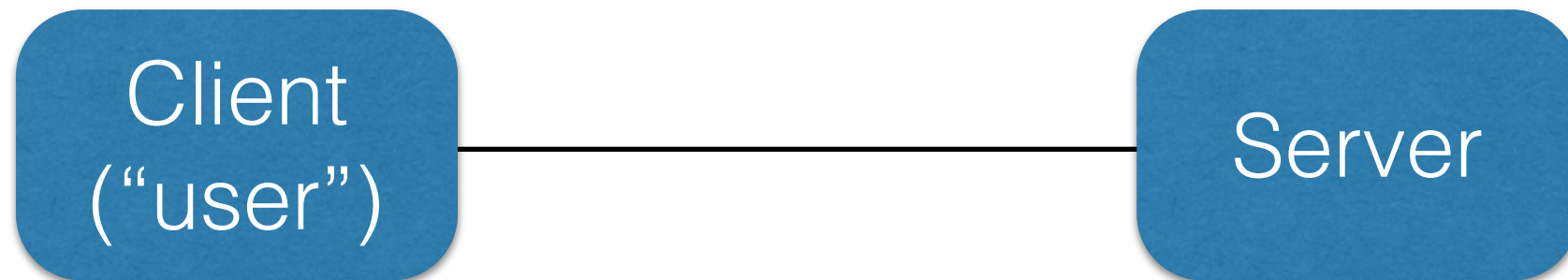


Proving who you are

Passwords and TLS

Basic, fundamental problem



**How do you prove to someone that
you are who you claim to be?**

Any system with access control must solve this

Users and servers prove themselves in different ways

- Different attack models
- Different assumptions about what they can feasibly do

User authentication

Attack model

- Attacker can eavesdrop and tamper with
1. communication between client and server
(or cut the server or client out altogether)



2. Attacker can subsequently compromise the server, gaining access to all of its data and secret keys

Attacker's goal:

Learn a targeted (or *any*) user's password

**Our goal: prove to the server we are the user
without compromising our password after server
compromise**

(see notes for details of
how passwords are stored)

Storing passwords in Linux

Stored in /etc/shadow

username

salt

hash

seed: \$6\$5MfvmF0aDU\$CVt7...:14400:0:99999:7::

algorithm

- 1: MD5-based
- 2: Blowfish
- 5: SHA-256
- 6: SHA-512

last changed (days since 1/1/1970 :
min #days until must change :
max #days until must change :
#days before expire to warn :
(#days to wait from expire to disable) :
(#days this has been expired)

Thought experiment: Do we need good passwords?

- Is the attack:
 - **Online**: attacker can only guess by trying to log in
 - **Offline**: the attacker has access to `/etc/shadow`
- Does the attack target:
 - A single *targeted* user
 - Any user: the attacker just wants *a* way to log in

Thought experiment: Do we need good passwords?

- Is the attack:
 - **Online**: attacker can only guess by trying to log in
 - **Offline**: the attacker has access to `/etc/shadow`
- Does the attack target:
 - A single *targeted* user
 - Any user: the attacker just wants *a* way to log in

Online attack

Thought experiment: Do we need good passwords?

- Is the attack:
 - **Online**: attacker can only guess by trying to log in
 - **Offline**: the attacker has access to `/etc/shadow`
- Does the attack target:
 - A single *targeted* user
 - Any user: the attacker just wants *a* way to log in

Online attack

Passwords are 6-digit numbers (bad!)

After 3 bad login attempts, wait 24hrs

Give the attacker 10 years to guess

Thought experiment: Do we need good passwords?

- Is the attack:
 - **Online**: attacker can only guess by trying to log in
 - **Offline**: the attacker has access to `/etc/shadow`
- Does the attack target:
 - A single *targeted* user
 - Any user: the attacker just wants *a* way to log in

Online attack

Passwords are 6-digit numbers (bad!)

After 3 bad login attempts, wait 24hrs

Give the attacker 10 years to guess

$$3 \cdot 365 \cdot 10 \text{ attempts} \\ \approx 10^4$$

Thought experiment: Do we need good passwords?

- Is the attack:
 - **Online**: attacker can only guess by trying to log in
 - **Offline**: the attacker has access to `/etc/shadow`
- Does the attack target:
 - A single *targeted* user
 - Any user: the attacker just wants *a* way to log in

Online attack

Passwords are 6-digit numbers (bad!)

After 3 bad login attempts, wait 24hrs

Give the attacker 10 years to guess

$3 \cdot 365 \cdot 10$ attempts
 $\approx 10^4$

Chance of success:
 $10^4 / 10^6 \approx 1\%$

Passwords and Computer Security

- In 2012, **76% of network intrusions exploited weak or stolen credentials** (username/password)
 - Source: Verizon data breach investigations report
- First step after any successful intrusion: install **sniffer** or **keylogger** to steal more passwords
- Second step: run cracking tools on password files
 - Cracking needed because passwords are not (or at least *should not be*) stored in the clear

Gary McKinnon

- 2001 and 2002: hacked into 97 US military and NASA computers searching for evidence of free energy suppression and UFO coverups
 - “... shut down the entire US Army’s Military District of Washington network of over 2000 computers for 24 hrs”
 - “... rendered [US Naval Weapons Station Earle]’s entire network of over 300 computers inoperable at a critical time immediately following 11 September 2001”
- Method: Perl script randomly looking for **blank and default passwords** to administrator accounts

Old password surveys

- Klein (1990) and Spafford (1992)
 - 2.7% guessed in 15 minutes, 21% in a week
 - Much more computing power is available now
- Univ. of Michigan: 5% of passwords were “goblue”
- Zviran and Haga (1999)
 - Password usage at DoD facility in California
 - 80% of passwords were 4-7 characters in length; 80% used alphabetic characters only; 80% of the users had never changed their password

RockYou Hack (2009)

- Social gaming company
- Database with 32 million user passwords from partner social networks
- Passwords stored in the clear
- December 2009: entire database hacked using a SQL injection attack and posted on the Internet

Passwords in RockYou database

Password Popularity – Top 20

Rank	Password	Number of Users with Password (absolute)
1	123456	290731
2	12345	79078
3	123456789	76790
4	Password	61958
5	iloveyou	51622
6	princess	35231
7	rockyou	22588
8	1234567	21726
9	12345678	20553
10	abc123	17542

Rank	Password	Number of Users with Password (absolute)
11	Nicole	17168
12	Daniel	16409
13	babygirl	16094
14	monkey	15294
15	Jessica	15162
16	Lovely	14950
17	michael	14898
18	Ashley	14329
19	654321	13984
20	Qwerty	13856

GAWKER MEDIA WEBSITES HACKED, STAFF AND USER PASSWORDS LEAKED

GAWKER MEDIA, THE blog powerhouse built by Nick Denton, has been hacked.

After bringing the company's websites to a standstill Sunday, one or more hackers operating under the name Gnosis released a 500-MB file apparently containing Gawker's source code, commenter and staff passwords, and internal conversations between the company's employees.

The e-mail addresses and passwords of hundreds of thousands of Gawker users have been compromised, the hackers said.

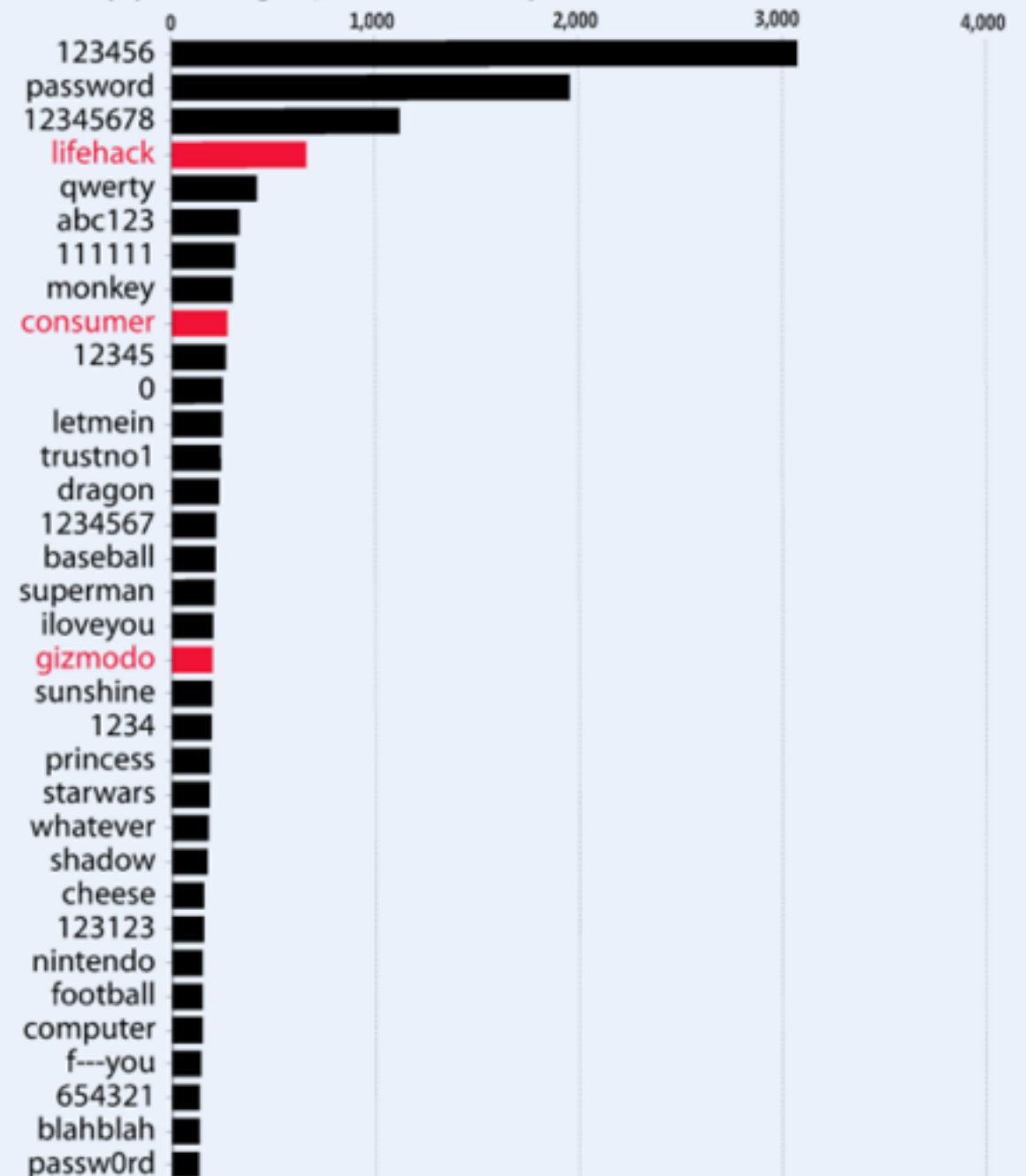
It's the worst security breach in New York-based Gawker's eight-year history, and a wake-up call to all web publishers. ([Click here to access your Wired.com profile](#) if you feel the need to change your password for this website.)

2010 - Gawker

Stored passwords encrypted 188,279 decrypted & released

Bet You Can Guess These

The most popular among 188,279 Gawker Media passwords that leaked online.



GAWKER MEDIA WEBSITES HACKED, STAFF AND USER PASSWORDS LEAKED

GAWKER MEDIA, THE blog powerhouse built by Nick Denton, has been hacked.

After bringing the company's websites to a standstill Sunday, one or more hackers operating under the name Gnosis released a 500-MB file apparently containing Gawker's source code, commenter and staff passwords, and internal conversations between the company's employees.

The e-mail addresses and passwords of hundreds of thousands of Gawker users have been compromised, the hackers said.

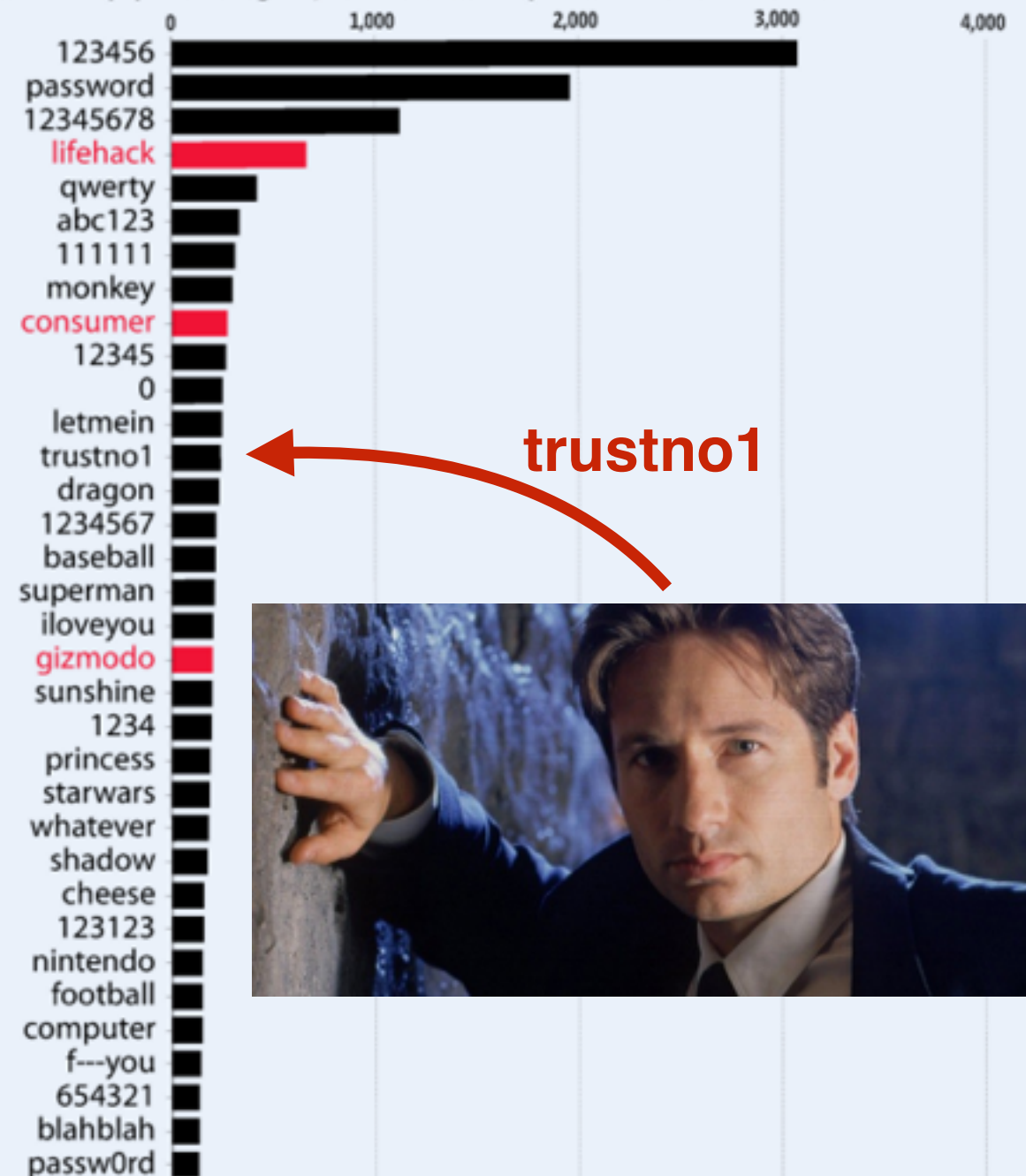
It's the worst security breach in New York-based Gawker's eight-year history, and a wake-up call to all web publishers. ([Click here to access your Wired.com profile](#) if you feel the need to change your password for this website.)

2010 - Gawker

Stored passwords encrypted 188,279 decrypted & released

Bet You Can Guess These

The most popular among 188,279 Gawker Media passwords that leaked online.



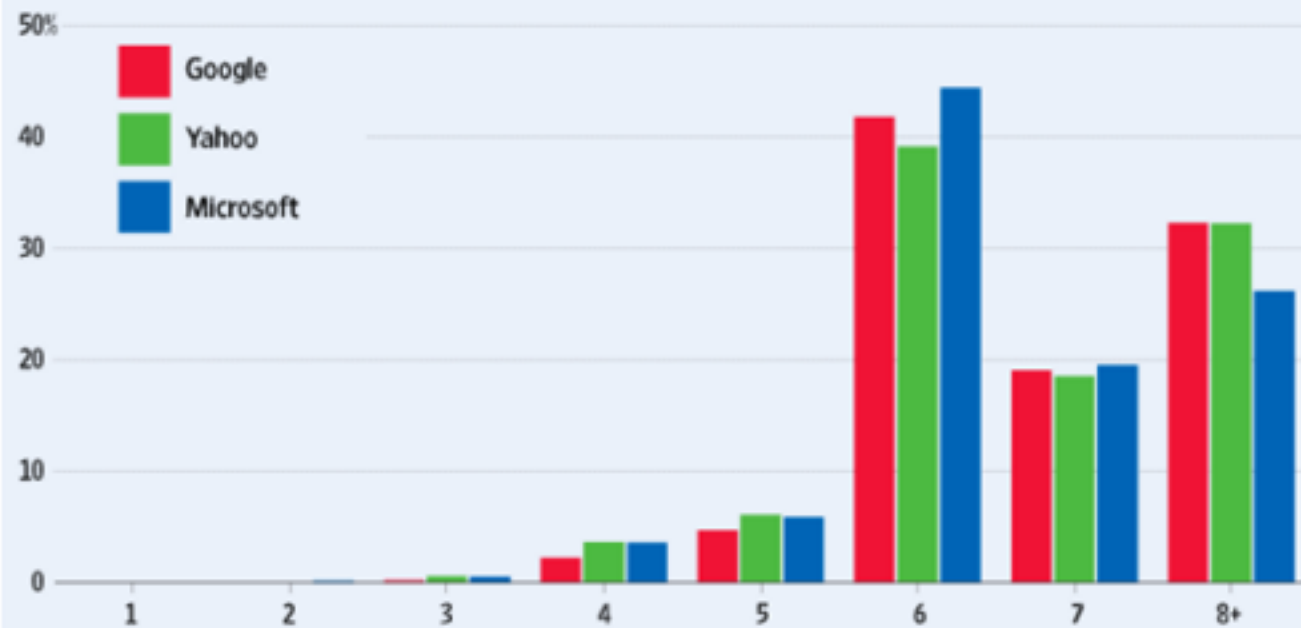
trustno1



Gawker Passwords (2010)

Gawker Media Password Lengths

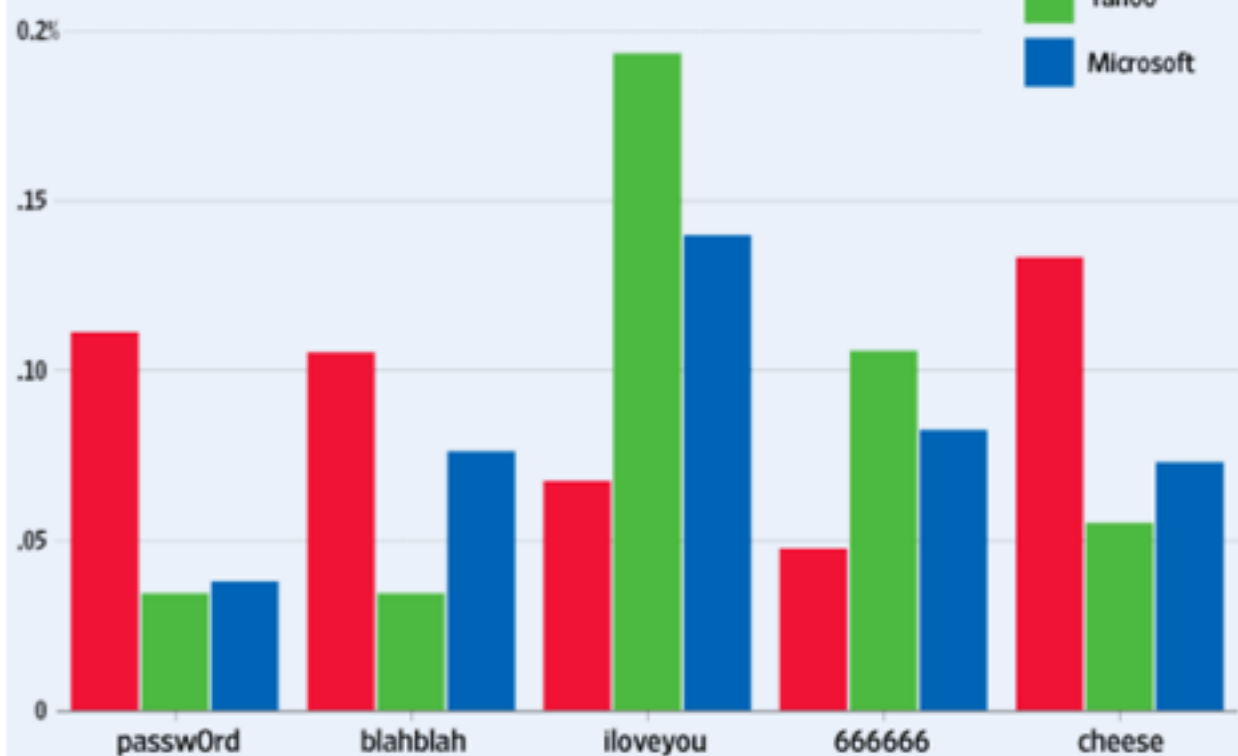
Percentage of passwords plotted against password length in characters. Divided by email provider.



Source: Anonymized set of 188,279 leaked Gawker Media passwords

What passwords did users have?

Percentage of users from each email service with each password



Source: Anonymized set of 188,279 leaked Gawker Media passwords

Server authentication

Attack model

- Attacker can eavesdrop and tamper with
1. communication between client and server
(or cut the server or client out altogether)

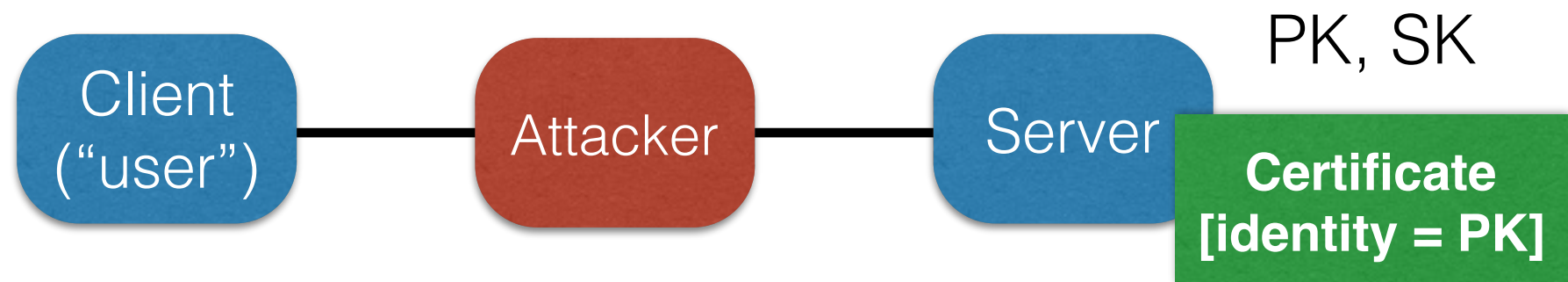


2. Attacker does *not* have access to the server's private key

Attacker's goal: Convince the Client that he is the 'Server'

**Our goal: Establish a session (symmetric) key
between the client and server**

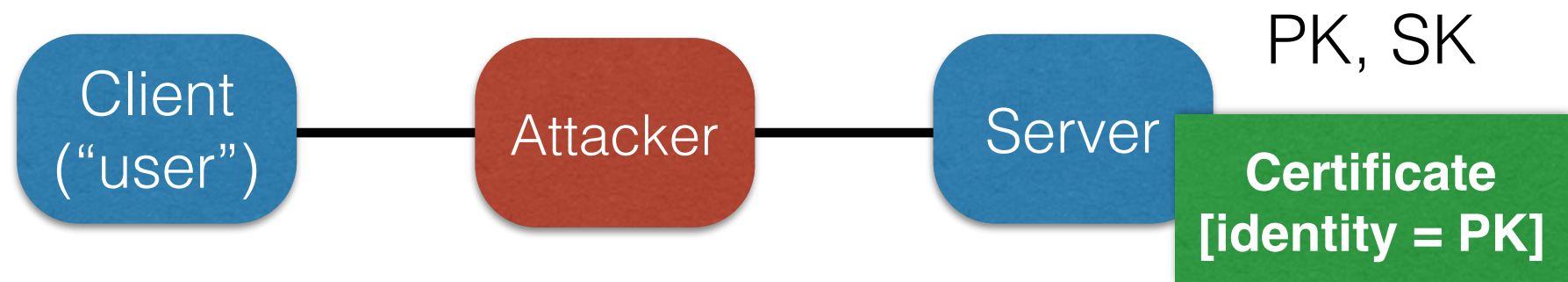
Server authentication: Certificates



Attacker does *not* have access to the server's private key

Recall: A certificate attests “the only person who knows the corresponding secret key is <identity>”

Server authentication: Certificates



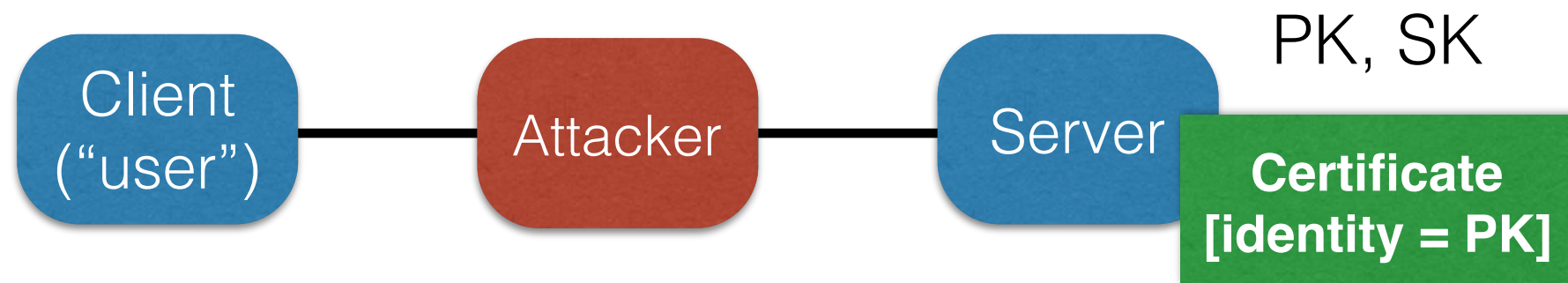
Attacker does *not* have access to the server's private key

Recall: A certificate attests “the only person who knows the corresponding secret key is <identity>”

Challenge #1

How does the server prove it knows its secret key without *giving away* its secret key?

Server authentication: Certificates



Attacker does *not* have access to the server's private key

Recall: A certificate attests “the only person who knows the corresponding secret key is <identity>”

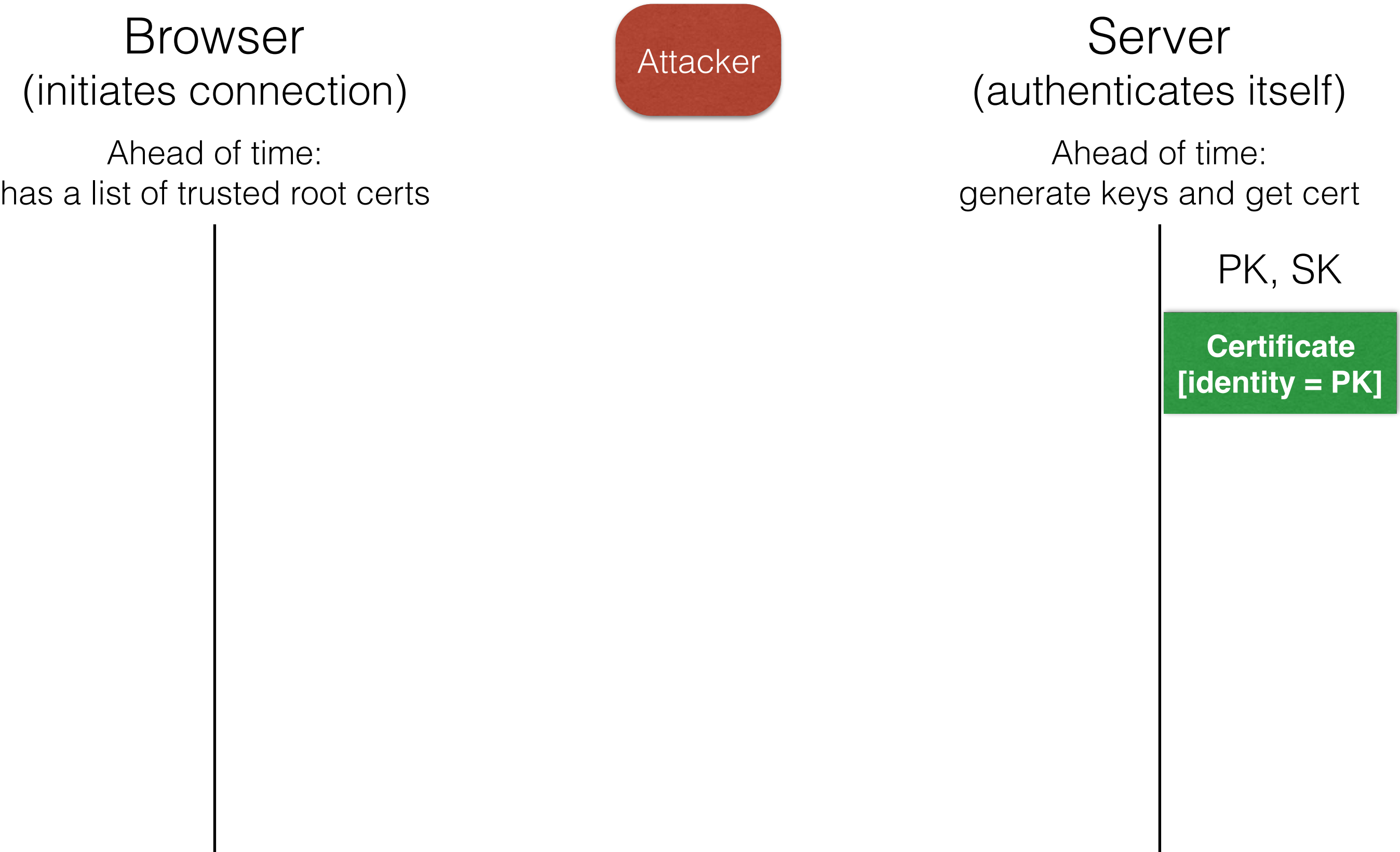
Challenge #1

How does the server prove it knows its secret key without *giving away* its secret key?

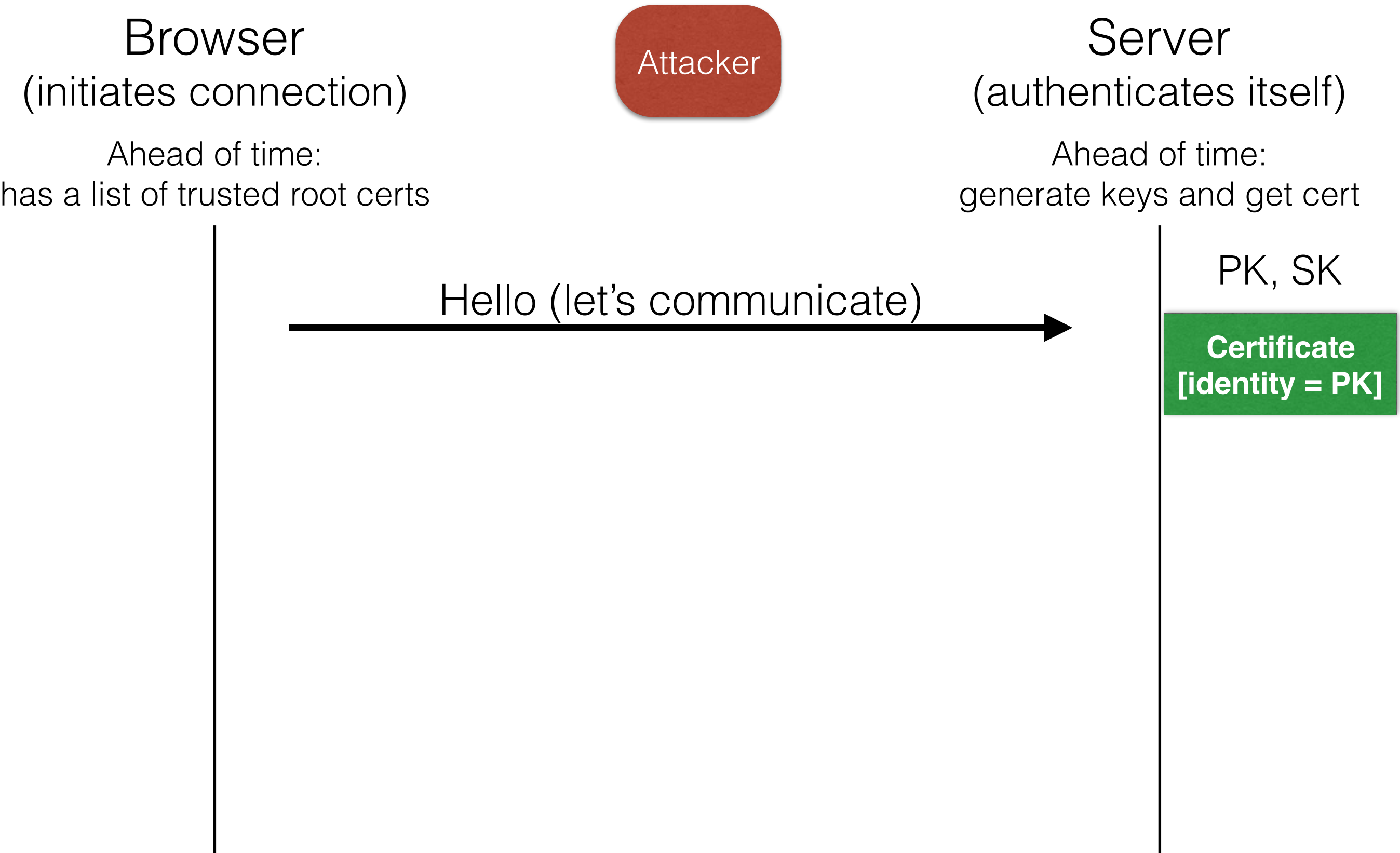
Challenge-response protocols:

The client issues a *challenge* that only someone with the secret knowledge could correctly react to

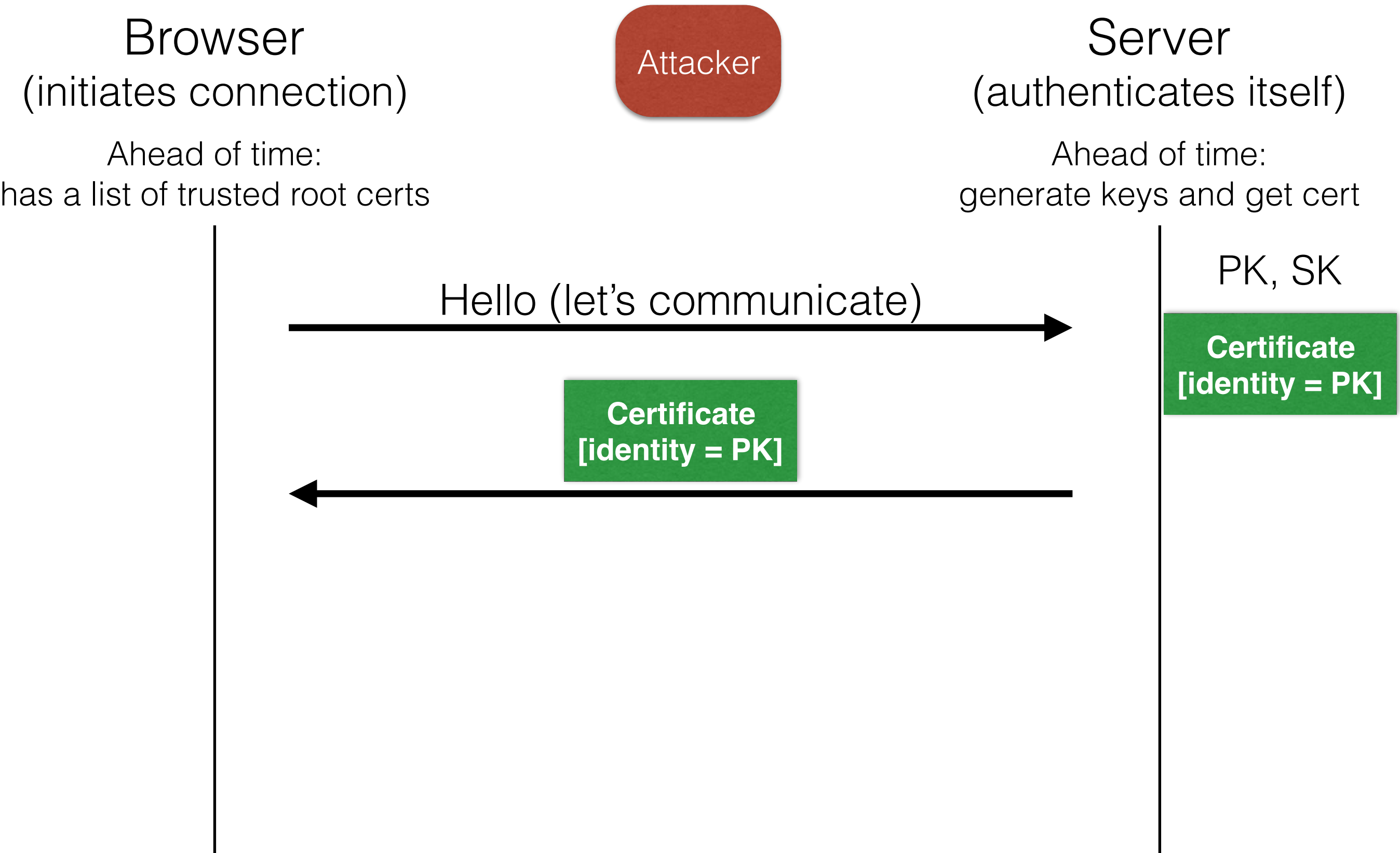
Server authentication with certificates



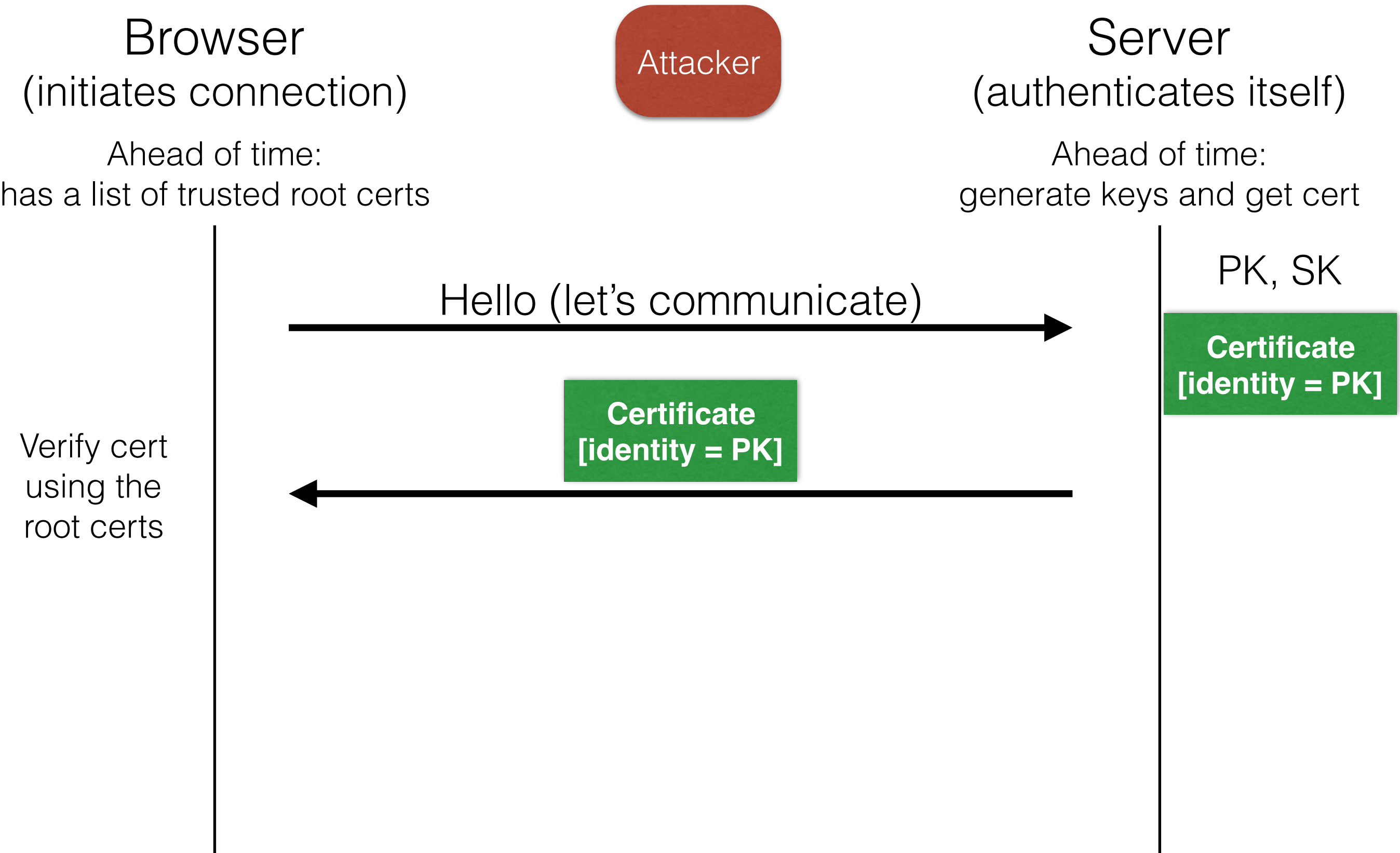
Server authentication with certificates



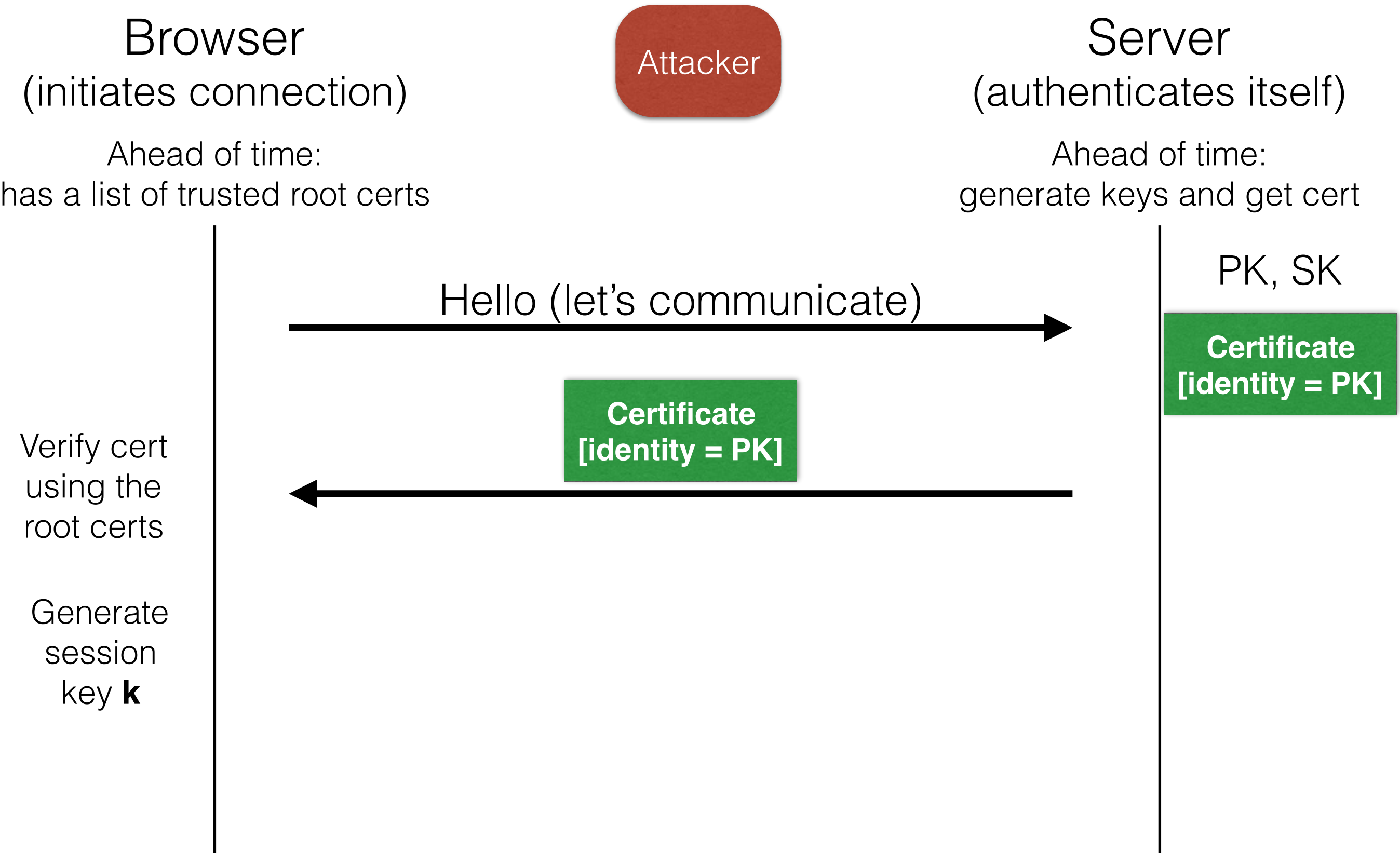
Server authentication with certificates



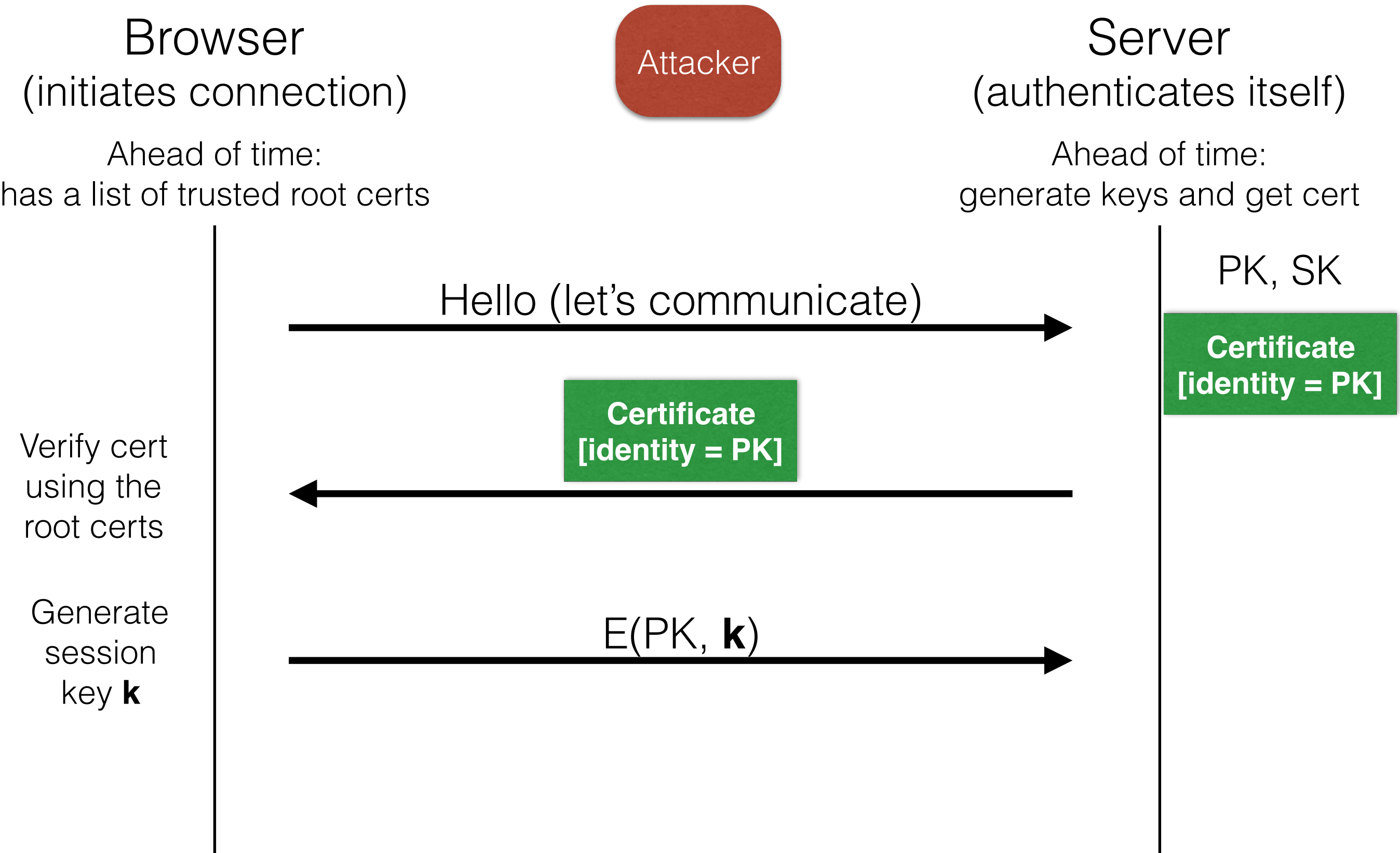
Server authentication with certificates



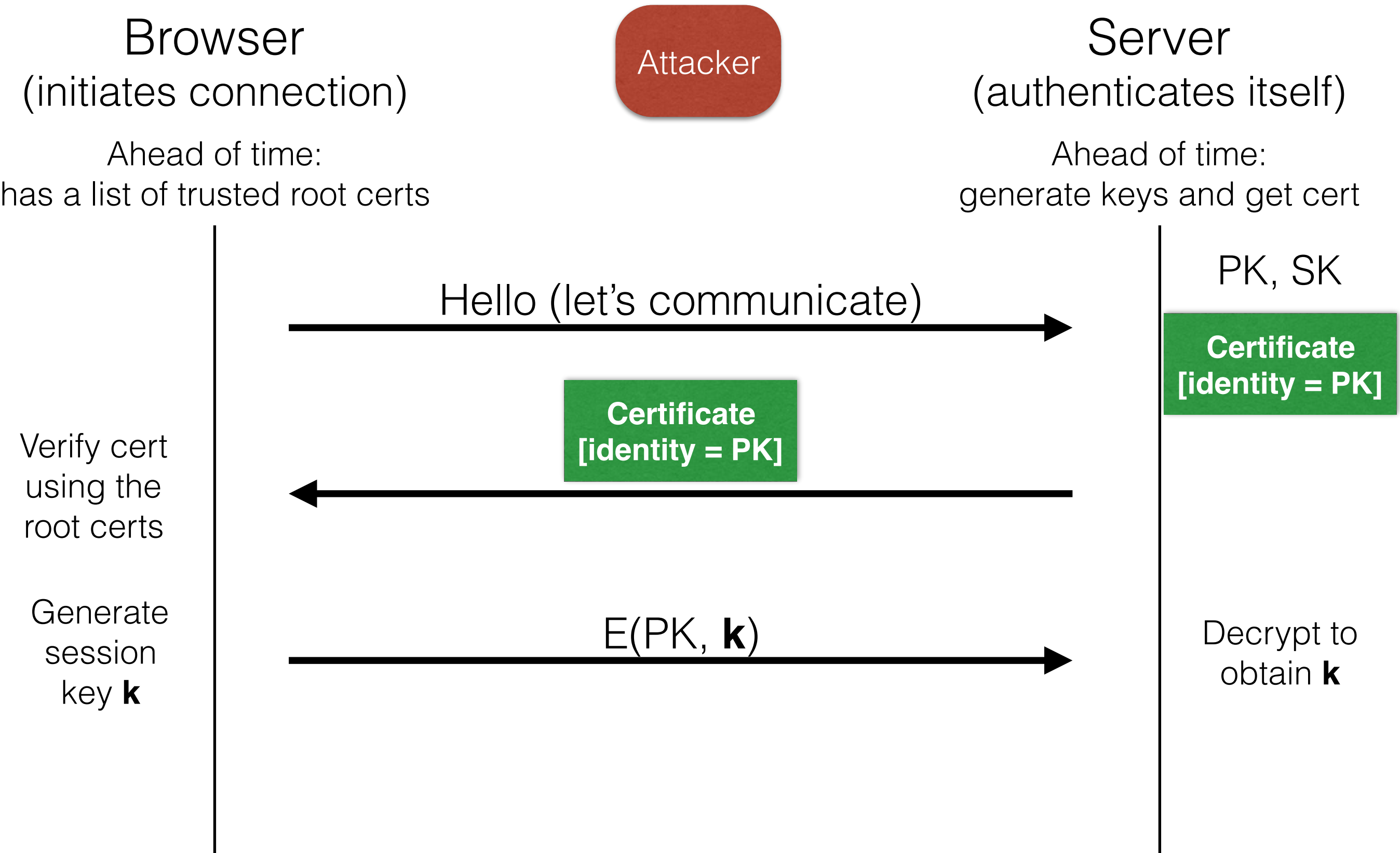
Server authentication with certificates



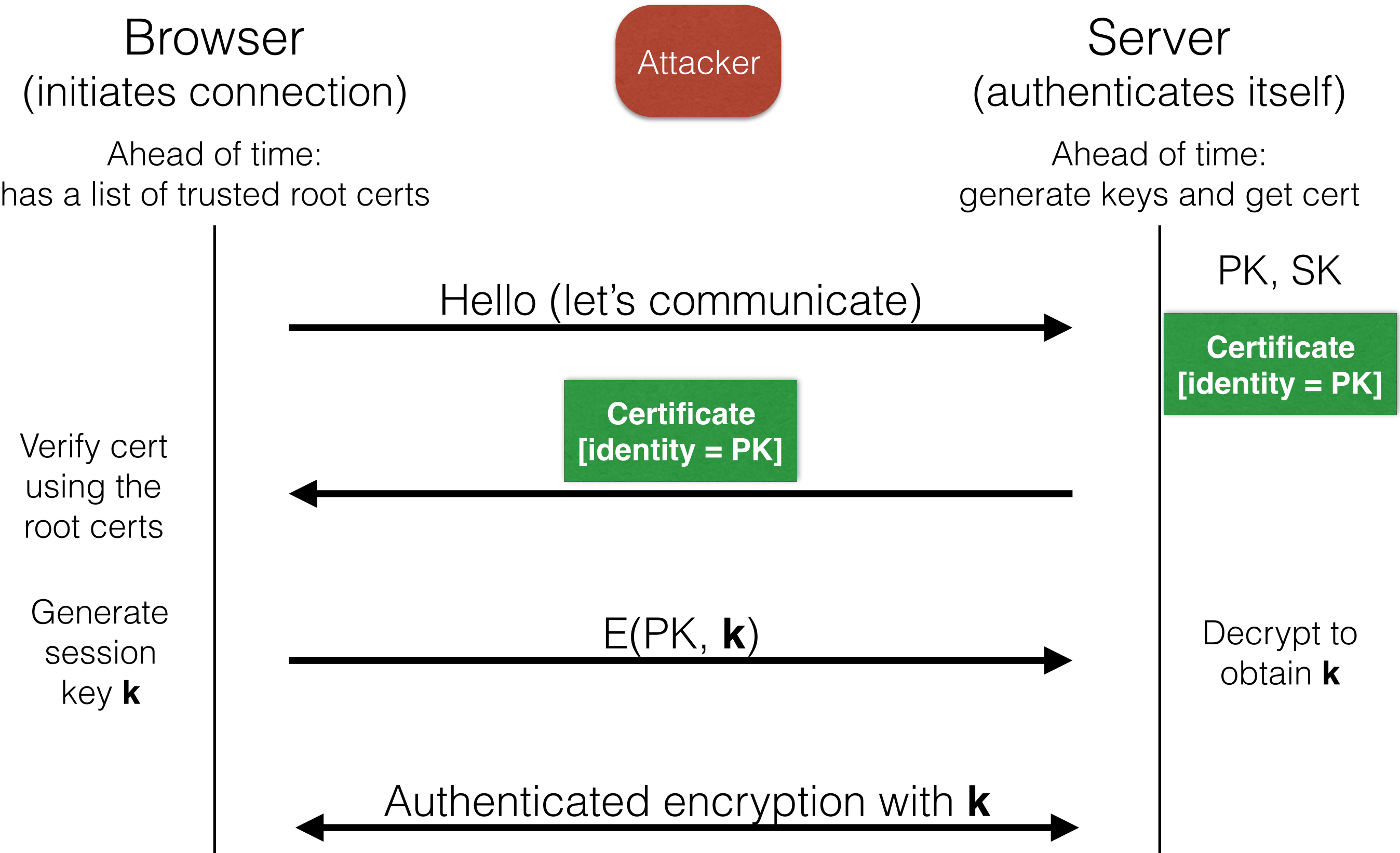
Server authentication with certificates



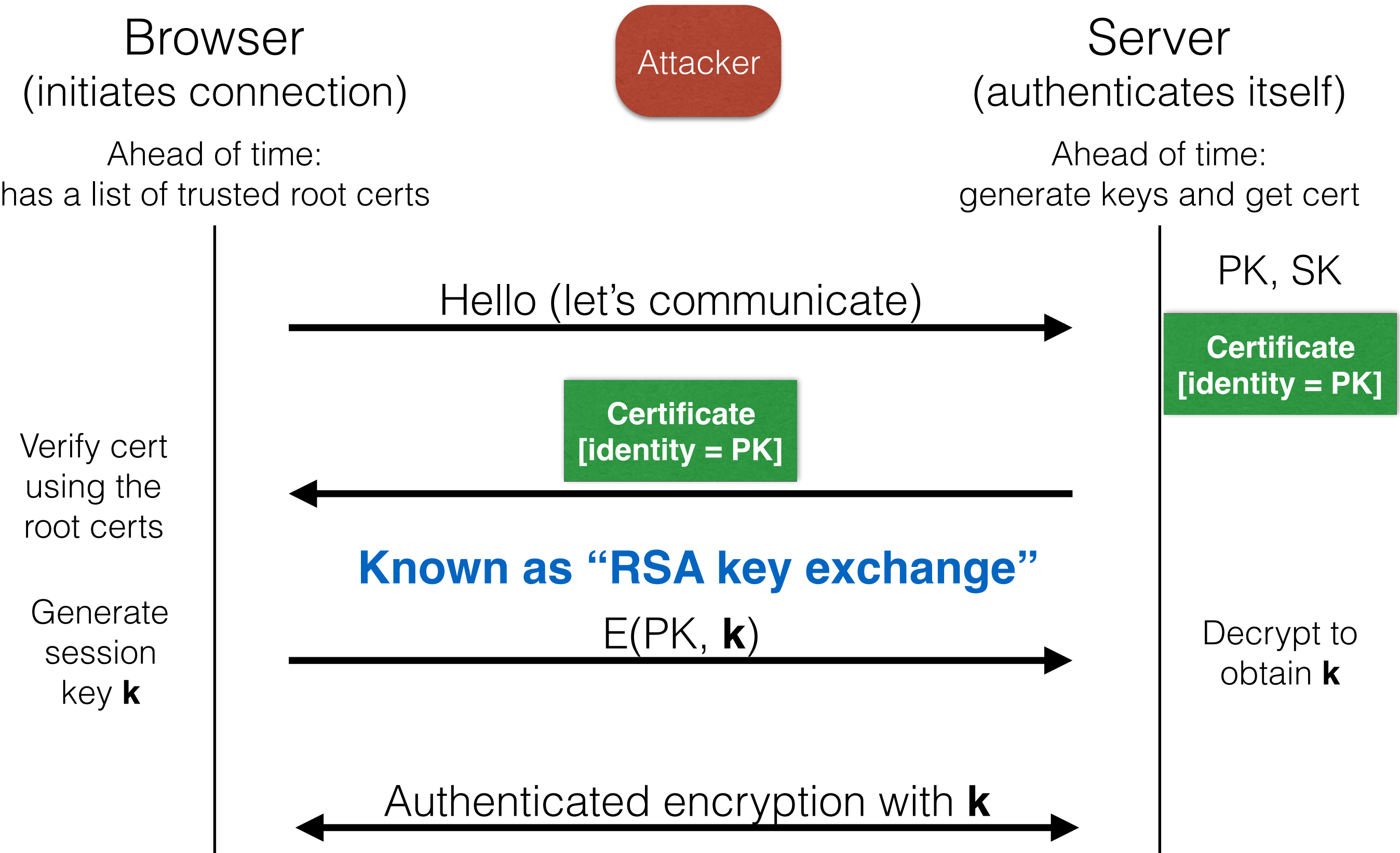
Server authentication with certificates



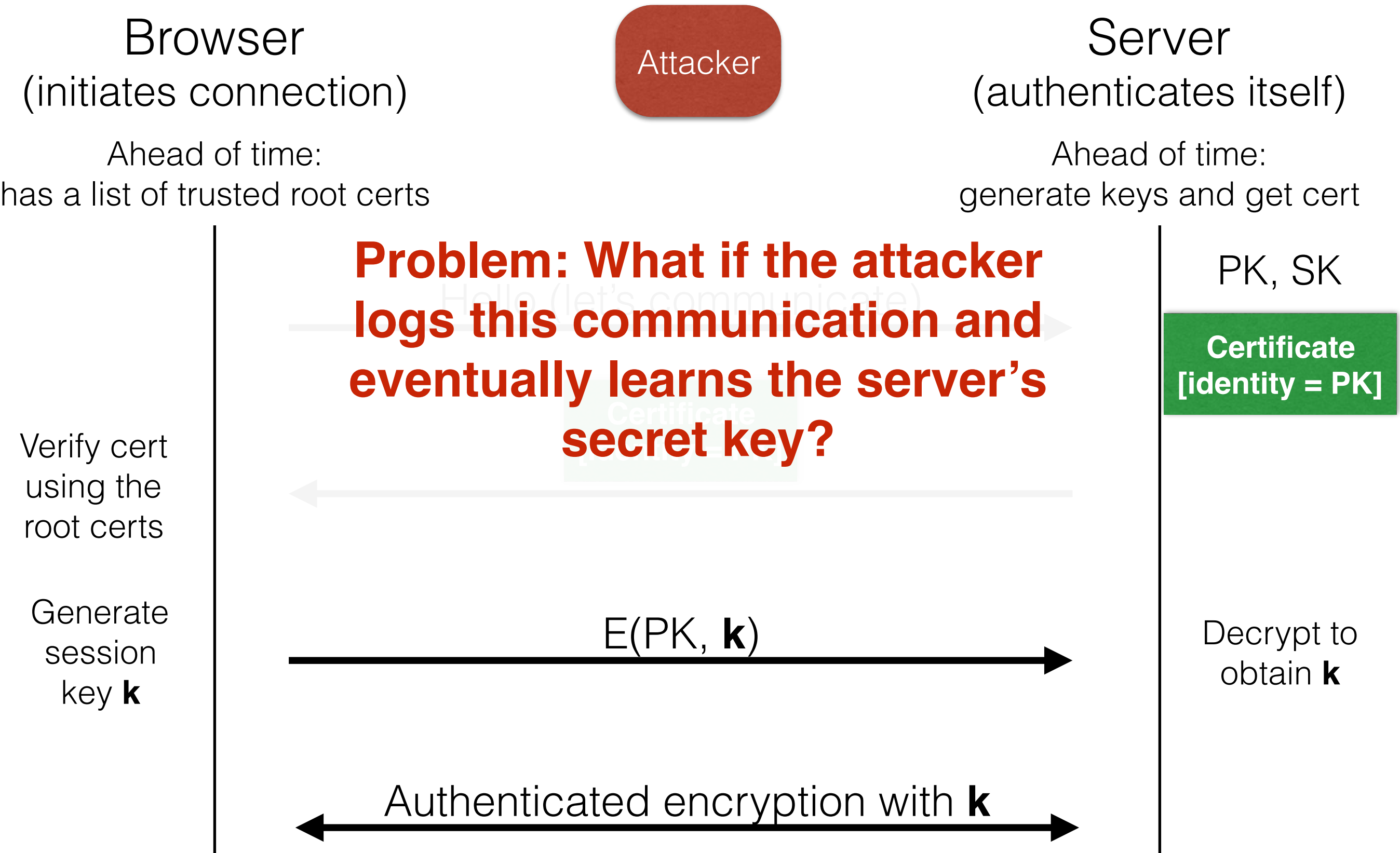
Server authentication with certificates



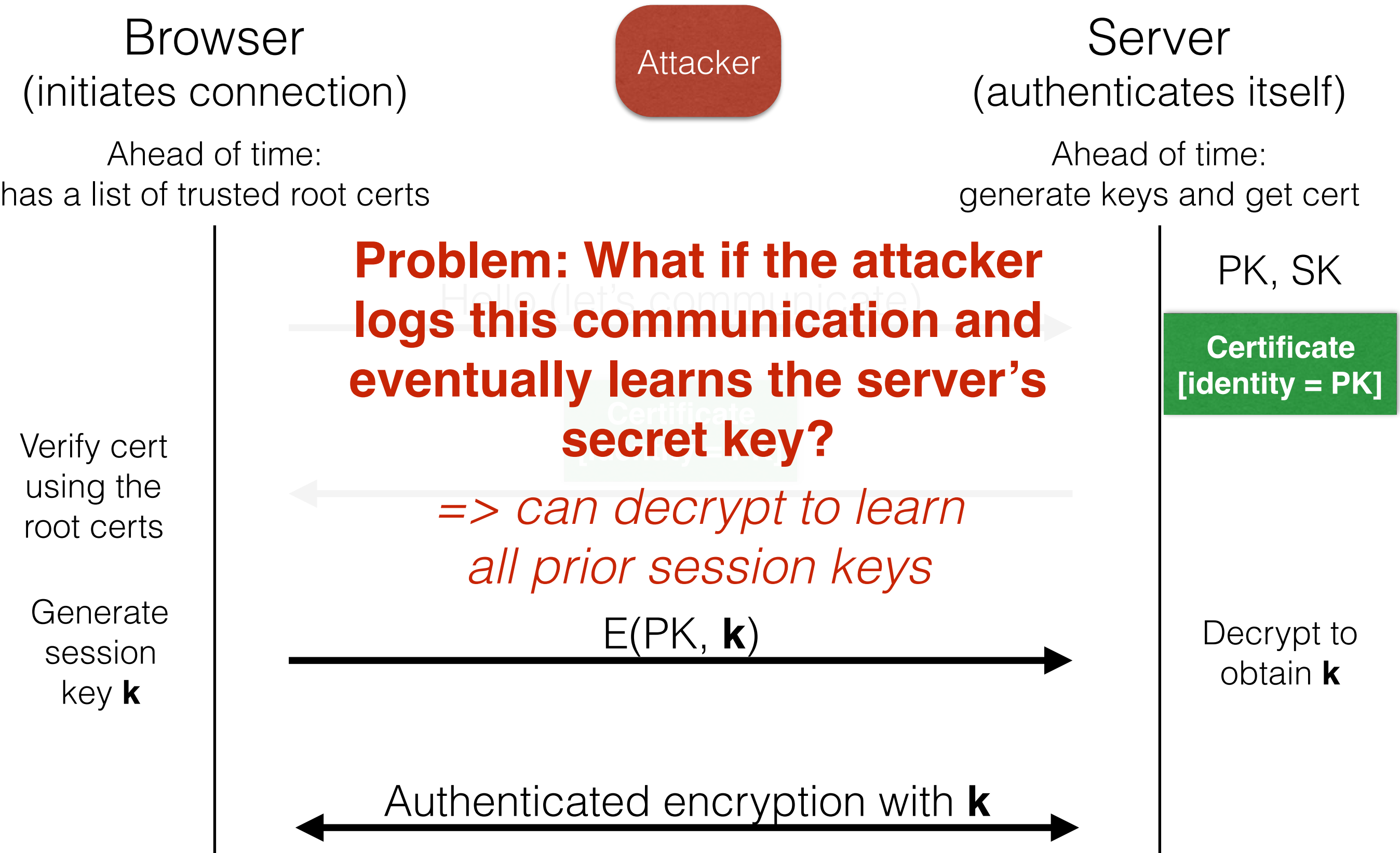
Server authentication with certificates



Server authentication with certificates



Server authentication with certificates



(Perfect) Forward Secrecy

**Compromising long-term keys
should not compromise past session keys**

Attack model

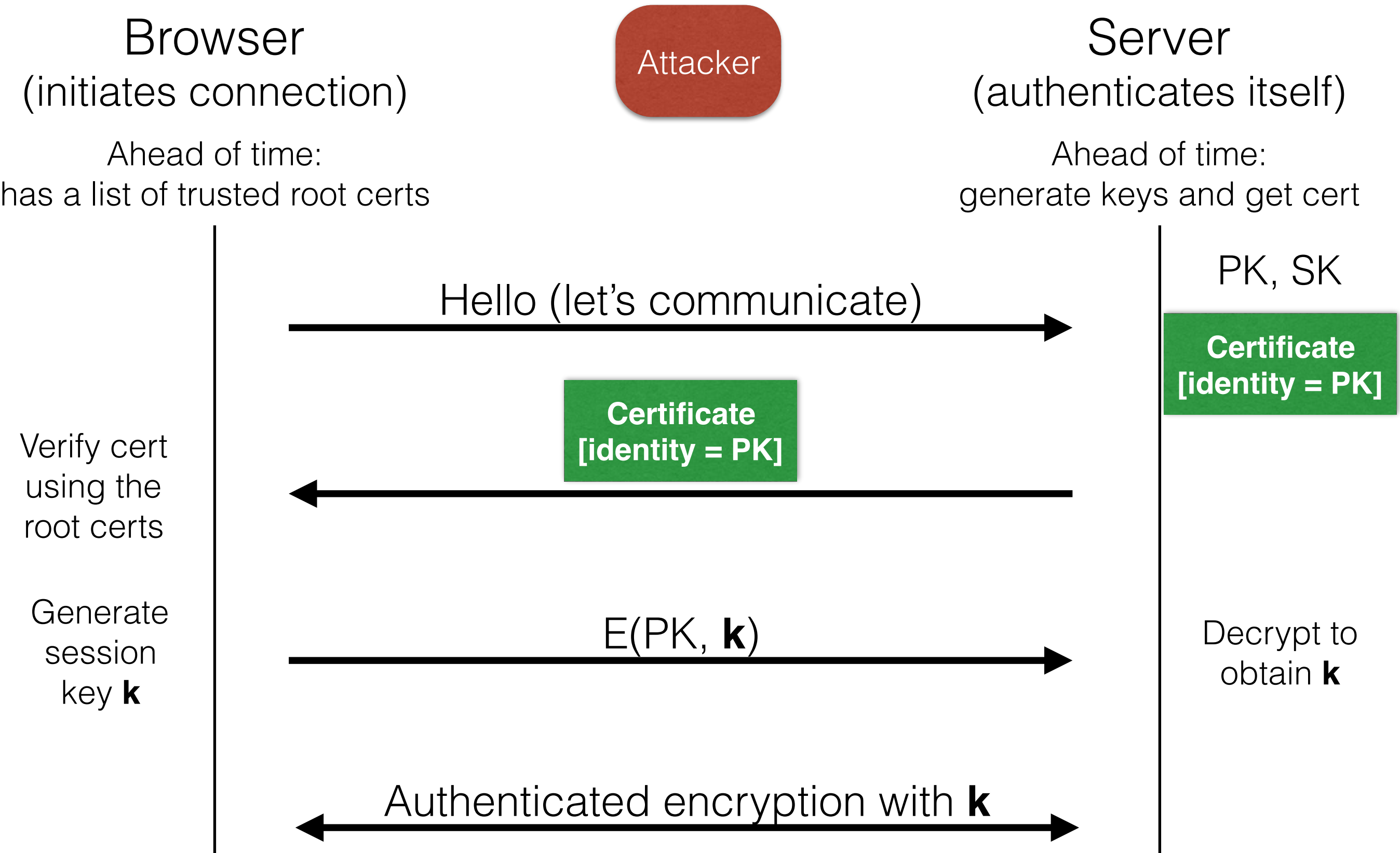
- Attacker can eavesdrop and tamper with
1. communication between client and server
(or cut the server or client out altogether)



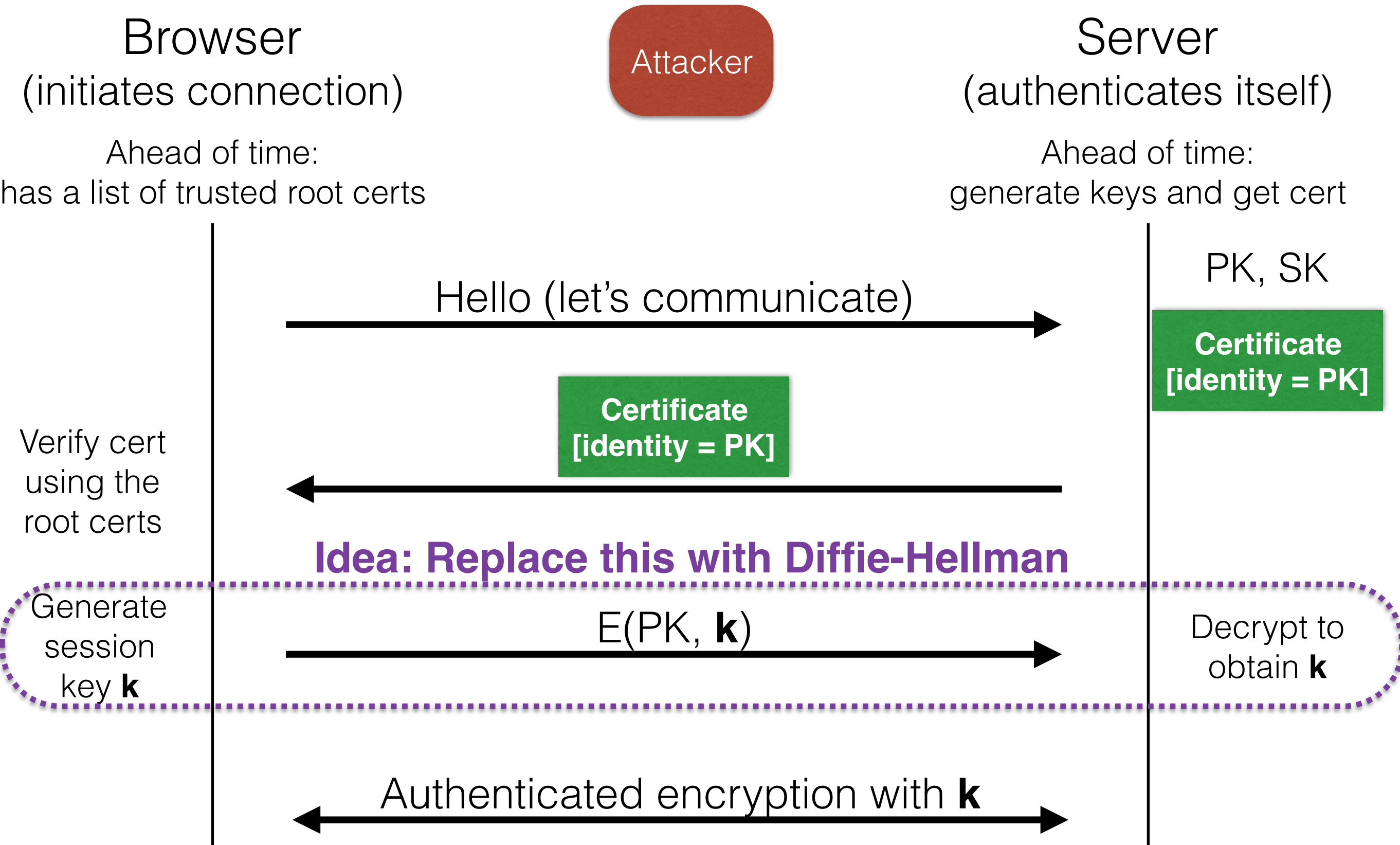
2. Attacker **subsequently** gains access to the server's private key

Attacker's goal: Learn all prior communication

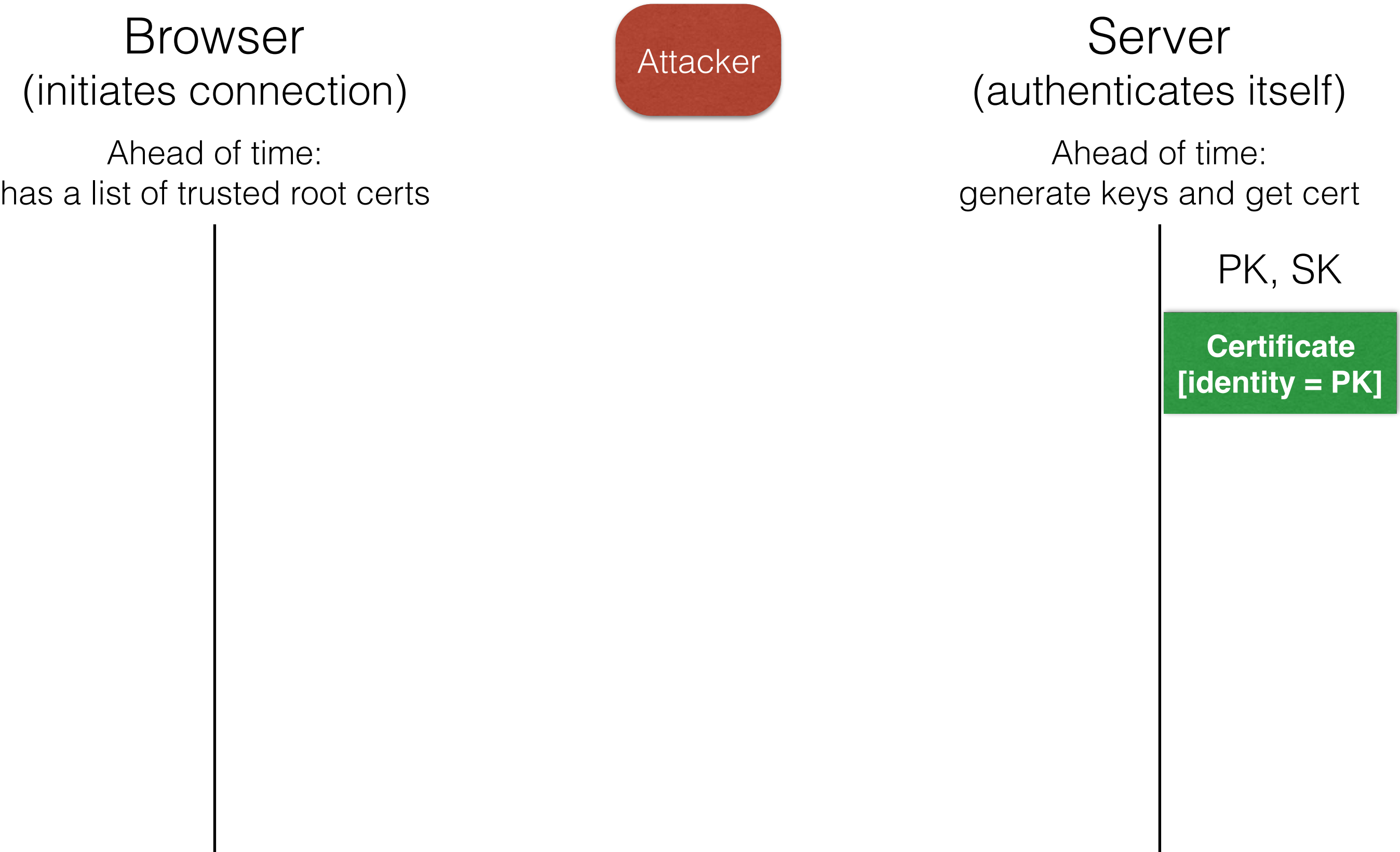
Server authentication with certificates



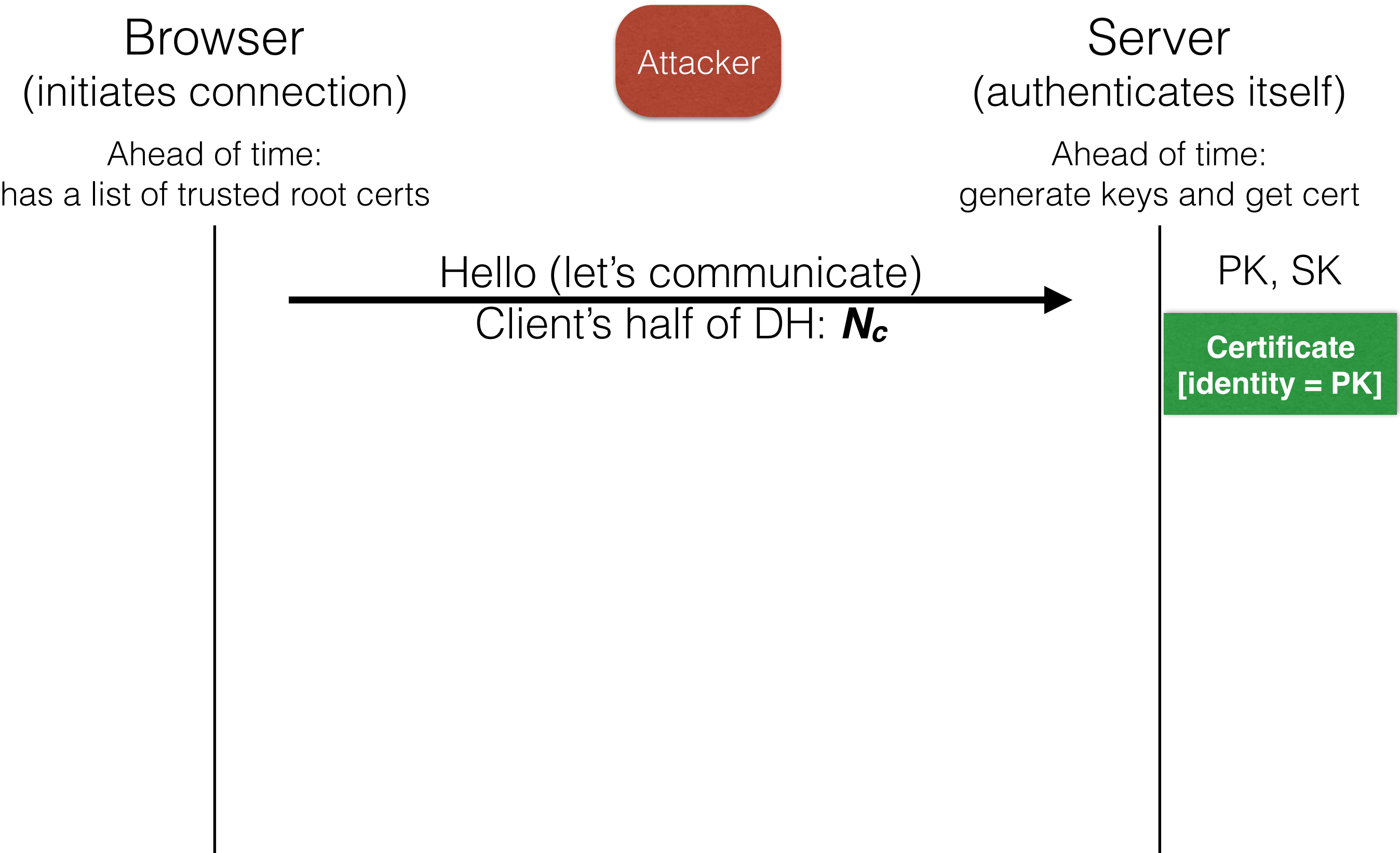
Server authentication with certificates



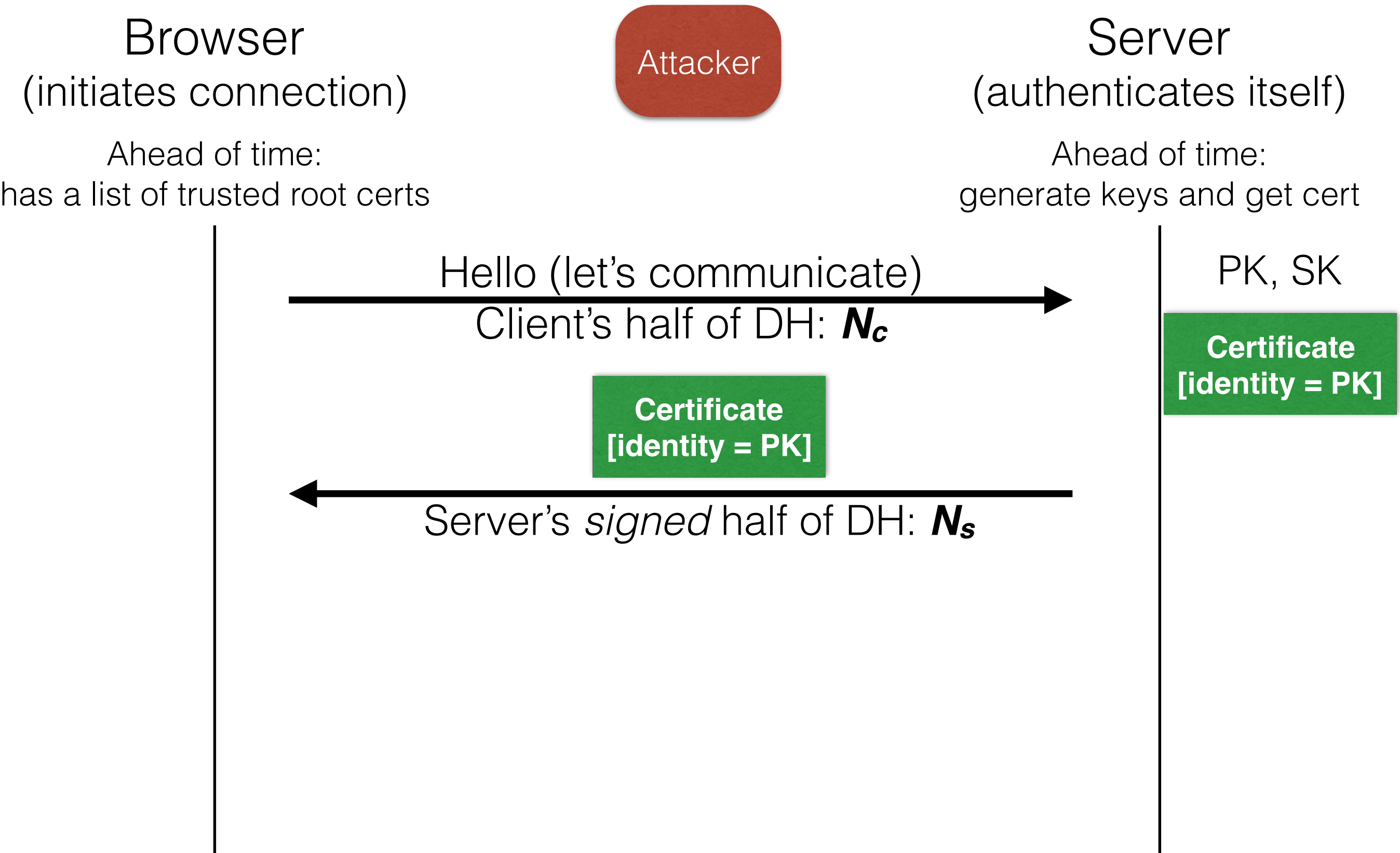
Server authentication with certificates



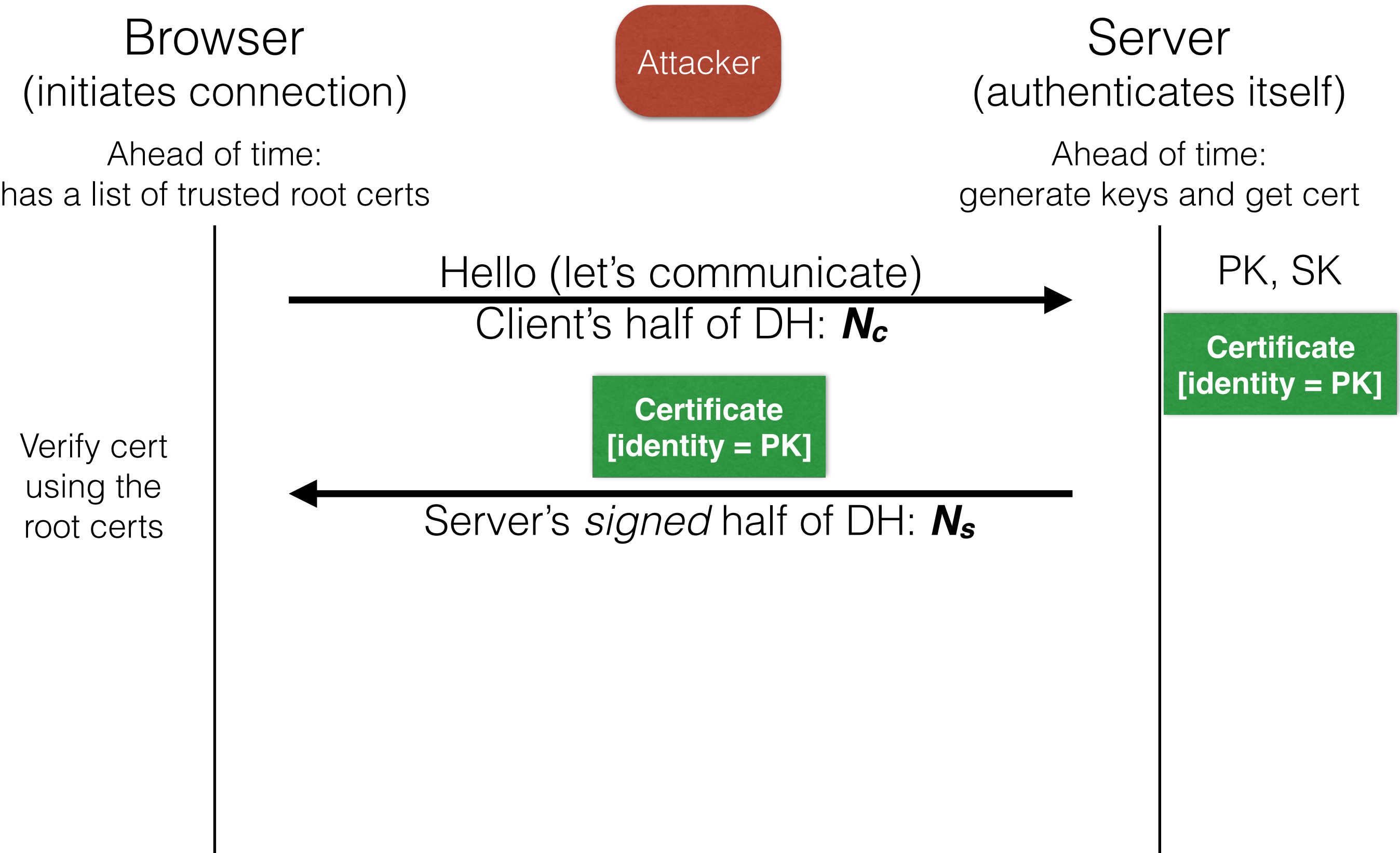
Server authentication with certificates



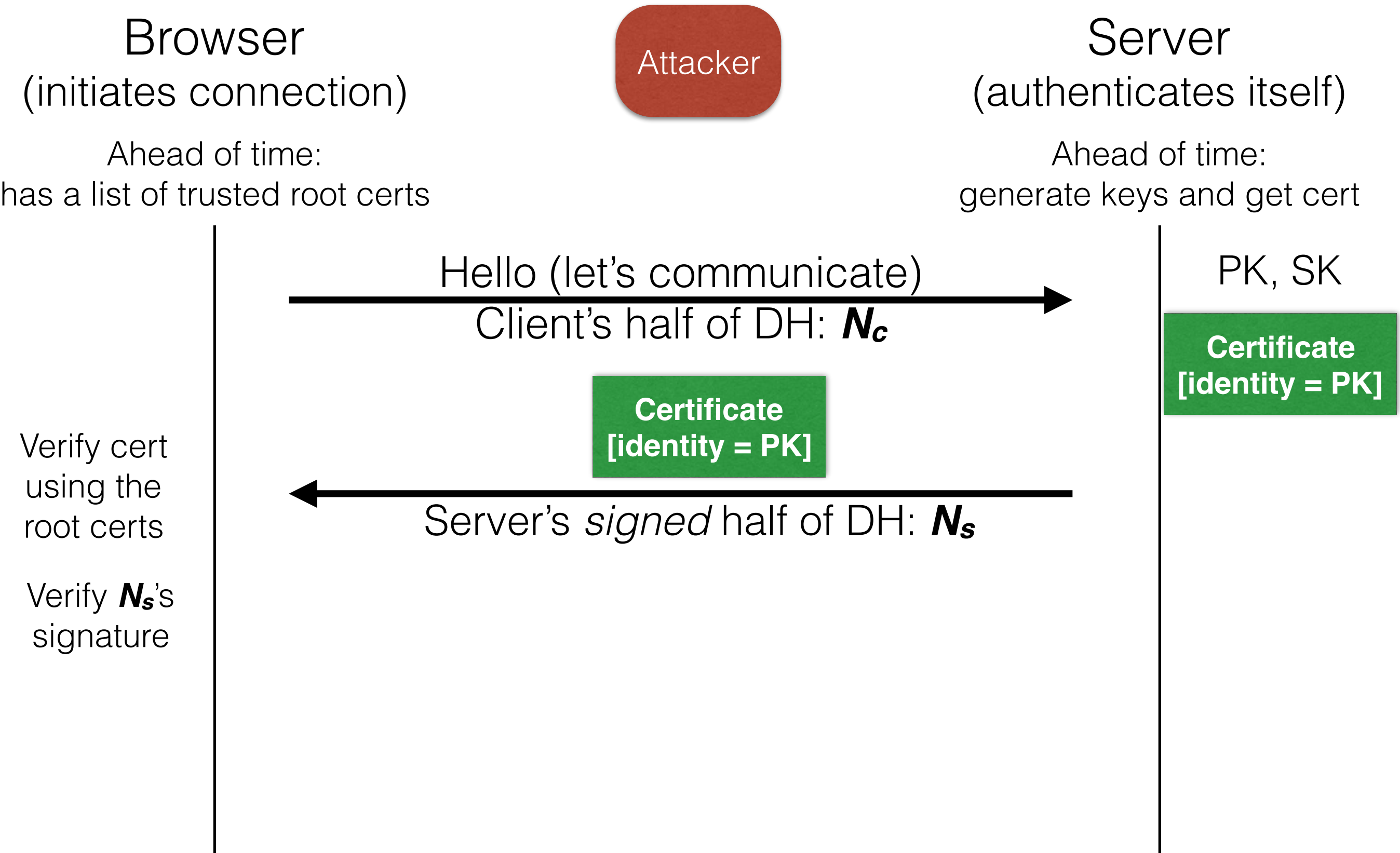
Server authentication with certificates



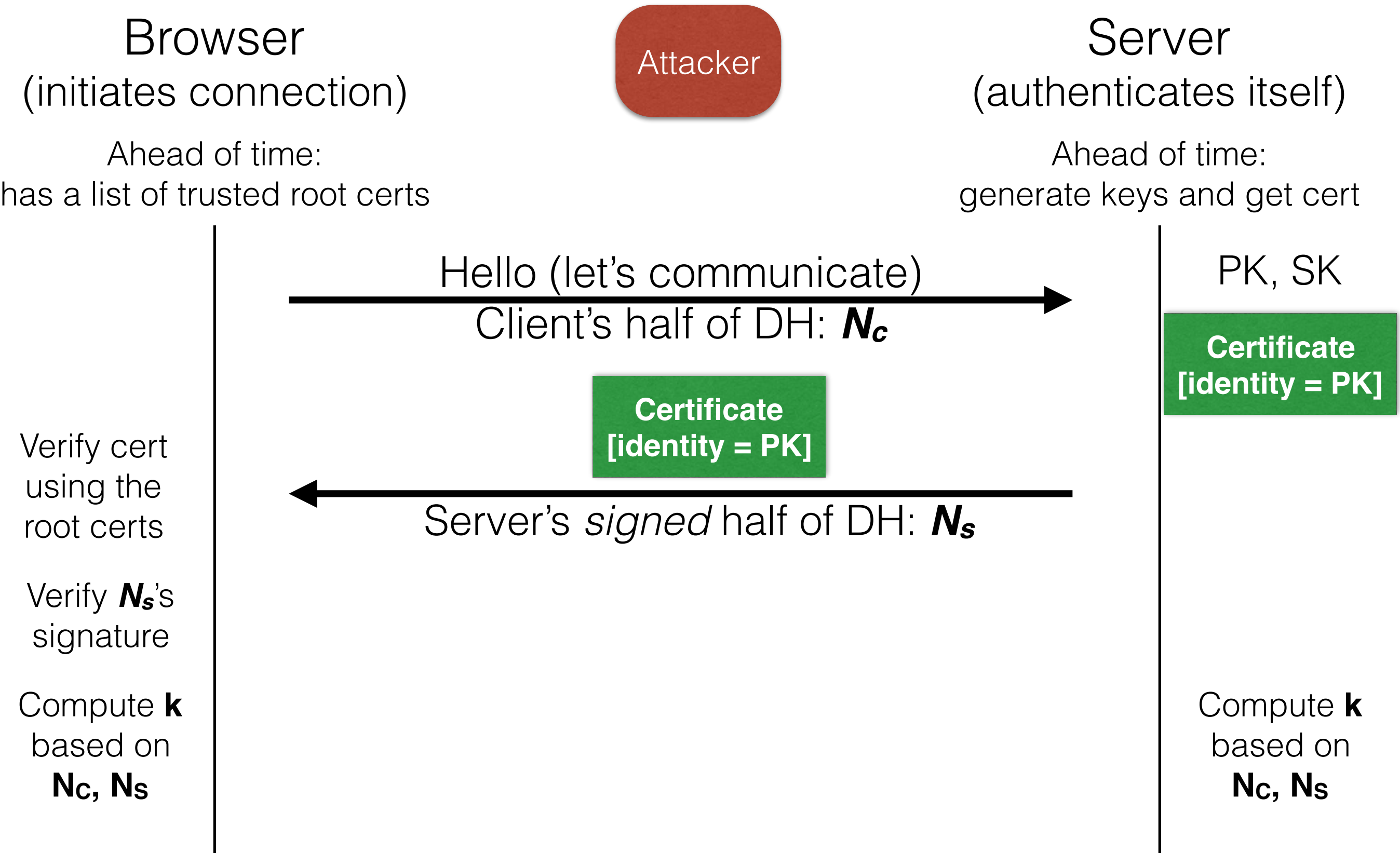
Server authentication with certificates



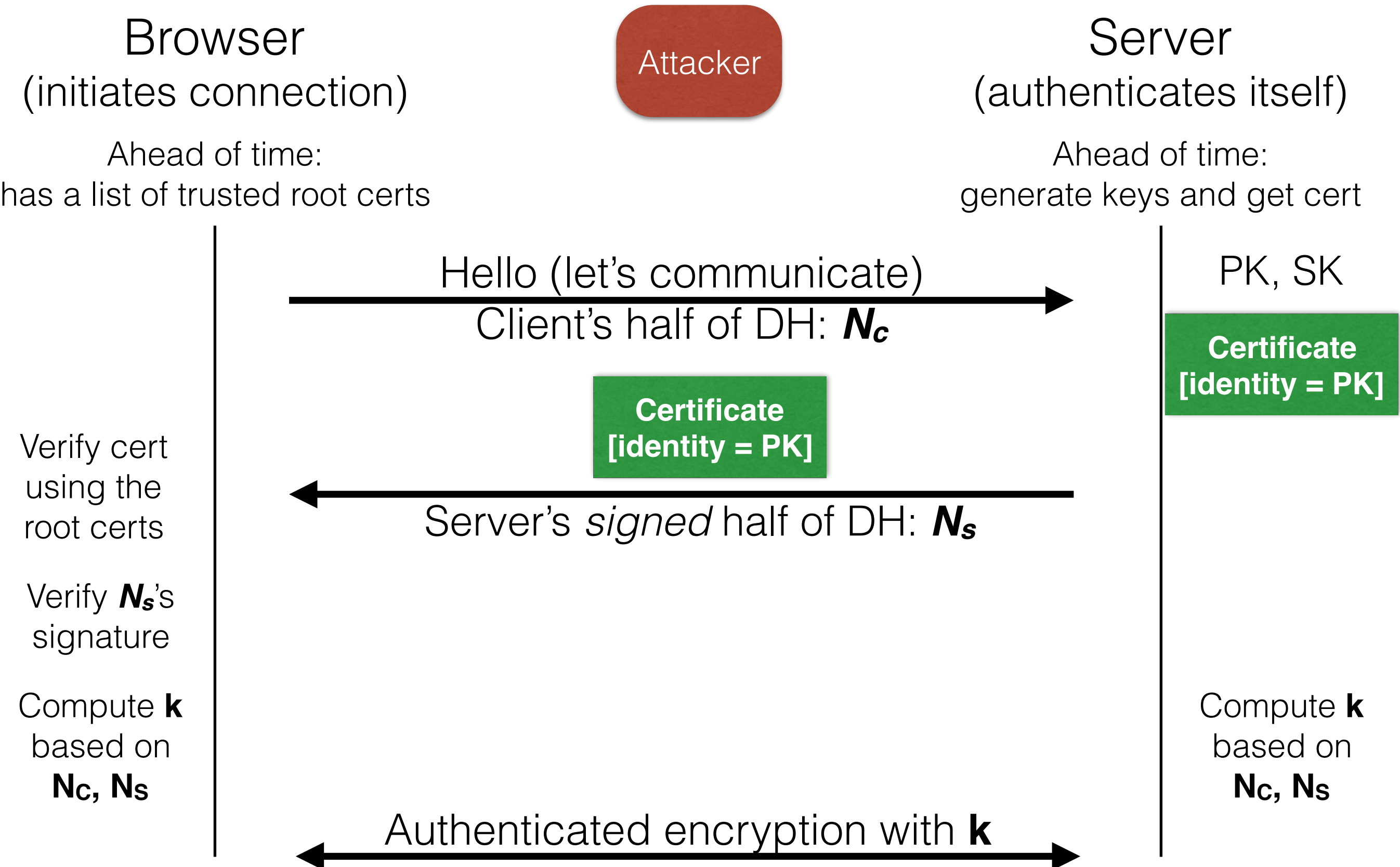
Server authentication with certificates



Server authentication with certificates



Server authentication with certificates



TLS/SSL “core” protocol (high level)

Browser

(initiates connection)

Server

(authenticates itself)

