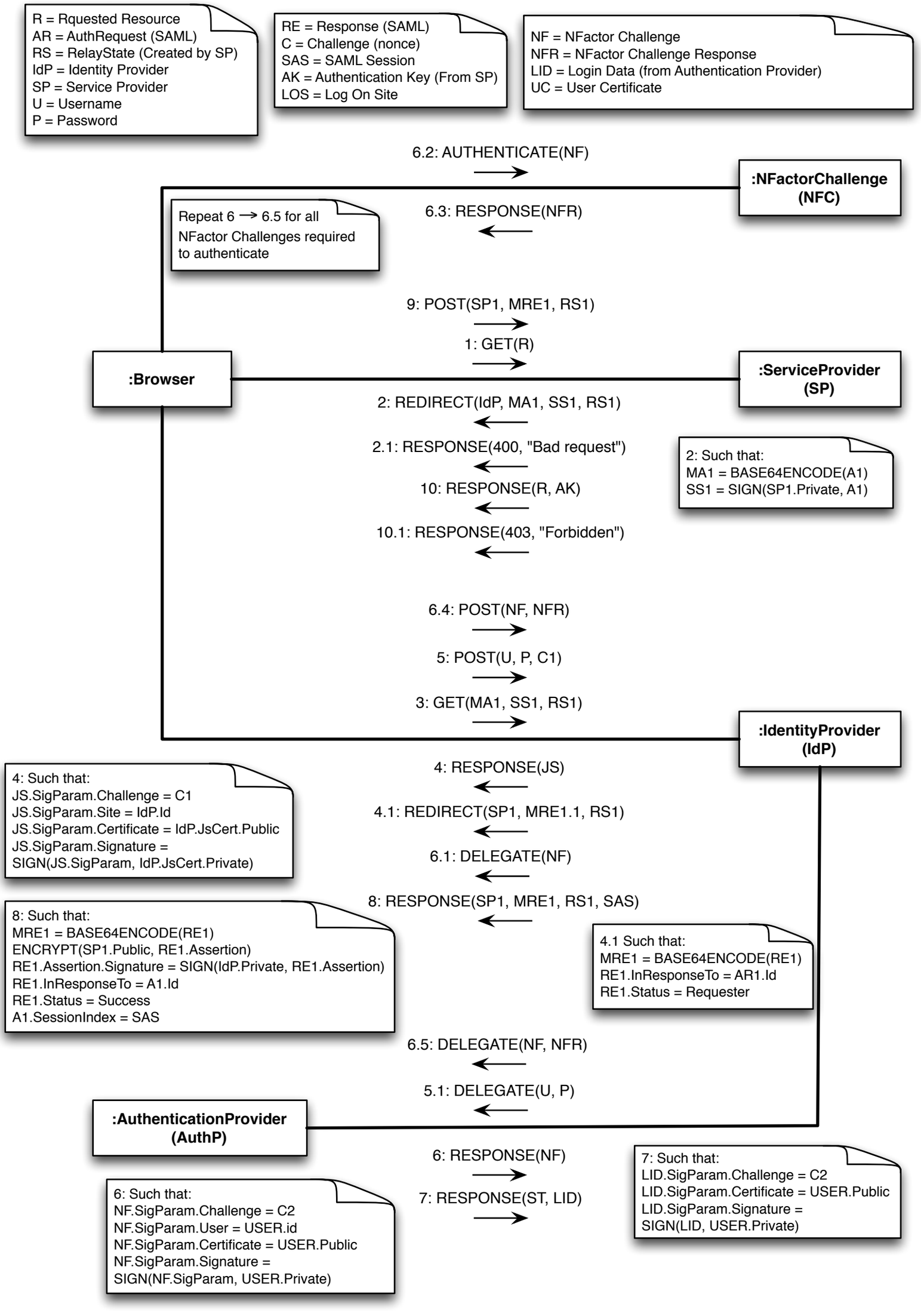
R = Rquested Resource
AR = AuthRequest (SAML)
RS = RelayState (Created by SP)
IdP = Identity Provider
SP = Service Provider
U = Username
P = Password
RE = Response (SAML)
C = Challenge (nonce)
SAS = SAML Session
AK = Authentication Key (From SP)
LOS = Log On Site
NF = NFactor Challenge
NFR = NFactor Challenge Response
LID = Login Data (from Authentication Provider)
UC = User Certificate
6.2: AUTHENTICATE(NF)
:NFactorChallenge (NFC)
6.3: RESPONSE(NFR)
Repeat 6 → 6.5 for all NFactor Challenges required to authenticate
9: POST(SP1, MRE1, RS1)
1: GET(R)
:Browser
:ServiceProvider (SP)
2: REDIRECT(IdP, MA1, SS1, RS1)
2.1: RESPONSE(400, "Bad request")
10: RESPONSE(R, AK)
10.1: RESPONSE(403, "Forbidden")
2: Such that:
MA1 = BASE64ENCODE(A1)
SS1 = SIGN(SP1.Private, A1)
6.4: POST(NF, NFR)
5: POST(U, P, C1)
3: GET(MA1, SS1, RS1)
:IdentityProvider (IdP)
4: RESPONSE(JS)
4.1: REDIRECT(SP1, MRE1.1, RS1)
6.1: DELEGATE(NF)
8: RESPONSE(SP1, MRE1, RS1, SAS)
4: Such that:
JS.SigParam.Challenge = C1
JS.SigParam.Site = IdP.Id
JS.SigParam.Certificate = IdP.JsCert.Public
JS.SigParam.Signature = SIGN(JS.SigParam, IdP.JsCert.Private)
8: Such that:
MRE1 = BASE64ENCODE(RE1)
ENCRYPT(SP1.Public, RE1.Assertion)
RE1.Assertion.Signature = SIGN(IdP.Private, RE1.Assertion)
RE1.InResponseTo = A1.Id
RE1.Status = Success
A1.SessionIndex = SAS
4.1 Such that:
MRE1 = BASE64ENCODE(RE1)
RE1.InResponseTo = AR1.Id
RE1.Status = Requester
6.5: DELEGATE(NF, NFR)
5.1: DELEGATE(U, P)
:AuthenticationProvider (AuthP)
6: RESPONSE(NF)
7: RESPONSE(ST, LID)
6: Such that:
NF.SigParam.Challenge = C2
NF.SigParam.User = USER.id
NF.SigParam.Certificate = USER.Public
NF.SigParam.Signature = SIGN(NF.SigParam, USER.Private)
7: Such that:
LID.SigParam.Challenge = C2
LID.SigParam.Certificate = USER.Public
LID.SigParam.Signature = SIGN(LID, USER.Private)