セッション盗用

セッション・タイムアウトの時間は IIS 4.0，IIS 5.0 ともデフォルトでは 20 分に設定されている。

言い換えると，ユーザがあるASPアプリケーションとの会話をうち切ったつもりでいても，20分間はそのセッションが生き残っているということである。もし，本人がWWWブラウザを終了させずに席を離れた隙に他人が操作をして当該ASPアプリケーションのページを呼び出すことができれば，生きているセッションに再接続することができ，各種の情報が漏れてしまうことになる。

また，ネットワークの盗聴やクロスサイトスクリプティングといった手口でセッションIDを盗んで，セッションがタイムアウトする前に別のコンピュータから再接続しても，他人のセッションを乗っ取れる（**図２**）。

図２　Webアプリケーションのセッション・タイムアウトを余分に大きな値にすると，ユーザのセッションIDを不正に入手した不正行為者によってなりすまされる危険性が高くなる



安全のためのセッション管理

セッションが有効に継続している最中に別のコンピュータから割り込まれる問題には，セッションID以外のクライアント側認証データの導入など異なる方法で対処しなくてはならないが，ユーザは終了したつもりでもセッションが消滅しておらず他人に再接続される問題に対しては，セッション・タイムアウト時間をなるべく小さく設定することである程度対処できる。

次のような対策を施すことを推奨する。

(1) 必要以上にセッションを継続させないため，それが可能な箇所では次のようにSession.Abandonメソッドを実行してセッションを破棄する。Sessionはセッション管理機能にアクセスするためのASPの組み込みオブジェクトである。

```
Session.Abandon
```

(2) セッションの継続が必要で，ユーザに閲覧させる情報・入力させる項目があまり多くないページでは，Session.Timeoutプロパティに小さな値を設定し，無理のない範囲でタイムアウトの時間を短くする。Session.Timeoutプロパティには１分刻みでタイムアウト時間を設定できる。その最小値は１分である。次の代入文は，セッション・タイムアウトを３分に設定する例である。

```
Session.Timeout = 3
```

Session.Timeoutに設定したタイムアウト時間は，次に設定し直すまで複数のASPページにわたって有効である。なお，設定の効果は同じセッションに属すページに限られる。

(3) 入力欄の多いユーザ登録ページなどでは，ユーザが入力に手間取ることを考慮して，あまり大きくない範囲で，タイムアウトの値に余裕を持たせる。

(4) 入力欄があまりに多い場合は，ページを分割することも考慮する。

(5) ユーザに閲覧させる情報が多い，またはユーザがゆっくりと情報を閲覧するようなページ（契約条項のページなど）などでは，一度セッション自体を切断すること（Session.Abandon）も考慮する。

(6)「不正アクセスを防止するためにタイムアウトの値を細かく設定しています」のような案内を，Web アプリケーションの要所要所に配置する。

ハートビート

短い一定時間ごとに通信を送って相手先の「生存」を問い合わせたり発信側の「生存」を通知する手法を「ハートビート」（心臓の鼓動）と呼ぶ。

バナー公告などを別フレーム化しているサイトの場合，その特定フレーム部分を WWW ブラウザで繰り返しリロードするハートビートでセッションを継続させるとともに，セッション・タイムアウトの時間を最小の値である 1 分に設定することも可能である（**図 3**）。

(1) Session.Timeout の値を最小の 1 分に指定する。

```
<% Session.Timeout = 1 %>
```

ASP プログラムの文頭でタイムアウトを最小の 1 分に設定する

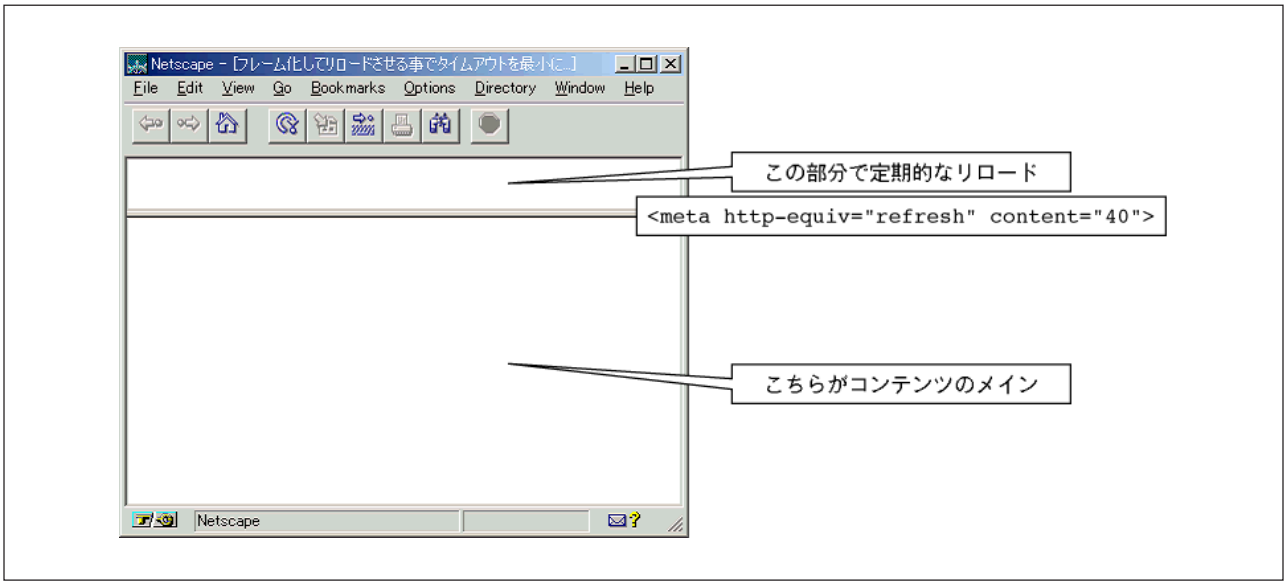
(2) WWW ブラウザには次の <meta> タグを含む HTML を送り，定期的にリロードさせる。

```
<meta  http-equev="refresh"  content="40">
```

40 秒おきにリロードさせるように指定

ただし，クライアントからサーバへのハートビートが行われるということ，すなわち定期的にクライアントからのリロード要求が発生するということは，ネットワーク，およびサーバに対する負荷が高くなるということであり，ネットワーク帯域，マシンの処理能力，およびソフトウェア設計について十分な考慮が必要である。

図 3　ページをフレーム化し，1 つのフレームをサーバとの間で定期的にリロードさせること（ハートビート）で，ページが閲覧されている状態でのタイムアウトを防ぎつつ，タイムアウトの値を最小値にすることできる。



まとめ

セッション盗用対策の一環として，Webアプリケーションごとに，またはWebページごとに十分に小さなセッション・タイムアウト時間を設定しよう。必要なら，積極的にセッションを破棄することも考慮に入れるべきである。

関連記事

『1-2. クロスサイトスクリプティング』

参考文献

『ASP デベロッパーズガイド』
Andrew M.Fedorchek，David K. Rensin 著，SE 編集部 訳，河端善博 監修，翔泳社