# AddressChecking

## checkAddress

The ServiceProvider software prior to version 1.3c includes an Application-level setting in ShibbolethXml in the `<Sessions>` element called `checkAddres`
, which defaults to `true` if not present. As of version 1.3, this setting is `false` in the default file distributed with the software.

This single setting, in versions prior to 1.3c, has the combined semantic of forcing comparison of two distinct sets of client addresses:

- the address of the client placed into the initial SAML assertion created by the !IdP and the address of the client that delivers the SAML assertion to the !SP to create a session
- the address of the client making a resource request and the address of the client that was given the session cookie that corresponds to the current request

As you can see, these are two somewhat distinct concepts. More problematic is that the first comparison is often difficult to perform because it is often the case in modern networks that a client will be seen to have distinct addresses when it accesses two disparate web servers, as in many cases the !IdP and !SP are. As a result, many applications require that `checkAddress` be set to `false` to function effectively.

Under the premise that the second comparison is less likely to cause user complaints than the first comparison, and because the second comparison is arguably much more important in terms of preventing session hijacking through cookie theft, a second setting has been created.

## consistentAddress

As of version 1.3c, the software includes a new Application-level setting for the `<Sessions>` element called `consistentAddress` . It defaults to `true` , including when not present, which means that sites upgrading to this version will automatically enforce this setting even if `checkAddress` is set to `false` . This setting can be added after the upgrade, but this is not recommended unless absolutely required.

This setting ensures that once a session cookie is issued to a client, any further use of that session cookie must be from a client with the same network address. This raises the bar for session hijackers to the level of network address spoofing, which may or may not be simple to do, but is definitely harder than stealing cookies and relies on a different set of attacking skills.