Securitatea poștei electronice

Prof. Dr. Adrian Atanasiu

March 15, 2017

Permite utilizatorilor să comunice prin mesaje, folosind facilitățile oferite de rețelele de calculatoare existente.

Permite utilizatorilor să comunice prin mesaje, folosind facilitățile oferite de rețelele de calculatoare existente.
Serviciul de e-mail este cea mai utilizată aplicație, oferind un grad relativ redus de securitate.

Permite utilizatorilor să comunice prin mesaje, folosind facilitățile oferite de rețelele de calculatoare existente.
Serviciul de e-mail este cea mai utilizată aplicație, oferind un grad relativ redus de securitate.
Mesajele se criptează folosind diverse produse: Privacy Enhanced Mail (*PEM*), MIME Object Security Services (*MOSS*), *X*.400, *PGP* sau *S/MIME*.

PGP este o specificație, iar *S/MIME* este un protocol; ambele sunt compatibile cu serviciile Internet actuale.

PGP

Cel mai utilizat sistem de securitate utilizat de serviciile de poștă electronică.

PGP

Cel mai utilizat sistem de securitate utilizat de serviciile de poștă electronică.Motive:

1. Este accesibil free în versiuni care funcționează pe o gamă largă de platforme (WINDOWS, UNIX, MacIntosh etc.).

PGP

Cel mai utilizat sistem de securitate utilizat de serviciile de poștă electronică.Motive:

1. Este accesibil free în versiuni care funcționează pe o gamă largă de platforme (WINDOWS, UNIX, MacIntosh etc.).
2. Este bazat pe algoritmi criptogafici considerați siguri: RSA, DSS și $ElGamal$ (pentru criptarea cu cheie publică), $CAST-128, IDEA, 3DES$ (pentru criptarea simetrică) și $SHA-1$ (ca funcție de dispersie).

PGP

Cel mai utilizat sistem de securitate utilizat de serviciile de poștă electronică.Motive:

1. Este accesibil free în versiuni care funcționează pe o gamă largă de platforme (WINDOWS, UNIX, MacIntosh etc.).
2. Este bazat pe algoritmi criptogafici considerați siguri: RSA, DSS și $ElGamal$ (pentru criptarea cu cheie publică), $CAST-128, IDEA, 3DES$ (pentru criptarea simetrică) și $SHA-1$ (ca funcție de dispersie).
3. Are o arie largă de aplicabilitate.

PGP

Cel mai utilizat sistem de securitate utilizat de serviciile de poștă electronică.Motive:

1. Este accesibil free în versiuni care funcționează pe o gamă largă de platforme (WINDOWS, UNIX, MacIntosh etc.).
2. Este bazat pe algoritmi criptogafici considerați siguri: RSA, DSS și $ElGamal$ (pentru criptarea cu cheie publică), $CAST-128, IDEA, 3DES$ (pentru criptarea simetrică) și $SHA-1$ (ca funcție de dispersie).
3. Are o arie largă de aplicabilitate.
4. Nu a fost creat, dezvoltat sau controlat de nici un organism guvernamental sau organizație de standarde.

Protocoalele *PGP* asigură cinci servicii:

Protocoalele *PGP* asigură cinci servicii:

- Autentificare;

Protocoalele *PGP* asigură cinci servicii:

- Autentificare;
- Confidențialitate;

Protocoalele *PGP* asigură cinci servicii:

- Autentificare;
- Confidențialitate;
- Compresie;

Protocoalele *PGP* asigură cinci servicii:

- Autentificare;
- Confidențialitate;
- Compresie;
- Compatibilitate e-mail;

Protocoalele *PGP* asigură cinci servicii:

- Autentificare;
- Confidențialitate;
- Compresie;
- Compatibilitate e-mail;
- Segmentare.

Autentificare

Serviciu de semnătură digitală cu appendix, oferit de *PGP*.

Autentificare

Serviciu de semnătură digitală cu appendix, oferit de *PGP*.

Intrare: Mesajul m este trimis de *Alice* către *Bob*.

- *Alice*, în calitate de expeditor, efectuează următorii pași:

Autentificare

Serviciu de semnătură digitală cu appendix, oferit de *PGP*.

Intrare: Mesajul m este trimis de *Alice* către *Bob*.

- *Alice*, în calitate de expeditor, efectuează următorii pași:
  1. Generează o amprentă $h(m)$ a mesajului, folosind o funcție de dispersie criptografică.

Autentificare

Serviciu de semnătură digitală cu appendix, oferit de *PGP*.

Intrare: Mesajul m este trimis de *Alice* către *Bob*.

- *Alice*, în calitate de expeditor, efectuează următorii pași:
  1. Generează o amprentă $h(m)$ a mesajului, folosind o funcție de dispersie criptografică.
  2. Semnează folosind cheia sa secretă: $d_{Alice}(h(m))$.

Autentificare

Serviciu de semnătură digitală cu appendix, oferit de *PGP*.

Intrare: Mesajul m este trimis de *Alice* către *Bob*.

- *Alice*, în calitate de expeditor, efectuează următorii pași:
  1. Generează o amprentă $h(m)$ a mesajului, folosind o funcție de dispersie criptografică.
  2. Semnează folosind cheia sa secretă: $d_{Alice}(h(m))$.
  3. Aplică o funcție de compresie Z perechii

$$(m, d_{Alice}(h(m)))$$

La recepția mesajului $Z(m, d_{Alice}(h(m)),\ Bob$:

La recepția mesajului $Z(m, d_{Alice}(h(m))$, *Bob*:

1. Decomprimă (cu Z^{-1}) și află mesajul m precum și $d_{Alice}(h(m))$.

La recepția mesajului $Z(m, d_{Alice}(h(m)), Bob$:

1. Decomprimă (cu Z^{-1}) și află mesajul m precum și $d_{Alice}(h(m))$.
2. Folosind cheia publică a lui *Alice*, determină

$$h(m) = e_{Alice}(d_{Alice}(h(m)))$$

La recepția mesajului $Z(m, d_{Alice}(h(m)), Bob$:

1. Decomprimă (cu Z^{-1}) și află mesajul m precum și $d_{Alice}(h(m))$.
2. Folosind cheia publică a lui *Alice*, determină

$$h(m) = e_{Alice}(d_{Alice}(h(m)))$$

3. Aplică funcția de dispersie h lui m și compară rezultatul cu $h(m)$.

La recepția mesajului $Z(m, d_{Alice}(h(m)), Bob$:

1. Decomprimă (cu Z^{-1}) și află mesajul m precum și $d_{Alice}(h(m))$.
2. Folosind cheia publică a lui *Alice*, determină

$$h(m) = e_{Alice}(d_{Alice}(h(m)))$$

3. Aplică funcția de dispersie h lui m și compară rezultatul cu $h(m)$.
   Dacă cele două secvențe coincid, mesajul m este acceptat ca autentic.

Securitate

Securitatea se bazează pe securitatea sistemelor folosite (*SHA*/*RSA* sau *SHA*/*DSA*).

Securitate

Securitatea se bazează pe securitatea sistemelor folosite (*SHA/RSA* sau *SHA/DSA*).
De obicei semnăturile sunt atașate mesajului (sau fișierului) pe care-l semnează.
Sunt posibile și semnături detașate. Acestea sunt păstrate de expeditor pentru a fi folosite în diverse scopuri.

Securitate

Securitatea se bazează pe securitatea sistemelor folosite (*SHA/RSA* sau *SHA/DSA*).
De obicei semnăturile sunt atașate mesajului (sau fișierului) pe care-l semnează.
Sunt posibile și semnături detașate. Acestea sunt păstrate de expeditor pentru a fi folosite în diverse scopuri.

Exemplu

Dacă mai multe părți semnează un contract, fiecare semnătură este independentă și se aplică doar documentului inițial.
Semnătura unui program executabil poate detecta o posibilă virusare ulterioară.

Confidențialitate

Se folosește $CAST - 128$ (alternative – $IDEA$, $3DES$); criptarea se efectuează în modul CFB pe 64 biți.
În implementări se folosește RSA sau $ElGamal$ pentru criptarea cu cheie publică.

Confidențialitate

Se folosește *CAST* – 128 (alternative – *IDEA*, *3DES*); criptarea se efectuează în modul *CFB* pe 64 biți.
În implementări se folosește *RSA* sau *ElGamal* pentru criptarea cu cheie publică.

Timpul de criptare se optimizează combinând cele două tipuri de criptare (un sistem de criptare simetric este mult mai rapid decât unul cu cheie publică).

Fie mesajul *m* trimis de *Alice* către *Bob*.
Alice efectuează următorii pași:

Fie mesajul m trimis de *Alice* către *Bob*.
Alice efectuează următorii pași:

1. Arhivează m cu un protocol de arhivare Z; se obține $Z(m)$.

Fie mesajul m trimis de *Alice* către *Bob*.
Alice efectuează următorii pași:

1. Arhivează m cu un protocol de arhivare Z; se obține $Z(m)$.
2. Generează aleator o cheie de sesiune K pe 128 biți.

Fie mesajul m trimis de *Alice* către *Bob*.
Alice efectuează următorii pași:

1. Arhivează m cu un protocol de arhivare Z; se obține $Z(m)$.
2. Generează aleator o cheie de sesiune K pe 128 biți.
3. Efectuează criptarea $e_K(Z(m))$.

Fie mesajul m trimis de *Alice* către *Bob*.
Alice efectuează următorii pași:

1. Arhivează m cu un protocol de arhivare Z; se obține $Z(m)$.
2. Generează aleator o cheie de sesiune K pe 128 biți.
3. Efectuează criptarea $e_K(Z(m))$.
4. Criptează cheia de sesiune folosind cheia publică a lui *Bob*: $e_{Bob}(K)$.

Fie mesajul m trimis de *Alice* către *Bob*.
Alice efectuează următorii pași:

1. Arhivează m cu un protocol de arhivare Z; se obține $Z(m)$.
2. Generează aleator o cheie de sesiune K pe 128 biți.
3. Efectuează criptarea $e_K(Z(m))$.
4. Criptează cheia de sesiune folosind cheia publică a lui *Bob*: $e_{Bob}(K)$.
5. Trimite cuplul $\alpha = (e_{Bob}(K), e_K(Z(m)))$.

La primirea mesajului α, *Bob*:

La primirea mesajului α, *Bob*:

1. Află cheia de sesiune $K = d_{Bob}(e_{Bob}(K))$.

La primirea mesajului α, *Bob*:

1. Află cheia de sesiune $K = d_{Bob}(e_{Bob}(K))$.
2. Decriptează partea a doua a mesajului primit:
$$d_K(e_K(Z(m))) = Z(m).$$

La primirea mesajului α, *Bob*:

1. Află cheia de sesiune $K = d_{Bob}(e_{Bob}(K))$.
2. Decriptează partea a doua a mesajului primit:
$$d_K(e_K(Z(m))) = Z(m).$$
3. Dezarhivează cu Z^{-1} și află mesajul m.

Cheie de sesiune

Cheia de sesiune este de fapt o cheie one-time.
Protocolul ei de distribuție nu este necesar, deoarece nu se solicită o confirmare a cheii de către *Bob*.

Cheie de sesiune

Cheia de sesiune este de fapt o cheie one-time.
Protocolul ei de distribuție nu este necesar, deoarece nu se solicită o confirmare a cheii de către *Bob*.

Generarea unei chei de sesiune: *Alice* introduce o parolă, folosind tastatura sau mouse-ul.
PGP folosește parola și timpii de scriere la tastatură (respectiv de mișcare a mouse-ului) pentru a genera o cheie aleatoare care va fi folosită de un sistem simetric de criptare.

Dacă sunt necesare ambele servicii *PGP* (autenticitate și confidențialitate), *Alice* va semna întâi mesajul cu cheia sa secretă, apoi îl va cripta cu cheia de sesiune, iar pe aceasta o va cripta cu componenta publică a cheii lui *Bob*.

Dacă sunt necesare ambele servicii *PGP* (autenticitate și confidențialitate), *Alice* va semna întâi mesajul cu cheia sa secretă, apoi îl va cripta cu cheia de sesiune, iar pe aceasta o va cripta cu componenta publică a cheii lui *Bob*.
Motivele pentru care se preferă această ordine:

1. În general este mai convenabilă stocarea semnăturii unui text clar, și nu a unui text criptat (a cărui cheie ar trebui reținută și ea).

Dacă sunt necesare ambele servicii *PGP* (autenticitate și confidențialitate), *Alice* va semna întâi mesajul cu cheia sa secretă, apoi îl va cripta cu cheia de sesiune, iar pe aceasta o va cripta cu componenta publică a cheii lui *Bob*.
Motivele pentru care se preferă această ordine:

1. În general este mai convenabilă stocarea semnăturii unui text clar, și nu a unui text criptat (a cărui cheie ar trebui reținută și ea).
2. Dacă se cere verificarea de către o terță parte, aceasta nu trebuie implicată în procesul de decriptare, ci doar în cel de verificare a semnăturii.

Compresie

PGP face o compresie a mesajului după semnarea lui, dar înaintea criptării.

Aceasta duce la o optimizare a spațiului folosit atât în mesajele e-mail cât și în stocarea fișierelor.

Compresie

PGP face o compresie a mesajului după semnarea lui, dar înaintea criptării.

Aceasta duce la o optimizare a spațiului folosit atât în mesajele e-mail cât și în stocarea fișierelor.

Algoritmul de compresie folosit de *PGP* este *ZIP*.

Semnătura este generată înaintea compresiei deoarece:

Semnătura este generată înaintea compresiei deoarece:

1. Este preferabil să semnăm un mesaj necomprimat, deoarece pentru verificări va fi oferit textul clar.

Semnătura este generată înaintea compresiei deoarece:

1. Este preferabil să semnăm un mesaj necomprimat, deoarece pentru verificări va fi oferit textul clar.
   În caz contrar sau se va păstra în memorie doar o versiune comprimată a mesajului, sau acesta va fi comprimat ori de câte ori se va solicita verificarea semnăturii.

Semnătura este generată înaintea compresiei deoarece:

1. Este preferabil să semnăm un mesaj necomprimat, deoarece pentru verificări va fi oferit textul clar.
   În caz contrar sau se va păstra în memorie doar o versiune comprimată a mesajului, sau acesta va fi comprimat ori de câte ori se va solicita verificarea semnăturii.
2. Chiar dacă este ușor de recomprimat mesajul ori de câte ori este necesară verificarea, apare o dificultate datorită faptului că *ZIP* este un algoritm nedeterminist: diverse implementări duc la rate de compresie diferite și deci la forme diferite.

Semnătura este generată înaintea compresiei deoarece:

1. Este preferabil să semnăm un mesaj necomprimat, deoarece pentru verificări va fi oferit textul clar.
   În caz contrar sau se va păstra în memorie doar o versiune comprimată a mesajului, sau acesta va fi comprimat ori de câte ori se va solicita verificarea semnăturii.
2. Chiar dacă este ușor de recomprimat mesajul ori de câte ori este necesară verificarea, apare o dificultate datorită faptului că *ZIP* este un algoritm nedeterminist: diverse implementări duc la rate de compresie diferite și deci la forme diferite.
   Aceste versiuni sunt totuși interoperabile: orice versiune poate decompresa corect ieșirea din oricare altă versiune. Aplicând funcția de dispersie și semnătura după compresie, vom obliga ca toate implementările *PGP* să conducă la aceeași compresie.

radix – 64

Multe sisteme de e-mail permit doar transmiterea blocurilor ASCII. De aceea *PGP* efectuează și o conversie a octeților la secvențe de caractere ASCII printabile.

radix – 64

Multe sisteme de e-mail permit doar transmiterea blocurilor ASCII. De aceea *PGP* efectuează și o conversie a octeților la secvențe de caractere ASCII printabile.
Conversia folosită este *radix* – 64: fiecare grup de 3 octeți binari este transformat în patru caractere ASCII.

radix – 64

Multe sisteme de e-mail permit doar transmiterea blocurilor ASCII. De aceea *PGP* efectuează și o conversie a octeților la secvențe de caractere ASCII printabile.
Conversia folosită este *radix* – 64: fiecare grup de 3 octeți binari este transformat în patru caractere ASCII.

Folosirea sistemului *radix* – 64 mărește mesajul cu 33%, extensie compensată de rata de compresie (*ZIP* are de obicei o rată de compresie de 50%).

radix – 64

Multe sisteme de e-mail permit doar transmiterea blocurilor ASCII. De aceea *PGP* efectuează și o conversie a octeților la secvențe de caractere ASCII printabile.
Conversia folosită este *radix* – 64: fiecare grup de 3 octeți binari este transformat în patru caractere ASCII.

Folosirea sistemului *radix* – 64 mărește mesajul cu 33%, extensie compensată de rata de compresie (*ZIP* are de obicei o rată de compresie de 50%).
Astfel, dacă un fișier are lungimea n, după compresie și conversie, lungimea lui va fi de aproximativ $1,33 \times 0,5 \times n = 0,665 \times n$.

Algoritmul *radix* – 64 convertește tot șirul de intrare, considerat ca o secvență binară de octeți.
Aceasta îl face necitibil unui intrus ocazional; deci oferă o anumită confidențialitate mesajului (și uneori înlocuiește complet criptarea).

Algoritmul *radix* – 64 convertește tot șirul de intrare, considerat ca o secvență binară de octeți.
Aceasta îl face necitibil unui intrus ocazional; deci oferă o anumită confidențialitate mesajului (și uneori înlocuiește complet criptarea).

Există opțiunea ca *PGP* să convertească (cu *radix* – 64) numai componenta de semnătură a mesajului; în acest fel *Bob* îl va putea citi direct, fără a folosi *PGP*.
PGP va fi utilizat numai pentru a verifica semnătura.

Segmentare și reasamblare

Există adesea o restricție referitoare la lungimea mesajelor transmise prin e-mail.

Segmentare și reasamblare

Există adesea o restricție referitoare la lungimea mesajelor transmise prin e-mail.

PGP segmentează automat un mesaj mai lung în mesaje care sunt trimise separat prin e-mail. Blocurile sunt puse în ordine în fișiere cu extensia .*as*1, .*as*2 etc.

Segmentare și reasamblare

Există adesea o restricție referitoare la lungimea mesajelor transmise prin e-mail.

PGP segmentează automat un mesaj mai lung în mesaje care sunt trimise separat prin e-mail. Blocurile sunt puse în ordine în fișiere cu extensia .*as*1, .*as*2 etc.
Segmentarea este realizată după încheierea celorlaltor operații.
Deci cheia de sesiune și semnătura vor apare o singură dată, la începutul primului segment.

Segmentare și reasamblare

Există adesea o restricție referitoare la lungimea mesajelor transmise prin e-mail.

PGP segmentează automat un mesaj mai lung în mesaje care sunt trimise separat prin e-mail. Blocurile sunt puse în ordine în fișiere cu extensia .*as*1, .*as*2 etc.
Segmentarea este realizată după încheierea celorlaltor operații.
Deci cheia de sesiune și semnătura vor apare o singură dată, la începutul primului segment.
După recepția tuturor segmentelor, *PGP* face reasamblarea lor în ordinea indicată de extensii, eliminând headerele și alte detalii de e-mail.

Chei utilizate de *PGP*

PGP-ul folosește patru tipuri de chei:

Chei utilizate de *PGP*

PGP-ul folosește patru tipuri de chei:

- chei de sesiune (simetrice one-time),

Chei utilizate de *PGP*

PGP-ul folosește patru tipuri de chei:

- chei de sesiune (simetrice one-time),
- chei publice,

Chei utilizate de *PGP*

PGP-ul folosește patru tipuri de chei:

- chei de sesiune (simetrice one-time),
- chei publice,
- chei private,

Chei utilizate de *PGP*

PGP-ul folosește patru tipuri de chei:

- chei de sesiune (simetrice one-time),
- chei publice,
- chei private,
- chei simetrice (bazate pe parolă).

Cerințe:

1. O modalitate de a genera aleator chei de sesiune.

Cerințe:

1. O modalitate de a genera aleator chei de sesiune.
2. Fiecare utilizator trebuie să aibă mai multe perechi de chei publice/private care se schimbă periodic.

Cerințe:

1. O modalitate de a genera aleator chei de sesiune.
2. Fiecare utilizator trebuie să aibă mai multe perechi de chei publice/private care se schimbă periodic.
   Utilizatorul poate opta pentru mai multe perechi de chei, la un moment dat, fie pentru a interacționa cu mai mulți destinatari simultan, fie doar pentru a îmbunătăți securitatea, limitând utilizarea unei anumite chei la doar o porțiune din text.
   Dificultatea în aceste cazuri este determinată de faptul că nu există o modalitate de relaționare între utilizatori și cheile lor publice.

Cerințe:

1. O modalitate de a genera aleator chei de sesiune.
2. Fiecare utilizator trebuie să aibă mai multe perechi de chei publice/private care se schimbă periodic.
   Utilizatorul poate opta pentru mai multe perechi de chei, la un moment dat, fie pentru a interacționa cu mai mulți destinatari simultan, fie doar pentru a îmbunătăți securitatea, limitând utilizarea unei anumite chei la doar o porțiune din text.
   Dificultatea în aceste cazuri este determinată de faptul că nu există o modalitate de relaționare între utilizatori și cheile lor publice.
3. Un utilizator trebuie să păstreze un fișier cu propriile sale perechi de chei publice/private, și un fișier cu cheile publice.

Generarea cheilor de sesiune

Fiecare cheie de sesiune este asociată unui singur mesaj și este folosită exclusiv pentru criptarea și decriptarea acestuia.

Generarea cheilor de sesiune

Fiecare cheie de sesiune este asociată unui singur mesaj și este folosită exclusiv pentru criptarea și decriptarea acestuia.

PGP-ul menține un buffer de 256 octeți aleatori.

Generarea cheilor de sesiune

Fiecare cheie de sesiune este asociată unui singur mesaj și este folosită exclusiv pentru criptarea și decriptarea acestuia.

PGP-ul menține un buffer de 256 octeți aleatori.De fiecare dată când *PGP*-ul așteaptă apăsarea unei taste, înregistrează timpul, în format de 32 de biți de la care începe așteptarea.

Generarea cheilor de sesiune

Fiecare cheie de sesiune este asociată unui singur mesaj și este folosită exclusiv pentru criptarea și decriptarea acestuia.

PGP-ul menține un buffer de 256 octeți aleatori.De fiecare dată când *PGP*-ul așteaptă apăsarea unei taste, înregistrează timpul, în format de 32 de biți de la care începe așteptarea.
Când o tastă este apăsată, *PGP*-ul înregistrează momentul la care a avut loc acest eveniment, precum și valoarea pe 8 biți a caracterului tastat.

Datele referitoare la timp și caracter sunt utilizate pentru a genera o cheie care – la rândul ei – este folosită pentru a cripta valoare curentă a buffer-ului de octeți aleatori.

Datele referitoare la timp și caracter sunt utilizate pentru a genera o cheie care – la rândul ei – este folosită pentru a cripta valoare curentă a buffer-ului de octeți aleatori.

Numărul aleator obținut este combinat cu cheia de sesiune rezultată din algoritmul *CAST* – 128 pentru a forma noua intrare a generatorului.

Identificatori de chei

Cheia de sesiune este criptată cu cheia publică a lui *Bob*, astfel încât doar acesta poate să recupereze cheia de sesiune și să decripteze mesajul.

Identificatori de chei

Cheia de sesiune este criptată cu cheia publică a lui *Bob*, astfel încât doar acesta poate să recupereze cheia de sesiune și să decripteze mesajul.
Dacă *Bob* are mai multe perechi de chei publice/private, atunci el nu va ști ce cheie să folosească pentru a recupera cheia de sesiune.

Identificatori de chei

Cheia de sesiune este criptată cu cheia publică a lui *Bob*, astfel încât doar acesta poate să recupereze cheia de sesiune și să decripteze mesajul.
Dacă *Bob* are mai multe perechi de chei publice/private, atunci el nu va ști ce cheie să folosească pentru a recupera cheia de sesiune.
Soluția adoptată de *PGP*: se alocă câte un *ID* fiecărei chei publice, unic (cu probabilitate mare) în raport cu un anumit destinatar.
ID-ul fiecărei chei publice *KP* constă din cei mai puțin semnificativi 64 biți ai acesteia:

$$KP \ (mod 2^{64})$$

Identificatori de chei

Cheia de sesiune este criptată cu cheia publică a lui *Bob*, astfel încât doar acesta poate să recupereze cheia de sesiune și să decripteze mesajul.
Dacă *Bob* are mai multe perechi de chei publice/private, atunci el nu va ști ce cheie să folosească pentru a recupera cheia de sesiune.
Soluția adoptată de *PGP*: se alocă câte un *ID* fiecărei chei publice, unic (cu probabilitate mare) în raport cu un anumit destinatar.
ID-ul fiecărei chei publice *KP* constă din cei mai puțin semnificativi 64 biți ai acesteia:

$$KP\ (mod 2^{64})$$

Aceasta este o dimensiune suficientă pentru ca probabilitatea existenței duplicatelor să fie redusă.

Servere de chei

Cheile sunt stocate și organizate într-un mod arborescent, care să permită o utilizare efectivă și sistematică de către toate părțile.

Servere de chei

Cheile sunt stocate și organizate într-un mod arborescent, care să permită o utilizare efectivă și sistematică de către toate părțile.

PGP-ul furnizează fiecărui nod o pereche de structuri de date: una pentru a stoca perechile de chei publice/private deținute de nodul respectiv și una pentru a stoca cheile publice corespunzătoare ale celorlalți utilizatori.

Servere de chei

Cheile sunt stocate și organizate într-un mod arborescent, care să permită o utilizare efectivă și sistematică de către toate părțile.

PGP-ul furnizează fiecărui nod o pereche de structuri de date: una pentru a stoca perechile de chei publice/private deținute de nodul respectiv și una pentru a stoca cheile publice corespunzătoare ale celorlalți utilizatori.

Aceste structuri de date sunt cunoscute sub numele de "*serverul de chei private*" și respectiv "*serverul de chei publice*".

Serverul de chei private

Timestamp	ID cheie	Cheie publică	Cheie privată	ID utilizator
⋮	⋮	⋮	⋮	⋮
T_i	$PU_i \ (mod \ 2^{64})$	PU_i	$e_{h(P_i)}(PR_i)$	Utilizator i
⋮	⋮	⋮	⋮	⋮

O linie conține o pereche de chei publice/private deținute de utilizator.

Serverul de chei private

Timestamp	ID cheie	Cheie publică	Cheie privată	ID utilizator
⋮	⋮	⋮	⋮	⋮
T_i	$PU_i \ (mod \ 2^{64})$	PU_i	$e_{h(P_i)}(PR_i)$	Utilizator i
⋮	⋮	⋮	⋮	⋮

O linie conține o pereche de chei publice/private deținute de utilizator.

- **Timestamp**: data/ora la care a fost generată perechea.

Serverul de chei private

Timestamp	ID cheie	Cheie publică	Cheie privată	ID utilizator
⋮	⋮	⋮	⋮	⋮
T_i	$PU_i \ (mod \ 2^{64})$	PU_i	$e_{h(P_i)}(PR_i)$	Utilizator i
⋮	⋮	⋮	⋮	⋮

O linie conține o pereche de chei publice/private deținute de utilizator.

- **Timestamp**: data/ora la care a fost generată perechea.
- *ID* **cheie**: cei mai puțin semnificativi 64 biți.

Serverul de chei private

Timestamp	ID cheie	Cheie publică	Cheie privată	ID utilizator
⋮	⋮	⋮	⋮	⋮
T_i	$PU_i \ (mod \ 2^{64})$	PU_i	$e_{h(P_i)}(PR_i)$	Utilizator i
⋮	⋮	⋮	⋮	⋮

O linie conține o pereche de chei publice/private deținute de utilizator.

- **Timestamp**: data/ora la care a fost generată perechea.
- *ID* **cheie**: cei mai puțin semnificativi 64 biți.
- **Cheia publică**: componenta publică a perechii de chei.

Serverul de chei private

Timestamp	ID cheie	Cheie publică	Cheie privată	ID utilizator
⋮	⋮	⋮	⋮	⋮
T_i	$PU_i \ (mod\ 2^{64})$	PU_i	$e_{h(P_i)}(PR_i)$	Utilizator i
⋮	⋮	⋮	⋮	⋮

O linie conține o pereche de chei publice/private deținute de utilizator.

- **Timestamp**: data/ora la care a fost generată perechea.
- *ID* **cheie**: cei mai puțin semnificativi 64 biți.
- **Cheia publică**: componenta publică a perechii de chei.
- **Cheia privată**: componenta privată a perechii de chei; acest câmp este criptat.

Serverul de chei private

Timestamp	ID cheie	Cheie publică	Cheie privată	ID utilizator
⋮	⋮	⋮	⋮	⋮
T_i	$PU_i \ (mod\ 2^{64})$	PU_i	$e_{h(P_i)}(PR_i)$	Utilizator i
⋮	⋮	⋮	⋮	⋮

O linie conține o pereche de chei publice/private deținute de utilizator.

- **Timestamp**: data/ora la care a fost generată perechea.
- *ID* **cheie**: cei mai puțin semnificativi 64 biți.
- **Cheia publică**: componenta publică a perechii de chei.
- **Cheia privată**: componenta privată a perechii de chei; acest câmp este criptat.
- *ID* **utilizator**: de obicei conține adresa e-mail a utilizatorului.

Deoarece valoarea cheii private trebuie păstrată cât mai sigur, ea este criptată și apoi înregistrată pe server.

Deoarece valoarea cheii private trebuie păstrată cât mai sigur, ea este criptată și apoi înregistrată pe server.
Procedura de criptare:

1. Utilizatorul alege o parolă.

Deoarece valoarea cheii private trebuie păstrată cât mai sigur, ea este criptată și apoi înregistrată pe server.
Procedura de criptare:

1. Utilizatorul alege o parolă.
2. De fiecare dată când sistemul generează o nouă pereche de chei folosind *RSA*, este cerută parola. Utilizând $SHA-1$, este generată o amprentă pe 160 de biți, iar parola se șterge.

Deoarece valoarea cheii private trebuie păstrată cât mai sigur, ea este criptată și apoi înregistrată pe server.
Procedura de criptare:

1. Utilizatorul alege o parolă.
2. De fiecare dată când sistemul generează o nouă pereche de chei folosind *RSA*, este cerută parola. Utilizând $SHA-1$, este generată o amprentă pe 160 de biți, iar parola se șterge.
3. Sistemul criptează cheia privată folosind $CAST-128$ cu amprenta – trunchiată pe 128 biți – drept cheie.

Deoarece valoarea cheii private trebuie păstrată cât mai sigur, ea este criptată și apoi înregistrată pe server.
Procedura de criptare:

1. Utilizatorul alege o parolă.
2. De fiecare dată când sistemul generează o nouă pereche de chei folosind *RSA*, este cerută parola. Utilizând $SHA-1$, este generată o amprentă pe 160 de biți, iar parola se șterge.
3. Sistemul criptează cheia privată folosind $CAST-128$ cu amprenta – trunchiată pe 128 biți – drept cheie.
   Amprenta este apoi ștearsă și cheia privată criptată este stocată pe serverul de chei private.

Serverul de chei publice

Este folosit pentru stocarea cheilor publice ale celorlalți utilizatori conectați cu posesorul serverului de chei.

Serverul de chei publice

Este folosit pentru stocarea cheilor publice ale celorlalți utilizatori conectați cu posesorul serverului de chei.
Conține toate câmpurile serverului de chei private:
- Timestamp,

Serverul de chei publice

Este folosit pentru stocarea cheilor publice ale celorlalți utilizatori conectați cu posesorul serverului de chei.
Conține toate câmpurile serverului de chei private:
- Timestamp,
- *ID* cheie,

Serverul de chei publice

Este folosit pentru stocarea cheilor publice ale celorlalți utilizatori conectați cu posesorul serverului de chei.
Conține toate câmpurile serverului de chei private:
- Timestamp,
- *ID* cheie,
- Cheia publică,

Serverul de chei publice

Este folosit pentru stocarea cheilor publice ale celorlalți utilizatori conectați cu posesorul serverului de chei.
Conține toate câmpurile serverului de chei private:
- Timestamp,
- *ID* cheie,
- Cheia publică,
- *ID* - utilizator (un utilizator poate avea mai multe *ID*-uri asociate unei singure chei).

Cele două servere sunt construite în conformitate cu o autoritate centrală *PGP* care eliberează certificate.

Cele două servere sunt construite în conformitate cu o autoritate centrală *PGP* care eliberează certificate.
Fiecare intrare în serverul cheilor publice reprezintă o cheie publică certificată.

Cele două servere sunt construite în conformitate cu o autoritate centrală *PGP* care eliberează certificate.
Fiecare intrare în serverul cheilor publice reprezintă o cheie publică certificată.
Fiecărei linii din tabel îi este asociat un câmp **Legitimare Cheie**, care indică în ce măsură *PGP*-ul va considera că aceasta reprezintă o cheie publică validă pentru utilizator, și cât de ridicat este nivelul de încredere.
Acest câmp este calculat de *PGP*.

Cele două servere sunt construite în conformitate cu o autoritate centrală *PGP* care eliberează certificate.
Fiecare intrare în serverul cheilor publice reprezintă o cheie publică certificată.
Fiecărei linii din tabel îi este asociat un câmp **Legitimare Cheie**, care indică în ce măsură *PGP*-ul va considera că aceasta reprezintă o cheie publică validă pentru utilizator, și cât de ridicat este nivelul de încredere.
Acest câmp este calculat de *PGP*.

De asemenea, fiecărei linii i se asociază una sau mai multe semnături ale certificatului.

La rândul ei, fiecare semnătură are asociat un câmp **Signature Trust** care indică în ce măsură are încredere utilizatorul că semnătura respectivă certifică o cheie publică.

La rândul ei, fiecare semnătură are asociat un câmp **Signature Trust** care indică în ce măsură are încredere utilizatorul că semnătura respectivă certifică o cheie publică.

Un câmp **Owner trust** este inclus pentru a indica în ce măsură cheia publică este de încredere în semnarea altor certificate de chei publice; acest nivel de încredere este stabilit de utilizator.

Un fișier *PGP* este alcătuit din:

- Un pachet mesaj;
- Un pachet semnătură;
- Un pachet cheie de sesiune.

Pachetul mesaj conține datele care au importanță pentru utilizator și care vor fi trimise sau stocate, precum și un header care conține informații generate de *PGP*: numele fișierului și o ștampilă de timp (care indică data creerii mesajului sau fișierului).

Pachetul mesaj conține datele care au importanță pentru utilizator și care vor fi trimise sau stocate, precum și un header care conține informații generate de *PGP*: numele fișierului și o ștampilă de timp (care indică data creerii mesajului sau fișierului).

Această componentă constă dintr-un singur pachet de caractere alfabetice.

Pachetul semnătură conține o combinație între informații legate de cheia publică a lui *Alice* și o amprentă a mesajului (obținută prin aplicarea unei funcții de dispersie criptografică asupra componentei mesaj).

Pachetul semnătură conține o combinație între informații legate de cheia publică a lui *Alice* și o amprentă a mesajului (obținută prin aplicarea unei funcții de dispersie criptografică asupra componentei mesaj).

Pachetelor *mesaj* și *semnătură* li se poate aplica o compresie folosind *ZIP* și pot fi criptate cu ajutorul unei chei de sesiune.

Pachetul de chei de sesiune include cheia de sesiune și identificatorul cheii publice a lui *Bob* – care a fost utilizată de *Alice* pentru a cripta cheia de sesiune.

Pachetul de chei de sesiune include cheia de sesiune și identificatorul cheii publice a lui *Bob* – care a fost utilizată de *Alice* pentru a cripta cheia de sesiune.
În interiorul pachetului principal se află un pachet cu cheia publică de sesiune criptată.

Pachetul de chei de sesiune include cheia de sesiune și identificatorul cheii publice a lui *Bob* – care a fost utilizată de *Alice* pentru a cripta cheia de sesiune.
În interiorul pachetului principal se află un pachet cu cheia publică de sesiune criptată.
Pachetele de date criptate identic sunt precedate de un pachet ce conține cheia publică de sesiune criptată – pentru fiecare cheie utilizată în criptarea mesajului.

Pachetul de chei de sesiune include cheia de sesiune și identificatorul cheii publice a lui *Bob* – care a fost utilizată de *Alice* pentru a cripta cheia de sesiune.
În interiorul pachetului principal se află un pachet cu cheia publică de sesiune criptată.
Pachetele de date criptate identic sunt precedate de un pachet ce conține cheia publică de sesiune criptată – pentru fiecare cheie utilizată în criptarea mesajului.
Mesajul este criptat cu cheia de sesiune, care este la rândul ei criptată și stocată în pachetul cheii de sesiune.

Corpul pachetului cheii de sesiune este alcătuit din:

1. Un octet reprezentând cifra 3.

Corpul pachetului cheii de sesiune este alcătuit din:

1. Un octet reprezentând cifra 3.
2. 8 octeți ai *ID*-ului cheii publice cu care este criptată cheia de sesiune.

Corpul pachetului cheii de sesiune este alcătuit din:

1. Un octet reprezentând cifra 3.
2. 8 octeți ai *ID*-ului cheii publice cu care este criptată cheia de sesiune.
3. 8 octeți care indică algoritmul folosit pentru cheia publică.

Corpul pachetului cheii de sesiune este alcătuit din:

1. Un octet reprezentând cifra 3.
2. 8 octeți ai *ID*-ului cheii publice cu care este criptată cheia de sesiune.
3. 8 octeți care indică algoritmul folosit pentru cheia publică.
4. Un șir de octeți reprezentând cheia publică criptată. Conținutul acestui șir este dependent de algoritmul utiizat pentru cheia publică.

Corpul pachetului cheii de sesiune este alcătuit din:

1. Un octet reprezentând cifra 3.
2. 8 octeți ai *ID*-ului cheii publice cu care este criptată cheia de sesiune.
3. 8 octeți care indică algoritmul folosit pentru cheia publică.
4. Un șir de octeți reprezentând cheia publică criptată. Conținutul acestui șir este dependent de algoritmul utiizat pentru cheia publică.

Dacă se folosește compresia, atunci criptarea se aplică după compresia pachetului format din pachetul mesaj și pachetul semnătură.

Comentarii finale *PGP*

Periodic sunt lansate versiuni noi de *PGP* și unele falii de securitate – dacă există – sunt rezolvate pe parcurs.

Comentarii finale *PGP*

Periodic sunt lansate versiuni noi de *PGP* și unele falii de securitate – dacă există – sunt rezolvate pe parcurs.

Exemplu

În 2006, guvernul SUA – găsind aproape imposibilă accesarea fișierelor criptate cu PGP ale unui inculpat – i-a cerut acestuia să le furnizeze parola; acest lucru însă contravine amendamentului 5, care dă dreptul unui acuzat să nu se incrimineze singur.

Comentarii finale *PGP*

Periodic sunt lansate versiuni noi de *PGP* și unele falii de securitate – dacă există – sunt rezolvate pe parcurs.

Exemplu

În 2006, guvernul SUA – găsind aproape imposibilă accesarea fișierelor criptate cu PGP ale unui inculpat – i-a cerut acestuia să le furnizeze parola; acest lucru însă contravine amendamentului 5, care dă dreptul unui acuzat să nu se incrimineze singur.

Având în vedere utilizarea îndelungată a *PGP*-ului, îmbunătățirile sistematice care i-au fost aduse, precum și rezistența confirmată la atacuri prin criptanaliză, sistemul s-a dovedit sigur până acum.

MIME

MIME (Multipurpose Internet Mail Extension) este un standard care are ca scop redefinirea formatelor mesajelor e-mail, permițând:

MIME

MIME (Multipurpose Internet Mail Extension) este un standard care are ca scop redefinirea formatelor mesajelor e-mail, permițând:

- Folosirea în mesaje de caractere dinafara setului *ASCII*;

MIME

MIME (Multipurpose Internet Mail Extension) este un standard care are ca scop redefinirea formatelor mesajelor e-mail, permițând:

- Folosirea în mesaje de caractere dinafara setului *ASCII*;
- Un set extins de formate pentru mesaje (înafara modului text);

MIME

MIME (Multipurpose Internet Mail Extension) este un standard care are ca scop redefinirea formatelor mesajelor e-mail, permițând:

- Folosirea în mesaje de caractere dinafara setului *ASCII*;
- Un set extins de formate pentru mesaje (înafara modului text);
- Mesaje compuse (*Multipart message bodies*);

MIME

MIME (Multipurpose Internet Mail Extension) este un standard care are ca scop redefinirea formatelor mesajelor e-mail, permițând:

- Folosirea în mesaje de caractere dinafara setului *ASCII*;
- Un set extins de formate pentru mesaje (înafara modului text);
- Mesaje compuse (*Multipart message bodies*);
- Headere cu informații care folosesc caractere dinafara setului *ASCII*.

Standardul apare ca urmare a necesității transmiterii – prin intermediul poștei electronice – de imagini, înregistrări video sau mesaje text scrise în alte limbi decât cea engleză.

Standardul apare ca urmare a necesității transmiterii – prin intermediul poștei electronice – de imagini, înregistrări video sau mesaje text scrise în alte limbi decât cea engleză.

MIME extinde poșta electronică la caractere *UNICODE* într-o manieră simplă, compatibilă cu versiunile anterioare și totodată deschisă posibilității de extindere.

Standardul apare ca urmare a necesității transmiterii – prin intermediul poștei electronice – de imagini, înregistrări video sau mesaje text scrise în alte limbi decât cea engleză.

MIME extinde poșta electronică la caractere *UNICODE* într-o manieră simplă, compatibilă cu versiunile anterioare și totodată deschisă posibilității de extindere.

Formatul *MIME* este folosit și în *WWW* pentru definirea tipului de date hypertext și pentru specificarea scripturilor *HTML*.

RFC 822

Formatul mesajelor *MIME* se bazează pe formatul *RFC* 822 (folosit și astăzi).

RFC 822

Formatul mesajelor *MIME* se bazează pe formatul *RFC* 822 (folosit și astăzi).
Este formatul standard pentru mesajele text trimise prin poșta electronică. În contextul *RFC* 822 mesajele sunt privite similar scrisorilor obișnuite: un text pus într-un plic.

RFC 822

Formatul mesajelor *MIME* se bazează pe formatul *RFC* 822 (folosit și astăzi).
Este formatul standard pentru mesajele text trimise prin poșta electronică. În contextul *RFC* 822 mesajele sunt privite similar scrisorilor obișnuite: un text pus într-un plic.
Plicul conține informația necesară pentru realizarea unei transmisii și livrări corecte.

RFC 822

Formatul mesajelor *MIME* se bazează pe formatul *RFC* 822 (folosit și astăzi).
Este formatul standard pentru mesajele text trimise prin poșta electronică. În contextul *RFC* 822 mesajele sunt privite similar scrisorilor obișnuite: un text pus într-un plic.
Plicul conține informația necesară pentru realizarea unei transmisii și livrări corecte.
Conținutul este obiectul care va fi livrat destinatarului. Standardul *RFC* 822 se aplică doar conținutului.

RFC 822

Formatul mesajelor *MIME* se bazează pe formatul *RFC* 822 (folosit și astăzi).
Este formatul standard pentru mesajele text trimise prin poșta electronică. În contextul *RFC* 822 mesajele sunt privite similar scrisorilor obișnuite: un text pus într-un plic.
Plicul conține informația necesară pentru realizarea unei transmisii și livrări corecte.
Conținutul este obiectul care va fi livrat destinatarului. Standardul *RFC* 822 se aplică doar conținutului.
Conținutul include câmpuri de antet (headere), utilizate de sistem pentru crearea plicului; standardul facilitează accesarea automată a acestor informații de către diverse programe.

Un mesaj în format *RFC* 822 are două părți:

Un mesaj în format *RFC* 822 are două părți:

- un antet (*header*) – utilizat de agentul care asigură serviciul de e-mail,

Un mesaj în format *RFC* 822 are două părți:

- un antet (*header*) – utilizat de agentul care asigură serviciul de e-mail,
- un text ASCII (*corp*).

Un mesaj în format *RFC* 822 are două părți:

- un antet (*header*) – utilizat de agentul care asigură serviciul de e-mail,
- un text ASCII (*corp*).

O linie de antet are forma:

$$< \textit{cuvant cheie} >:< \textit{atribute} >$$

Un mesaj în format *RFC* 822 are două părți:

- un antet (*header*) – utilizat de agentul care asigură serviciul de e-mail,
- un text ASCII (*corp*).

O linie de antet are forma:

$$< \textit{cuvant cheie} >:< \textit{atribute} >$$

Formatul permite ca o linie mai mare să fie scrisă pe mai multe linii.

Un mesaj în format *RFC* 822 are două părți:

- un antet (*header*) – utilizat de agentul care asigură serviciul de e-mail,
- un text ASCII (*corp*).

O linie de antet are forma:

$$< \textit{cuvant cheie} >:< \textit{atribute} >$$

Formatul permite ca o linie mai mare să fie scrisă pe mai multe linii.
Cele mai utilizate cuvinte cheie sunt: *From*, *To*, *Subject* și *Date*.

Un mesaj în format *RFC* 822 are două părți:

- un antet (*header*) – utilizat de agentul care asigură serviciul de e-mail,
- un text ASCII (*corp*).

O linie de antet are forma:

$$< \textit{cuvant cheie} >:< \textit{atribute} >$$

Formatul permite ca o linie mai mare să fie scrisă pe mai multe linii.
Cele mai utilizate cuvinte cheie sunt: *From*, *To*, *Subject* și *Date*.
În funcție de mesaj, pot apare și alte linii de antet, cum ar fi *CC*, *List Info* etc.

Exemplu

From: Cipher Editor < cipher − editor@ieee − security.org >
Subject: Cipher Newsletter, Issue 60, May 18, 2009
Date: Tue, 18 May 2009 13:20:49 -0600
To: < cipher@mailman.xmission.com >
CC: < aadrian@gmail.com >
List Info: "subscriptions for the IEEE online newsletter, Cipher"
Show details...

Exemplu

From: Cipher Editor < cipher − editor@ieee − security.org >
Subject: Cipher Newsletter, Issue 60, May 18, 2009
Date: Tue, 18 May 2009 13:20:49 -0600
To: < cipher@mailman.xmission.com >
CC: < aadrian@gmail.com >
List Info: "subscriptions for the IEEE online newsletter, Cipher"
Show details...

Alt câmp care apare frecvent în antetele formatului *RFC* 822 este *Message ID*.

Exemplu

From: Cipher Editor < cipher − editor@ieee − security.org >
Subject: Cipher Newsletter, Issue 60, May 18, 2009
Date: Tue, 18 May 2009 13:20:49 -0600
To: < cipher@mailman.xmission.com >
CC: < aadrian@gmail.com >
List Info: "subscriptions for the IEEE online newsletter, Cipher"
Show details...

Alt câmp care apare frecvent în antetele formatului *RFC* 822 este *Message ID*.
Această linie conține un identificator unic asociat mesajului.

MIME este o extensie a lui *RFC* 822, care elimină unele restricții ale protocolului *SMTP* (*Simple Mail Transfer Protocol*):

MIME este o extensie a lui *RFC* 822, care elimină unele restricții ale protocolului *SMTP* (*Simple Mail Transfer Protocol*):

1. *SMTP* nu poate transmite fișiere executabile sau alte obiecte binare.

MIME este o extensie a lui *RFC* 822, care elimină unele restricții ale protocolului *SMTP* (*Simple Mail Transfer Protocol*):

1. *SMTP* nu poate transmite fișiere executabile sau alte obiecte binare.
2. *SMTP* este limitat la ASCII pe 7 biți.

MIME este o extensie a lui *RFC* 822, care elimină unele restricții ale protocolului *SMTP* (*Simple Mail Transfer Protocol*):

1. *SMTP* nu poate transmite fișiere executabile sau alte obiecte binare.
2. *SMTP* este limitat la ASCII pe 7 biți.
3. Serverele *SMTP* acceptă doar mesaje limitate ca mărime.

În plus, unele implementări *SMTP* nu sunt pe deplin conforme cu standardele *SMTP* definite în *RFC* 822.
Cele mai frecvente probleme se referă la:

În plus, unele implementări *SMTP* nu sunt pe deplin conforme cu standardele *SMTP* definite în *RFC* 822.
Cele mai frecvente probleme se referă la:

- Ștergerea, adăugarea sau reordonarea comenzilor $<$ *carriage return* $>$ și $<$ *linefeed* $>$.

În plus, unele implementări *SMTP* nu sunt pe deplin conforme cu standardele *SMTP* definite în *RFC* 822.
Cele mai frecvente probleme se referă la:

- Ștergerea, adăugarea sau reordonarea comenzilor $<$ *carriage return* $>$ și $<$ *linefeed* $>$.
- Trunchierea sau ștergerea liniilor mai lungi de 76 caractere;

În plus, unele implementări *SMTP* nu sunt pe deplin conforme cu standardele *SMTP* definite în *RFC* 822.
Cele mai frecvente probleme se referă la:

- Ștergerea, adăugarea sau reordonarea comenzilor $<$ *carriage return* $>$ și $<$ *linefeed* $>$.
- Trunchierea sau ștergerea liniilor mai lungi de 76 caractere;
- Eliminarea spațiilor albe (*Tab*-ul sau caracterul spațiu);

În plus, unele implementări *SMTP* nu sunt pe deplin conforme cu standardele *SMTP* definite în *RFC* 822.
Cele mai frecvente probleme se referă la:

- Ștergerea, adăugarea sau reordonarea comenzilor $<$ *carriage return* $>$ și $<$ *linefeed* $>$.
- Trunchierea sau ștergerea liniilor mai lungi de 76 caractere;
- Eliminarea spațiilor albe (*Tab*-ul sau caracterul spațiu);
- Aranjarea liniilor din mesaj la o lungime standard;

În plus, unele implementări *SMTP* nu sunt pe deplin conforme cu standardele *SMTP* definite în *RFC* 822.
Cele mai frecvente probleme se referă la:

- Ștergerea, adăugarea sau reordonarea comenzilor $<$ *carriage return* $>$ și $<$ *linefeed* $>$.
- Trunchierea sau ștergerea liniilor mai lungi de 76 caractere;
- Eliminarea spațiilor albe (*Tab*-ul sau caracterul spațiu);
- Aranjarea liniilor din mesaj la o lungime standard;
- Conversia caracterului *Tab* într-un multiplu de caractere spațiu.

O specificație *MIME* conține:

O specificație *MIME* conține:

1. Cinci linii noi în antet, care oferă informații despre corpul mesajului.

O specificație *MIME* conține:

1. Cinci linii noi în antet, care oferă informații despre corpul mesajului.
2. Reprezentări standardizate care suportă elemente multimedia pentru poșta electronică.

O specificație *MIME* conține:

1. Cinci linii noi în antet, care oferă informații despre corpul mesajului.
2. Reprezentări standardizate care suportă elemente multimedia pentru poșta electronică.
3. Codificări de transfer care permit conversia oricărui format într-un format standard, protejat la modificări efectuate de sistemul de mail.

Liniile de antet din *MIME*:

- *MIME – Version*: este însoțit de parametrul 1.0. Arată că mesajul este codificat conform *RFC* 2045 și *RFC* 2046.

Liniile de antet din *MIME*:

- *MIME – Version*: este însoțit de parametrul 1.0. Arată că mesajul este codificat conform *RFC* 2045 și *RFC* 2046.
- *Content – Type*: Descrie datele din mesaj, pentru ca serverul destinație să selecteze un mecanism care să reprezinte utilizatorului datele respective într-o manieră adecvată.

Liniile de antet din *MIME*:

- *MIME – Version*: este însoțit de parametrul 1.0. Arată că mesajul este codificat conform *RFC* 2045 și *RFC* 2046.
- *Content – Type*: Descrie datele din mesaj, pentru ca serverul destinație să selecteze un mecanism care să reprezinte utilizatorului datele respective într-o manieră adecvată.
- *Content – Transfer – Encoding*: Indică tipul de transformare folosit pentru reprezentarea mesajului într-o manieră acceptabilă pentru transfer.

Liniile de antet din *MIME*:

- *MIME – Version*: este însoțit de parametrul 1.0. Arată că mesajul este codificat conform *RFC* 2045 și *RFC* 2046.
- *Content – Type*: Descrie datele din mesaj, pentru ca serverul destinație să selecteze un mecanism care să reprezinte utilizatorului datele respective într-o manieră adecvată.
- *Content – Transfer – Encoding*: Indică tipul de transformare folosit pentru reprezentarea mesajului într-o manieră acceptabilă pentru transfer.
- *Content – ID*: Este folosit pentru a identifica în mod unic entitățile *MIME*, indiferent de contextul în care se află.

Liniile de antet din *MIME*:

- *MIME – Version*: este însoțit de parametrul 1.0. Arată că mesajul este codificat conform *RFC* 2045 și *RFC* 2046.
- *Content – Type*: Descrie datele din mesaj, pentru ca serverul destinație să selecteze un mecanism care să reprezinte utilizatorului datele respective într-o manieră adecvată.
- *Content – Transfer – Encoding*: Indică tipul de transformare folosit pentru reprezentarea mesajului într-o manieră acceptabilă pentru transfer.
- *Content – ID*: Este folosit pentru a identifica în mod unic entitățile *MIME*, indiferent de contextul în care se află.
- *Content – Description*: O descriere a obiectului care formează mesajul; util când mesajul nu este de tip text.

Liniile de antet din *MIME*:

- *MIME – Version*: este însoțit de parametrul 1.0. Arată că mesajul este codificat conform *RFC* 2045 și *RFC* 2046.
- *Content – Type*: Descrie datele din mesaj, pentru ca serverul destinație să selecteze un mecanism care să reprezinte utilizatorului datele respective într-o manieră adecvată.
- *Content – Transfer – Encoding*: Indică tipul de transformare folosit pentru reprezentarea mesajului într-o manieră acceptabilă pentru transfer.
- *Content – ID*: Este folosit pentru a identifica în mod unic entitățile *MIME*, indiferent de contextul în care se află.
- *Content – Description*: O descriere a obiectului care formează mesajul; util când mesajul nu este de tip text.

Ultimele două cuvinte cheie sunt opționale.

Tipurile de date *Content – Type*

Tip	Subtip	Descriere
Text	Plain	Un text neformatat (de exemplu *ASCII*)
	Enriched	Oferă o flexibilitate sporită a formatului.
Multipart	Mixed	Diferitele componente ale mesajului sunt independente dar sunt transmise împreună. Ele sunt prezentate destinatarului în ordinea în care apar în mesaj.
	Parallel	Diferă de *Mixed* prin faptul că ordinea componentelor în mesaj este aleatoare.
	Alternative	Componentele sunt versiuni alternative pentru aceeași informație. Ele sunt ordonate crescător după similitudinea cu originalul, iar sistemul de mail al destinatarului va lista varianta "cea mai bună".
	Digest	Diferă de *Mixed* prin faptul că tipul principal al fiecărei părți este *message/rfc*822.
Message	rfc822	Corpul este un mesaj conform cu *RFC* 822.
	Partial	Permite fragmentarea unui mesaj mare într-un mod accesibil sistemului de mail al destinatarului
	External-body	Un pointer la un obiect care există în altă parte.
Image	jpeg	Imaginea este în format *JPEG*, codificată *JFIF*.
	gif	Imaginea este în format *GIF*.
Video	mpeg	Format *MPEG*.
Audio	Basic	Se folosește un canal *ISDN* pe 8 biți, codificat la o rată standard de 8 kHz.
Application	PostScript	Postscript Adobe
	octet-stream	Date reprezentate în binar pe 8 biți.

MIME: *Content - Transfer - Encoding*

O definire a codificării pentru corpul mesajului.

MIME: *Content - Transfer - Encoding*

O definire a codificării pentru corpul mesajului.

7bit	Datele sunt reprezentate prin linii scurte de caractere *ASCII*.
8bit	Liniile sunt scurte, dar pot fi caractere non-*ASCII*.
binary	Liniile nu sunt obligatoriu suficient de scurte pentru transferul *SMTP*.
quoted − printable	Dacă datele conțin suficient de mult text *ASCII*, acesta rămâne accesibil unei citiri directe.
base64	Fiecare bloc de 6 biți se codifică într-un bloc de 8 biți – caracter *ASCII* printabil.
x − token	Codificare nestandard.

MIME: *Content - Transfer - Encoding*

O definire a codificării pentru corpul mesajului.

7bit	Datele sunt reprezentate prin linii scurte de caractere *ASCII*.
8bit	Liniile sunt scurte, dar pot fi caractere non-*ASCII*.
binary	Liniile nu sunt obligatoriu suficient de scurte pentru transferul *SMTP*.
quoted − printable	Dacă datele conțin suficient de mult text *ASCII*, acesta rămâne accesibil unei citiri directe.
base64	Fiecare bloc de 6 biți se codifică într-un bloc de 8 biți – caracter *ASCII* printabil.
x − token	Codificare nestandard.

Atributele $7bit, 8bit, binary$ arată că nu se aplică nici o codificare. Pentru transferul *SMTP* forma *7bit* este suficient de sigură.

MIME: *Content - Transfer - Encoding*

O definire a codificării pentru corpul mesajului.

7bit	Datele sunt reprezentate prin linii scurte de caractere *ASCII*.
8bit	Liniile sunt scurte, dar pot fi caractere non-*ASCII*.
binary	Liniile nu sunt obligatoriu suficient de scurte pentru transferul *SMTP*.
quoted − printable	Dacă datele conțin suficient de mult text *ASCII*, acesta rămâne accesibil unei citiri directe.
*base*64	Fiecare bloc de 6 biți se codifică într-un bloc de 8 biți – caracter *ASCII* printabil.
x − token	Codificare nestandard.

Atributele *7bit*, *8bit*, *binary* arată că nu se aplică nici o codificare.
Pentru transferul *SMTP* forma *7bit* este suficient de sigură.
MIME utilizează două metode de codificare a datelor:
quoted − printable și *base*64.

MIME: *Content - Transfer - Encoding*

O definire a codificării pentru corpul mesajului.

7bit	Datele sunt reprezentate prin linii scurte de caractere *ASCII*.
8bit	Liniile sunt scurte, dar pot fi caractere non-*ASCII*.
binary	Liniile nu sunt obligatoriu suficient de scurte pentru transferul *SMTP*.
quoted − printable	Dacă datele conțin suficient de mult text *ASCII*, acesta rămâne accesibil unei citiri directe.
*base*64	Fiecare bloc de 6 biți se codifică într-un bloc de 8 biți – caracter *ASCII* printabil.
x − token	Codificare nestandard.

Atributele $7bit, 8bit, binary$ arată că nu se aplică nici o codificare.
Pentru transferul *SMTP* forma *7bit* este suficient de sigură.
MIME utilizează două metode de codificare a datelor:
quoted − printable și *base*64.
Prima permite un transfer ușor de citit, iar a doua oferă o securitate sporită tuturor tipurilor de date.

Încercarea de a combina protocoalele *PGP* și *MIME* este naturală, dar a întâmpinat mai multe probleme, cea mai semnificativă fiind incapacitatea de a recupera conținutul mesajelor semnate fără analiza structurilor de date specifice *PGP*-ului.

Încercarea de a combina protocoalele *PGP* și *MIME* este naturală, dar a întâmpinat mai multe probleme, cea mai semnificativă fiind incapacitatea de a recupera conținutul mesajelor semnate fără analiza structurilor de date specifice *PGP*-ului.

PGP-ul poate genera în urma criptării mesajelor fie caractere *ASCII*, fie șiruri arbitrare de octeți, generând și o semnătură sau extrăgând datele cheii publice.

Încercarea de a combina protocoalele *PGP* și *MIME* este naturală, dar a întâmpinat mai multe probleme, cea mai semnificativă fiind incapacitatea de a recupera conținutul mesajelor semnate fără analiza structurilor de date specifice *PGP*-ului.

PGP-ul poate genera în urma criptării mesajelor fie caractere *ASCII*, fie șiruri arbitrare de octeți, generând și o semnătură sau extrăgând datele cheii publice.

Protocolul de transmitere a datelor solicită ca acestea să fie în format *ASCII*.

Încercarea de a combina protocoalele *PGP* și *MIME* este naturală, dar a întâmpinat mai multe probleme, cea mai semnificativă fiind incapacitatea de a recupera conținutul mesajelor semnate fără analiza structurilor de date specifice *PGP*-ului.

PGP-ul poate genera în urma criptării mesajelor fie caractere *ASCII*, fie șiruri arbitrare de octeți, generând și o semnătură sau extrăgând datele cheii publice.

Protocolul de transmitere a datelor solicită ca acestea să fie în format *ASCII*.
Dacă mesajul trebuie transmis în mai multe părți, trebuie folosit mecanismul *MIME* de transmitere parțială, în locul formatului *PGP ASCII* multipart.

Înainte de criptarea cu *PGP*, datele trebuie trecute în forma canonică *MIME*.

Înainte de criptarea cu *PGP*, datele trebuie trecute în forma canonică *MIME*.
Criptarea cu *PGP* este anunțată prin antetul *content type multipart/encrypted* și trebuie să aibă valoarea parametrului de protocol *application/pgp-encrypted*.

Înainte de criptarea cu *PGP*, datele trebuie trecute în forma canonică *MIME*.
Criptarea cu *PGP* este anunțată prin antetul *content type multipart/encrypted* și trebuie să aibă valoarea parametrului de protocol *application/pgp-encrypted*.
Corpul mesajului criptat cu *MIME* trebuie să fie compus din două părți, prima parte având antetul *application/pgp-encrypted* și conținând informația de control.

Înainte de criptarea cu *PGP*, datele trebuie trecute în forma canonică *MIME*.
Criptarea cu *PGP* este anunțată prin antetul *content type multipart/encrypted* și trebuie să aibă valoarea parametrului de protocol *application/pgp-encrypted*.
Corpul mesajului criptat cu *MIME* trebuie să fie compus din două părți, prima parte având antetul *application/pgp-encrypted* și conținând informația de control.
A doua parte trebuie să conțină mesajul criptat; ea este etichetată cu antetul *application/octet-stream*.

Pentru generarea semnăturii digitale cu *PGP*:

Pentru generarea semnăturii digitale cu *PGP*:

- Mesajul care trebuie semnat este adus la forma canonică specifică antetului *content type*.

Pentru generarea semnăturii digitale cu *PGP*:

- Mesajul care trebuie semnat este adus la forma canonică specifică antetului *content type*.
- Se aplică o criptare de tipul *Content Transfer Encoding*. În particular, capetele de linie ale mesajului vor folosi secvențe $< CR >< LF >$.

Pentru generarea semnăturii digitale cu *PGP*:

- Mesajul care trebuie semnat este adus la forma canonică specifică antetului *content type*.
- Se aplică o criptare de tipul *Content Transfer Encoding*. În particular, capetele de linie ale mesajului vor folosi secvențe $< CR >< LF >$.
- Se adaugă antetele de conținut *MIME*.

Pentru generarea semnăturii digitale cu *PGP*:

- Mesajul care trebuie semnat este adus la forma canonică specifică antetului *content type*.
- Se aplică o criptare de tipul *Content Transfer Encoding*. În particular, capetele de linie ale mesajului vor folosi secvențe $< CR >< LF >$.
- Se adaugă antetele de conținut *MIME*.
- Din mesajul semnat se înlătură spațiile.

Pentru generarea semnăturii digitale cu *PGP*:

- Mesajul care trebuie semnat este adus la forma canonică specifică antetului *content type*.
- Se aplică o criptare de tipul *Content Transfer Encoding*. În particular, capetele de linie ale mesajului vor folosi secvențe $< CR >< LF >$.
- Se adaugă antetele de conținut *MIME*.
- Din mesajul semnat se înlătură spațiile.
- Semnătura se calculează atât pentru datele care urmează să fie semnate cât și pentru mulțimea antetelor de conținut.

Pentru generarea semnăturii digitale cu *PGP*:

- Mesajul care trebuie semnat este adus la forma canonică specifică antetului *content type*.
- Se aplică o criptare de tipul *Content Transfer Encoding*. În particular, capetele de linie ale mesajului vor folosi secvențe $< CR >< LF >$.
- Se adaugă antetele de conținut *MIME*.
- Din mesajul semnat se înlătură spațiile.
- Semnătura se calculează atât pentru datele care urmează să fie semnate cât și pentru mulțimea antetelor de conținut.
- Semnătura trebuie să fie detașată de mesajul pe care-l însoțește (pentru a se evita modificarea mesajului).

La primirea unui mesaj semnat, *Bob* trebuie:

La primirea unui mesaj semnat, *Bob* trebuie:

- Să aducă la forma canonică $< LC >< LF >$ capetele de linie înainte de verificarea semnăturii.

La primirea unui mesaj semnat, *Bob* trebuie:

- Să aducă la forma canonică $< LC >< LF >$ capetele de linie înainte de verificarea semnăturii.
- Să trimită serviciului de verificare al semnăturii mesajul semnat și antetele de conținut, împreună cu semnătura *PGP*.

Dacă se dorește atât semnarea digitală, cât și criptarea mesajului, sunt două modalități pentru realizarea acestui lucru:

Dacă se dorește atât semnarea digitală, cât și criptarea mesajului, sunt două modalități pentru realizarea acestui lucru:

- Mesajul este întâi semnat conform antetului *multipart/signature* și apoi criptat conform antetului *multipart/encrypted*, pentru a forma corpul final al mesajului.

Dacă se dorește atât semnarea digitală, cât și criptarea mesajului, sunt două modalități pentru realizarea acestui lucru:

- Mesajul este întâi semnat conform antetului *multipart/signature* și apoi criptat conform antetului *multipart/encrypted*, pentru a forma corpul final al mesajului. Acesta este cel mai folosit standard pentru trimiterea mesajelor.

Dacă se dorește atât semnarea digitală, cât și criptarea mesajului, sunt două modalități pentru realizarea acestui lucru:

- Mesajul este întâi semnat conform antetului *multipart/signature* și apoi criptat conform antetului *multipart/encrypted*, pentru a forma corpul final al mesajului. Acesta este cel mai folosit standard pentru trimiterea mesajelor.
- Pachetul *PGP* descrie o modalitate pentru semnarea și criptarea datelor într-un singur mesaj *PGP*.

Dacă se dorește atât semnarea digitală, cât și criptarea mesajului, sunt două modalități pentru realizarea acestui lucru:

- Mesajul este întâi semnat conform antetului *multipart/signature* și apoi criptat conform antetului *multipart/encrypted*, pentru a forma corpul final al mesajului. Acesta este cel mai folosit standard pentru trimiterea mesajelor.
- Pachetul *PGP* descrie o modalitate pentru semnarea și criptarea datelor într-un singur mesaj *PGP*.
  Această metodă favorizează reducerea supraprocesării și crește compatibilitatea cu implementările non-*MIME* ale *PGP*-ului.

Dacă se dorește atât semnarea digitală, cât și criptarea mesajului, sunt două modalități pentru realizarea acestui lucru:

- Mesajul este întâi semnat conform antetului *multipart/signature* și apoi criptat conform antetului *multipart/encrypted*, pentru a forma corpul final al mesajului. Acesta este cel mai folosit standard pentru trimiterea mesajelor.
- Pachetul *PGP* descrie o modalitate pentru semnarea și criptarea datelor într-un singur mesaj *PGP*.
  Această metodă favorizează reducerea supraprocesării și crește compatibilitatea cu implementările non-*MIME* ale *PGP*-ului.
  Este permis în mod explicit unui agent de mail să decripteze un mesaj combinat și să-l rescrie ca un obiect multipart/signed folosind semnătura încorporată în versiunea criptată.

Controverse *MIME*

- Criptarea imbricată: prezentă în primele versiuni; s-a renunțat deoarece structura de ansamblu a mesajului nu putea fi vizibilă fără o decodificare prealabilă.

Controverse *MIME*

- Criptarea imbricată: prezentă în primele versiuni; s-a renunțat deoarece structura de ansamblu a mesajului nu putea fi vizibilă fără o decodificare prealabilă.
  Pot exista criptări imbricate când aceiași informație este codificată de mai multe ori.

Controverse *MIME*

- Criptarea imbricată: prezentă în primele versiuni; s-a renunțat deoarece structura de ansamblu a mesajului nu putea fi vizibilă fără o decodificare prealabilă.
  Pot exista criptări imbricate când aceiași informație este codificată de mai multe ori.
- Uuencode: A fost luat în considerare pentru a înlocui *base* 64.

Controverse *MIME*

- Criptarea imbricată: prezentă în primele versiuni; s-a renunțat deoarece structura de ansamblu a mesajului nu putea fi vizibilă fără o decodificare prealabilă.
  Pot exista criptări imbricate când aceiași informație este codificată de mai multe ori.
- Uuencode: A fost luat în considerare pentru a înlocui *base* 64.A fost respins deoarece:
  - nu era bine specificat;

Controverse *MIME*

- Criptarea imbricată: prezentă în primele versiuni; s-a renunțat deoarece structura de ansamblu a mesajului nu putea fi vizibilă fără o decodificare prealabilă.
  Pot exista criptări imbricate când aceiași informație este codificată de mai multe ori.
- Uuencode: A fost luat în considerare pentru a înlocui *base* 64.A fost respins deoarece:
  - nu era bine specificat;
  - rezultatul în urma aplicării nu era robust pentru unele noduri gateway.

Controverse *MIME*

- Criptarea imbricată: prezentă în primele versiuni; s-a renunțat deoarece structura de ansamblu a mesajului nu putea fi vizibilă fără o decodificare prealabilă.
  Pot exista criptări imbricate când aceiași informație este codificată de mai multe ori.
- Uuencode: A fost luat în considerare pentru a înlocui *base* 64.A fost respins deoarece:
  - nu era bine specificat;
  - rezultatul în urma aplicării nu era robust pentru unele noduri gateway.
  - uneori fișierele Uuencode ajung în puncte diferite într-o formă care nu poate fi decodificată.

Continuare

- Compresia: Nu s-a adoptat datorită situației legale incerte.

Continuare

- Compresia: Nu s-a adoptat datorită situației legale incerte.
- Numărul atributelor Content-type.

S/MIME

S/MIME (*Secure MIME*) este o suplimentare (bazată pe tehnologia *RSA* Data Security) a securității formatului standard *MIME* destinat creșterii securității mesajelor poștei electronice.

S/MIME

S/MIME (*Secure MIME*) este o suplimentare (bazată pe tehnologia *RSA* Data Security) a securității formatului standard *MIME* destinat creșterii securității mesajelor poștei electronice.

În general specificația *S/MIME* este utilizată în standardul pentru uzul comercial și instituțional, pe când *PGP* este folosit pentru utilizatorii obișnuiți ai e-mailului.

Principala deosebire între *S/MIME* și *PGP* este accea că *PGP* permite utilizatorilor să se certifice între ei.

Principala deosebire între *S/MIME* și *PGP* este accea că *PGP* permite utilizatorilor să se certifice între ei.

S/MIME are o utilizare limitată doar la poșta electronică, în timp ce *PGP*-ul are și alte aplicații cum ar fi în *VPN* (*Virtual Private Network*), criptarea volumelor hard-disk-ului, arhivarea și criptarea fișierelor, batch processing etc.

Principala deosebire între *S/MIME* și *PGP* este accea că *PGP* permite utilizatorilor să se certifice între ei.

S/MIME are o utilizare limitată doar la poșta electronică, în timp ce *PGP*-ul are și alte aplicații cum ar fi în *VPN* (*Virtual Private Network*), criptarea volumelor hard-disk-ului, arhivarea și criptarea fișierelor, batch processing etc.

Lungimea cheilor este importantă pentru *PGP* și *S/MIME*; securitatea oferită de *S/MIME* este mult mai scăzută: chei de până la 1024 biți în timp ce specificația *ANSI* recomandă chei de minim 1024 biți pentru *RSA* și Diffie-Hellman (*DH*).

Cleptografie

Se pot construi versiuni de ale protocolului *S/MIME* prin care se implementează rutine de generare a cheilor *DH* sau *RSA* care produc chei aparent normale dar care conțin o trapă secretă prin care un terț poate recupera cheia privată.

Cleptografie

Se pot construi versiuni de ale protocolului *S/MIME* prin care se implementează rutine de generare a cheilor *DH* sau *RSA* care produc chei aparent normale dar care conțin o trapă secretă prin care un terț poate recupera cheia privată.
Acest proces reprezintă o extensie a conceptului de canale subliminale.

Cleptografie

Se pot construi versiuni de ale protocolului *S/MIME* prin care se implementează rutine de generare a cheilor *DH* sau *RSA* care produc chei aparent normale dar care conțin o trapă secretă prin care un terț poate recupera cheia privată.
Acest proces reprezintă o extensie a conceptului de canale subliminale.

Astfel de canale nu pot exista însă în cazul *PGP*-ului: orice inspectare a codului sursă ar depista rapid orice problemă de acest tip.

Securitate

Deci, din perspectiva securității, *PGP*-ul poate fi considerat mai sigur decât *S/MIME*, cel puțin din următoarele motive:

Securitate

Deci, din perspectiva securității, *PGP*-ul poate fi considerat mai sigur decât *S/MIME*, cel puțin din următoarele motive:

- *S/MIME* este limitat la chei publice de 1024 biți; *PGP*-ul poate utiliza de până la 4096 biți.

Securitate

Deci, din perspectiva securității, *PGP*-ul poate fi considerat mai sigur decât *S/MIME*, cel puțin din următoarele motive:

- *S/MIME* este limitat la chei publice de 1024 biți; *PGP*-ul poate utiliza de până la 4096 biți.
- Specificația *PGP*-ului este publică, fiind supusă unei permanente revizuiri de către toți specialiștii, în timp ce utilizatorii *S/MIME* pot doar să aibă încredere în eficiența implementării.

Funcțiile *S/MIME*

- Se poate cripta orice tip de conținut și orice cheie de criptare, pentru unul sau mai mulți destinatari (*enveloped data*).

Funcțiile *S/MIME*

- Se poate cripta orice tip de conținut și orice cheie de criptare, pentru unul sau mai mulți destinatari (*enveloped data*).
- Se pot semna date (*signed data*). Perechea formată din corpul mesajului și semnătură sunt apoi codificate cu *base*64.

Funcțiile *S/MIME*

- Se poate cripta orice tip de conținut și orice cheie de criptare, pentru unul sau mai mulți destinatari (*enveloped data*).
- Se pot semna date (*signed data*). Perechea formată din corpul mesajului și semnătură sunt apoi codificate cu *base*64.
- Se pot semna date în clar (*clear-signed data*).

Funcțiile *S/MIME*

- Se poate cripta orice tip de conținut și orice cheie de criptare, pentru unul sau mai mulți destinatari (*enveloped data*).
- Se pot semna date (*signed data*). Perechea formată din corpul mesajului și semnătură sunt apoi codificate cu *base*64.
- Se pot semna date în clar (*clear-signed data*).Ca și în cazul anterior, se generează o semnătură a corpului mesajului, după care se codifică cu *base*64 numai semnătura.

Funcțiile *S/MIME*

- Se poate cripta orice tip de conținut și orice cheie de criptare, pentru unul sau mai mulți destinatari (*enveloped data*).
- Se pot semna date (*signed data*). Perechea formată din corpul mesajului și semnătură sunt apoi codificate cu *base*64.
- Se pot semna date în clar (*clear-signed data*).Ca și în cazul anterior, se generează o semnătură a corpului mesajului, după care se codifică cu *base*64 numai semnătura.
- Se pot semna și cripta date (*signed and enveloped data*).

Funcțiile *S/MIME*

- Se poate cripta orice tip de conținut și orice cheie de criptare, pentru unul sau mai mulți destinatari (*enveloped data*).
- Se pot semna date (*signed data*). Perechea formată din corpul mesajului și semnătură sunt apoi codificate cu *base*64.
- Se pot semna date în clar (*clear-signed data*).Ca și în cazul anterior, se generează o semnătură a corpului mesajului, după care se codifică cu *base*64 numai semnătura.
- Se pot semna și cripta date (*signed and enveloped data*).Corpul mesajului (criptat sau nu) se semnează, apoi semnătura (singură sau împreună cu corpul mesajului – criptat sau nu) se criptează din nou.

Algoritmi criptografici

A. Pentru crearea unei amprente: Funcțiile de dispersie folosite sunt $SHA-1$ și $MD5$.

Algoritmi criptografici

A. Pentru crearea unei amprente: Funcțiile de dispersie folosite sunt $SHA-1$ și $MD5$.

B. Pentru criptarea amprentei (și generarea semnăturii):

Algoritmi criptografici

A. Pentru crearea unei amprente: Funcțiile de dispersie folosite sunt $SHA-1$ și $MD5$.

B. Pentru criptarea amprentei (și generarea semnăturii):

- Ambele părți folosesc *DSS*.

Algoritmi criptografici

A. Pentru crearea unei amprente: Funcțiile de dispersie folosite sunt $SHA-1$ și $MD5$.

B. Pentru criptarea amprentei (și generarea semnăturii):

- Ambele părți folosesc *DSS*.
- Expeditorul folosește *RSA* pentru criptare.

Algoritmi criptografici

A. Pentru crearea unei amprente: Funcțiile de dispersie folosite sunt $SHA-1$ și $MD5$.

B. Pentru criptarea amprentei (și generarea semnăturii):

- Ambele părți folosesc *DSS*.
- Expeditorul folosește *RSA* pentru criptare.
- Destinatarul trebuie să poată verifica semnături *RSA* cu chei până la 1024 biți.

Continuare

C. Pentru criptarea cheilor de sesiune:

Continuare

C. Pentru criptarea cheilor de sesiune:

- Ambele părți folosesc *ElGamal*.

Continuare

C. Pentru criptarea cheilor de sesiune:

- Ambele părți folosesc *ElGamal*.
- Expeditorul mai poate folosi pentru criptare *RSA* cu chei până la 1024 biți.

Continuare

C. Pentru criptarea cheilor de sesiune:

- Ambele părți folosesc *ElGamal*.
- Expeditorul mai poate folosi pentru criptare *RSA* cu chei până la 1024 biți.
- Destinatarul trebuie să poată efectua decriptări *RSA*.

Continuare

C. Pentru criptarea cheilor de sesiune:

- Ambele părți folosesc *ElGamal*.
- Expeditorul mai poate folosi pentru criptare *RSA* cu chei până la 1024 biți.
- Destinatarul trebuie să poată efectua decriptări *RSA*.

D. Pentru criptarea mesajelor cu chei de sesiune one-time:

Continuare

C. Pentru criptarea cheilor de sesiune:

- Ambele părți folosesc *ElGamal*.
- Expeditorul mai poate folosi pentru criptare *RSA* cu chei până la 1024 biți.
- Destinatarul trebuie să poată efectua decriptări *RSA*.

D. Pentru criptarea mesajelor cu chei de sesiune one-time:

- Expeditorul poate folosi criptarea cu $3DES$ sau $RC2/40$.

Continuare

C. Pentru criptarea cheilor de sesiune:

- Ambele părți folosesc *ElGamal*.
- Expeditorul mai poate folosi pentru criptare *RSA* cu chei până la 1024 biți.
- Destinatarul trebuie să poată efectua decriptări *RSA*.

D. Pentru criptarea mesajelor cu chei de sesiune one-time:

- Expeditorul poate folosi criptarea cu $3DES$ sau $RC2/40$.
- Expeditorul trebuie să poată decripta $3DES$ (sau eventual $RC2/40$).

Specificațiile *S/MIME* includ o discuție asupra procedurii de selecție a algoritmului de criptare.
De obicei, *Alice* trebuie să facă două alegeri:

Specificațiile *S/MIME* includ o discuție asupra procedurii de selecție a algoritmului de criptare.
De obicei, *Alice* trebuie să facă două alegeri:

1. Stabilește dacă *Bob* este capabil să decripteze un mesaj criptat cu un sistem de criptare dat.

Specificațiile *S/MIME* includ o discuție asupra procedurii de selecție a algoritmului de criptare.
De obicei, *Alice* trebuie să facă două alegeri:

1. Stabilește dacă *Bob* este capabil să decripteze un mesaj criptat cu un sistem de criptare dat.
2. Dacă *Bob* acceptă numai texte criptate cu sisteme slabe, *Alice* trebuie să decidă dacă poate trimite mesajul folosind o criptare slabă.

Specificațiile *S/MIME* includ o discuție asupra procedurii de selecție a algoritmului de criptare.
De obicei, *Alice* trebuie să facă două alegeri:

1. Stabilește dacă *Bob* este capabil să decripteze un mesaj criptat cu un sistem de criptare dat.
2. Dacă *Bob* acceptă numai texte criptate cu sisteme slabe, *Alice* trebuie să decidă dacă poate trimite mesajul folosind o criptare slabă.

Toți partenerii anunță în prealabil capacitățile solicitate pentru decriptarea mesajelor pe care le trimit, în ordinea descrescătoare a preferințelor.

Specificațiile *S/MIME* includ o discuție asupra procedurii de selecție a algoritmului de criptare.
De obicei, *Alice* trebuie să facă două alegeri:

1. Stabilește dacă *Bob* este capabil să decripteze un mesaj criptat cu un sistem de criptare dat.
2. Dacă *Bob* acceptă numai texte criptate cu sisteme slabe, *Alice* trebuie să decidă dacă poate trimite mesajul folosind o criptare slabă.

Toți partenerii anunță în prealabil capacitățile solicitate pentru decriptarea mesajelor pe care le trimit, în ordinea descrescătoare a preferințelor.
Fiecare destinatar poate stoca aceste informații pentru a le folosi ulterior.

Pentru a trimite un mesaj, *Alice* verifică:

Pentru a trimite un mesaj, *Alice* verifică:

1. Dacă are o listă de capacități de decriptare a lui *Bob*, ea va alege primul sistem de criptare pe care este capabilă să-l folosească.

Pentru a trimite un mesaj, *Alice* verifică:

1. Dacă are o listă de capacități de decriptare a lui *Bob*, ea va alege primul sistem de criptare pe care este capabilă să-l folosească.
2. Dacă nu are lista lui *Bob*, dar deține cel puțin un mesaj de la el, atunci mesajul trimis de *Alice* va folosi același algoritm de criptare folosit în mesajele respective.

Pentru a trimite un mesaj, *Alice* verifică:

1. Dacă are o listă de capacități de decriptare a lui *Bob*, ea va alege primul sistem de criptare pe care este capabilă să-l folosească.
2. Dacă nu are lista lui *Bob*, dar deține cel puțin un mesaj de la el, atunci mesajul trimis de *Alice* va folosi același algoritm de criptare folosit în mesajele respective.
3. Dacă nu deține nici o informație despre capacitățile de decriptare ale lui *Bob*, dar este dispusă să riște ca mesajul să nu poată fi citit, va trimite un mesaj criptat cu $3DES$.

Pentru a trimite un mesaj, *Alice* verifică:

1. Dacă are o listă de capacități de decriptare a lui *Bob*, ea va alege primul sistem de criptare pe care este capabilă să-l folosească.
2. Dacă nu are lista lui *Bob*, dar deține cel puțin un mesaj de la el, atunci mesajul trimis de *Alice* va folosi același algoritm de criptare folosit în mesajele respective.
3. Dacă nu deține nici o informație despre capacitățile de decriptare ale lui *Bob*, dar este dispusă să riște ca mesajul să nu poată fi citit, va trimite un mesaj criptat cu $3DES$.
4. Dacă nu deține nici o informație despre capacitățile de decriptare ale lui *Bob*, și nu vrea să riște ca mesajul să nu poată fi citit de acesta, atunci *Alice* va trimite un mesaj criptat cu $RC2/40$.

ZIP

Creat în 1977; funcțional, este echivalent cu PKZIP, sistemul creat de PKWARE și folosit de Windows.

ZIP

Creat în 1977; funcțional, este echivalent cu PKZIP, sistemul creat de PKWARE și folosit de Windows.
Se pare că este cel mai utilizat algoritm de compresie; este free și a fost implementat pentru toate sistemele actuale de calcul.

ZIP

Creat în 1977; funcțional, este echivalent cu PKZIP, sistemul creat de PKWARE și folosit de Windows.
Se pare că este cel mai utilizat algoritm de compresie; este free și a fost implementat pentru toate sistemele actuale de calcul.
Baza teoretică (definită în algoritmul *L*277, precursor al *ZIP*-ului): multe cuvinte dintr-un text (patternuri de imagini în cazul unui GIF etc) se repetă.
Această secvență care se repetă poate fi înlocuită cu un cod scurt.

Exemplu

the brown fox jumped over the brown foxy jumping frog

Exemplu

the brown fox jumped over the brown foxy jumping frog

Lungimea este de 53 octeți = 424 biți.

Exemplu

the brown fox jumped over the brown foxy jumping frog

Lungimea este de 53 octeți = 424 biți.Se parcurge textul.

Fiecare caracter este codificat într-o secvență de 9 biți, unde primul bit este 1, iar următorii opt biți conțin codul ASCII al caracterului.

Exemplu

the brown fox jumped over the brown foxy jumping frog

Lungimea este de 53 octeți = 424 biți.Se parcurge textul.

Fiecare caracter este codificat într-o secvență de 9 biți, unde primul bit este 1, iar următorii opt biți conțin codul ASCII al caracterului.

Algoritmul caută secvențele de lungime maximă care se repetă.
În cazul nostru, aceasta este **the brown fox**.

Exemplu

Se păstrează prima apariție a secvenței; celelalte sunt înlocuite de un pointer la această apariție și de un număr care dă lungimea secvenței.

Exemplu

Se păstrează prima apariție a secvenței; celelalte sunt înlocuite de un pointer la această apariție și de un număr care dă lungimea secvenței.
În cazul nostru, a doua apariție se înlocuiește cu $(26, 13)$ (secvența de aici repetă pe cea care a început 26 caractere mai devreme, pe o lungime de 13 caractere).

Exemplu

Se păstrează prima apariție a secvenței; celelalte sunt înlocuite de un pointer la această apariție și de un număr care dă lungimea secvenței.
În cazul nostru, a doua apariție se înlocuiește cu $(26, 13)$ (secvența de aici repetă pe cea care a început 26 caractere mai devreme, pe o lungime de 13 caractere).

Sunt două opțiuni de codificare a perechii (specificate la începutul perechii): un pointer pe 8 biți și un număr pe 4 biți (opțiunea 00) sau un pointer pe 12 biți și un număr pe 6 biți (opțiunea 01).

Exemplu

Se păstrează prima apariție a secvenței; celelalte sunt înlocuite de un pointer la această apariție și de un număr care dă lungimea secvenței.
În cazul nostru, a doua apariție se înlocuiește cu $(26, 13)$ (secvența de aici repetă pe cea care a început 26 caractere mai devreme, pe o lungime de 13 caractere).

Sunt două opțiuni de codificare a perechii (specificate la începutul perechii): un pointer pe 8 biți și un număr pe 4 biți (opțiunea 00) sau un pointer pe 12 biți și un număr pe 6 biți (opțiunea 01).
Deci a doua apariție a secvenței **the brown fox** este codificată $< 00_b >< 26_d >< 13_d >$ (b - binar, d - zecimal),

Exemplu

Se păstrează prima apariție a secvenței; celelalte sunt înlocuite de un pointer la această apariție și de un număr care dă lungimea secvenței.
În cazul nostru, a doua apariție se înlocuiește cu $(26, 13)$ (secvența de aici repetă pe cea care a început 26 caractere mai devreme, pe o lungime de 13 caractere).

Sunt două opțiuni de codificare a perechii (specificate la începutul perechii): un pointer pe 8 biți și un număr pe 4 biți (opțiunea 00) sau un pointer pe 12 biți și un număr pe 6 biți (opțiunea 01).
Deci a doua apariție a secvenței **the brown fox** este codificată $< 00_b >< 26_d >< 13_d >$ (b - binar, d - zecimal),sau
00 00011010 1101.

Exemplu

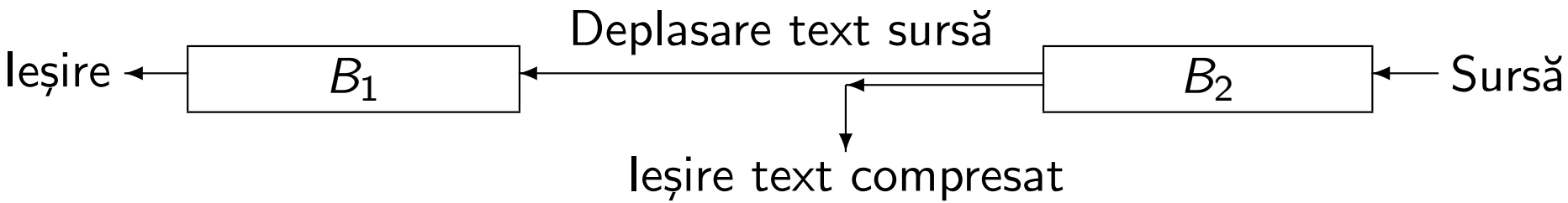
Din mesaj a rămas

the brown fox jumped over $0_b26_d13_d$ **y** $0_b27_d5_d$ **ing frog**

Textul compresat are 35 caractere de câte 9 biți și două coduri; în total $35 \times 9 + 2 \times 14 = 343$ biți; o rată de compresie de $1,24$.

Algoritmul folosește două buffere (B_1 și B_2).

Algoritmul folosește două buffere (B_1 și B_2).



B_1 conține ultimele n caractere ale sursei care au fost prelucrate, iar B_2 conține următoarele p caractere care urmează să fie prelucrate.

Algoritmul caută dacă un prefix α ($|\alpha| = s \geq 2$) din B_2 se află ca subșir în B_1.

Algoritmul caută dacă un prefix α ($|\alpha| = s \geq 2$) din B_2 se află ca subșir în B_1.
Dacă nu, primul caracter din B_2 iese codificat pe 9 biți în secvența compresată și – simultan – este deplasat în B_1.

Algoritmul caută dacă un prefix α ($|\alpha| = s \geq 2$) din B_2 se află ca subșir în B_1.
Dacă nu, primul caracter din B_2 iese codificat pe 9 biți în secvența compresată și – simultan – este deplasat în B_1.
Dacă α este în B_1, se continuă căutarea pentru a găsi cel mai lung subșir comun, de lungime k.
Acesta este scos ca text compresat sub forma unui triplet (indicator, pointer, k).

Algoritmul caută dacă un prefix α ($|\alpha| = s \geq 2$) din B_2 se află ca subșir în B_1.
Dacă nu, primul caracter din B_2 iese codificat pe 9 biți în secvența compresată și – simultan – este deplasat în B_1.
Dacă α este în B_1, se continuă căutarea pentru a găsi cel mai lung subșir comun, de lungime k.
Acesta este scos ca text compresat sub forma unui triplet (indicator, pointer, k).
Cele mai din stânga k caractere din B_1 sunt eliminate, iar cele k caractere din topul lui B_2 trec în B_1.

radix – 64

Deoarece sistemele de e-mail permit transmiterea și recepția doar a codurilor *ASCII* pe 8 biți, protocoalele de e-mail convertesc textul criptat alocând fiecărui grup de 6 biți, un caracter *ASCII* printabil.

radix – 64

Deoarece sistemele de e-mail permit transmiterea și recepția doar a codurilor *ASCII* pe 8 biți, protocoalele de e-mail convertesc textul criptat alocând fiecărui grup de 6 biți, un caracter *ASCII* printabil. Algoritmul de codificare folosit de *PGP* și *S/MIME* este numit *radix* – 64.

radix − 64

Deoarece sistemele de e-mail permit transmiterea și recepția doar a codurilor *ASCII* pe 8 biți, protocoalele de e-mail convertesc textul criptat alocând fiecărui grup de 6 biți, un caracter *ASCII* printabil. Algoritmul de codificare folosit de *PGP* și *S/MIME* este numit *radix* − 64.
MIME folosește protocolul de codificare *base*64 identic cu *radix* − 64, singura deosebire fiind adăgarea unei sume de control pe 24 biți, calculată înainte de codificare.

radix − 64

Deoarece sistemele de e-mail permit transmiterea și recepția doar a codurilor *ASCII* pe 8 biți, protocoalele de e-mail convertesc textul criptat alocând fiecărui grup de 6 biți, un caracter *ASCII* printabil.
Algoritmul de codificare folosit de *PGP* și *S/MIME* este numit *radix* − 64.
MIME folosește protocolul de codificare *base*64 identic cu *radix* − 64, singura deosebire fiind adăgarea unei sume de control pe 24 biți, calculată înainte de codificare.
Suma este codificată cu același algoritm și scrisă la începutul mesajului, separată de rest prin simbolul "=".

Caracteristici *radix* – 64

- Domeniul de definiție este mulțimea caracterelor reprezentabile binar (orice caracter reprezentabil în binar).

Caracteristici *radix* – 64

- Domeniul de definiție este mulțimea caracterelor reprezentabile binar (orice caracter reprezentabil în binar).
- Mulțimea valorilor este formată din 65 caractere printabile: unul este pentru legătură (pad), iar celelalte $2^6 = 64$ sunt reprezentate pe 6 biți.

Caracteristici *radix* – 64

- Domeniul de definiție este mulțimea caracterelor reprezentabile binar (orice caracter reprezentabil în binar).
- Mulțimea valorilor este formată din 65 caractere printabile: unul este pentru legătură (pad), iar celelalte $2^6 = 64$ sunt reprezentate pe 6 biți.
- Nu există caractere de control.

Caracteristici *radix* – 64

- Domeniul de definiție este mulțimea caracterelor reprezentabile binar (orice caracter reprezentabil în binar).
- Mulțimea valorilor este formată din 65 caractere printabile: unul este pentru legătură (pad), iar celelalte $2^6 = 64$ sunt reprezentate pe 6 biți.
- Nu există caractere de control.
- Caracterul "–" (cu semnificație în multe sisteme, inclusiv *RFC* 822) nu este folosit; deci el trebuie eliminat anterior.

Codificarea celor 64 caractere din sistemul *radix* − 64 (sunt folosite cele 52 litere, cele zece cifre și caracterele $+, /$):

Codificarea celor 64 caractere din sistemul *radix* − 64 (sunt folosite cele 52 litere, cele zece cifre și caracterele +, /):

Valoare 6-biți	Cod Caracter	Valoare 6-biți	Cod Caracter	Valoare 6-biți	Cod Caracter	Valoare 6-biți	Cod Caracter
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/
						pad	=

Procedura de codificare:

1. Fiecare secvență de 3 octeți (24 cifre binare) este împărțită în patru secvențe de 6 biți fiecare.

Procedura de codificare:

1. Fiecare secvență de 3 octeți (24 cifre binare) este împărțită în patru secvențe de 6 biți fiecare.
2. Apoi fiecare grup de 6 biți este codificat prin caracterul din tabel.

Procedura de codificare:

1. Fiecare secvență de 3 octeți (24 cifre binare) este împărțită în patru secvențe de 6 biți fiecare.
2. Apoi fiecare grup de 6 biți este codificat prin caracterul din tabel.

Deci o intrare de 24 biți este expandată la ieșire pe 32 biți.

Exemplu

Fie

$$00100011 \;\; 01011100 \;\; 10010001$$

($235C91$ în hexazecimal).

Exemplu

Fie

$$00100011 \;\; 01011100 \;\; 10010001$$

($235C91$ în hexazecimal).
Aranjată în grupuri de câte 6 biți:

$$001000 \;\; 110101 \;\; 110010 \;\; 010001.$$

Valorile lor zecimale sunt $8, 53, 50$ și 17.

Exemplu

Fie

$$00100011 \;\; 01011100 \;\; 10010001$$

($235C91$ în hexazecimal).
Aranjată în grupuri de câte 6 biți:

$$001000 \;\; 110101 \;\; 110010 \;\; 010001.$$

Valorile lor zecimale sunt $8, 53, 50$ și 17.
Folosind acum tabela *radix* $-$ 64 se obține **I1yR**.

Exemplu

Fie

$$00100011 \;\; 01011100 \;\; 10010001$$

($235C91$ în hexazecimal).
Aranjată în grupuri de câte 6 biți:

$$001000 \;\; 110101 \;\; 110010 \;\; 010001.$$

Valorile lor zecimale sunt $8, 53, 50$ și 17.
Folosind acum tabela *radix* $-$ 64 se obține **I1yR**.
Trecută în format *ASCII* (8 biți, cu 0 pe bitul de paritate), avem

$$01001001 \;\; 00110001 \;\; 01111001 \;\; 01010010$$

Exemplu

Fie

$$00100011 \;\; 01011100 \;\; 10010001$$

($235C91$ în hexazecimal).
Aranjată în grupuri de câte 6 biți:

$$001000 \;\; 110101 \;\; 110010 \;\; 010001.$$

Valorile lor zecimale sunt $8, 53, 50$ și 17.
Folosind acum tabela *radix* – 64 se obține **I1yR**.
Trecută în format *ASCII* (8 biți, cu 0 pe bitul de paritate), avem

$$01001001 \;\; 00110001 \;\; 01111001 \;\; 01010010$$

Sau – în hexazecimal – 49317952.

Exemplu

Fie

$$00100011 \;\; 01011100 \;\; 10010001$$

($235C91$ în hexazecimal).
Aranjată în grupuri de câte 6 biți:

$$001000 \;\; 110101 \;\; 110010 \;\; 010001.$$

Valorile lor zecimale sunt $8, 53, 50$ și 17.
Folosind acum tabela *radix* – 64 se obține **l1yR**.
Trecută în format *ASCII* (8 biți, cu 0 pe bitul de paritate), avem

$$01001001 \;\; 00110001 \;\; 01111001 \;\; 01010010$$

Sau – în hexazecimal – 49317952.
Deci, codificarea *radix* – 64 transformă $235C91$ în 49317952.

Sfârșit