

## CHANGE REQUEST

**33.220 CR 045** rev - Current version: **6.3.0**

For **HELP** on using this form, see bottom of this page or look at the pop-up text over the symbols.

Proposed change affects:  UICC apps  ME  Radio Access Network  Core Network

<b>Title:</b>	Key derivation function: character encoding		
<b>Source:</b>	Nokia		
<b>Work item code:</b>	SEC1-SC	<b>Date:</b>	02/02/2005
<b>Category:</b>	<b>C</b>	<b>Release:</b>	Rel-6
	<p>Use <u>one</u> of the following categories:</p> <p><b>F</b> (correction)</p> <p><b>A</b> (corresponds to a correction in an earlier release)</p> <p><b>B</b> (addition of feature),</p> <p><b>C</b> (functional modification of feature)</p> <p><b>D</b> (editorial modification)</p> <p>Detailed explanations of the above categories can be found in 3GPP <a href="#">TR 21.900</a>.</p>		<p>Use <u>one</u> of the following releases:</p> <p><i>Ph2</i> (GSM Phase 2)</p> <p><i>R96</i> (Release 1996)</p> <p><i>R97</i> (Release 1997)</p> <p><i>R98</i> (Release 1998)</p> <p><i>R99</i> (Release 1999)</p> <p><i>Rel-4</i> (Release 4)</p> <p><i>Rel-5</i> (Release 5)</p> <p><i>Rel-6</i> (Release 6)</p> <p><i>Rel-7</i> (Release 7)</p>

<b>Reason for change:</b>	How to encode input parameters for the key derivation function is unclear, e.g., how an input parameter which is a character string is encoded to an octet string. UTF-8 encoding shall be used in the encoding. To avoid confusion, the KDF input parameters are now separated by commas ",", instead of concatenation marks " ".
<b>Summary of change:</b>	Input parameter encoding is clarified, i.e., UTF-8 encoding shall be used. The KDF input parameters are separated by commas ",", instead of concatenation marks " ".
<b>Consequences if not approved:</b>	Input parameter encoding is unclear.

<b>Clauses affected:</b>	2, 4.5.2, 5.3.2, B.2.1 (new), B.3										
<b>Other specs affected:</b>	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">Y</td> <td style="padding: 2px;">N</td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/></td> <td style="padding: 2px;"><input checked="" type="checkbox"/></td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/></td> <td style="padding: 2px;"><input checked="" type="checkbox"/></td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/></td> <td style="padding: 2px;"><input checked="" type="checkbox"/></td> </tr> </table>	Y	N	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Other core specifications Test specifications O&M Specifications	
Y	N										
<input type="checkbox"/>	<input checked="" type="checkbox"/>										
<input type="checkbox"/>	<input checked="" type="checkbox"/>										
<input type="checkbox"/>	<input checked="" type="checkbox"/>										
<b>Other comments:</b>											

===== BEGIN CHANGE =====

---

## 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

- [1] 3GPP TS 31.102: "3rd Generation Partnership Project; Technical Specification Group Terminals; Characteristics of the USIM application".
- [2] 3GPP TS 33.102: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture".
- [3] Franks J., et al.: "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.
- [4] A. Niemi, et al.: "Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)", RFC 3310, September 2002.
- [5] 3GPP TS 33.221: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Generic Authentication Architecture (GAA); Support for Subscriber Certificates".
- [6] T. Dierks, et al.: "The TLS Protocol Version 1.0", RFC 2246, January 1999.
- [7] OMA: "Provisioning Content Version 1.1", Version 13-Aug-2003. Open Mobile Alliance.
- [8] 3GPP TS 23.228: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; IP Multimedia Subsystem (IMS); Stage 2 (Release 6)".
- [9] IETF RFC 3546 (2003): "Transport Layer Security (TLS) Extensions".
- [10] 3GPP TS 31.103: "3rd Generation Partnership Project; Technical Specification Group Terminals; Characteristics of the IP Multimedia Services Identity Module (ISIM) application".
- [11] 3GPP TS 23.003: "3rd Generation Partnership Project; Technical Specification Group Core Network; Numbering, addressing and identification".
- [12] IETF RFC 3548 (2003): "The Base16, Base32, and Base64 Data Encodings".
- [13] 3GPP TS 33.210: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Network domain security; IP network layer security".
- [14] IETF RFC 3588 (2003): "Diameter Base Protocol".
- [15] 3GPP TS 31.101: "3rd Generation Partnership Project; Technical Specification Group Terminals; UICC-terminal interface; Physical and logical characteristics".
- [16] 3GPP TS 33.203: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G security; Access security for IP-based services".
- [17] IETF RFC 3280 (2002): "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile".
- [18] IETF RFC 2818 (2000): "HTTP over TLS".

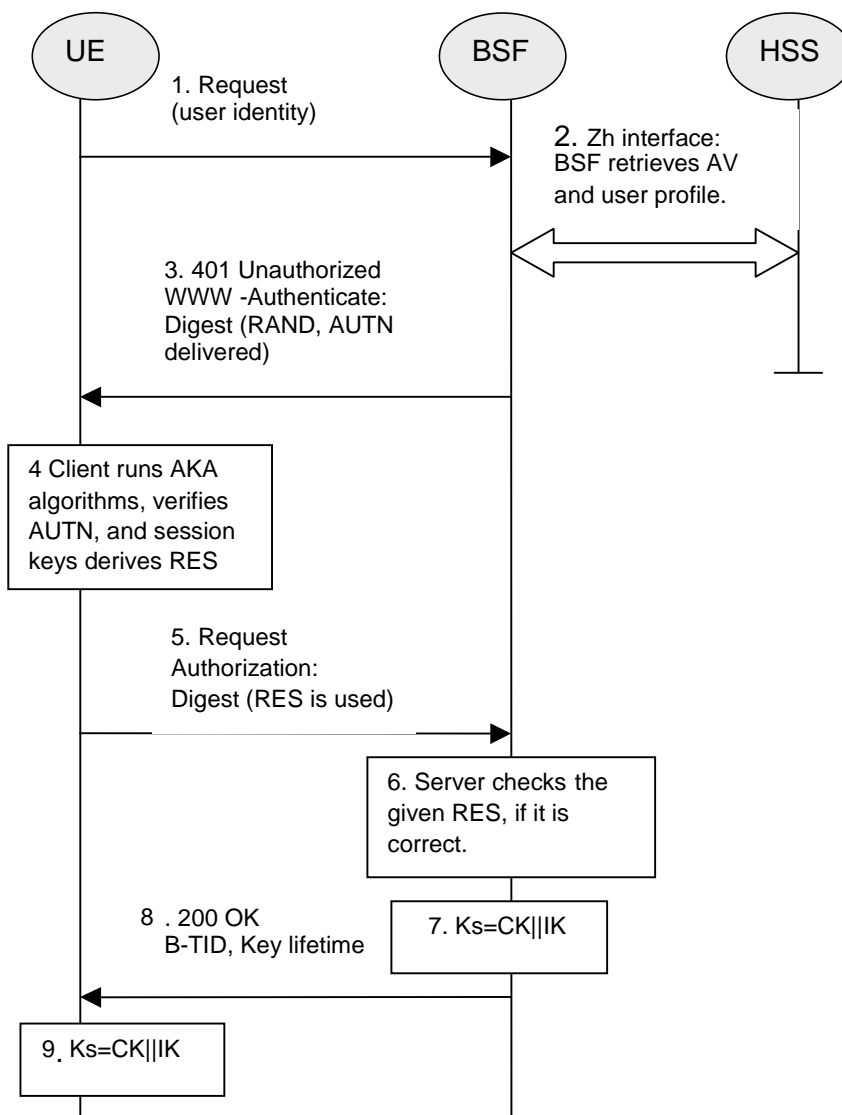
- [19] 3GPP TS 33.310: "3rd Generation Partnership Project; Technical Specification Group Service and System Aspects; Network Domain Security (NDS); Authentication Framework (AF)".
- [20] IETF RFC 2560 (1999): "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP".
- [21] FIPS PUB 180-2 (2002): "Secure Hash Standard".
- [22] IETF RFC 2104 (1997): "HMAC: Keyed-Hashing for Message Authentication".
- [23] ISO/IEC 10118-3:2004: "Information Technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions".
- [24] [IETF RFC 3629 \(2003\): "UTF-8, a transformation format of ISO 10646"](#).

===== BEGIN NEXT CHANGE =====

## 4.5.2 Bootstrapping procedures

When a UE wants to interact with a NAF, and it knows that the bootstrapping procedure is needed, it shall first perform a bootstrapping authentication (see figure 4.3). Otherwise, the UE shall perform a bootstrapping authentication only when it has received bootstrapping initiation required message or a bootstrapping negotiation indication from the NAF, or when the lifetime of the key in UE has expired (cf. subclause 4.5.3).

NOTE 1: The main steps from the specifications of the AKA protocol in TS 33.102 [2] and the HTTP digest AKA protocol in RFC 3310 [4] are repeated in figure 3 for the convenience of the reader. In case of any potential conflict, the specifications in TS 33.102 [2] and RFC 3310 [4] take precedence.



**Figure 4.3: The bootstrapping procedure**

1. The UE sends an HTTP request towards the BSF.
2. BSF retrieves the complete set of GBA user security settings and one Authentication Vector (AV, AV = RAND||AUTN||XRES||CK||IK) over the reference point Zh from the HSS.
3. Then BSF forwards the RAND and AUTN to the UE in the 401 message (without the CK, IK and XRES). This is to demand the UE to authenticate itself.
4. The UE checks AUTN to verify that the challenge is from an authorised network; the UE also calculates CK, IK and RES. This will result in session keys IK and CK in both BSF and UE.
5. The UE sends another HTTP request, containing the Digest AKA response (calculated using RES), to the BSF.
6. The BSF authenticates the UE by verifying the Digest AKA response.
7. The BSF generates key material  $K_s$  by concatenating CK and IK. The B-TID value shall be also generated in format of NAI by taking the base64 encoded [12] RAND value from step 3, and the BSF server name, i.e. `base64encode(RAND)@BSF_servers_domain_name`.
8. The BSF shall send a 200 OK message, including a B-TID, to the UE to indicate the success of the authentication. In addition, in the 200 OK message, the BSF shall supply the lifetime of the key  $K_s$ . The key material  $K_s$  is generated in UE by concatenating CK and IK.

9. Both the UE and the BSF shall use the Ks to derive the key material Ks\_NAF during the procedures as specified in clause 4.5.3. Ks\_NAF shall be used for securing the reference point Ua.

Ks\_NAF is computed as  $Ks\_NAF = KDF(Ks, "gba-me" \parallel RAND \parallel IMPI \parallel NAF\_Id)$ , where KDF is the key derivation function as specified in Annex B, and the key derivation parameters consist of the user's IMPI, the NAF\_Id and RAND. The NAF\_Id consists of the full DNS name of the NAF. KDF shall be implemented in the ME.

NOTE 2: To allow consistent key derivation based on NAF name in UE and BSF, at least one of the three following prerequisites shall be fulfilled:

- (1) The NAF is known in DNS under one domain name (FQDN) only, i.e. no two different domain names point to the IP address of the NAF. This has to be achieved by administrative means. This prerequisite is not specific to 3GPP, as it is necessary also under other circumstances, e.g. for TLS V1.0 without use of wildcard or multiple-name certificates.
- (2) Each DNS entry of the NAF points to a different IP address. The NAF responds to all these IP addresses. Each IP address is tied to the corresponding FQDN by NAF configuration. The NAF can see from the IP address, which FQDN to use for key derivation.
- (3) Ua uses a protocol which transfers the host name (FQDN of NAF as used by UE) to NAF (e.g. HTTP/1.1 with mandatory Host request header field). This requires the NAF to check the validity of the host name, to use this name in all communication with UE where appropriate, and to transfer this name to BSF to allow for correct derivation of Ks\_NAF. In case of a TLS tunnel this requires either multiple-identities certificates or the deployment of RFC 3546 [9] or other protocol means with similar purpose.

The UE and the BSF shall store the key Ks with the associated B-TID for further use, until the lifetime of Ks has expired, or until the key Ks is updated.

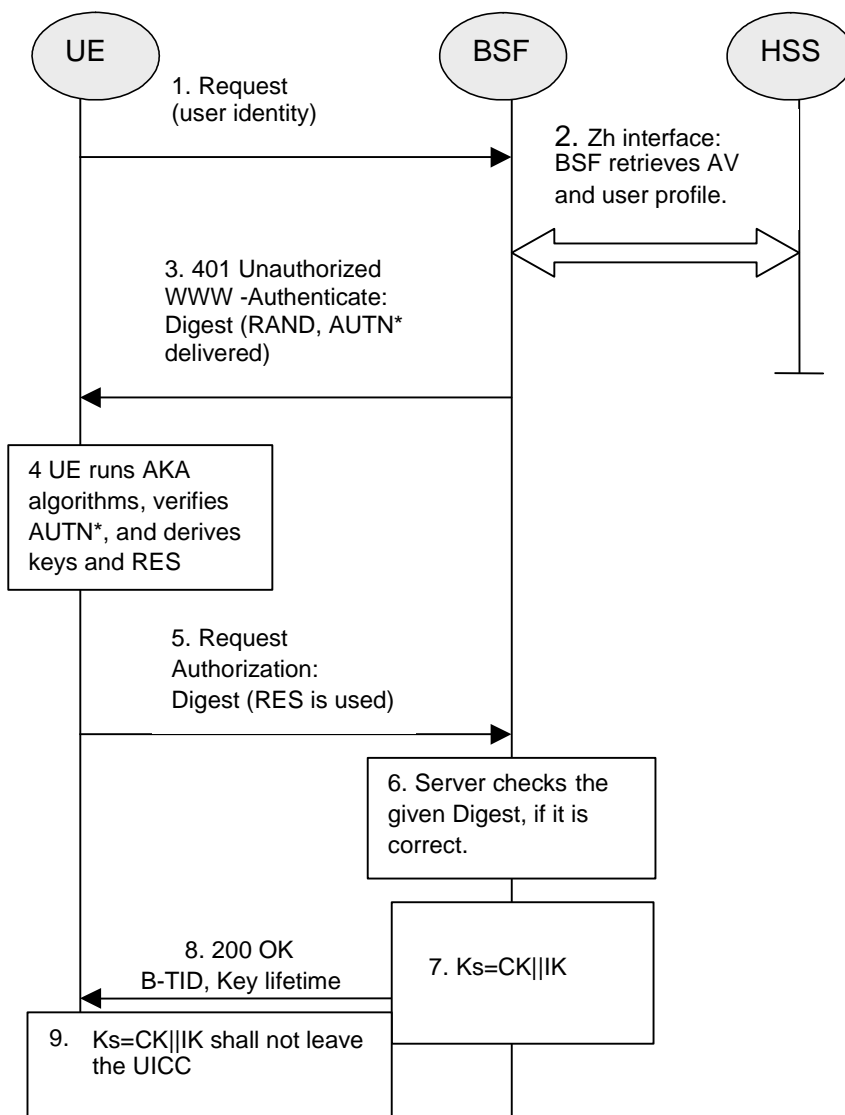
===== BEGIN NEXT CHANGE =====

## 5.3.2 Bootstrapping procedure

The procedure specified in this clause differs from the procedure specified clause 4.5.2 in the local handling of keys and Authentication Vectors in the UE and the BSF. The messages exchanged over the Ub reference point are identical for both procedures.

When a UE wants to interact with a NAF, and it knows that the bootstrapping procedure is needed, it shall first perform a bootstrapping authentication (see figure 5.1). Otherwise, the UE shall perform a bootstrapping authentication only when it has received bootstrapping initiation required message or a bootstrapping renegotiation indication from the NAF, or when the lifetime of the key in UE has expired (see clause 5.3.3).

NOTE: The main steps from the specifications of the AKA protocol in TS 33.102 [2] and the HTTP digest AKA protocol in RFC 3310 [4] are repeated in figure 5.1 for the convenience of the reader. In case of any potential conflict, the specifications in TS 33.102 [2] and RFC 3310 [4] take precedence.



**Figure 5.1: The bootstrapping procedure with UICC-based enhancements**

1. The ME sends an HTTP request towards the BSF.
2. The BSF retrieves the complete set of GBA user security settings and one Authentication Vector (AV, AV = RAND||AUTN||XRES||CK||IK) over the Zh reference point from the HSS. The BSF can then decide to perform GBA\_U, based on the user security settings (USSs). In this case, the BSF proceeds in the following way:

- BSF computes  $MAC^* = MAC \oplus \text{Trunc}(\text{SHA-1}(\text{IK}))$

NOTE: Trunc denotes that from the 160 bit output of SHA-1 [21], the 64 bits numbered as [0] to [63] are used within the \* operation to MAC.

The BSF stores the XRES after flipping the least significant bit.

3. Then BSF forwards the RAND and AUTN\* (where  $AUTN^* = SQN \oplus AK \parallel AMF \parallel MAC^*$ ) to the UE in the 401 message (without the CK, IK and XRES). This is to demand the UE to authenticate itself.
4. The ME sends RAND and AUTN\* to the UICC. The UICC calculates IK and MAC (by performing  $MAC = MAC^* \oplus \text{Trunc}(\text{SHA-1}(\text{IK}))$ ). Then the UICC checks AUTN (i.e.  $SQN \oplus AK \parallel AMF \parallel MAC$ ) to verify that the challenge is from an authorised network; the UICC also calculates CK and RES. This will result in session keys CK and IK in both BSF and UICC. The UICC then transfers RES (after flipping the least significant bit) to the ME and stores Ks, which is the concatenation of CK and IK, on the UICC.

5. The ME sends another HTTP request, containing the Digest AKA response (calculated using RES), to the BSF.
6. The BSF authenticates the UE by verifying the Digest AKA response.
7. The BSF generates the key Ks by concatenating CK and IK. The B-TID value shall be also generated in format of NAI by taking the base64 encoded [12] RAND value from step 3, and the BSF server name, i.e. base64encode(RAND)@BSF\_servers\_domain\_name.
8. The BSF shall send a 200 OK message, including the B-TID, to the UE to indicate the success of the authentication. In addition, in the 200 OK message, the BSF shall supply the lifetime of the key Ks.
9. Both the UICC and the BSF shall use the Ks to derive NAF-specific keys Ks\_ext\_NAF and Ks\_int\_NAF during the procedures as specified in clause 5.3.3, if applicable. Ks\_ext\_NAF and Ks\_int\_NAF are used for securing the Ua reference point.

Ks\_ext\_NAF is computed in the UICC as  $Ks\_ext\_NAF = KDF(Ks, \text{h1-key-derivation-parameters"gba-me", RAND, IMPI, NAF\_Id})$ , and Ks\_int\_NAF is computed in the UICC as  $Ks\_int\_NAF = KDF(Ks, \text{h1-key-derivation-parameters"gba-u", RAND, IMPI, NAF\_Id})$ , where KDF is the key derivation function as specified in Annex B, and the key derivation parameters include the user's IMPI, the NAF\_Id and RAND. The NAF\_Id consists of the full DNS name of the NAF. The key derivation parameters used for Ks\_ext\_NAF derivation must be different from those used for Ks\_int\_NAF derivation. This is done by adding a static string "gba-me" in Ks\_ext\_NAF and "gba-u" in Ks\_int\_NAF as an input parameter to the key derivation function.

NOTE: The NOTE 2 of clause 4.5.2 also applies here.

The UICC and the BSF store the key Ks with the associated B-TID for further use, until the lifetime of Ks has expired, or until the key Ks is updated.

===== BEGIN NEXT CHANGE =====

## Annex B (normative): Specification of the key derivation function KDF

### B.1 Introduction

This annex specifies the key derivation function (KDF) that is used in the NAF specific key derivation in both GBA (i.e. GBA\_ME) and GBA\_U. The key derivation function defined in the annex takes the following assumptions:

1. the input parameters to the key derivation functions are octet strings - not bit strings of arbitrary length:
2. a single input parameter will have lengths no greater than 65535 octets.

### B.2 Generic key derivation function

The input parameters and their lengths shall be concatenated into a string S as follows:

1. The length of each input parameter in octets shall be encoded into two-octet string:
  - a) express the number of octets in input parameter  $P_i$  as a number  $l$  in the range  $0 \leq l \leq 65535$ .
  - b)  $L_i$  is then a two-octet representation of the number  $l$ , with the most significant bit of the first octet of  $L_i$  equal to the most significant bit of  $l$ , and the least significant bit of the second octet of  $L_i$  equal to the least significant bit of  $l$ ,

EXAMPLE: If  $P_i$  contains 258 octets then  $L_i$  will be the two-octet string 0x01 0x02.

2. String S shall be constructed from  $n$  input parameters as follows:

$$S = FC \parallel P_0 \parallel L_0 \parallel P_1 \parallel L_1 \parallel P_2 \parallel L_2 \parallel P_3 \parallel L_3 \parallel \dots \parallel P_n \parallel L_n$$

where

FC is single octet used to distinguish between different instances of the algorithm,

P0 is a static ASCII-encoded string,

L0 is the two octet representation of the length of the P0,

P1 ... Pn are the n input parameters, and

L1 ... Ln are the two-octet representations of the corresponding input parameters.

- The final output, i.e. the derived key is equal to HMAC-SHA-256 (as specified in [22] and [23]) computed on the string S using the key Key:

derived key = HMAC-SHA-256 ( Key , S )

## B.2.1 Input parameter encoding

A character string shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [24].

---

## B.3 NAF specific key derivation in GBA and GBA\_U

In GBA and GBA\_U, the input parameters for the key derivation function shall be the following:

- FC = 0x01,
- P1 = RAND,
- L1 = length of RAND is 16 octets (i.e. 0x00 0x10),
- P2 = IMPI encoded to an octet string using UTF-8 encoding (see clause B.2.1),
- L2 = length of IMPI is variable (not greater than 65535),
- P3 = NAF\_ID encoded to an octet string using UTF-8 encoding (see clause B.2.1), and
- L3 = length of NAF\_ID is variable (not greater than 65535).

In the key derivation of Ks\_NAF as specified in clause 4 and Ks\_ext\_NAF as specified in clause 5,

- P0 = "gba-me" (i.e. 0x67 0x62 0x61 0x2d 0x6d 0x65), and
- L0 = length of P0 is 6 octets (i.e., 0x00 0x06).

In the key derivation of Ks\_int\_NAF as specified in clause 5,

- P0 = "gba-u" (i.e. 0x67 0x62 0x61 0x2d 0x75), and
- L0 = length of P0 is 5 octets (i.e., 0x00 0x05).

The Key to be used in key derivation shall be:

- Ks (i.e. CK || IK concatenated) as specified in clauses 4 and 5,

NOTE: In the specification this function is denoted as:

$Ks\_NAF = KDF (Ks, "gba-me" \parallel RAND \parallel IMPI \parallel NAF\_Id),$

$Ks\_ext\_NAF = KDF (Ks, "gba-me" \parallel RAND \parallel IMPI \parallel NAF\_Id),$  and

$Ks\_int\_NAF = KDF (Ks, "gba-u" \parallel RAND \parallel IMPI \parallel NAF\_Id).$

===== END CHANGE =====