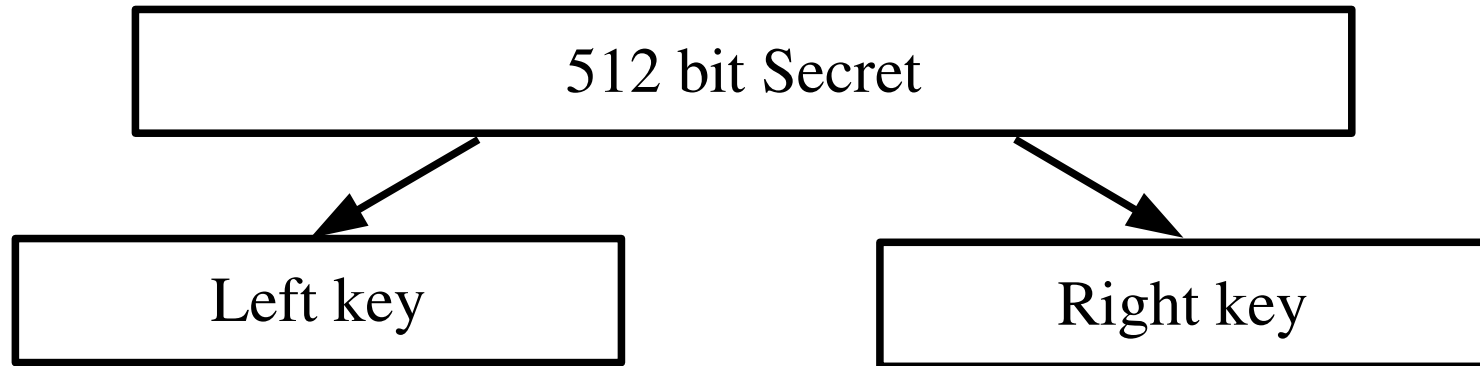


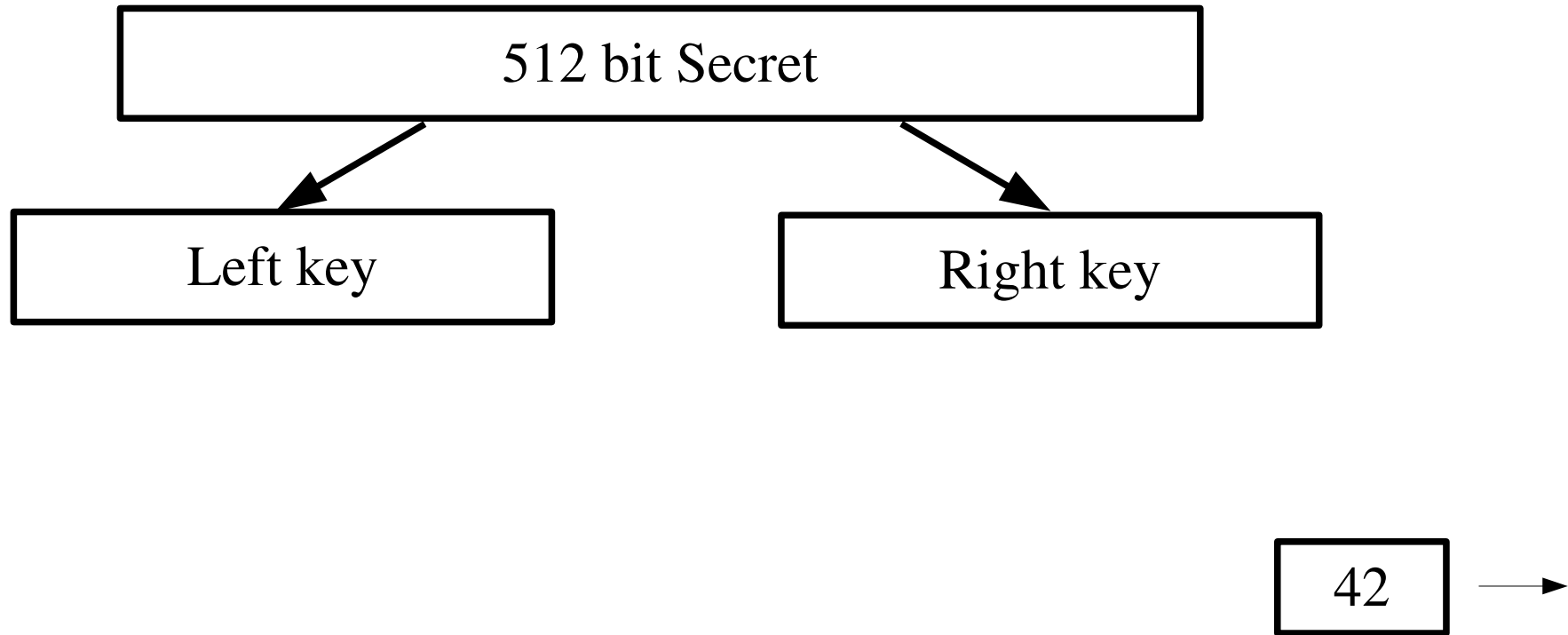
Monitor's Secret Key Crypto - KARN, encrypt

512 bit Secret

Monitor's Secret Key Crypto - KARN, encrypt

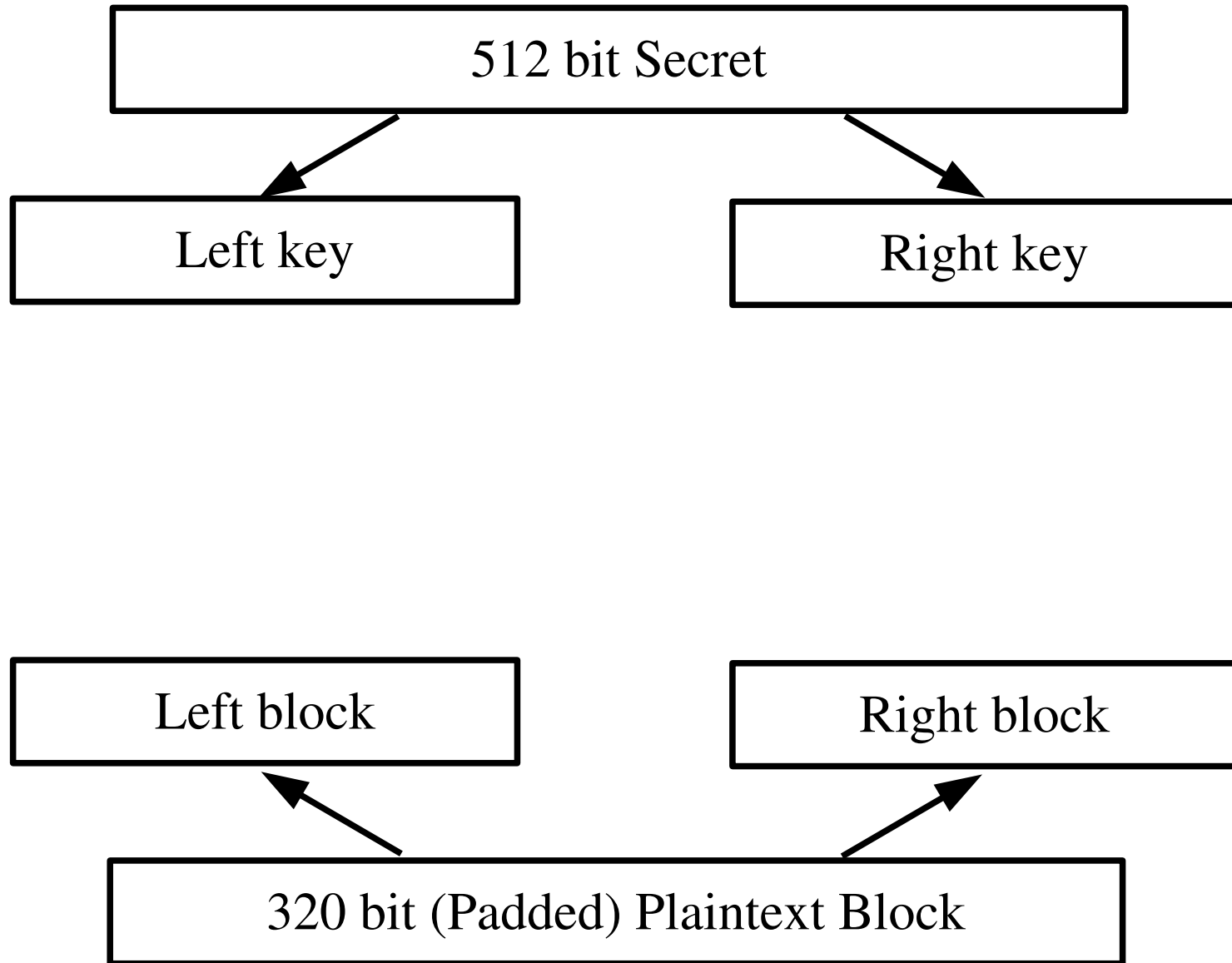


Monitor's Secret Key Crypto - KARN, encrypt



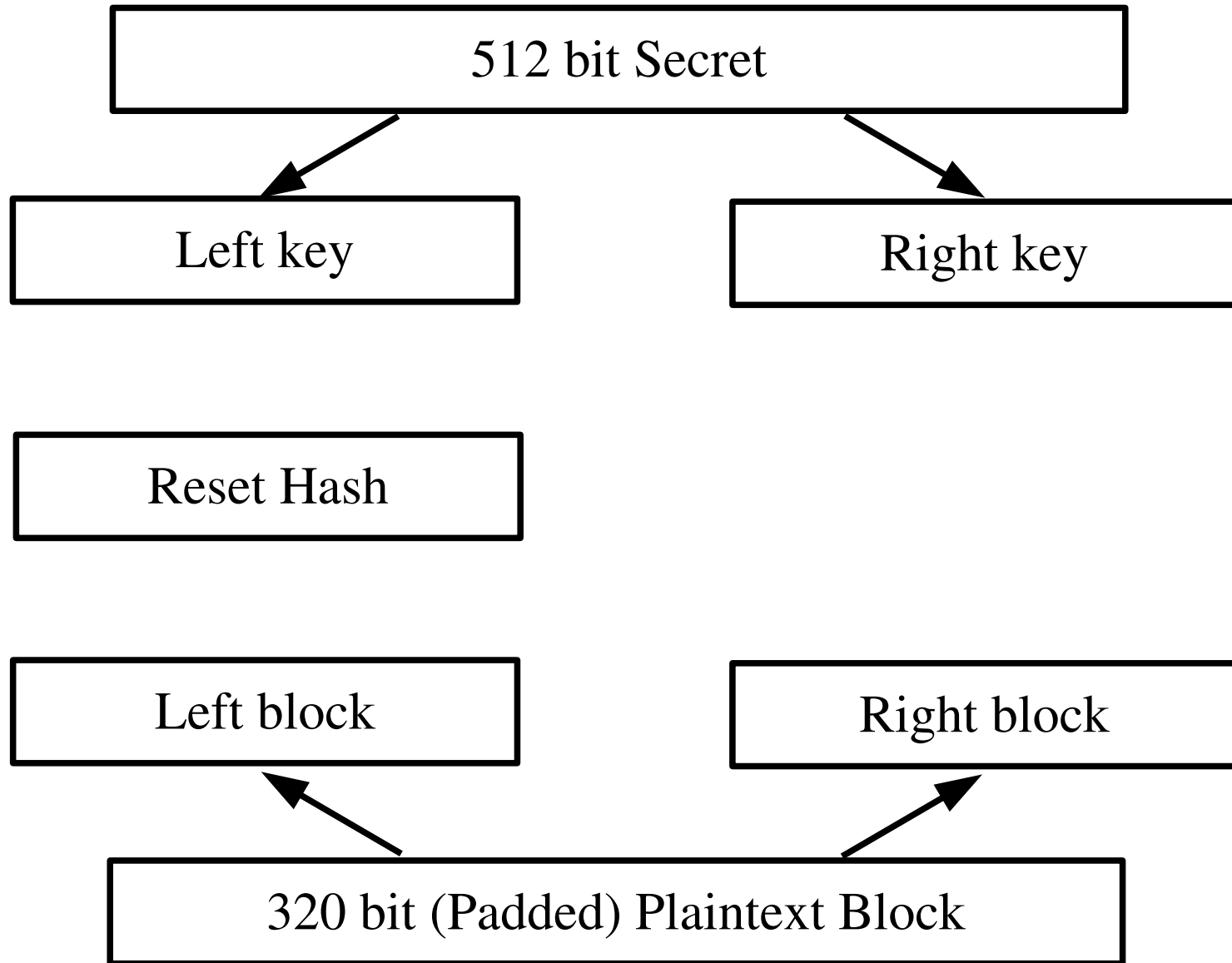
First: send out "guard byte" - the number 42 (00101010)

Monitor's Secret Key Crypto - KARN, encrypt



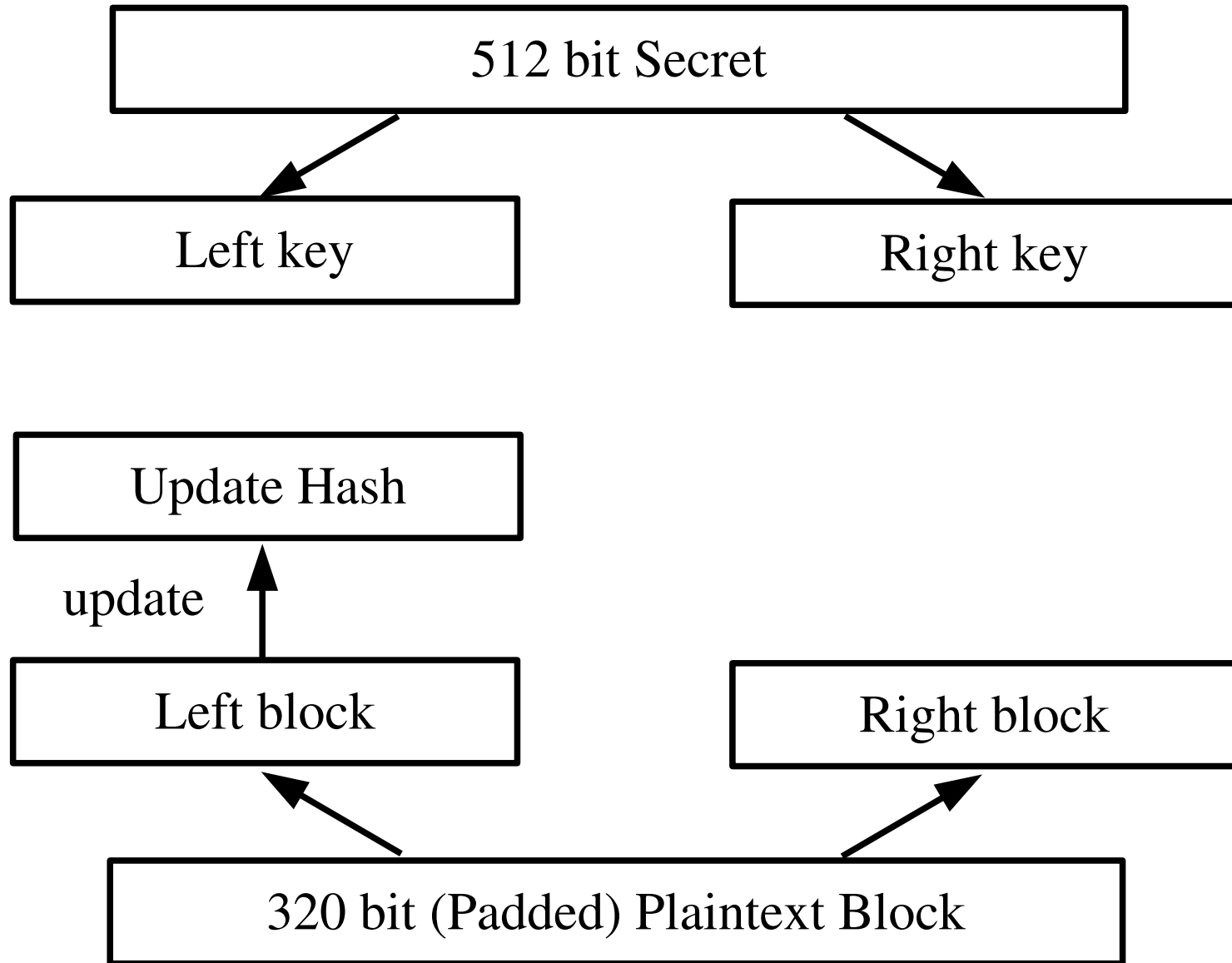
Split plaintext block into two halves

Monitor's Secret Key Crypto - KARN, encrypt



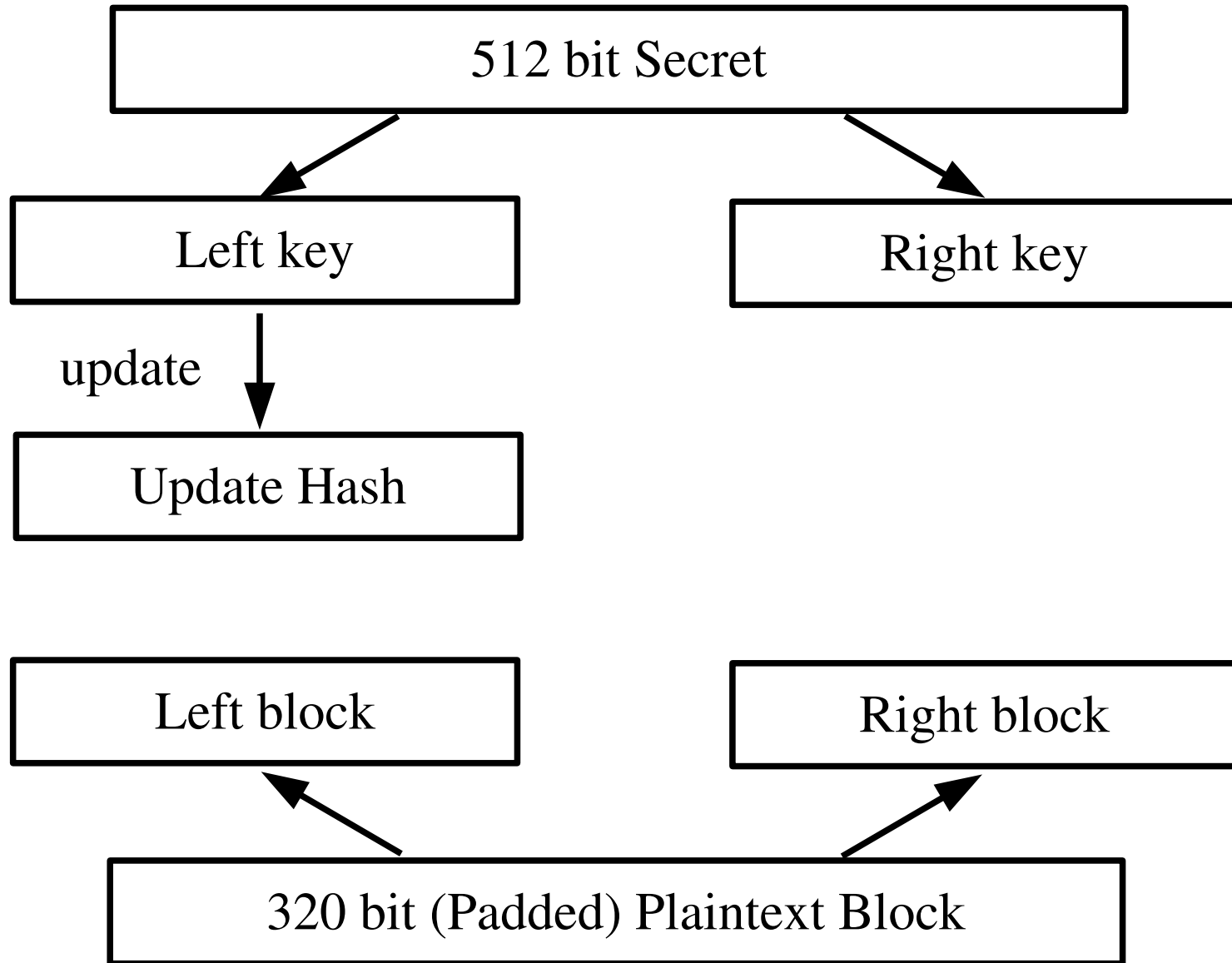
Reset a SHA message digest

Monitor's Secret Key Crypto - KARN, encrypt



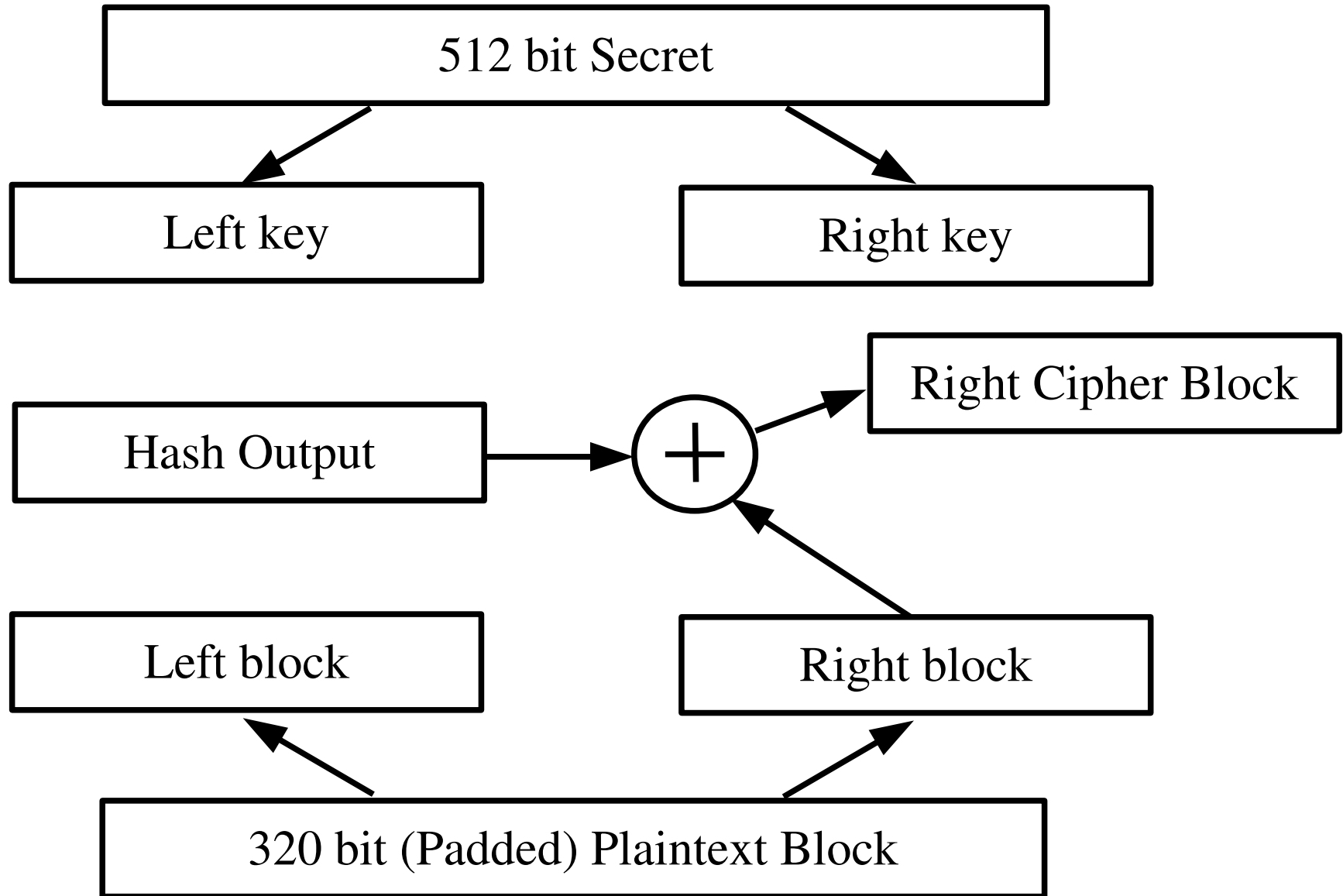
Hash sequence: left plaintext block then left key

Monitor's Secret Key Crypto - KARN, encrypt



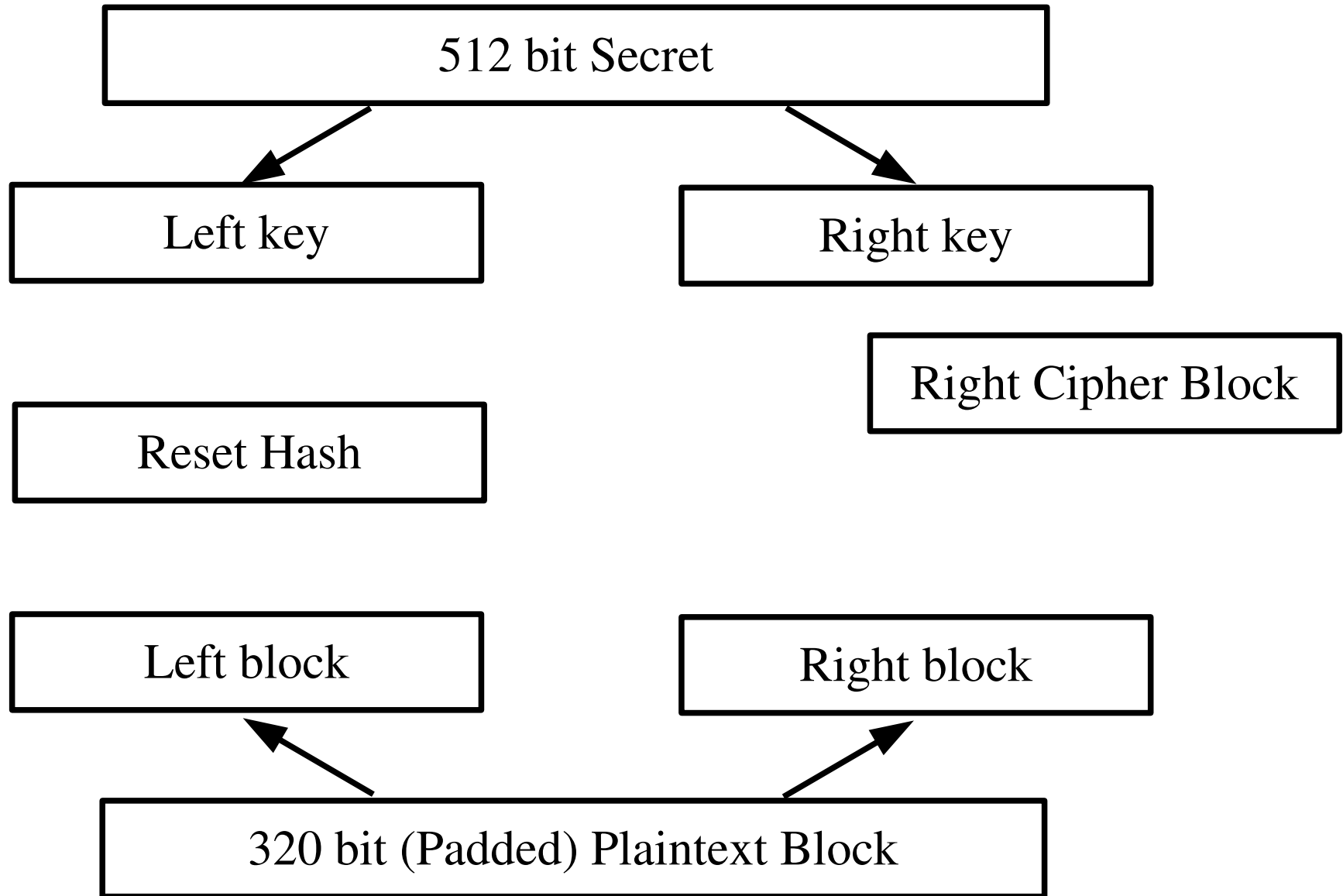
Hash sequence: left plaintext block then left key

Monitor's Secret Key Crypto - KARN, encrypt



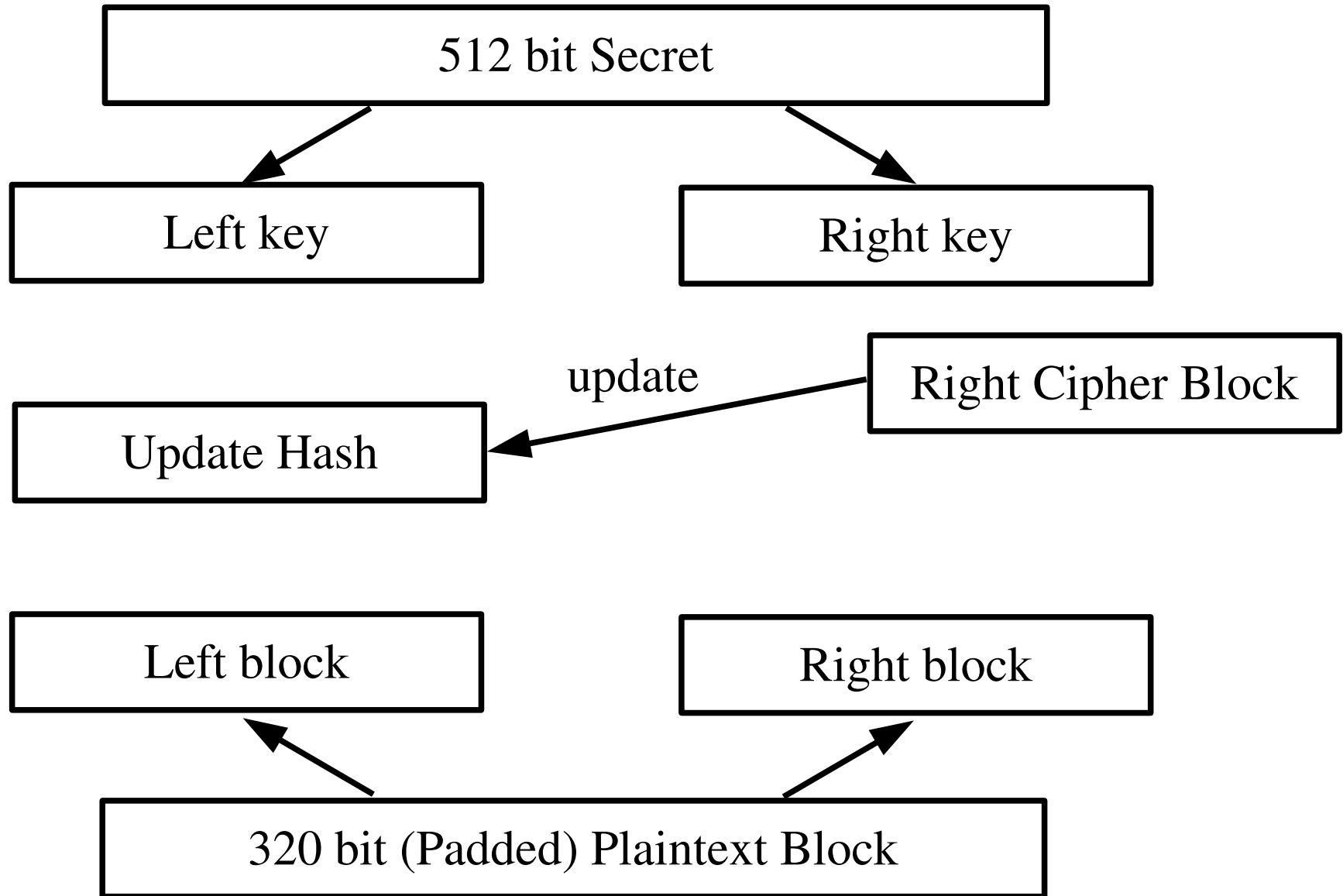
Form right cipher block from right plaintext block and hash

Monitor's Secret Key Crypto - KARN, encrypt



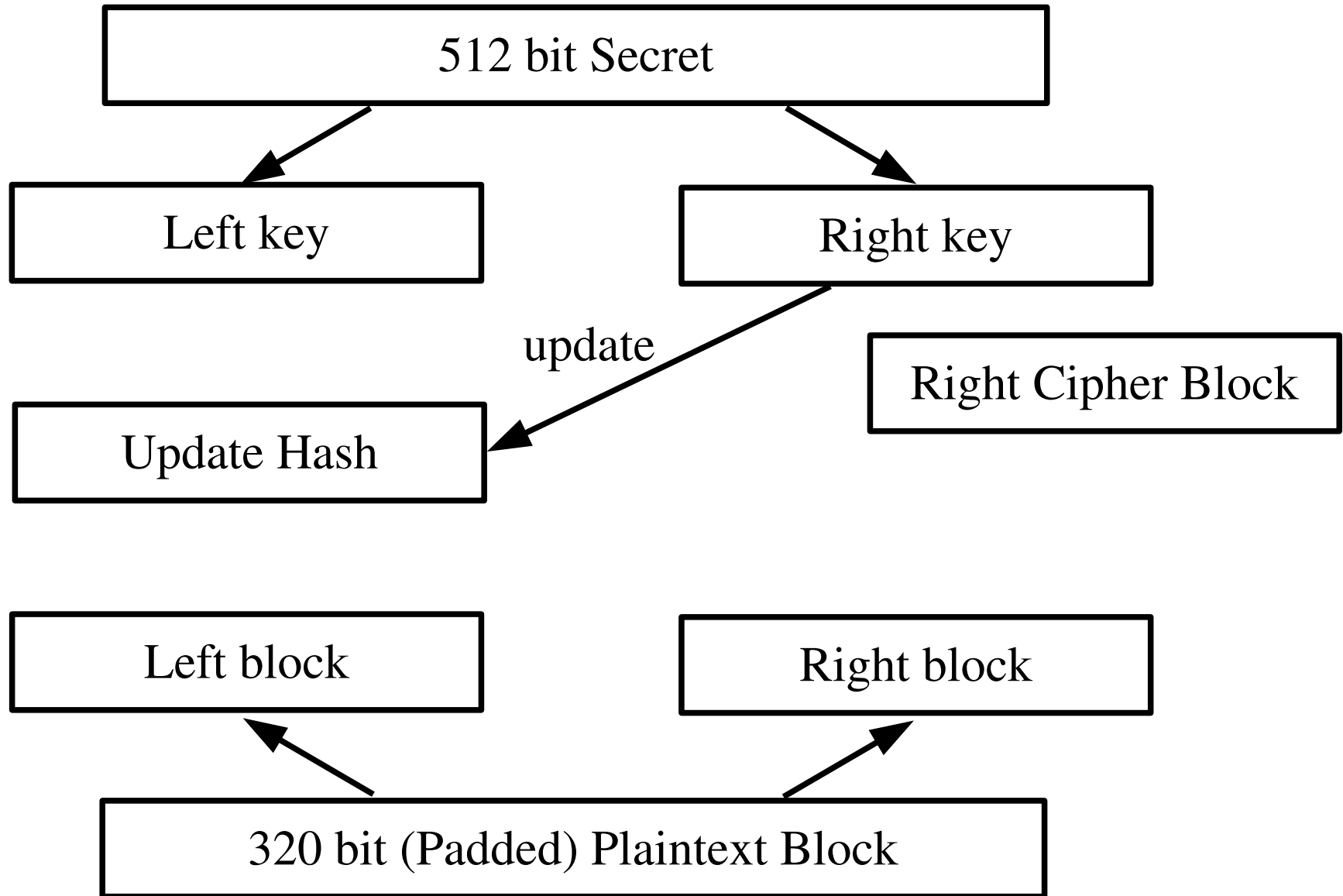
Reset Hash

Monitor's Secret Key Crypto - KARN, encrypt



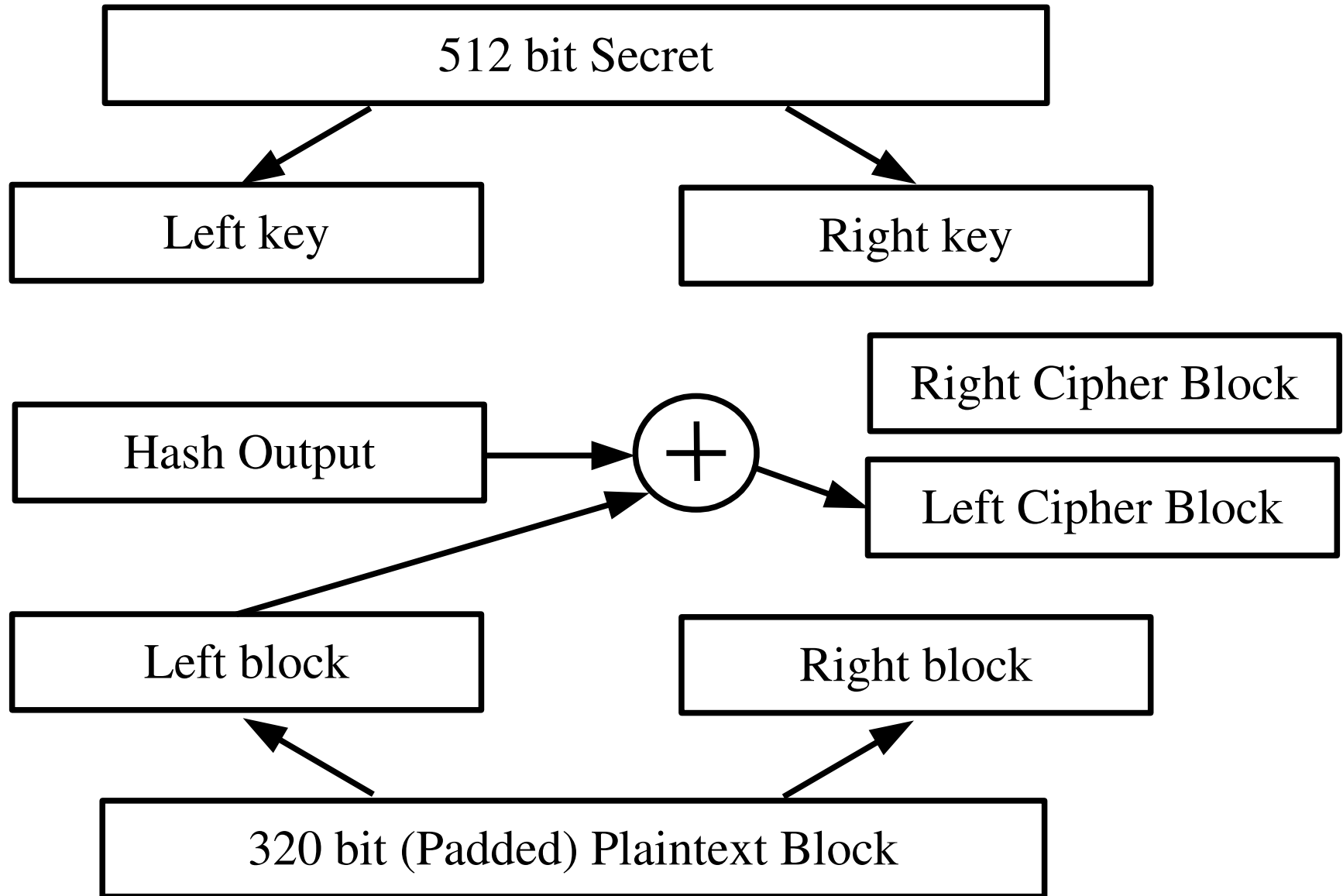
Hash sequence: right cipher block then right key

Monitor's Secret Key Crypto - KARN, encrypt



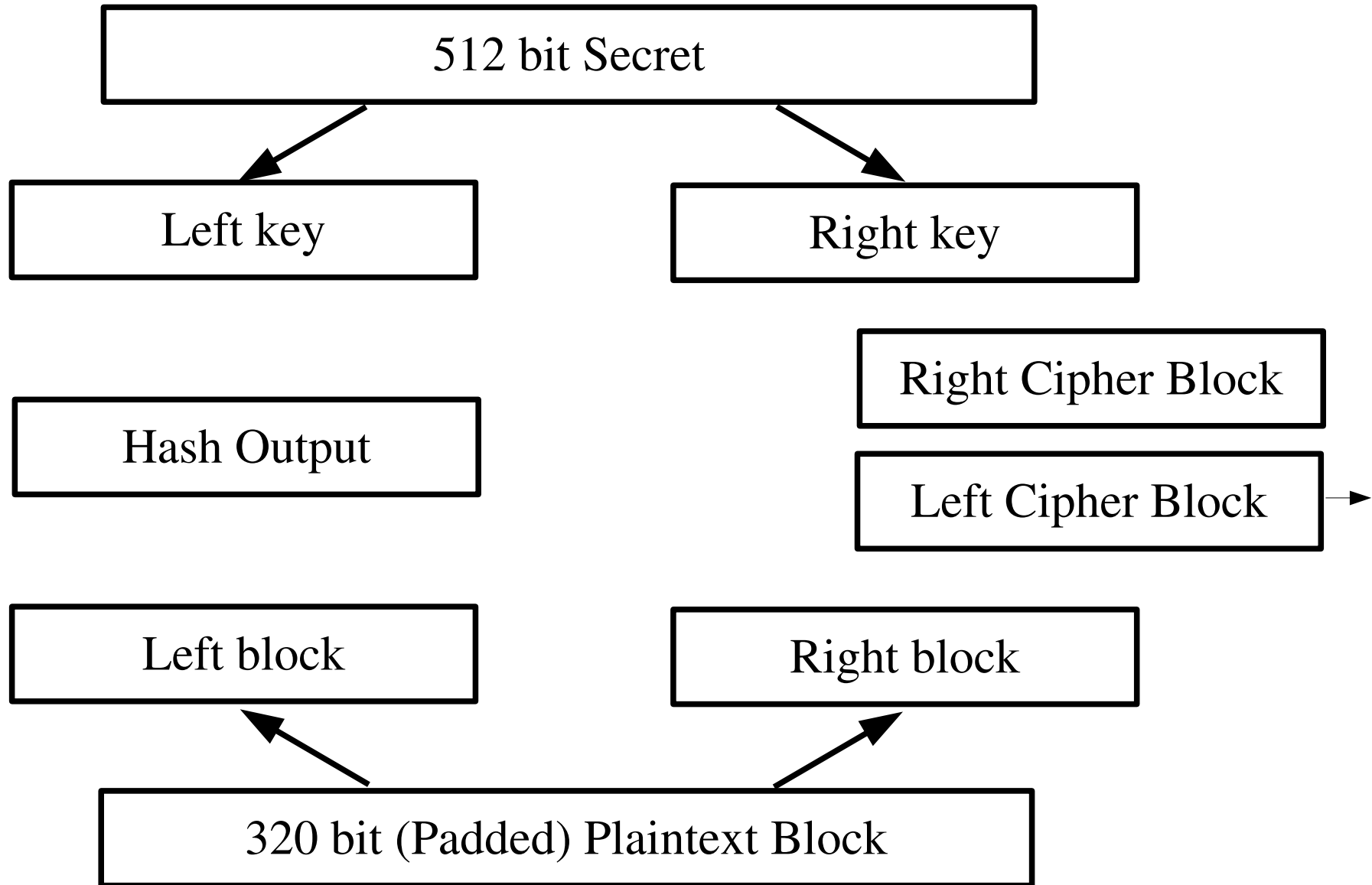
Hash sequence: right cipher block then right key

Monitor's Secret Key Crypto - KARN, encrypt



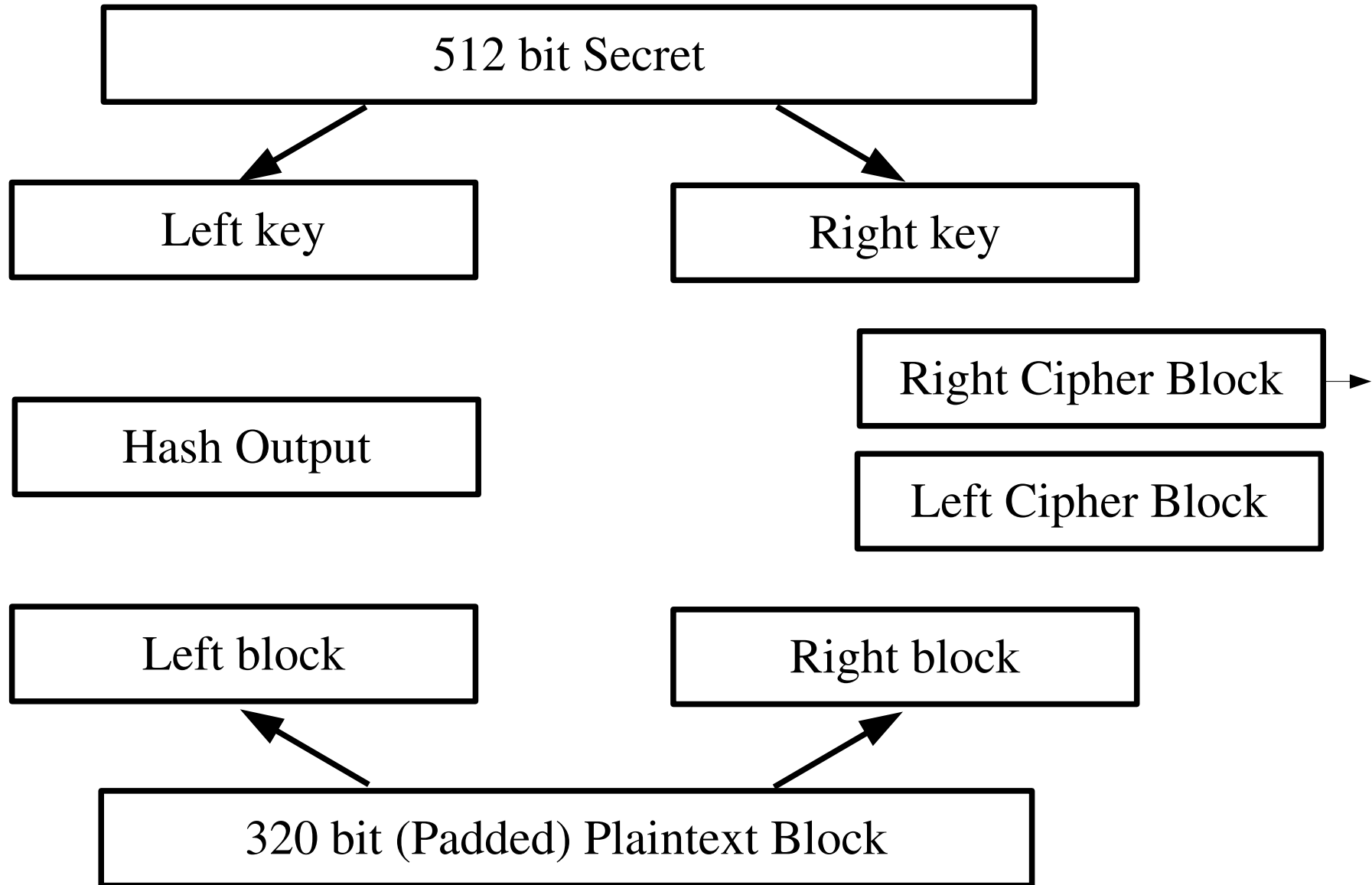
Form left cipher block from left plaintext block and hash

Monitor's Secret Key Crypto - KARN, encrypt



Output left cipher block

Monitor's Secret Key Crypto - KARN, encrypt



Output right cipher block

Monitor's Secret Key Crypto - KARN, encrypt

```
ByteArrayOutputStream buffer =  
    new ByteArrayOutputStream();  
  
String input = "The plaintext message to pad";  
byte scratch[] = input.getBytes();  
int len = input.length();  
buffer.write(scratch, 0, len);
```

< 320 bit Last Plaintext Block

Pad to last block, or pad a whole block

Monitor's Secret Key Crypto - KARN, encrypt

```
buffer.write(0);
```



Stick a 0 byte on the end

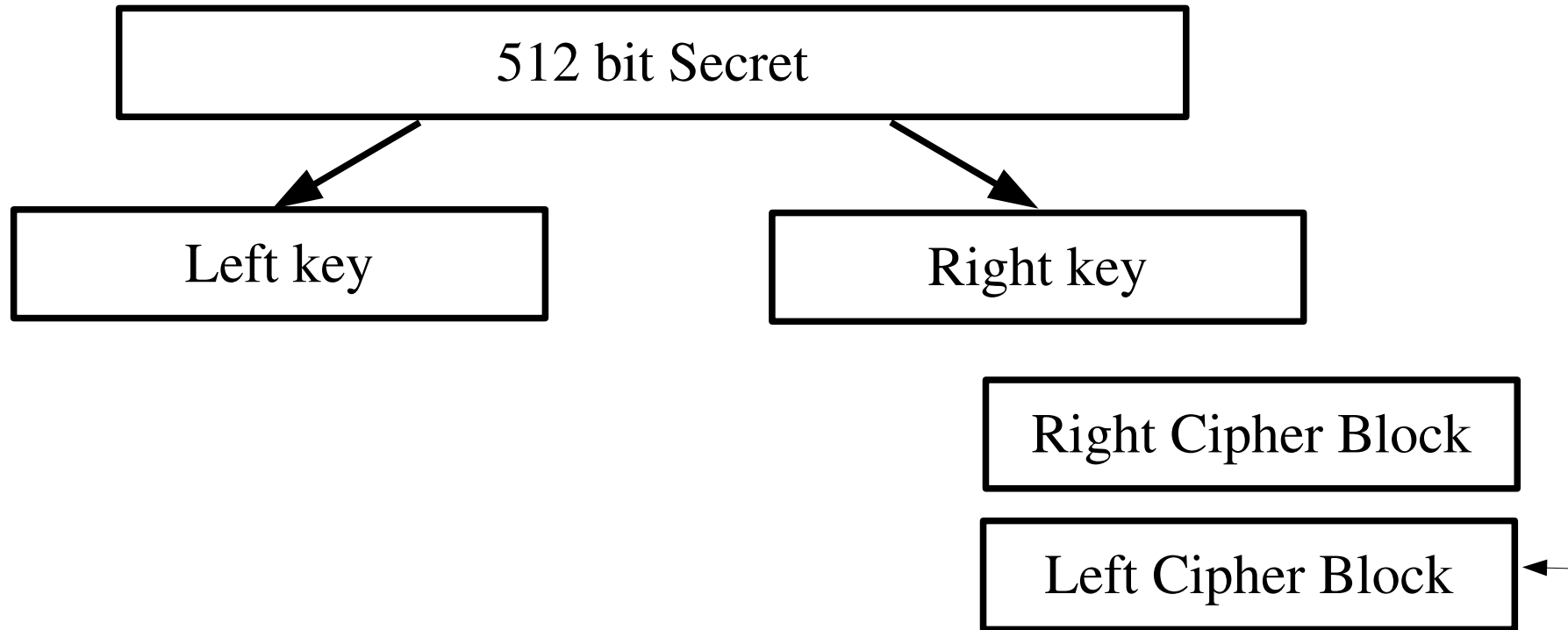
Monitor's Secret Key Crypto - KARN, encrypt

```
int padlen = BLOCKSIZE - ((len + 1) % BLOCKSIZE);  
scratch[] = new byte[padlen];  
SecureRandom sr = new SecureRandom();  
sr.nextBytes(scratch);  
buffer.write(scratch, 0, padlen);
```



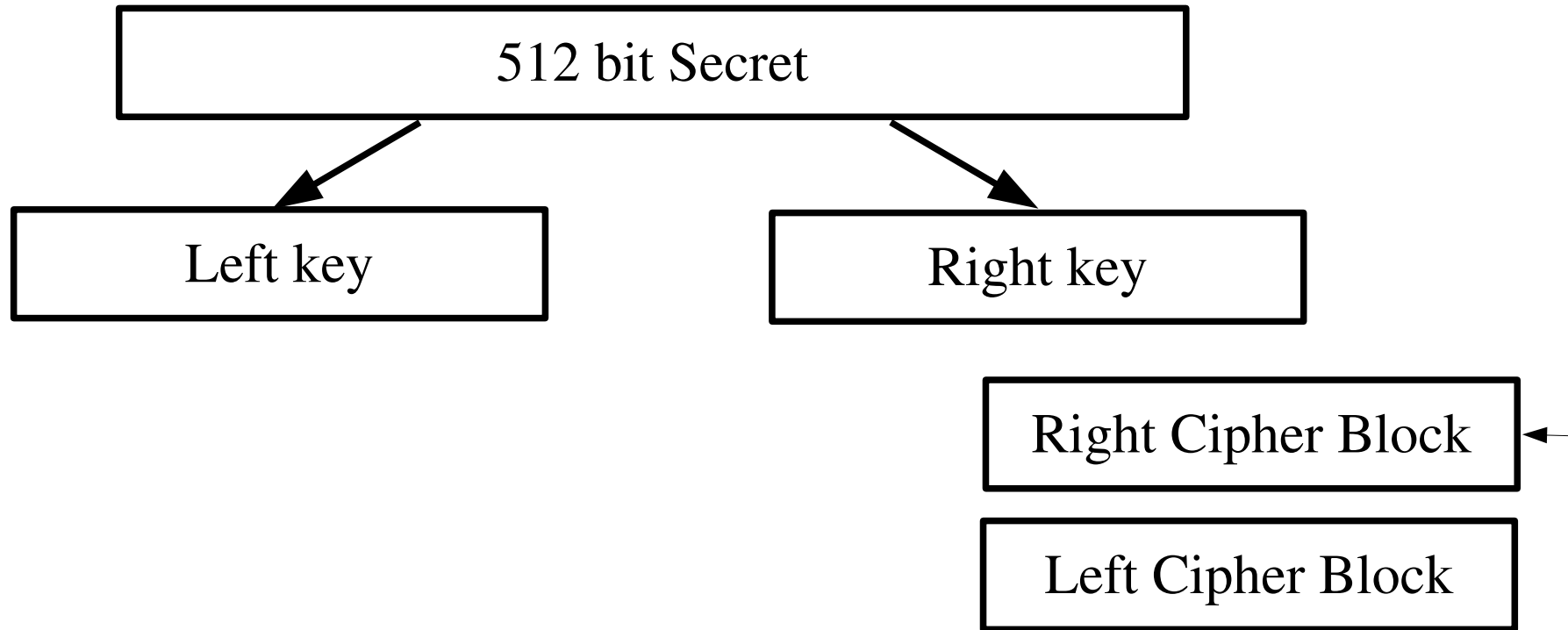
Remaining bytes are derived from random #

Monitor's Secret Key Crypto - KARN, decrypt



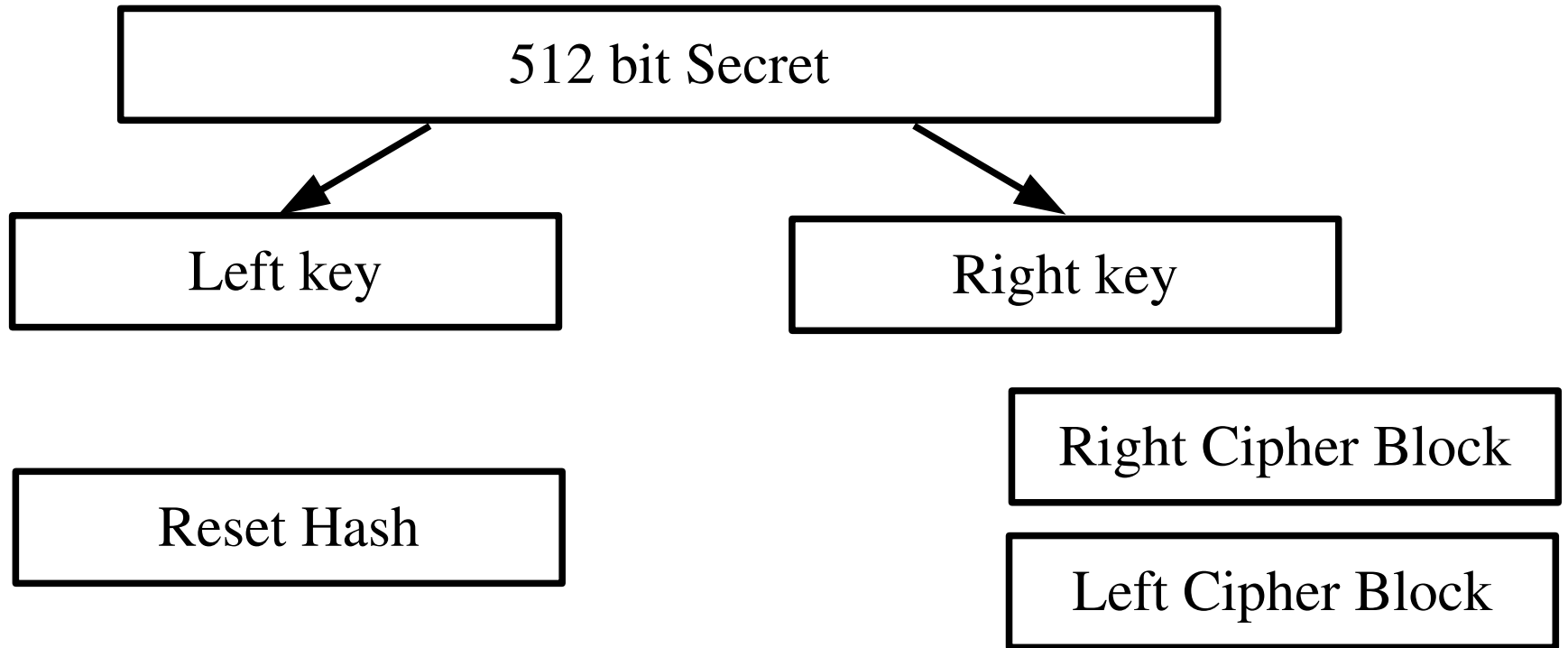
Input left cipher block

Monitor's Secret Key Crypto - KARN, decrypt

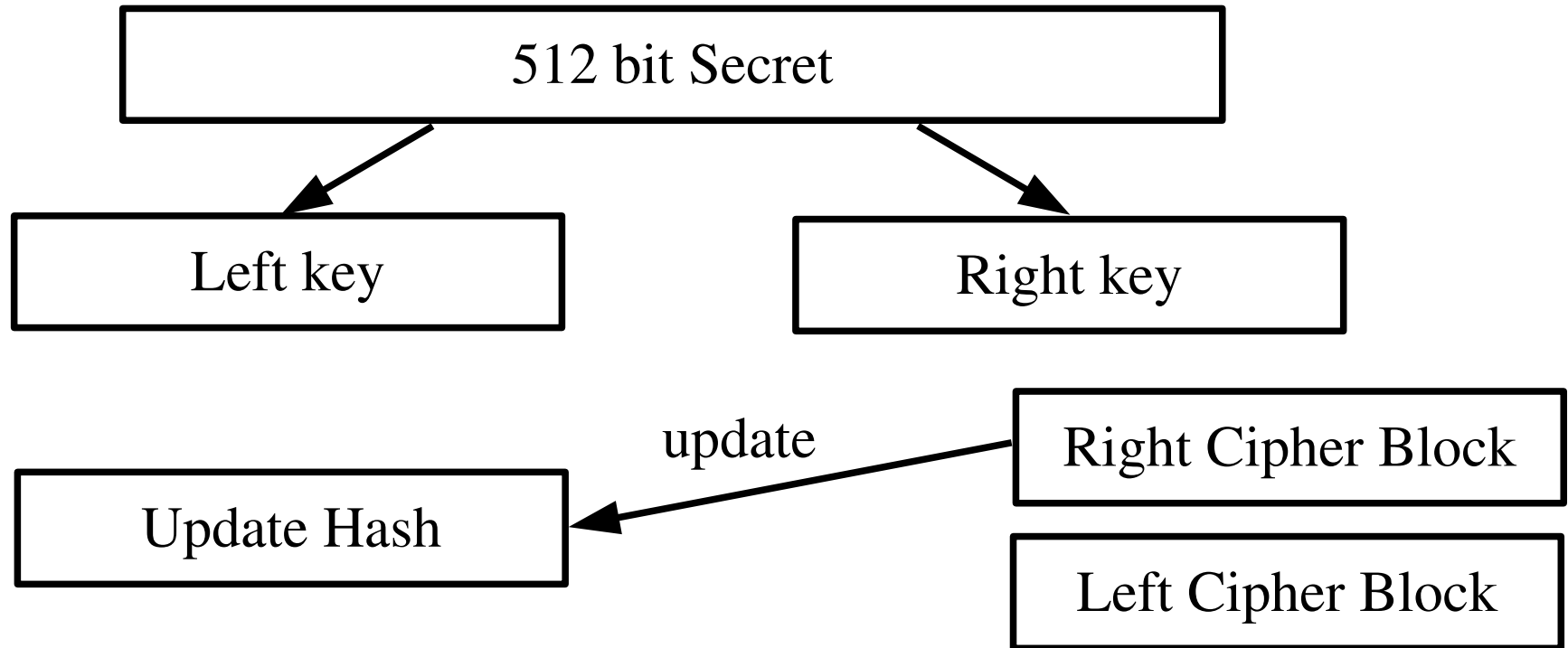


Input right cipher block

Monitor's Secret Key Crypto - KARN, decrypt

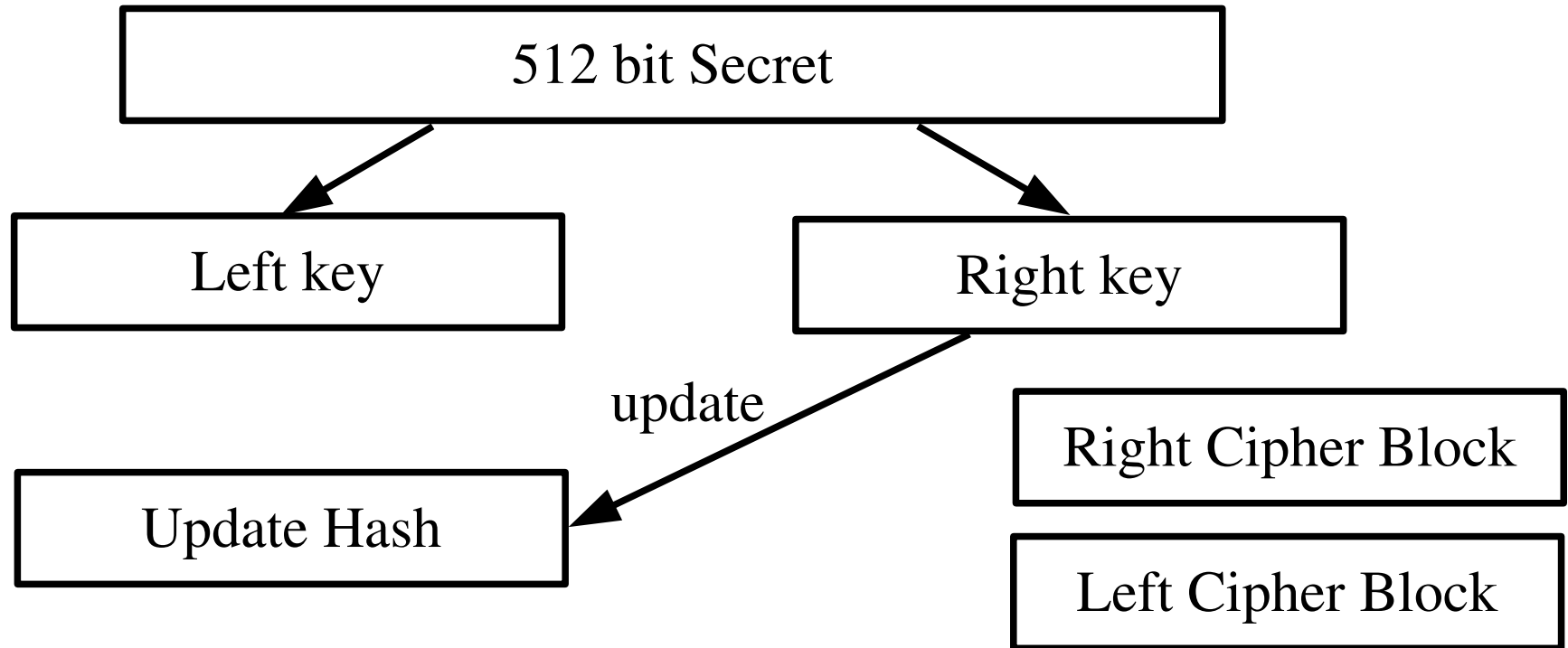


Monitor's Secret Key Crypto - KARN, decrypt



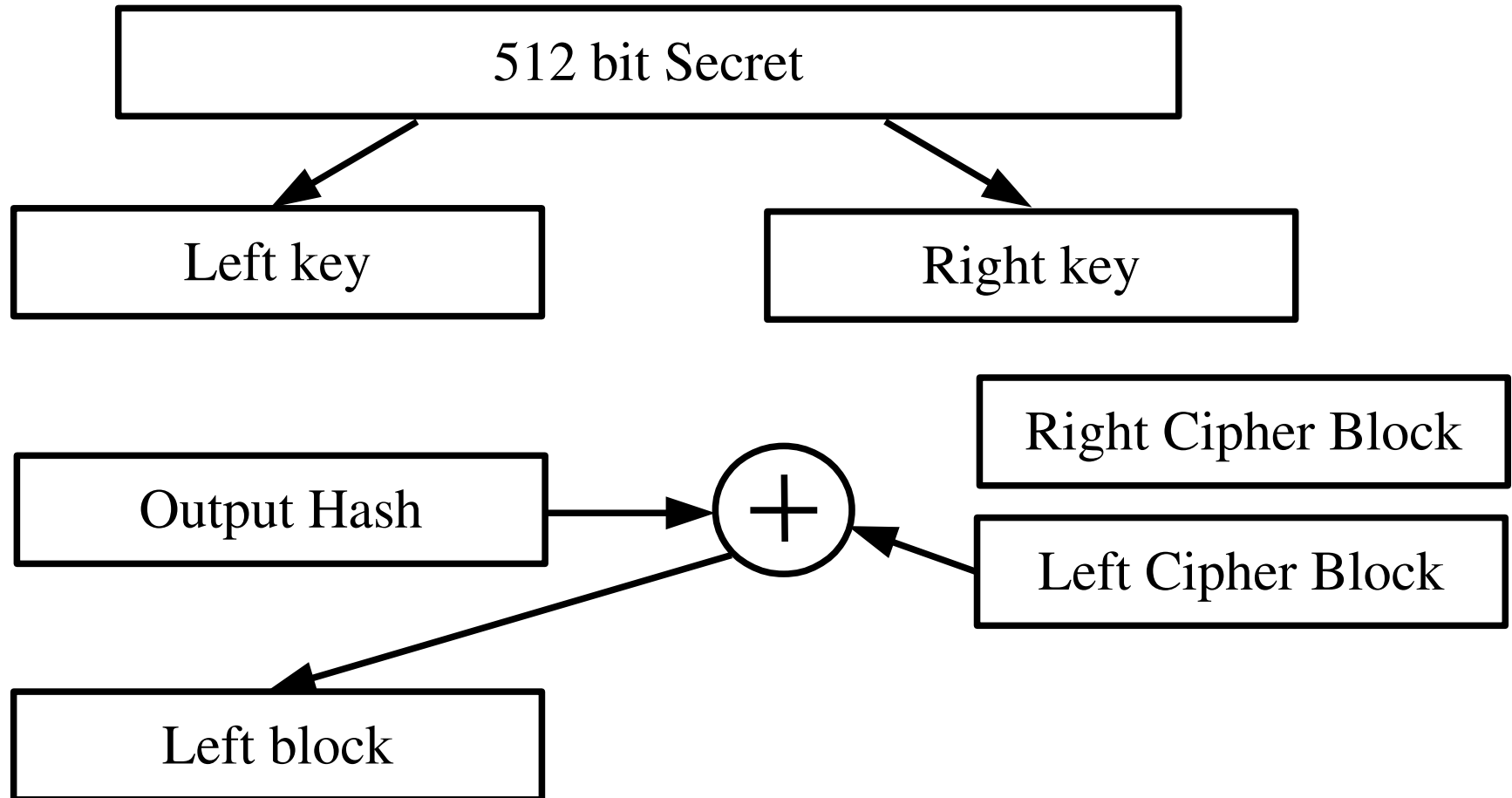
Hash sequence: right cipher block then right key

Monitor's Secret Key Crypto - KARN, decrypt



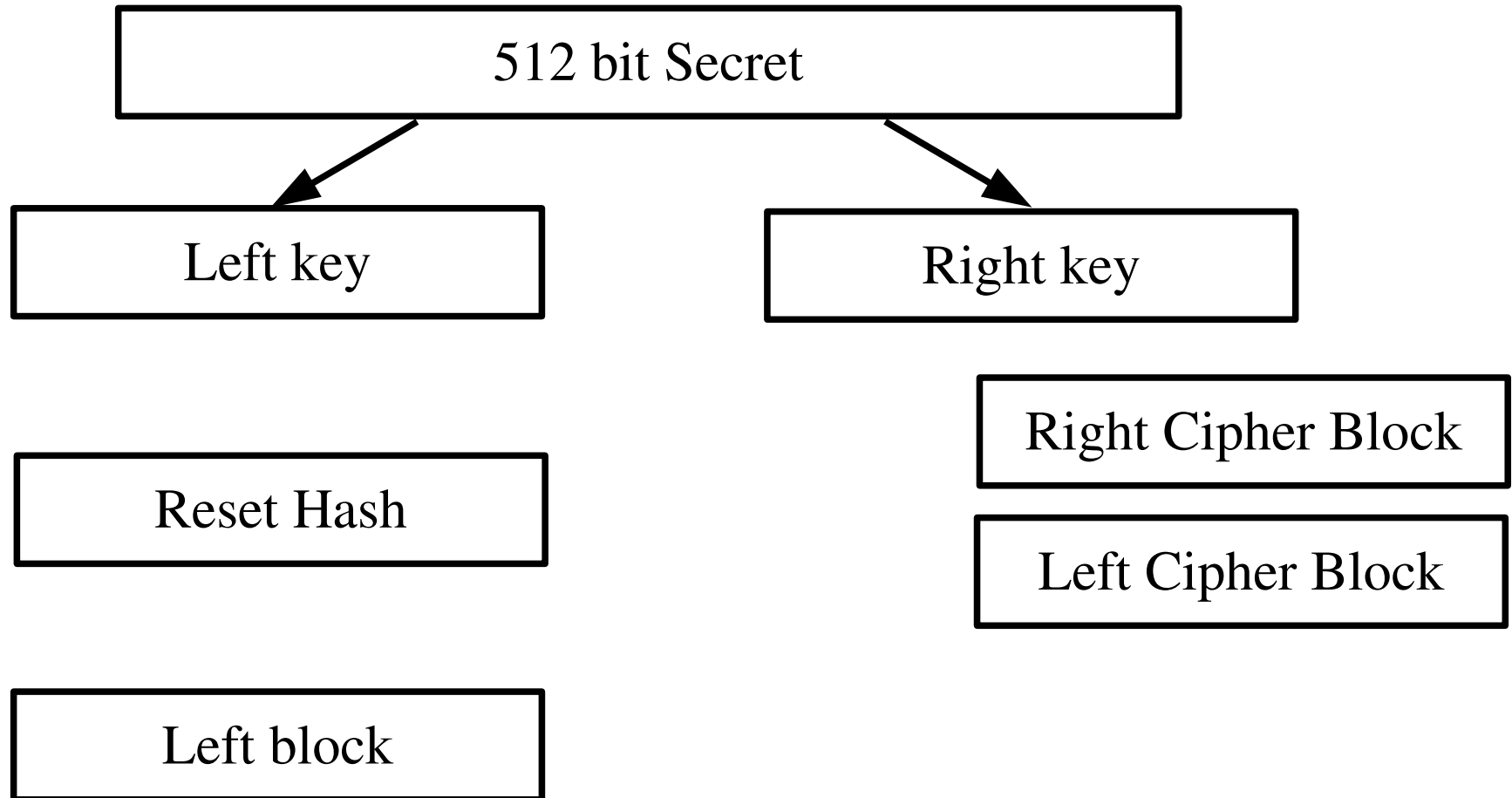
Hash sequence: right cipher block then right key

Monitor's Secret Key Crypto - KARN, decrypt

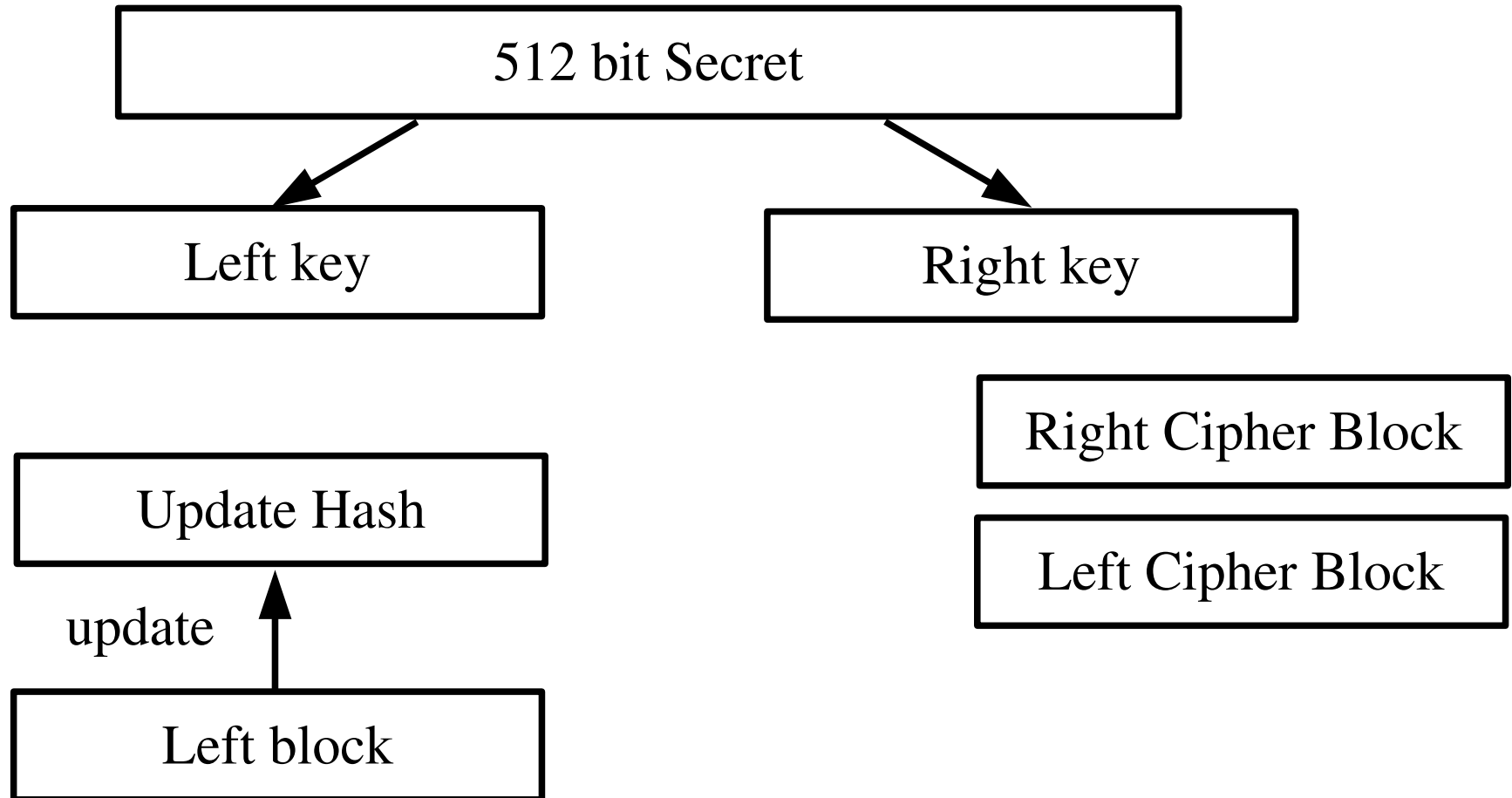


Form left plaintext block from left cipher block and hash

Monitor's Secret Key Crypto - KARN, decrypt

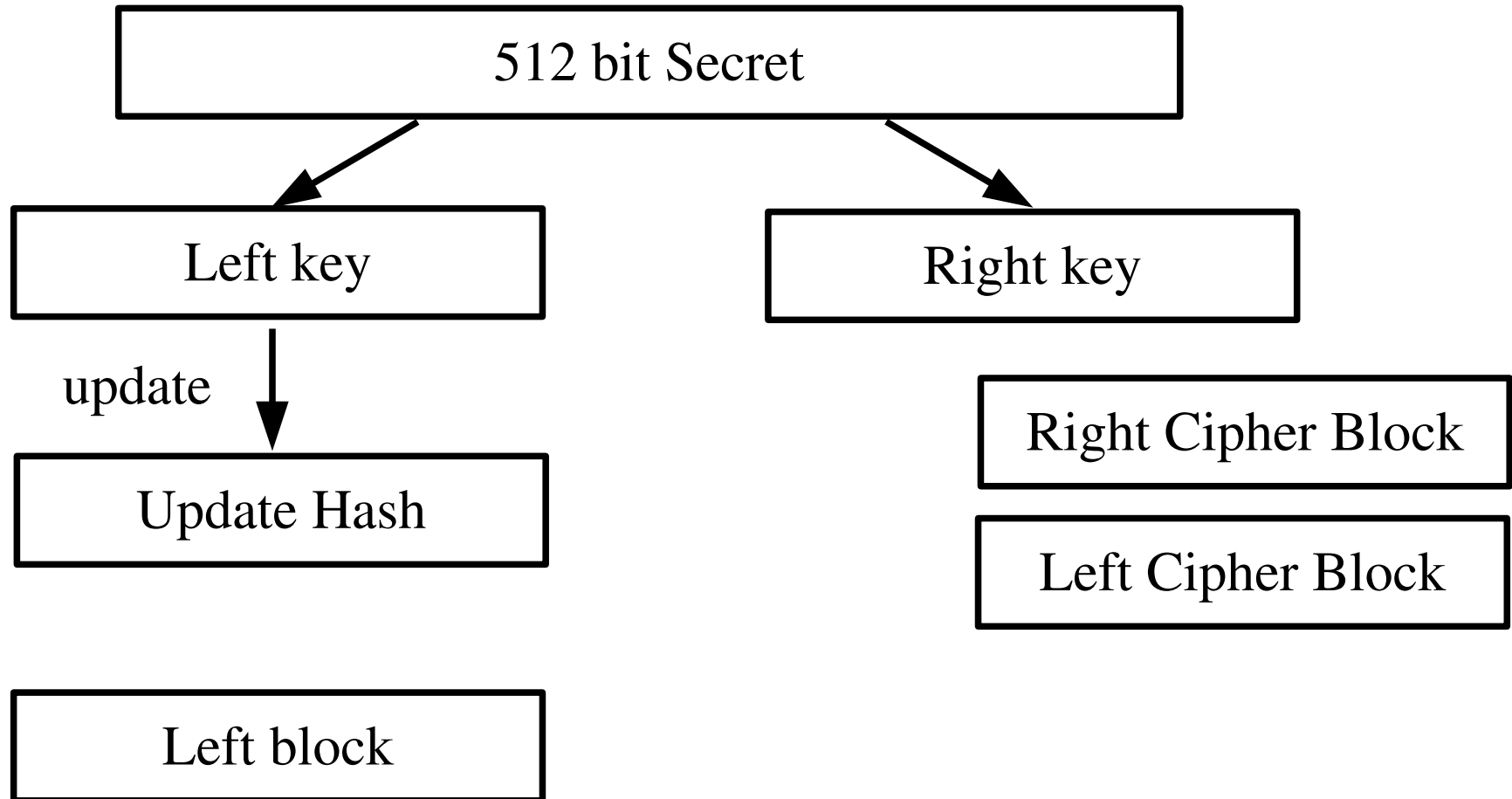


Monitor's Secret Key Crypto - KARN, decrypt



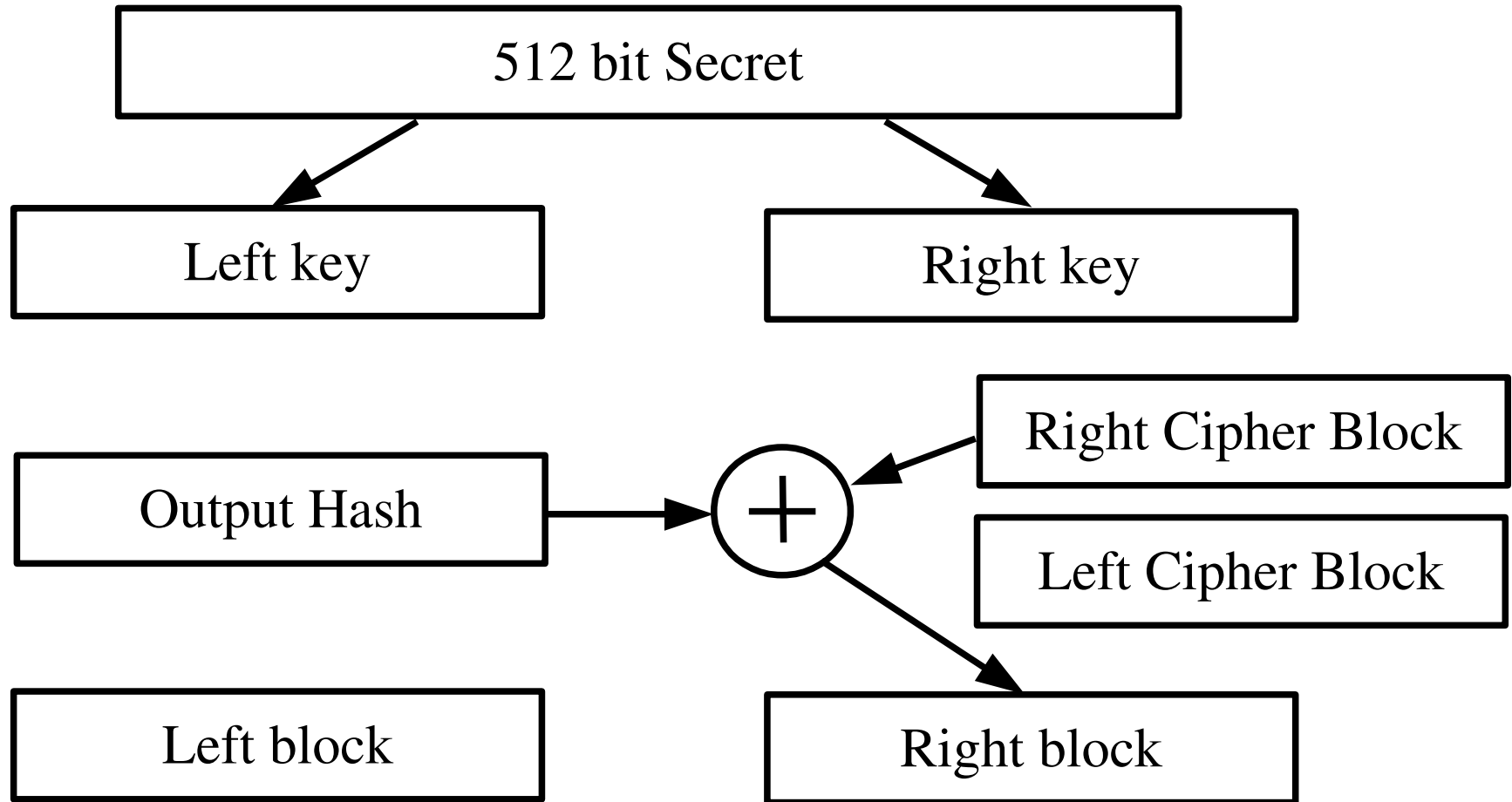
Hash sequence: left plaintext block then left key

Monitor's Secret Key Crypto - KARN, decrypt



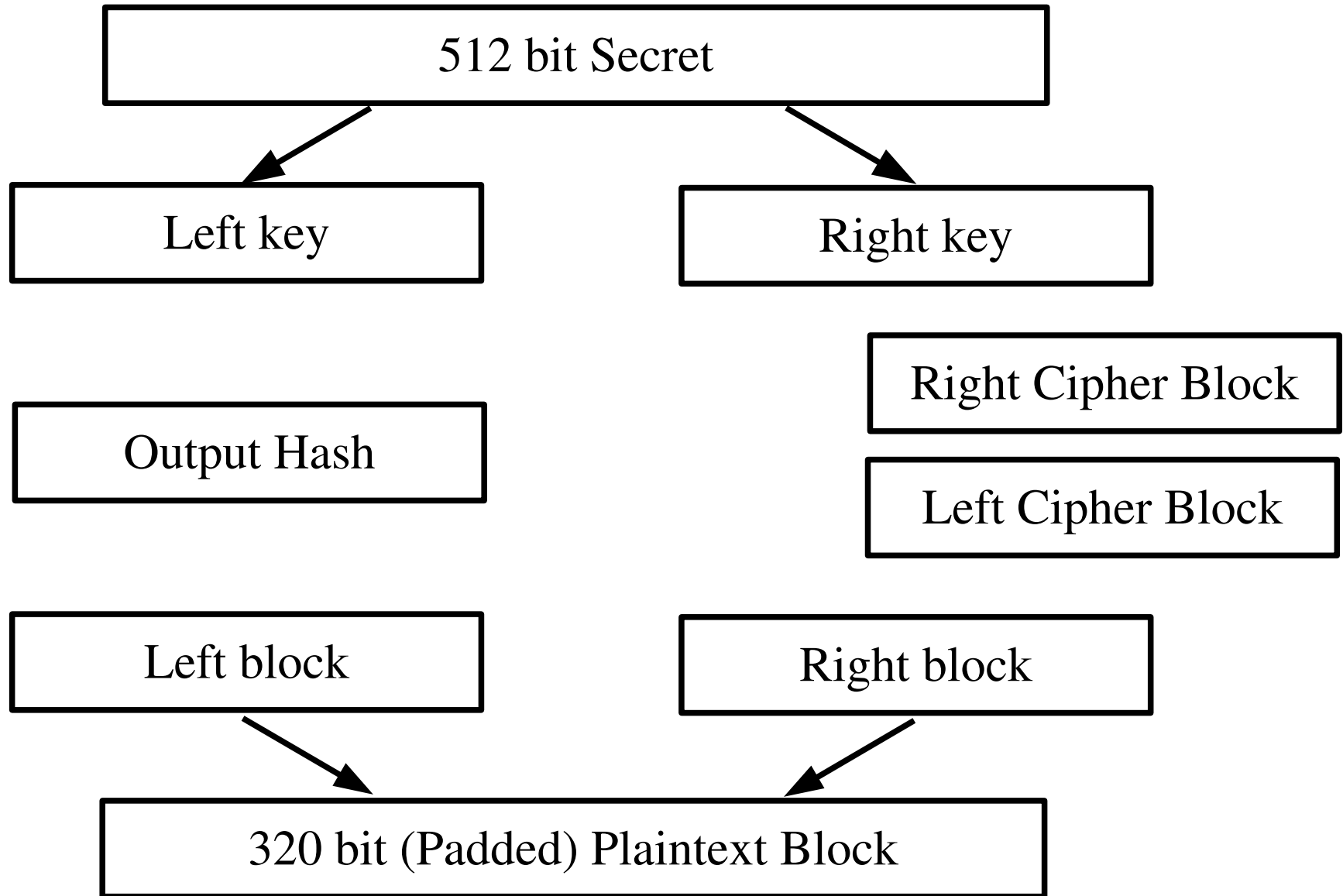
Hash sequence: left plaintext block then left key

Monitor's Secret Key Crypto - KARN, decrypt



Form right plaintext block from right cipher block and hash

Monitor's Secret Key Crypto - KARN, decrypt



Output Plaintext block