Algorithm 1 NDNOTA Consumer Side

1: **Output 1**: Handshake Authentication Request Interest (HAR) to AuthServ

2: **Output 2**: Authentication Request Interest (AR) to AuthServ

3: **Output 3**: Protected-Content Request Interest to Producer

4: Content Request Interest to Producer
5: *Data Packet* ← Producer
6: **if** *Data Packet type == 0 (Blob)* **then**
7: do nothing
8: **else if** *Data Packet type == 3 (NACK) && AuthMsg.Type = 1* **then**
9: Extract ***AuthServer list***
10: **if** *consumer has* ***no*** *AT of one of the AuthServ list* **then**
11: *AuthMsg* ← 2 (Handshake-Request State), (2, Consumer public key)
12: *ApplicationParameters* ← AuthMsg
13: *HAR Interest* ← (AuthServer Name + ApplicationParameters) & Sign Interest with Consumer Signature
14: Issue HAR to AuthServ → ***Output 1***
15: **else**
16: **go to 26**
17: **end if**
18: **end if**
19: *Data Packet* ← AuthServ
20: **switch** *Received Data Packet* **do**
21: **case** *Handshake-Reply (3) State*
22: Extract ***CT***, ***AuthServ public key***
18: *D-CT* ← Decrypt CT with Consumer private key
19: *CT* ← Encrypt D-CT with AuthServ public key
20: *AuthMsg* ← 4 (Authentication-Request State), (3, CT)
21: *ApplicationParameters* ← AuthMsg
22: *AR Interest* ← (AuthServer Name + ApplicationParameters) & Sign with Consumer Signature
23: Issue AR to AuthServ → ***Output 2***
24: **case** *Authentication-Reply (5) State*
25: Extract & save ***AT***
26: *AuthMsg* ← 6 (AuthToken State), (1, AuthServ), (4, AT)
27: *ApplicationParameters* ← AuthMsg
28: *Content Request Interest* ← (Producer Name + ApplicationParameters) & Sign with Consumer Signature
29: Issue Content Request Interest to Producer → ***Output 3***

Algorithm 2 NDNOTA Producer Side

1: **Input 1**: Content Request Interest from Consumer

2: **Output 1**: Is-Consumer-Authenticated Interest to AuthServ

3: **Output 2**: Data Packet to Consumer

4: **if** *ApplicationParameters = Empty* && *requested content!= protected* **then**
5: Issue Data packet that carries requested content
6: **else**
7: **if** *ApplicationParameters = Empty* && *requested content = protected* **then**
8: *AuthMsg* ← 1 (Redirect State), (1, list of AuthServs)
9: *Content* ← AuthMsg
10: *NACK Data Packet* ← (Content) & Sign with Producer Signature
11: **else if** *AuthMsg.Type = 6 && AuthMsg.Parameter[1].key = 4* **then**
12: Extract ***AT***, ***consumer key-locator***
13: **if** *consumer key-locator is cached* **then**
14: **if** *received AT = cached AT* && *NOT expired* **then**
15: **go to 27**
16: **else**
17: *AuthMsg* ← 7 (Is-Consumer-Authenticated State), (2, consumer key-locator), (4, AT)
18: *ApplicationParameters* ← AuthMsg
19: *Is-Consumer-Authenticated Interest* ← (AuthServer Name + ApplicationParameters) & Sign with Producer Signature
20: Issue Is-Consumer-Authenticated Interest to AuthServ → ***Output 1***
21: **end if**
22: **end if**
23: **end if**
24: *Data Packet* ← AuthServ
25: **if** *AuthMsg.Type = 8 && AuthMsg.Parameter[0].value = True* **then**
26: Store ***consumer key-locator*** & ***AT***
27: *Data Packet* ← (Protected Content) & Sign with Producer Signature
28: Issue Data Packet with protected content to Consumer → ***Output 2***
29: **else**
30: Issue NACK Data Packet to Consumer → ***Output 2***
31: **end if**

Algorithm 3 NDNOTA AuthServ Side

1: **Input 1**: Handshake Authentication Request Interest (HAR) from Consumer

2: **Output 1**: Handshake Reply (HR) Data Packet to Consumer

3: **Output 2**: Authentication Reply (AR) Data Packet to Consumer

4: **Output 3**: Authenticate Verification Data Packet to Producer

5: **if** *AuthMsg.Type* = *2* **then**
6: Extract ***consumer public key, key-locator***
7: *CT* ← Generate & encrypt a random token with consumer public key
8: *entry* ← Consumer key-locator, Plaintext CT, CT Expiry
9: *AuthMsg* ← 3 (Handshake-Reply State), (3, CT)
10: *Content* ← AuthMsg
11: *HR Data Packet* ← (Content) & Sign with AuthServ Signature
12: Issue HR Data Packet to Consumer → ***Output 1***
13: **end if**
14: **switch** *Received Interest Packet* **do**
15: **case** *Authentication-Request (4) State*
16: Extract ***CT***, c***onsumer key-locator***
17: *D-CT* ← Decrypt CT with AuthServ private key
18: **if** *D-CT* = *entry (CT)* **then**
19: *AT* ← Generate & encrypt a random token with AuthServ key
20: *entry* ← Consumer key-locator, AT, AT Expiry
21: *AuthMsg* ← 5 (Authentication-Reply State), (4, AT)
22: *Content* ← AuthMsg
23: *AR Data Packet* ← (Content) & Sign with AuthServ Signature
24: Issue AR Data Packet to Consumer → ***Output 2***
25: **else**
26: Issue NACK Data Packet to Consumer
27: **end if**
28: **case** *Is-Consumer-Authenticated (7) State*
29: Extract ***AT***, ***consumer key-locator***
30: **if** *AT* = *entry (AT)* **then**
31: *AuthMsg* ← 8 (Authenticated State), (5, True)
32: *Authenticate Confirmation Data Packet*← (Content) & Sign with AuthServ Signature
33: Issue Authenticate Confirmation Data Packet to Producer → ***Output 3***
34: **else**
35: *AuthMsg* ← 8 (Authenticated State), (5, False)
36: **end if**
37: *Content* ← AuthMsg
38: *Authenticate Verification Data Packet*← (Content) & Sign with AuthServ Signature
39: Issue Authenticate Confirmation Data Packet to Producer → ***Output 3***