

Algorithm 1: Quantum-resistant ECDSA Key Generation $KeyGen_q$.

```

1 begin
2    $G \leftarrow$  ECC base point;  $P \leftarrow$  ECC order;
3   Generate secret pre-image  $\rho$ ;
4   Compute private key  $K_s = H(\rho)$ ;
5   Compute public key  $K_p = (x, y)$  where  $K_p \equiv K_s \cdot G \bmod P$ ;
6   destroy  $K_s$ ;
7   return  $\rho, K_p$ ;
8 end

```

100

FIG. 1**Algorithm 2:** Quantum-resistant ECDSA signing $Sign_q$.

```

1 begin
2    $G \leftarrow$  ECC base point;  $P \leftarrow$  ECC order;
3    $\rho \leftarrow$  secret pre-image;
4    $M \leftarrow$  message;
5   Generate signature random  $r$ ;
6   Compute  $r^{-1}$  where  $r * r^{-1} \equiv 1 \bmod P$ ;
7   Compute  $R = (R_x, R_y)$  where  $R \equiv r \cdot G \bmod P$ ;
8   Compute hash of message  $H(M)$ ;
9   Enumerate ZKBoo proof  $\pi =$  begin
10    Zero-knowledge computation of private key  $K_s$  where  $K_s = H(\rho)$ ;
11    Zero-knowledge computation of public key  $K_p$  where
12      $K_p \equiv K_s \cdot G \bmod P$ ;
13    Compute  $s$  where  $s \equiv r^{-1} * (H(M) + R_x * K_s) \bmod P$ ;
14    Commit  $R_x, s$  in the proof;
15  end
16  destroy  $r, r^{-1}, K_s$ ;
17  return  $\sigma = \{R_x, s\}, \pi$ ;

```

200

FIG. 2

Algorithm 3: Quantum-resistant ECDSA verification $Verify_{y_q}$.

```

1 begin
2    $G \leftarrow$  ECC base point;  $P \leftarrow$  ECC order;
3    $K_p \leftarrow$  ECC public key;
4    $R_x, s \leftarrow$  signature  $\sigma$ ;  $\pi \leftarrow$  proof;  $M \leftarrow$  message;
5   Compute  $s^{-1}$  where  $s * s^{-1} \equiv 1 \bmod P$ ;
6   Compute hash of message  $H(M)$ ;
7   Compute  $u_1 = s^{-1} * H(M) \bmod P$ ;
8   Compute  $u_2 = s^{-1} * R_x \bmod P$ ;
9   Compute  $V = (V_x, V_y)$  where  $V = u_1 \cdot G + u_2 \cdot K_p \bmod P$ ;
10  if  $V_x \neq R_x$  then
11    | return "Failed Signature Verification"
12  end
13  else
14    | Verify ZKBoo proof  $\pi =$  begin
15    |   | Check that  $R_x, s$  is committed in the proof;
16    |   | Check that zero-knowledge computation of  $K_p$  from unknown
17    |   | pre-image is correct;
18    |   | if Check Failed then
19    |   |   | return "Failed Proof Verification"
20    |   | end
21    | end
22  return success;
23 end

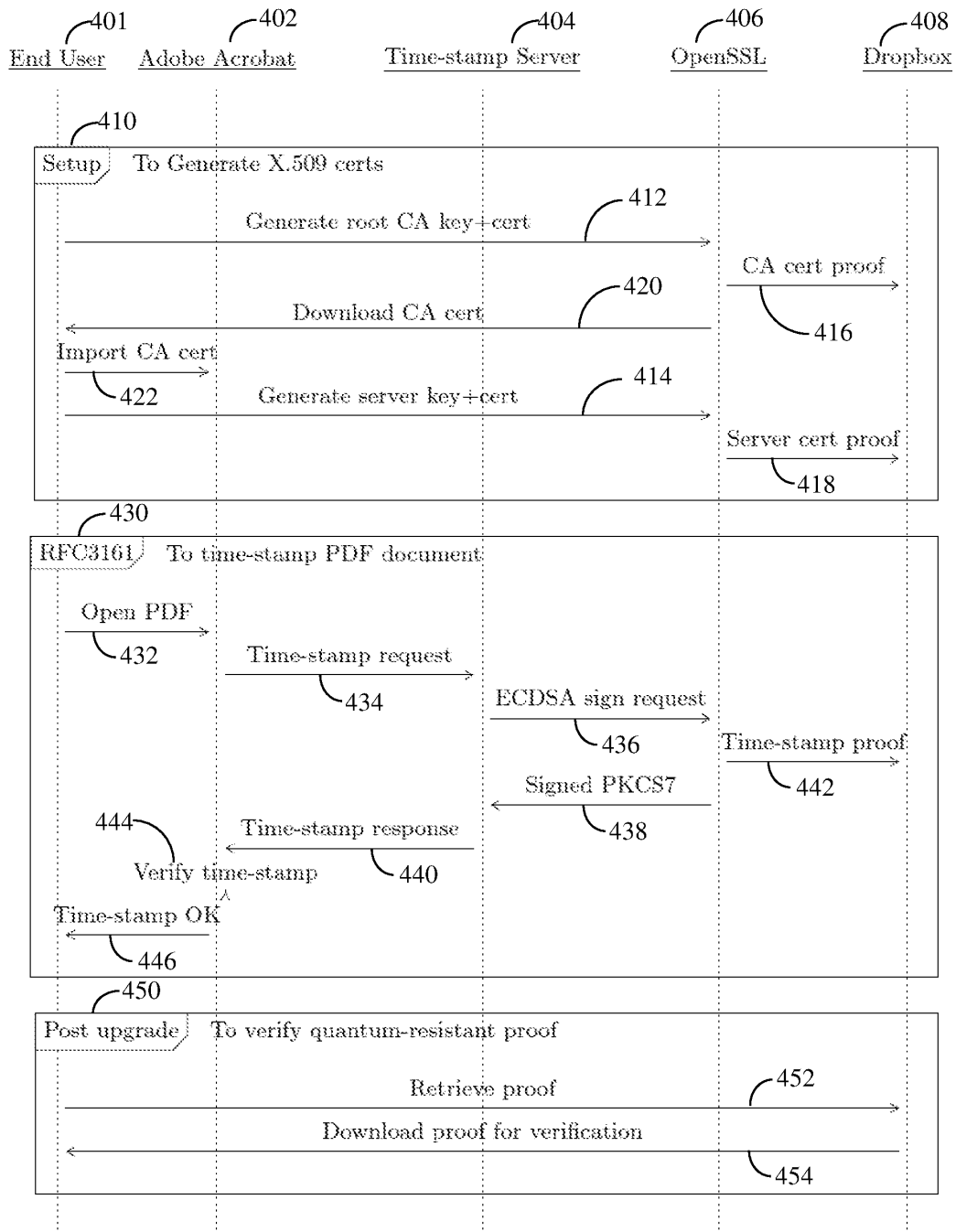
```

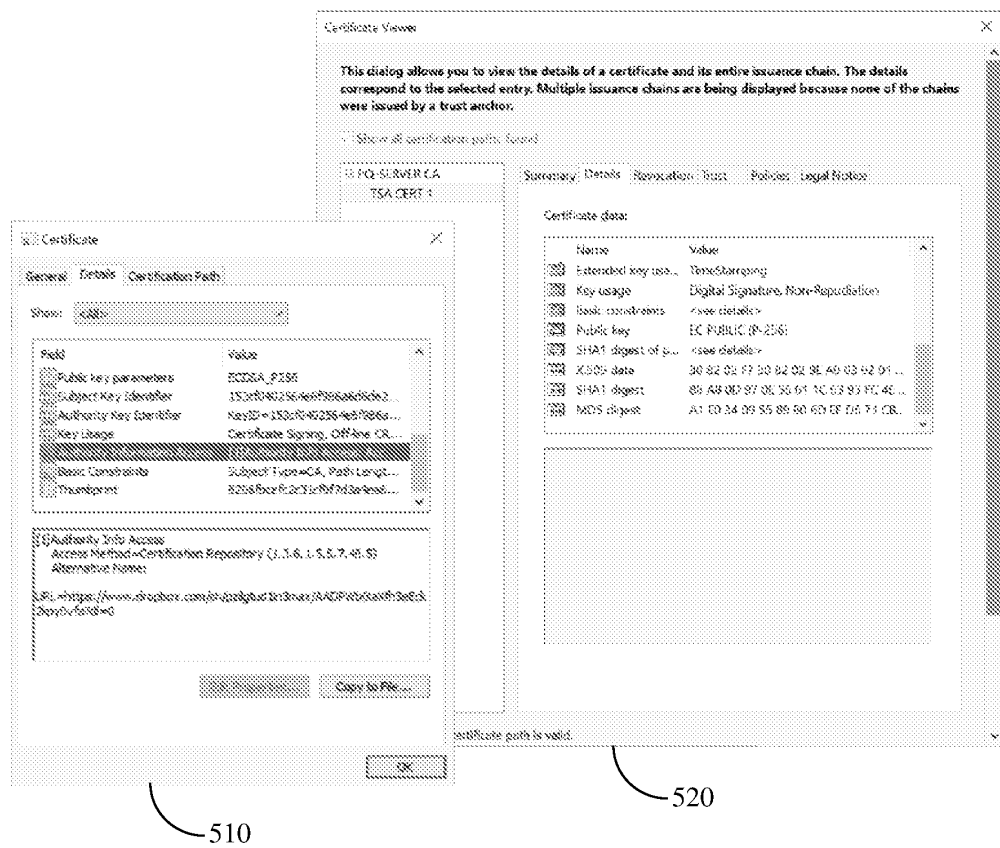
} 310

} 320

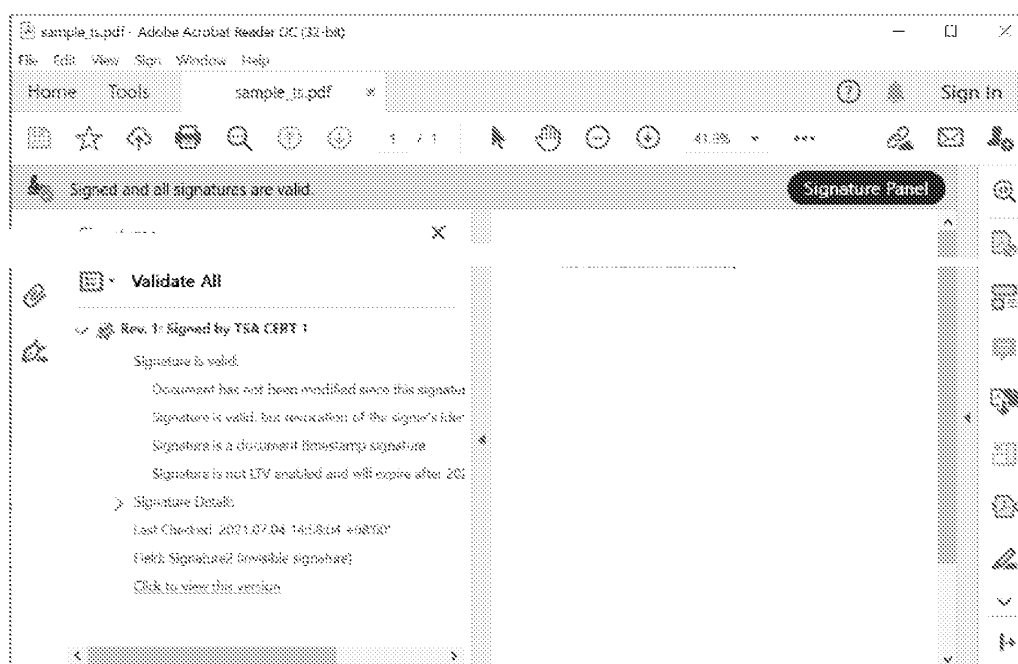
300

FIG. 3

400 **FIG. 4**



500
FIG. 5

600**FIG. 6**