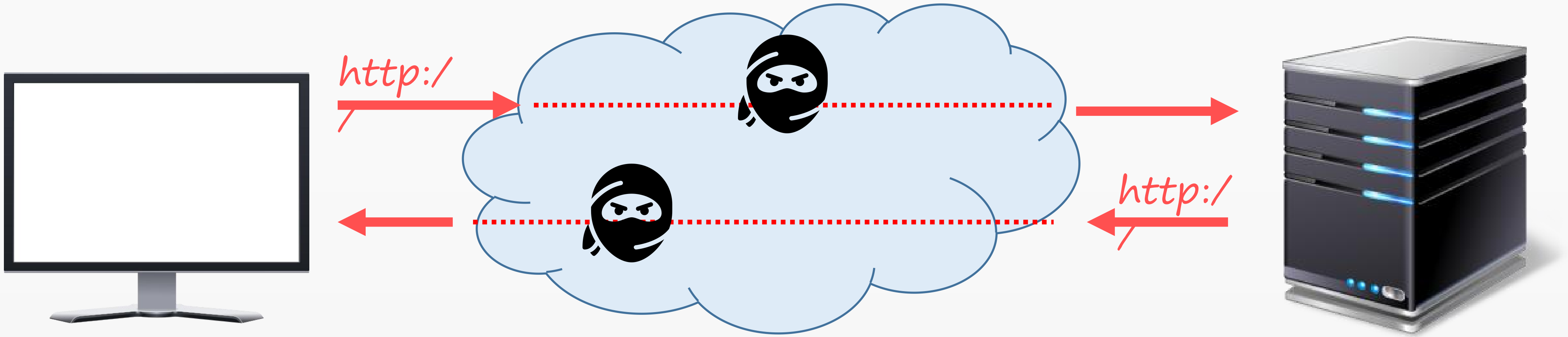
BASIC HTTP AUTH

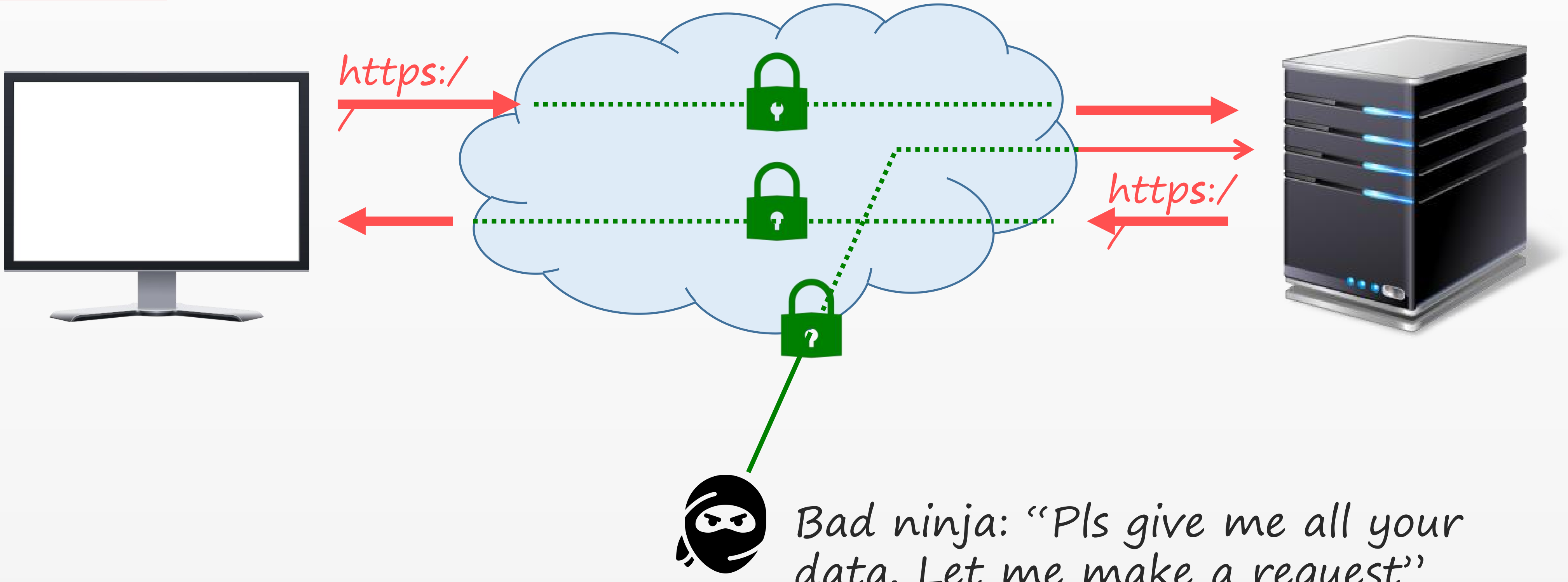
1. Communication with server using an <u>unsafe protocol</u> (like HTTP)



This data is not safe, so a bad person (ninja in the middle) can read the data

BASIC HTTP AUTH

2. Communication with a server using the safe HTTPS protocol



Bad ninja: "Pls give me all your data. Let me make a request"

The problem: the server still does not know who is authenticated

BASIC HTTP AUTH

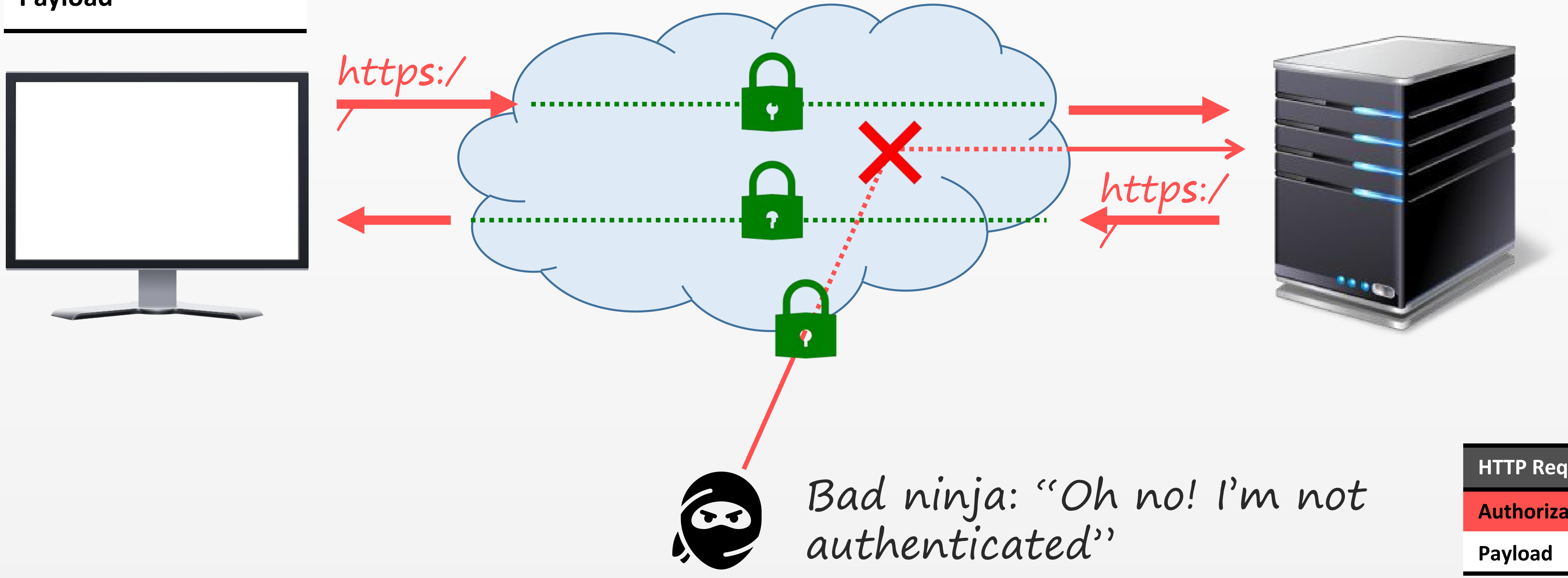
3. Communication with a server using BASIC HTTP Authentication

HTTP Request

Authorization Header

Payload

Auth information is now sent along with the request



Bad ninja: "Oh no! I'm not authenticated"

HTTP Request

Authorization Header

Payload

BASIC HTTP AUTH

Authorization Header

1. convert the username to base64

   e.g. wallyWarthog base64 encoded is d2FsbHlXYXJ0aG9n

2. convert the password to base64

   e.g. “loveGrass” base64 encoded is bG92ZUdyYXNz

3. join them together with a : in the middle

   e.g. d2FsbHlXYXJ0aG9n:bG92ZUdyYXNz

4. create the Authorization header (send in the HTTP request)

   Authorization: Basic d2FsbHlXYXJ0aG9n:bG92ZUdyYXNz

BASIC HTTP AUTH

Remember, for every request, you will be sending the same data (in our example, username & password) to the server

This is not ideal

Ideally you would rather work with a code (or token) only once

But Basic Auth still gets the job done, and it works well if you set it up correctly from the get-go