

The Authentication Server

Reference: RFC 931

RFC 931 Case Study

- This is an example of an *application-level* protocol.
- We will not worry about:
 - How the application data is transferred between client and server.
- We will focus on:
 - The rules that govern the data that is exchanged between client and server.

Before we start

- The authentication service described:
 - Can be useful
 - Can slow things down
 - Is not widely supported
 - Is not widely used
- Just because it has a *fancy* name doesn't mean it's an important protocol!

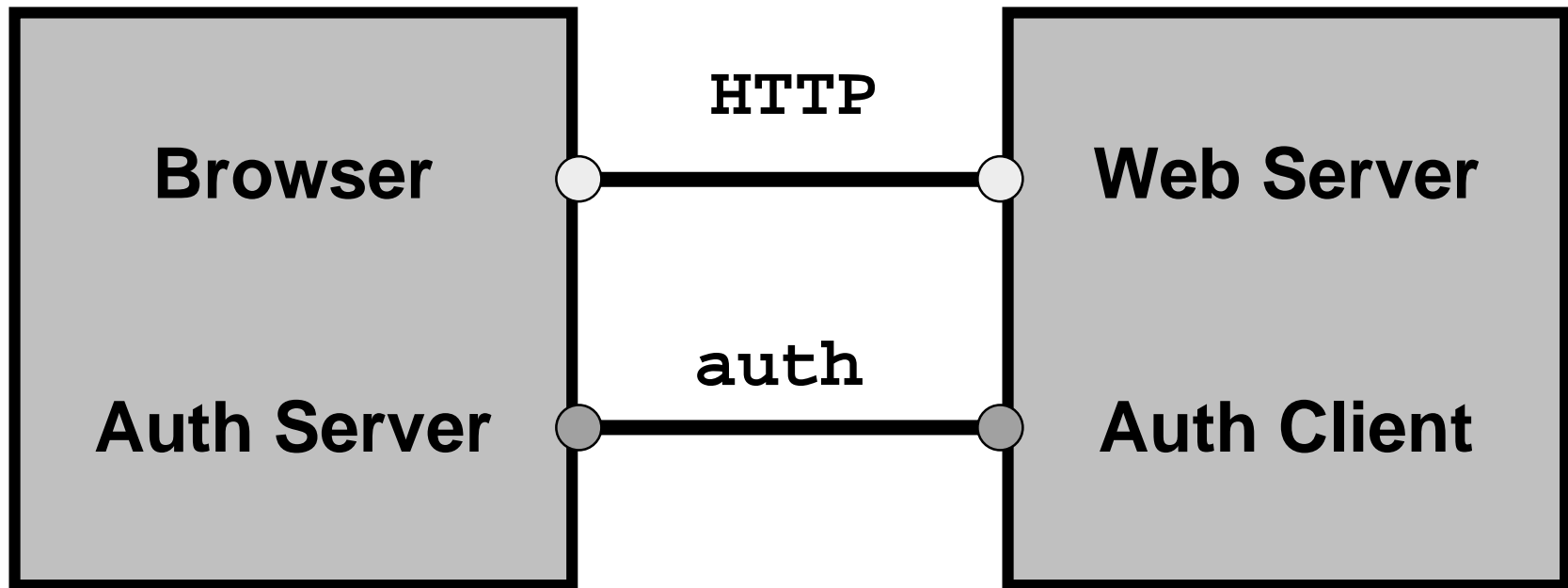
Authentication Server Protocol

- TCP based network service
- Server listens on port 113_{10}
- Server provides usernames associated with other TCP connections on the server machine

One possible use

Host A

Host B



The Web Server want to know who is running the browser.

Problems with `auth`

- In general there is no reason to "trust" the response provided by the `auth` server.
- Most *clients* are now running on PCs
 - Don't usually have `auth` server running.
- Lots of firewall issues
 - Don't allow TCP requests from outside the protected zone.

The application protocol

- All data sent is ASCII text.
 - no network byte order issue, we send strings.
- Request is a single line of text.
 - text identifies the active TCP connection that the `auth` client is interested in.

Request Format

`<local-port>, <foreign-port>`

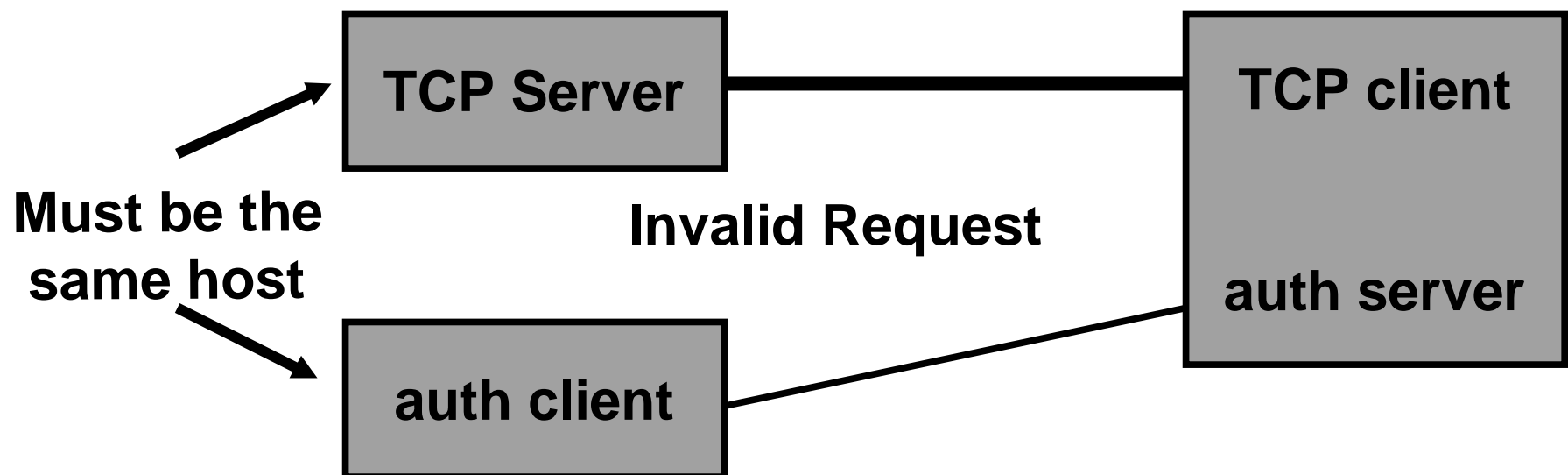
- The local port identifies the TCP port number on the auth server machine.
- The foreign port identifies the TCP port number on the auth client machine.

The Service

- The auth server can find out the IP address of the auth client
 - `getpeername()`
- The auth server asks the O.S. for:
 - pid of process using the local-port.
 - IP address of the other host connected to the local-port

The service (cont.)

- If the auth client IP address does not match the remote host IP address, the auth server sends an error message and closes the connection



Valid Request

- If the request is valid the auth server looks up the username of the process attached to the TCP connection.
- The server sends back a response that includes the username

Response Format

- One line of ASCII text:

`<local-port>,<foreign-port> : <response-type> : <info>`

- Response type can be: “USERID” or “ERROR”

- info depends on “Response Type”

Response-type: ERROR

- If the response indicates an error, the info string can be:
 - “invalid-port”
 - “no-user”
 - “unknown-error”

Response-type: USERID

- Info contains:
 <OPERATING-SYSTEM> : <USERNAME>
- Different Operating Systems have different formats for usernames...

Example Sessions

client: "1829, 7654\n"

server: "1829, 7654 : USERID : Unix : sally\n"

client: "1829, 7654\n"

server: "1829, 7654 : ERROR : Invalid Port\n"