УДК 681.3: 004.056: 004.414.3: 004.057.2
С.В. Мартыненко, к.ф-м.н.
С.В. Белов

К вопросу реализации стандартов Diffie-Hellman в технических спецификациях Национальной системы электронной подписи Украины

Аннотация: Рассматривается ряд вопросов использования и реализации алгоритма обмена ключами **Diffie-Hellman** в проекте технических спецификаций «Технічні специфікації форматів криптографічних повідомлень», разработанном ГСССЗИ Украины (актуальная версия на 05.02.2010 обсуждалась на общественных слушаниях, организованных Госкомпредпринимательства по вопросам принятия ряда технических спецификаций Национальной системы электронной цифровой подписи). Показано, что необходимо решение ряда вопросов в рамках данной спецификации с учетом действующих международных стандартов. Указанные проблемы и предложенные пути их решения являются необходимым условием при построении национального инфраструктуры электронной подписи и обеспечения совместимости программных продуктов разных разработчиков при проектировании программного обеспечения, использующего шифрование сообщений в системах электронного документооборота Украины.

В данной статье рассмотрены формулировки и определения последней редакции документа «Технічні специфікації форматів криптографічних повідомлень»[1], обсуждавшейся на совещании в Госкомпредпринимательства 4-5.02.2010, дан ряд предложений по решению проблемных вопросов технических спецификаций. Замечания и предложения приводятся в порядке нумерации пунктов указанных спецификаций.

1. Не вполне корректно название документа – оно не соответствует изложенному в нем материалу.

Поскольку в документе изложен лишь тип «enveloped-data» («захищені дані») (п.2.5 Спецификаций), название документа следует изменить на:

«Технічні специфікації форматів криптографічних повідомлень. **Захищені дані».**

2. **п. 2.8**

«Формат повідомлення "підписані дані" ("signed-data") встановлюється технічними специфікаціями форматів підписаних електронних даних»

Здесь необходимо определить OID "signed-data"

3. **п. 4.1** – необходимо определить **UnprotectedAttributes**

UnprotectedAttributes упоминаются в документе, но их определение и описание отсутствует:

«EnvelopedData ::= SEQUENCE {

version	*CMSVersion,*
originatorInfo	*[0] IMPLICIT OriginatorInfo OPTIONAL,*
recipientInfos	*RecipientInfos,*
encryptedContentInfo	*EncryptedContentInfo,*
unprotectedAttrs	*[1] IMPLICIT UnprotectedAttributes OPTIONAL}»*

4. **п. 4.4.2.2.1 ошибочен,**

«При застосуванні механізму автономного узгодження ключів типу Діффі-Геллмана, то в якості ідентифікатора відправника повинні використовуватися ім'я відправника та серійний номер сертифікату відкритого ключа відправника "issuerAndSerialNumber" або ключ шифрування відправника "subjectKeyIdentifier".»

В RFC 3852 [10], 5.3. SignerInfo Type определено:

issuerAndSerialNumber alternative identifies the sender's certificate, and thereby the sender's public key, by the **issuer's distinguished name and the certificate serial number**.

subjectKeyIdentifier alternative identifies the sender's certificate, and thereby the sender's public key, by a key identifier. When an X.509 certificate is referenced, the key identifier matches the X.509 subjectKeyIdentifier extension value.

Т.е. в *issuerAndSerialNumber* верно будет не «ім'я відправника», а должно быть «ім'я видавника», т.е ЦСК, который выпустил сертификат.

и

subjectKeyIdentifier – это **идентификатор ключа субъекта** (ідентифікатор відкритого ключа відправника) и он не может быть «ключем шифрування відправника».

5. **п.4.4.2.2.2**

«При застосуванні механізму автономного узгодження ключів типу Ель-Гамаля, то в якості ідентифікаційних даних відправника застосовується його відкритий ключ (маркер) сеансу "originatorKey".»

следует исключить или привести его OID (механизм Эль-Гамаля) и соответствующие выкладки алгоритмов.

Необходимо учесть, что это «уникальный» алгоритм с точки зрения его стандартизации, т.к. **он не приведен ни в одном соответствующем** (Cryptographic Message Syntax) **RFC** и поэтому его реализация вообще не желательна, т.к. только усложняет решение и делает проблематичным **совместимость разных решений**.

Если ссылка на реализацию алгоритма «Эль-Гамаля» будет сохранена в спецификациях, то также следует обязательно указать, является ли «Эль-Гамаль» обязательным или дополнительным (наряду с обязательным Диффи-Хеллмана).

6. **п.4.4.2.2.4**

«При використанні статичного механізму узгодження ключа, поле "originator" повинно містити ідентифікаційні дані відправника (ім'я видавника сертифікату та серійний номер чи ідентифікатор відкритого ключа видавника).»

содержит ошибку:

Вместо «серійній номер чи ідентифікатор відкритого ключа видавника» должно быть «серійній номер **сертифікату відправника** чи ідентифікатор відкритого ключа видавника».

7. **п.4.4.2.6** сказано, что имеется альтернатива:

«Поле "KeyAgreeRecipientIdentifier" є структурою з вибором альтернативи "issuerAndSerialNumber", що вказує за розпізнавальним ім'ям на центр сертифікації ключів та серійний номер сертифікату відкритого ключа одержувача, що використовується відправником при генерації узгодженого ключа КШК в протоколі Діффі-Геллмана узгодження ключа.»

но она не указана, поэтому необходимо **дополнить описанием второй альтернативы**, т.е. RecipientKeyIdentifier

KeyAgreeRecipientIdentifier ::= CHOICE {
issuerAndSerialNumber IssuerAndSerialNumber,
rKeyId [0] IMPLICIT RecipientKeyIdentifier }

8. **п. 4.4.3.1. ошибочен**:
«Ідентифікатор протоколу (алгоритму) узгодження ключа вказується в полі "EnvelopedData RecipientInfos KeyAgreeRecipientInfo Originator originatorKey algorithm".»

В этом пункте сказано, что идентификатор протокола согласования ключа указывается в поле originatorKey – **что в корне не верно** по следующим причинам:

Согласно RFC 3370 [8], 4.1 Key Agreement Algorithms

Key agreement algorithm identifiers are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm.

Key wrap algorithm identifiers are located in the KeyWrapAlgorithm parameters within the EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm

RFC 3370 4.1.1 X9.42 Ephemeral-Static Diffie-Hellman

keyEncryptionAlgorithm MUST be the id-alg-ESDH algorithm identifier. The algorithm identifier parameter field for id-alg-ESDH is KeyWrapAlgorithm, and this parameter MUST be present. The KeyWrapAlgorithm denotes the symmetric encryption algorithm used to encrypt the content-encryption key with the pairwise key-encryption key generated using the X9.42 Ephemeral-Static Diffie-Hellman key agreement algorithm. Triple-DES and RC2 key wrap algorithms are described in RFC 3217 [WRAP]. The id-alg-ESDH algorithm identifier and parameter syntax is:

id-alg-ESDH OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
alg(3) 5 }

RFC 3370 4.1.2 X9.42 Static-Static Diffie-Hellman

keyEncryptionAlgorithm MUST be the id-alg-SSDH algorithm identifier. The algorithm identifier parameter field for id-alg-SSDH is KeyWrapAlgorihtm, and this parameter MUST be present. The KeyWrapAlgorithm denotes the symmetric encryption algorithm used to encrypt the content-encryption key with the pairwise key-encryption key generated using the X9.42 Static-Static Diffie-Hellman key agreement algorithm. Triple-DES and RC2 key wrap algorithms are described in RFC 3217 [WRAP]. The id-alg-SSDH algorithm identifier and parameter syntax is:

id-alg-SSDH OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
alg(3) 10 }

Из приведенного следует, что идентификатор протокола согласования ключа указывается в поле keyEncryptionAlgorithm, а в параметрах этого идентификатора указывается идентификатор KeyWrapAlgorihtm.

9. **п.п. 4.4.3.2-4.4.3.4**
«4.4.3.2. У Технічних специфікаціях визначаються алгоритми узгодження ключа "id-ESDH-ua" (у циклічній групі простого поля) та "id-ECDH-ua" (в групі точок еліптичної кривої). Алгоритм узгодження ключа ESDH визначається у пункті 5.4

Технічних специфікацій. Алгоритм узгодження ключа ECDH визначається у пункті 5.3 Технічних специфікацій.

4.4.3.3. На використання алгоритму узгодження ключа "id-ESDH-ua" вказує такий об'єктний ідентифікатор:
id-ESDH-ua OBJECT IDENTIFIER ::= {iso(1) member-body(2) Ukraine(804) root(2) security(1) cryptography(1) pki(1) pki-alg(1) pki-alg-asym (3) ESDH-ua (4)}

4.4.3.4. На використання алгоритму узгодження ключа "id-ECDH-ua" вказує такий об'єктний ідентифікатор:
id-ECDH-ua OBJECT IDENTIFIER ::= {iso(1) member-body(2) Ukraine(804) root(2) security(1) cryptography(1) pki(1) pki-alg(1) pki-alg-asym(3) ECDH-ua(3)}»

следует дополнить:

Из предыдущего пункта 9 Замечаний следует, что **требуется определить для режима статик-статик его OID**-ы (id-alg-SSDH-ua для ГОСТ 34.310, соответственно - для ДСТУ 4145 может быть пара **OID: статик-статик и динамик-статик**).

Примечание. Определение двух OID-ов для эллиптических кривых не обязательно, т.к. международные стандарты не предусматривают разные OID-ы для статик-статик и динамик-статик режимов. При этом коллизий не будет, т.к. режим однозначно определяется значением поля originatorKey. Значение OID-а при этом может рассматриваться как дополнительный контроль.

Можно сохранить и текущую редакцию спецификаций, но при этом следует четко указать, что каждый из идентификаторов (*id-ESDH-ua, id-ECDH-ua)* применяется как для статического, так и для динамического режимов.

10. п. 4.4.3.5 **исправить терминологию** (и по тексту спецификаций также):
«Параметри алгоритму ESDH:

ESDHParams::=SEQUENCE {
P INTEGER,
Q(q) INTEGER,
A(a) INTEGER
x0 INTEGER OPTIONAL,
c INTEGER OPTIONAL,
d INTEGER OPTIONAL}»

Указан алгоритм ESDH. Но это не обозначение алгоритма, а обозначение только одного из его режимов, а именно **Ephemeral-Static Diffie-Hellman,** т.е. второй режим SSDH (**Static-Static Diffie-Hellman**) автоматически выпадает.

Вместо «алгоритм ESDH» следует использовать (как это принято в международных стандартах) «алгоритм DH» или «алгоритм FFC DH» (FFC - Finite Field Cryptography). Соответственно, для эллиптических кривых принято использовать «алгоритм ECDH» или «алгоритм ECC DH» (ECC - Elliptic Curve Cryptography).

При разработке спецификаций необходимо учесть стандарт NIST SP 800-56A «Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography», March, 2007 [13].

11. **п. 4.4.3.5** (указан выше) **и 4.4.3.6 –**
«4.4.3.6. Значення полів структури "ESDHParams" наведено у таблиці 3.

Таблиця 3.

P	*характеристика основного поля*
Q	*порядок циклічної підгрупи*
A	*твірний елемент циклічної підгрупи*
x0	*початковий стан, що використовувався для генерації p, q*
C	*параметр датчика, що використовувався для генерації p, q.*
D	*довільне число, що використовувалося для генерації a, 1< d < p – 1*

»

следует исключить. Эти пункты далее дублируются в п.п. 4.5.3.2, 4.5.3.3, но в других обозначениях, что приведет только к путанице, не добавив никакой информативности.

«4.5.3.2. Параметри алгоритму "ESDH" повинні бути представлені такою структурою:

```
ESDHParameters ::=  SEQUENCE {
p                   INTEGER,
q                   INTEGER,
a                   INTEGER,
x0                  INTEGER OPTIONAL,
c                   INTEGER OPTIONAL,
d                   INTEGER OPTIONAL }
```

4.5.3.3. Значення полів структури "ESDHParameters" наведено у таблиці 1.

Таблиця 1.

p	*модуль, просте число 21020 < p < 21024*
q	*порядок циклічної групи, просте число 2254 < q < 2256, є дільником для (p – 1)*
a	*твірний елемент циклічної групи, 1< a < p – 1, при цьому aq(mod p) = 1*
x0	*початковий стан, що використовувався для генерації p, q*
c	*параметр датчика, що використовувався для генерації p, q.*
d	*довільне число, що використовувалося для генерації a, 1< d < p – 1*

»

12. **п.4.4.3.7 сформулирован крайне некорректно**, указана лишь часть необходимой информации, а относительно второй чести - «догадайся мол сам…». Во-вторых, это относится исключительно к FFC DH (т.е. ГОСТ 34.310 и DSA) и не относится к ECC DH (т.е. ДСТУ 4145 и ECDSA).

«4.4.3.7. При використанні динамічного механізму узгодження ключа (згідно з пунктом 6.2 національного стандарту України ДСТУ ISO/IEC 11770-3), поле "originatorKey publicKey" повинно містити відкритий ключ відправника (маркер), що має такий формат:

PublicKey:: = INTEGER, що інкапсулюється в BIT STRING»

Такое изложение только вносит путаницу и не дает никакой ясности, что обязательно в дальнейшем приведет к несовместимости решений.

Т.к. originatorKey :: = [1] OriginatorPublicKey

```
OriginatorPublicKey ::= SEQUENCE {
    algorithm   AlgorithmIdentifier,
    publicKey   BIT STRING }
```

то следует определить не только publicKey, но и algorithm

Для FFC DH имеем:

Согласно RFC 3370 [8], п. 4.1.1 X9.42 Ephemeral-Static Diffie-Hellman

originator MUST be the originatorKey alternative. The originatorKey algorithm field MUST contain the dh-public-number object identifier **with absent parameters**. The originatorKey publicKey field MUST contain the sender's ephemeral public key. The dh-public-number object identifier is:

dh-public-number OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) ansi-x942(10046) number-type(2) 1 }

Откуда следует, что идентификатор алгоритма (для DSA) должен быть dh-public-number и параметры должны отсутствовать (with absent parameters).

Важное примечание: Отсутствие параметров – это не ASN1Null, а отсутствие поля параметров в целом, и это надо четко указать.

Для ECC DH имеем:

RFC 3278 [5]:

The originatorKey algorithm field MUST contain the id-ecPublicKey object identifier (see Section 8.1) with NULL parameters. The originatorKey publicKey field MUST contain the DER-encoding of a value of the ASN.1 type ECPoint (see Section 8.2), which represents the sending agent's ephemeral EC public key.

8.1 Algorithm identifiers

When the object identifier id-ecPublicKey is used here with an algorithm identifier, the associated parameters contain NULL.

ansi-X9-62 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) 10045 }

id-public-key-type OBJECT IDENTIFIER ::= { ansi-X9.62 2 }

id-ecPublicKey OBJECT IDENTIFIER ::= { id-publicKeyType 1 }

Откуда следует, что идентификатор алгоритма (для ECDSA) должен быть id-ecPublicKey, а параметрами должен быть ASN1Null.

Относительно ГОСТ 34.310 и ДСТУ 4145 см. ниже.

Как общее предложение по структуре спецификаций – рекомендуется спецификацию четко разделить на два раздела – один для FFC DH (т.е. ГОСТ 34.310 и DSA), второй для ECC DH (т.е. ДСТУ 4145 и ECDSA).

13. **п.4.4.3.9 содержит ошибку**

«4.4.3.9. При використанні статичного механізму узгодження ключа згідно з пунктом 6.1 національного стандарту України ДСТУ ISO/IEC 11770-3, поле "originator" повинно містити ідентифікаційні дані відправника (ім'я видавника, який виготовив сертифікат) та серійний номер чи ідентифікатор відкритого ключа видавника.»

Ошибка та же, что и в п.4.4.2.2.4:

Вместо «серійній номер чи ідентифікатор відкритого ключа видавника» должно быть «серійній номер **сертифікату відправника** чи ідентифікатор відкритого ключа видавника».

14. п.4.5.1 **и ошибочен, и не корректен**:

«4.5.1. Формат сертифікату шифрування повинен відповідати формату сертифіката відкритого ключа, визначеному в розділі 1 Технічних специфікацій форматів представлення базових об'єктів національної системи електронного цифрового підпису, затверджених наказом Департаменту спеціальних телекомунікаційних систем та захисту інформації Служби безпеки України та Державного департаменту з питань зв'язку та інформатизації Міністерства транспорту та зв'язку України від 11.09.2006 № 99/166, за виключенням полів "subjectPublicKeyInfo".

В сертифікаті шифрування додатково від сертифікату відкритого ключа повинно бути в розширенні "Використання ключа" ("KeyUsage") встановлено значення "Узгодження ключа" ("keyAgreement").»

Последние слова первого абзаца, а именно: «за виключенням полів SubjectPublicKeyInfo», должны быть исключены, т.к. нет никакого «виключення» для SubjectPublicKeyInfo.

Во втором абзаце необходимо сформулировать фразу так, чтобы она не требовала «перевода» (перекладу):

Вместо не имеющей смысла фразы «додатково від сертифікату відкритого ключа» сформулировать «У сертифікаті шифрування повинно бути в розширенні …» и далее по тексту.

Относительно сертификата ключа шифрования – следует также указать, что параметры открытого ключа, в том числе ДКЕ, должны (<u>обязательно</u>) присутствовать (согласно «Форматов объектов» - они опциональны).

15. п. 4.5.2 необходимо кардинально переработать.

«Відкритий ключ та параметри криптографічних алгоритмів розміщуються в полі "subjectPublicKeyInfo".»

В том виде, в котором он сформулирован, он не имеет смысла.

По определению

```
SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm         AlgorithmIdentifier,
    subjectPublicKey  BIT STRING }
```

т.е. открытый ключ содержится (обязательно!) в поле subjectPublicKey, а параметры и идентификатор ключа содержатся в поле algorithm (обязательно!). Для ГОСТ 34.310 параметры алгоритма – это полные доменные параметры, для ДСТУ 4145 параметры алгоритма – это либо доменные параметры, либо OID кривой.

В этом пункте следует четко объяснить**, что такое сертификат ключа Диффи-Хеллмана**, т.к. опыт показывает, что у большинства разработчиков в этом вопросе возникает непонимание.

Во всех стандартах однозначно сказано, что, например, для ECC DH,

RFC 3279 [6], 2.3.5 ECDSA and ECDH Keys

The ECDSA and ECDH specifications use the same OIDs and parameter encodings.

When certificates contain an ECDSA or ECDH public key, the id-ecPublicKey algorithm identifier MUST be used.

This OID is used in public key certificates for both ECDSA signature keys and ECDH encryption keys. The intended application for the key may be indicated in the key usage field (see [RFC 3280]).

This OID is used in public key certificates for both ECDSA signature keys and ECDH encryption keys. The intended application for the key may be indicated in the key usage field (see [RFC 3280]).

Откуда следует, что сертификаты ключей ECDSA или ECDH отличаются между собой лишь полем keyUsage. Значение OID у них одинаково и соответствует OID ECDSA. Следовательно, применяется сертификат ECDSA ключа с указанным в нем keyUsage для ECDH. Именно keyUsage определяет то, что этот ключ ECDSA является ключом ECDH.

Подобное разъяснение должно быть приведено в этом пункте Спецификаций 4.5.2:

Сертификатами Diffie-Hellman являются сертификаты DSA и ГОСТ 34.310 (для FFC DH), и ECDSA и ДСТУ 4145 (для ECC DH) с указанным обязательно использованием ключа keyUsage согласно п.4.5.1.

Примечание. В последней версии стандарта RFC 5480 (Elliptic Curve Cryptography Subject Public Key Information, март 2009 [12]) предусмотрено три варианта OID поля SubjectPublicKeyInfo для сертификатов шифрования:

Первый – как указано выше (идентификатор id-ecPublicKey) – обязательное решение, показывает, что алгоритмы, которые могут использовать открытый ключ, не ограничены (indicates that the algorithms that can be used with the subject public key **are unrestricted**)

Второй – идентификатор id-ecDH – дополнительное решение, показывает, что алгоритмы, которые могут использовать открытый ключ, ограничены - только ECDH алгоритм может применяться для этого сертификата (indicates that the algorithm that can be used with the subject public key is restricted to the Elliptic Curve Diffie-Hellman algorithm. See Section 2.1.2. id-ecDH MAY be supported)

Третий – идентификатор id-ecMQV - дополнительное решение, показывает, что алгоритмы, которые могут использовать открытый ключ, ограничены - только ECDH алгоритм может применяться для этого сертификата (indicates that the algorithm that can be used with the subject public key is restricted to the Elliptic Curve Menezes-Qu-Vanstone key agreement algorithm. See Section 2.1.2. id-ecMQV MAY be supported).

Таким образом, используя в сертификатах идентификатор id-ecPublicKey, мы реализуем:

1. Обязательное решение (согласно требованиям стандартов)
2. Получаем возможность применять как для ECDSA, так и для ДСТУ 4145
3. Уходим от сложной задачи формирования запроса на получение сертификата, которая имеет место для второго и третьего вариантов (см. RFC 5273: Certificate Management over CMS (CMC), - June 2008).

Аналогичное решение должно быть применено и для FFC DH (DSA, ГОСТ 34.310).

16. п. **4.5.3.2**

«4.5.3.2. Параметри алгоритму “ESDH” повинні бути представлені такою структурою:

```
ESDHParameters ::= SEQUENCE {
p                  INTEGER,
q                  INTEGER,
a                  INTEGER,
x0                 INTEGER OPTIONAL,
```

c *INTEGER OPTIONAL,*
d *INTEGER OPTIONAL }»*

необходимо изменить структуру параметров, следуя международным стандартам. Не следует изобретать «**национальный Diffie-Hellman»!!!**

Согласно RFC 3370 [8]:

```
DHDomainParameters ::= SEQUENCE {
    p             INTEGER,  -- odd prime, p=jq +1
    g             INTEGER,  -- generator, g
    q             INTEGER,  -- factor of p-1
    j             INTEGER OPTIONAL,  -- subgroup factor
    validationParms    ValidationParms OPTIONAL }


ValidationParms ::= SEQUENCE {
    seed       BIT STRING,
    pgenCounter INTEGER }
```

Эта структура параметров соответствует Diffie-Hellman на ключах DSA алгоритма. Для адаптации структуры параметров к ГОСТ 34.310 необходимо определить соответствие так:

p – «характеристика основного поля»
g - «твірний елемент»
q – «порядок циклічної групи»
j (subgroup factor) – не используется и должен быть опущен.

```
validationParms  ::=  GOST34310ValidationParms OPTIONAL

GOST34310ValidationParms ::= SEQUENCE {
   x0  INTEGER, -- unconfigured state
    c   INTEGER, -- parameter of detector/ sensor, odd
    d   INTEGR OPTIONAL -- parameter of procedure C
```

Следует ли здесь вообще упоминать о параметрах валидации, которые отсутствуют в основной спецификации «Форматы базовых объектов …» и об обработке (как и когда?), о которых нигде ничего не сказано (вопрос «куда их вставить…?»)?

Поэтому GOST34310ValidationParms могут быть опущены или требуется четкое описание процедуры валидации (кто, когда и как выполняет).

17. **Аналогично замечание по п.4.5.3.2 относится и к п. 4.5.4.2** - не следует изобретать «**национальный Diffie-Hellman»!!!**

Согласно RFC 5480 [12], 2.1.1. Unrestricted Algorithm Identifier and Parameters

The parameter for id-ecPublicKey is as follows and MUST always be present:

```
ECDHParameters ::= CHOICE {
    namedCurve         OBJECT IDENTIFIER
    -- implicitCurve    NULL
    -- specifiedCurve   SpecifiedECDomain
 }
 -- implicitCurve and specifiedCurve MUST NOT be used in PKIX.
```

Откуда следует, что в качестве параметров указывается исключительно лишь OID кривой, а не ее полные доменные параметры.

18. **п.4.5.4.4**

«4.5.4.4. Кодування полів параметрів еліптичної кривої, що мають тип "OCTET STRING".

4.5.4.4.1. Кодування коефіцієнту B еліптичної кривої, базової точки еліптичної кривої "bp", а також відкритого ключа, здійснюється за форматом "Little-Endian". Представлення байтів повинно здійснюватися у прямому порядку.

4.5.4.4.2. На застосування алгоритму "ECDH" у поліноміальному базисі вказує такий об'єктний ідентифікатор:
id-ECDH-ua PB(1)

4.5.4.4.3. На застосування алгоритму "ECDH" у оптимальному нормальному базисі вказує такий об'єктний ідентифікатор:
id-ECDH-ua ONB(2)

4.5.4.4.4. Для зображення базової точки еліптичної кривої "bp" використовується такий формат:
bp OCTET STRING

4.5.4.4.5. Базова точка еліптичної кривої "bp" кодується згідно з національним стандартом України ДСТУ 4145-2002 та представляє собою послідовність байтів, що становить елемент основного поля (згідно з пунктом 5.3 національного стандарту України ДСТУ 4145-2002), який є стиснутим зображенням (згідно з пунктом 6.9 національного стандарту України ДСТУ 4145-2002) точки на еліптичній кривій (залежить від базису, що використовується). Розмір зображення у байтах дорівнює m/8 заокругленого до найближчого цілого у більшу сторону.

4.5.4.4.6. Для зображення коефіцієнта "B" еліптичної кривої використовується такий формат:
b OCTET STRING

4.5.4.4.7. Коефіцієнт "B" еліптичної кривої кодується згідно з національним стандартом України ДСТУ 4145-2002. Це послідовність байтів, яка становить елемент основного поля (згідно з пунктом 5.3 національного стандарту України ДСТУ 4145-2002). Розмір зображення в байтах дорівнює m/8 заокругленого до найближчого цілого у більшу сторону.

4.5.4.4.9. Для зображення відкритого ключа використовується формат:
PublicKey:: = OCTET STRING , що інкапсулюється в BIT STRING

4.5.4.4.10. Відкритий ключ кодується згідно з національним стандартом України ДСТУ 4145-2002. Це послідовність байтів, обчислена відповідно до пункту 9.2 ДСТУ 4145-2002, яка становить елемент основного поля (згідно з пунктом 5.3 національного стандарту України ДСТУ 4145-2002), який є стиснутим зображенням (згідно з пунктом 6.9 національного стандарту України ДСТУ 4145-2002) точки на еліптичній кривій, що відображає відкритий ключ електронного цифрового підпису. Розмір зображення в байтах дорівнює m/8 заокруглене до найближчого цілого у більшу сторону.

4.5.4.4.11. Особистий ключ обчислюється відправником для кожного повідомлення відповідно до пункту 9.1 національного стандарту України ДСТУ 4145-2002. «

необходимо исключить большинство подпунктов (с учетом п. 4.5.4.3)

В п.4.5.4.4.2-4.5.4.4.2 введены OID-ы для разных базисов (полиномиальный/нормальный). Это не имеет смысла, т.к. согласно п.17 этих замечаний **в качестве параметров должен указываться OID, а не полные доменные параметры**. А OID кривой однозначно определяет ее базис.

Следует оставить лишь п.п.4.5.4.4.9-4.5.4.4.11, но привести их в соответствие стандартам.

Следует также привести в соответствие стандартам форму представления ключа – согласно ДСТУ 4145, основной формой является **не сжатая, но допускается (?) (см. п.9.2 ДСТУ 4145) и сжатая форма:**

«Припускається зберігання й передача відкритого ключа цифрового підпису у стисненому вигляді. Стискання відкритого ключа цифрового підпису виконують згідно з 6.9, відновлення відкритого ключа виконують згідно з 6.10».

Т.е. из определения стандарта ДСТУ 4145-2002 следует, что сжатая форма является «***допустимой***», и в таком случае **«несжатая» является основной.**

Такой подход соответствует **стандарту RFC 3279** [6]:

Implementations that support elliptic curve according to this specification MUST support the uncompressed form and MAY support the compressed form.

Такой подход соответствует и **стандарту RFC 5008** [18]:

The originatorKey publicKey field MUST contain the message originator's ephemeral public key, which is a DER-encoded ECPoint (see Section 3). The ECPoint SHOULD be represented in uncompressed form.

Такой подход также соответствует **стандарту RFC 5480** [12], 2.2. Subject Public Key:

The subjectPublicKey from SubjectPublicKeyInfo is the ECC public key. ECC public keys have the following syntax:

ECPoint ::= OCTET STRING

Implementations of Elliptic Curve Cryptography according to this document **MUST support the uncompressed form and MAY support the compressed form** of the ECC public key. The hybrid form of the ECC public key from [X9.62] MUST NOT be used.

При этом и порядок (прямой/ обратный) кодирования также однозначно определен (в отличие от ДСТУ 4145) стандартами:

The elliptic curve public key (an OCTET STRING) is mapped to a subject public key (a BIT STRING) as follows: the most significant bit of the OCTET STRING becomes the most significant bit of the BIT STRING, and the least significant bit of the OCTET STRING becomes the least significant bit of the BIT STRING.

Таким образом, следуя стандартам, основной должна быть несжатая форма (uncompressed form), сжатая форма допускается как дополнительная.

Однако, несмотря на то, что несжатая форма открытого ключа, вероятно, все же определена Стандартом ДСТУ 4145-2002, в Технических спецификациях базовых объектов (Приказ № 99/166 [16]) определен **только** формат «сжатого» ключа (п.1.3.11.5 «Відкритий ключ»):

«Відкритий ключ ДСТУ 4145 2002 – це послідовність байтів, яка являє собою елемент основного поля (згідно пункту 5.3 ДСТУ 4145-2002), який є стиснутим зображенням (згідно пункту 6.9 ДСТУ 4145-2002) точки на еліптичній кривій, що відображає відкритий ключ ЕЦП»

К тому же в форматах Спецификаций **отсутствует байт признака** формы представления открытого ключа, что не соответствует международным стандартам, в которых четко определено, что сжатая форма – признак 02 или 03 (в зависимости от знака), несжатая – 04, гибридная – 06.

В международных стандартах, посвященных реализации алгоритмов ECDSA и Diffie-Hellman (RFC 5480 [12], ANSI X9.62 [17] п. 4.3.6 Point-to-Octet-String Conversion) проверка структуры сертификата и наличия признаков является обязательным, как и программным обеспечением мировых разработчиков (Microsoft, SUN, Oracle Java), которое воспринимает такой «урезанный» формат как ошибку в структуре и прекращает какие-либо действия с открытым ключом. Здесь можно сказать только одно – разработчикам национальных спецификаций не помешало бы изучение соответствующих международных стандартов, тогда было бы меньше проблем с реализациями национальных спецификаций.

19. **п.п.4.5.4.5-4.5.4.6**

«4.5.4.5. Параметри алгоритму ECDH:

```
ECDHParams::= SEQUENCE {
version     [0] EXPLICIT INTEGER    DEFAULT 0,
f           BinaryField,
a           INTEGER (0..1),
b           OCTET STRING,
n           INTEGER,
bp          CHOICE {OCTET STRING, NULL}}

BinaryField ::= SEQUENCE {
        M                   INTEGER,
        CHOICE {
                    Trinomial,
                    Pentanomial}

Trinomial::= INTEGER

Pentanomial::= SEQUENCE {
K       INTEGER,
j       INTEGER,
l       INTEGER }
```

Значення полів структури “ECDHParams” наведено у таблиці 4.

Таблиця 4.

f	*основне поле*
a	*коефіцієнт A еліптичної кривої*
b	*коефіцієнт B еліптичної кривої*
n	*порядок базової точки (додатне ціле)*
bp	*базова точка еліптичної кривої або ознака її відсутності*
M INTEGER,	*ступінь розширення основного поля*
Trinomial	*примітивний тричлен*
Pentanomial	*примітивний п’ятичлен*

4.5.4.6. Кодування полів здійснюється відповідно до пункту 4.5.4.4 Технічних специфікацій.»

следует исключить, как не соответствующие стандарту, и к тому же лишние - дублируют п. 4.5.4.2.

20. п.4.6.1. ошибочен:

«4.6.1. Ідентифікатор алгоритму захисту ключа шифрування даних "KeyWrapAlgorithm" та пов'язані з ним параметри вказуються в полі "EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm". Ключ узгодження КШК формується за механізмами (протоколами) узгодження ключа ESDH або ECDH:

KeyWrapAlgorithm ::= AlgorithmIdentifier

»

Детали указаны в п.8 этих замечаний.

Следовательно, идентификатор алгоритма защиты ключа шифрования данных указывается не в поле keyEncryptionAlgorithm. В этом поле указывается **идентификатор протокола согласования ключа, а в параметрах идентификатора протокола согласования ключа указывается идентификатор KeyWrapAlgorihtm.**

Кроме того, в последнем предложении первого абзаца сказано, что ключ согласования КШК формируется по механизмам … ESDH или ECDH, тем самым мы ограничили решение только динамическим режимом, а статический (статик-статик) выпал.

Это последнее предложение следует исключить.

21. п.4.6.2-4.6.5 **требуют существенной доработки.**

4.6.2. "KeyWrapAlgorithm algorithm" повинен містити ідентифікатор алгоритму id-Key-Wrap-ua:

id-Key-Wrap-ua OBJECT IDENTIFIER ::= {iso(1) member-body(2) Ukraine(804) roo t (2)security(1) cryptography(1) pki(1) pki-alg(1) pki-alg-extra (4) key-wrap-algo (1) }

4.6.3. Синтаксис поля "KeyWrapAlgorithm algorithm parameters" такий:

```
UAKeyWrapParameters ::= SEQUENCE {
        dke                 Gost28147-89-DKE,
        shiftBits           INTEGER { gost28147-89-block(64) }
}
Gost28147-89-DKE ::= OCTET STRING (SIZE (64))
```

4.6.4. Значення полів структури "Gost28147-89-ParamSet" наведено у таблиці 5.

Таблиця 5.

Dke	*довгостроковий ключовий елемент*
ShiftBits	*Параметри шифрування*

4.6.5. Довгостроковий ключовий елемент обирається з додатку 1 до Інструкції про порядок постачання і використання ключів до засобів криптографічного захисту інформації, затвердженої наказом Адміністрації Держспецзв'язку від 12.06.2007 № 114, зареєстрованим в Міністерстві юстиції України за № 729/13996 від 25.06.2007.

Довгостроковий ключовий елемент кодується в упакованому форматі, відповідно до пункту 1.3.12.1 Технічних специфікацій форматів представлення базових об'єктів.

»

В том виде, в котором изложен алгоритм KeyWrapAlgorihtm, **эти пункты бессмысленны, т.к. не дают никакого представления и даже намека** о том, какой же следует применять базовый алгоритм из трех возможных – простой замены, гаммирования или гаммирования с обратной связью, или вообще иной?

Во-вторых, предлагается использовать термины, применяемые в стандарте ГОСТ 28147 или их аналоги по международным стандартам.

Так в п.4.6.4 используется термин «параметр шифрования». Какое-либо разъяснение этого «параметра» отсутствует. Может быть это «синхропосылка» (согласно ГОСТ 28147), или по международным стандартам – это «инициализирующий вектор», указанный в п.4.7.2?

В-третьих, лишено здравого смысла в рамках одного **базового** алгоритма (ГОСТ 28147) изобретать две разные структуры параметров, как это предлагается согласно п.4.6.3 и п.4.7.2.

В-четвертых, мы уже имеем один ДКЕ из открытого ключа сертификата отправителя (для статик режима) или можем его взять из сертификата/ ключа получателя (для динамического режима). Какой смысл утяжелять конверт еще одним ДКЕ в параметрах KeyWrapAlgorihtm (выше было сказано о том, что все равно следует четко сформулировать требования к сертификату ключа шифрования, в которых необходимо указать, что ДКЕ обязателен!)?

Однозначный вывод – как лишний ДКЕ, так и «параметр шифрования» должны быть исключены, т.е. в результате параметры KeyWrapAlgorihtm должны отсутствовать.

Предлагаем в Спецификациях учесть международные аналоги алгоритма Key Wrap:

RFC 2630 [2], 12.3.3.1 **Triple-DES Key Wrap**
RFC 2630, 12.3.3.2 **RC2 Key Wrap**
RFC 3394 [9], **AES Key Wrap** Algorithm

22. **п.4.6.6 –**

«4.6.6. Зашифрований симетричний ключ КШД розміщується в полі “EnvelopedData RecipientInfos KeyAgreeRecipientInfo RecipientEncryptedKeys encryptedKey”.
EncryptedKey ::= OCTET STRING

Поле “encryptedKey” повинно інкапсулювати “Gost28147-89-EncryptedKey”
Gost28147-89-EncryptedKey ::= SEQUENCE {
encryptedKey Gost28147-89-Key,
macKey Gost28147-89-MAC}

Gost28147-89-MAC ::= OCTET STRING (SIZE (1..4))»

следует кардинально изменить (или описать назначение и правила обработки лишних «наворотов» типа macKey с длиной от 1 до 4 байт (что это?). Если это имитовставка, т.е. MAC, то длина не может быть такой, которая указана!)

Этим пунктом 4.6.6 предлагается зашифрованный ключ помещать дополнительно в некоторую ASN1 структуру. Если назначением этой структуры является проверка правильности расшифрования ключа, то это решается в рамках алгоритма Key Wrap (снова рекомендуем ознакомиться с RFC 2630, где контрольная сумма, checksum, добавляется к зашифрованному ключу в виде конкатенации, что не требует формирования дополнительной ASN1 структуры), который и следует детально описать, как сказано выше. Тогда не потребуется строить лишние ASN1 структуры.

23. **п.5.3.2**

«5.3.2. КШК становить собою 256-бітове геш-значення, що обчислюється від 1024-бітового значення розділеної таємниці згідно з пунктом 6.1 або пунктом 6.2 національного стандарту України ДСТУ ISO/IEC 11770-3) у такій послідовності.»

необходимо уточнить или исправить:

Указано, что длина общего секрета (shared secret) равна 1024 бита. Но если мы используем ключи длины 512 бит (что допускается стандартом), то мы не можем получить общий секрет длины 1024 бита!

24. **п.5.3.2.2.1**

«5.3.2.2.1. При статичному механізмі узгодження ключа, КШК формується таким чином:

КШК(x,y) = ГОСТ 34.311 (K(x,y) | UKM);

Під час конкатенації ціле число K(x,y) використовуються, як послідовність байт у форматі LittleEndian (тобто, старші розряди числа містяться у байтах з більшими адресами). При цьому, не враховуються старші нульові байти.
ДКЕ (заповнення S-Box) обирається з "KeyWrapAlgorithm KeyWrapParameters dke";
В якості вектора ініціалізації ГОСТ 34.311 використовується нульовий вектор.
UKM – 8 випадкових октетів (байт).»

предлагается привести в соответствие международным стандартам.

Вычисление хеш функции от K(x,y)|UKM, где K(x,y) есть общий секрет (shared secret) и UKM - случайное 8 байтовое значение, не соответствует стандарту:

Согласно RFC 2630 [2]**, 2.1.2. Generation of Keying Material:**

X9.42 provides an algorithm for generating an essentially arbitrary amount of keying material from ZZ. Our algorithm is derived from that algorithm by mandating some optional fields and omitting others.

KM = H (ZZ || OtherInfo)

H is the message digest function SHA-1

ZZ is the shared secret value computed in Section 2.1.1. Leading zeros MUST be preserved, so that ZZ occupies as many octets as p. For instance, if p is 1024 bits, ZZ should be 128 bytes long.

OtherInfo is the DER encoding of the following structure:

```
OtherInfo ::= SEQUENCE {
  keyInfo KeySpecificInfo,
  partyAInfo [0] OCTET STRING OPTIONAL,
  suppPubInfo [2] OCTET STRING

}

KeySpecificInfo ::= SEQUENCE {
  algorithm OBJECT IDENTIFIER,
  counter OCTET STRING SIZE (4..4) }
```

Эта функция применяется для FFC DH. Для ECC DH используется такая (X9.63, RFC 3278 [5], RFC 5008 [18]):

KM = Hash (Z || Counter || ECC-CMS-SharedInfo)

```
ECC-CMS-SharedInfo  ::=  SEQUENCE {
      keyInfo      AlgorithmIdentifier,
      entityUInfo  [0] EXPLICIT OCTET STRING OPTIONAL,
      suppPubInfo  [2] EXPLICIT OCTET STRING }
```

Рекомендуется не изобретать новый Diffie-Hellman, а следовать стандартам, применяемым в CMS (Cryptographic Message Syntax).

25. **п.5.3.2.2.2 предлагается привести в соответствие международным стандартам**

«5.3.2.2.2. При динамічному механізмі узгодження ключа, КШК формується таким чином:

КШК(x,y) = ГОСТ 34.311 (K(x,y) | UKM);

Порядок обчислення КШК(x,y) наведено у п. 4.3.2.2.1.
ключові пари (x,a^x) та (y,a^y) повинні відповідати ГОСТ 34.310-95 (для 1020< p<1024 біт).»

Аналогично предыдущему пункту замечаний.

Дополнительно, в этом пункте впервые (как бы вскользь) указано ограничение по длине ключа для ГОСТ 34.310 – от 1020 до 1024 бит, но только для динамического режима. Касается ли это ограничение статического режима?

Это принципиальный момент, который должен быть вынесен в отдельный пункт (при начальном определении режимов).

26. **п.5.4.4.2**

«5.4.4.2. Обчислюється 256-бітове геш-значення:

КШК(A,B) = ГОСТ 34.311 (x_K | UKM).

Під час конкатенації координата точки еліптичної кривої x_K використовуються, як послідовність байт у форматі LittleEndian (тобто, старші розряди містяться у байтах з більшими адресами). При цьому, не враховуються старші нульові байти.
ДКЕ обирається з "KeyWrapAlgorithm KeyWrapParameters dke".
В якості вектора ініціалізації gost34.311 використовується нульовий вектор.
UKM – 8 випадкових октетів (байт).»

предлагается привести в соответствие международным стандартам

Во-первых, относительно вычисления хеш функции от K(x,y)|UKM см. выше пункт 24 замечаний.

Во-вторых, относительно выбора ДКЕ. Мы уже имеем один ДКЕ в ключе получателя. Какой смысл утяжелять конверт еще одним ДКЕ в параметрах KeyWrapAlgorihtm и брать его оттуда (см. также выше замечания по формату сертификата ключа шифрования).

Необходимо указать, что ДКЕ берется из ключа получателя, если же допускается отсутствие ДКЕ в ключе – то берется №1 из рекомендованных (по умолчанию).

27. Очень важный момент **– нет определения, в чем же отличие двух режимов** – статического и динамического, и **отсутствуют критерии их выбора**.

Согласно стандартов, статический режим применим только тогда, когда доменные параметры ключей отправителя и получателя эквивалентны. В наших отечественных стандартах к доменным параметрам добавляется еще и ДКЕ.

Вопрос: ДКЕ отправителя и получателя должны быть одинаковы или нет для статик режима? Или можно в случае их отличия использовать ДКЕ (как и доменные параметры) получателя?

Требуется четкий ответ в виде отдельного пункта в спецификации.

28. **п.5.4.5 исключить**, т.к. он не несет в себе никакой дополнительной информации и только утяжеляет документ.

«5.4.5. Порядок формування КШК при динамічному механізмі узгодження ключа.»

Общий вывод:

Реализация по указанной спецификации невозможна, требуется существенная доработка, в том числе приведение в соответствие стандартам.

29. **Дополнение: один из наиболее важных моментов – это описание OID-ов в структуре EnvelopedData для Diffie-Hellman**.

Чтобы наступила полная ясность в этом вопросе, необходимо четко определить назначение каждого из OID-ов структуры KeyAgreeRecipientInfo, откуда, в свою очередь, будет следовать область его применения.

Итак, имеем два режима:
Ephemeral-Static Diffie-Hellman
Static-Static Diffie-Hellman

Рассмотрим Ephemeral-Static Diffie-Hellman

Динамический ключ передается в структуре OriginatorPublicKey.

```
OriginatorIdentifierOrKey ::= CHOICE {
    issuerAndSerialNumber   IssuerAndSerialNumber,
    subjectKeyIdentifier    [0] SubjectKeyIdentifier,
    originatorKey           [1] OriginatorPublicKey }

OriginatorPublicKey ::= SEQUENCE {
     algorithm   AlgorithmIdentifier,
     publicKey   BIT STRING }
```

Здесь имеем **первый OID** (алгоритма открытого ключа отправителя), равный (пока берем только международные алгоритмы):
dh-public-number для DSA/DH, и
id-ecPublicKey для ECDSA/ECDH.

Этот OID, т.к. он характеризует открытый ключ, предназначен для создания объекта «открытый ключ» заданного через этот OID алгоритма. В дальнейшем этот открытый ключ будет использоваться для генерации общего секрета (shared secret).

Назовем этот OID как «OID открытого ключа отправителя».

Для того, чтобы четко определить, какой из ключей необходимо использовать, например для алгоритмов FFC DH - DSA или ГОСТ 34.310, достаточно в спецификации в качестве «OID открытого ключа отправителя» для ГОСТ 34.310 использовать непосредственно OID ГОСТ 34.310.

Такой же подход применяется в решении КриптоПро (RFC 4490 [11], 4.1.1. Key Agreement Algorithms Based on GOST R 34.10-94/2001 Public Keys):

The originator MUST be the originatorKey alternative. The originatorKey algorithm field MUST contain the object identifier id-GostR3410-94 or id-GostR3410-2001 and corresponding parameters (defined in Sections 2.3.1, 2.3.2 of [CPPK]).

The originatorKey publicKey field MUST contain the sender's public key.

Таким образом, в качестве «OID открытого ключа отправителя» используется OID «родительского» алгоритма, т.е. ГОСТ 34.310, что однозначно определяет формат открытого динамичного ключа. В статическом режиме алгоритм содержится в сертификате и, как сказано выше, должен быть также OID «родительского» алгоритма, т.е. ГОСТ 34.310. Аналогично следует делать для ДСТУ 4145. При этом для ДСТУ вопрос более актуален, т.к. мы имеем четыре формата открытого ключа (ДСТУ ПБ/ ДСТУ ОНБ в прямой и обратно кодировке каждый), т.е. соответственно четыре OID-ов.

Второй OID структуры KeyAgreeRecipientInfo передается в keyEncryptionAlgorithm:

```
keyEncryptionAlgorithm  KeyEncryptionAlgorithmIdentifier

KeyEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier

AlgorithmIdentifier  ::= SEQUENCE{
   algorithm  OBJECT IDENTIFIER,
   parameters ANY DEFINED BY algorithm OPTIONAL
 }
```

Этот OID, как указано выше, может принимать следующие значения:

1) для FFC DH (DSA) (RFC 3370 [8])
   i. id-alg-ESDH
   ii. id-alg-SSDH
2) Для ECC DH (ECDSA) (RFC 3278 [5])
   i. dhSinglePass-stdDH-sha1kdf-scheme
   ii. dhSinglePass-cofactorDH-sha1kdf-scheme
   iii. mqvSinglePass-sha1kdf-scheme
   а также (RFC 5008 [18])
   iv. dhSinglePass-stdDH-sha256kdf-scheme
   v. dhSinglePass-stdDH-sha384kdf-scheme
   а также (SEC1 v.2 [20])
   vi. dhSinglePass-stdDH-sha224kdf-scheme
   vii. dhSinglePass-stdDH-sha512kdf-scheme
   viii. dhSinglePass-cofactorDH-sha224kdf-scheme
   ix. dhSinglePass-cofactorDH-sha256kdf-scheme
   x. dhSinglePass-cofactorDH-sha384kdf-scheme
   xi. dhSinglePass-cofactorDH-sha512kdf-scheme
   xii. … (для MQV)

Согласно RFC 3278 [5]:

keyEncryptionAlgorithm MUST contain the dhSinglePass-stdDH-sha1kdf-scheme object identifier (see Section 8.1) if standard ECDH primitive is used, or the dhSinglePass-cofactorDH-sha1kdf-scheme object identifier (see Section 8.1) if the cofactor ECDH primitive is used. The parameters field contains KeyWrapAlgorithm. The KeyWrapAlgorithm is the algorithm identifier that indicates the symmetric encryption algorithm used to encrypt the content-encryption key (CEK) with the key-encryption key (KEK).

When using 1-Pass ECMQV - keyEncryptionAlgorithm MUST be the mqvSinglePass-sha1kdf-scheme algorithm identifier (see Section 8.1), with the parameters field KeyWrapAlgorithm. The KeyWrapAlgorithm indicates the symmetric encryption algorithm used to encrypt the CEK with the KEK generated using the 1-Pass ECMQV algorithm.

Отсюда следует, что второй OID (который передается в keyEncryptionAlgorithm) – это OID, который указывает на алгоритм согласования ключа (key agreement algorithms), применяемую функцию примитива/ генерации общего секрета и функцию выработки ключа КШК.

Назовем этот идентификатор как «OID схемы КЕК/ выработки ключа КШК» (или «OID алгоритма согласования ключа»). Подчеркнем, что схема КЕК состоит из двух компонент – функции генерации примитива/ общего секрета и непосредственно функции выработки КШК из примитива, так называемой kdf-scheme (Key Derivation Function).

Для алгоритмов FFC DH идентификаторы схемы КЕК ссылаются на режим (статик или динамик) и в этом смысле не несут никакой дополнительной информации, т.к. схемы КЕК одна и та же (см. RFC 2631 [3]), а режим статик или динамик легко и однозначно определяется через OriginatorIdentifierOrKey:

```
OriginatorIdentifierOrKey ::= CHOICE {
    issuerAndSerialNumber   IssuerAndSerialNumber,
    subjectKeyIdentifier    [0] SubjectKeyIdentifier,
    originatorKey           [1] OriginatorPublicKey }
```

Если имеем OriginatorPublicKey, - то это динамический режим, иначе – статический. Поэтому «OID схемы КЕК» (id-alg-ESDH или id-alg-SSDH**)** может рассматриваться лишь как дополнительный (второй) контроль режима.

Таким образом, идентификаторы id-alg-ESDH, id-alg-SSDH в части схемы КЕК являются эквивалентными и указывают на применение схемы, которая определена для CMS-DH (RFC 2631 [3]**).**

Следует также подчеркнуть, что стандартами определен ряд (не одна!) схем kdf-scheme (например, схемы в RFC 2631, в NIST SP 800-56A [13], в ДСТУ ISO 15946-3 [14] и т.д.).

В Спецификации предлагаются национальные kdf-scheme – они отличаются не только хеш функцией (ГОСТ 34.311 вместо SHA-1), а также структурой/ величиной параметра, от которого вычисляется хеш.

Категорически не рекомендуется это делать (изменять структуру/ величину параметра, от которого вычисляется хеш.), т.к. с точки зрения реализации это приведет к необходимости «утяжелять» решение еще одной функцией (классом) kdf-scheme.

Таким образом, введение нового OID, а именно id-alg-ESDH-ua в качестве схемы КЕК, указывает на применяемую схему КЕК для алгоритма ГОСТ 34.310. Отличие от стандартной (RFC 2631) должно быть только в применении ГОСТ 34.311 вместо SHA-1.

Для ECC DH (ECDSA) рассматриваемый второй OID однозначно определяет схему КЕК для ECDH (одну из двух возможных, ECMQV не следует рассматривать). По умолчанию используется dhSinglePass-cofactorDH-sha1kdf-scheme.

Поэтому введение нового OID схемы КЕК для ECC DH с ДСТУ 4145, а именно id-ECDH-ua, должно быть «привязано» к dhSinglePass-cofactorDH-sha1kdf-scheme с одним лишь отличием – может применяться хеш ГОСТ 34.311 вместо SHA-1.

В целом, применение хеш ГОСТ 34.311 вместо SHA-1 ничем не аргументировано и является нецелесообразным.

Следует однозначно решить вопрос об использовании для реализации алгоритма Diffie-Hellman SHA-1 вместо ГОСТ 34.311, что дает возможность реализации схем Diffie-Hellman в ПОЛНОМ соответствии с международными стандартами.

В частности, предлагаемые Спецификации ссылаются на стандарт ДСТУ ISO/IEC 10118-3 [15], который также предусматривает использование SHA-1.

Для применения же ГОСТ 34.311 вместо SHA-1 в Diffie-Hellman должны быть приведены четкие криптографические аргументы, т.к. это приводит только к утяжелению решения, в том числе ряд дополнительных вопросов с выбором ДКЕ.

Литература

1. Технічні специфікації форматів криптографічних повідомлень», разработанном ГСССЗИ Украины (актуальная версия на 08.02.2010, впервые опубликовано на сайте ГСССЗИ Украины *23 листопада 2009*).
2. RFC 2630 - Cryptographic Message Syntax.
3. RFC 2631 - Diffie-Hellman Key Agreement Method.
4. RFC 3217 - Triple-DES and RC2 Key Wrapping.
5. RFC 3278 - Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS).
6. RFC 3279 - Algorithms and Identifiers for the Internet X_509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.
7. RFC 3280 - Internet X.509 Public Key Infrastructure.Certificate and Certificate Revocation List (CRL) Profile.
8. RFC 3370 - Cryptographic Message Syntax (CMS) Algorithms.
9. RFC 3394 - Lightweight Directory Access Protocol version 2 (LDAPv2) to Historic Status.
10. RFC 3852 - Cryptographic Message Syntax (CMS).
11. RFC 4490 - Using the GOST 28147-89, GOST R 34.11-94, GOST R 34.10-94, and GOST R 34.10-2001 Algorithms with Cryptographic Message Syntax (CMS).
12. RFC 5480 - Elliptic Curve Cryptography Subject Public Key Information.
13. NIST SP 800-56A «Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography», March, 2007.
14. ДСТУ ISO/IEC 15946-3. Інформаційні технології. Методи захисту. Криптографічні методи, що ґрунтуються на еліптичних кривих. Частина 3. Установлення ключів (ISO/IEC 15946-3:2002, IDT)/
15. ДСТУ ISO/IEC 10118-3:2005. Інформаційні технології. Методи захисту. Геш-функції. Частина 3. Спеціалізовані геш-функції (ISO/IEC 10118-3:2004, IDT)
16. Технічні специфікації форматів представлення базових об’єктів (спільний Наказ ДСТСЗІ СБУ та Департаменту інформатизації МТЗ №99/166 від 11.09.2006 р.)
17. ANSI X9.62-1998. Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)
18. RFC 5008: Suite B in Secure/Multipurpose Internet Mail Extensions (S/MIME), September 2007

19. ANSI X9.63-199x. Public Key Cryptography For The Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography, January 8, 1999
20. SEC1. Standards for Efficient Cryptography 1 (SEC 1): Elliptic Curve Cryptography, - Certicom Research, May 21, 2009, Version 2.0, Certicom Corp.

Вопросы по этим замечаниям и предложениям просьба направлять:
martyn@itsway.kiev.ua (Мартыненко Сергей Васильевич)