IV Međunarodno savjetovanje o telekomunikacijama "BIHTEL 2002"
"Telekomunikacije – usluge i tehnologije"

NAPREDNI ENKRIPCIJSKI STANDARD (AES) – RIJNDAEL ALGORITAM KOMPARATIVNA ANALIZA

ADVANCED ENCRYPTION STANDARD (AES) – RIJNDAEL ALGORITHM COMPARATIVE ANALYSIS

Saša Mrdović, dipl.ing.el.
Elektrotehnički Fakultet, Univerzitet u Sarajevu
e-mail: sasa.mrdovic@etf.unsa.ba

Sažetak: *Rad opisuje napredni enkripcijski standard (Advanced Encryption Standard - AES), zvanični standard vlade SAD za zaštitu elektronskih dokumenta. Dat je kratak pregled simetričnih blok algoritama šifriranja. Posebno su opisani finalisti za izbor AES (MARS, RC6, Rijndael, Serpent i Twofish). Izabrani algoritam Rijndael je prikazan sa više detalja. Prezentirani su uporedni rezultati testiranja različitih karakteristika finalista i drugih industrijskih algoritama. Napravljen je osvrt kako se AES, kao i drugi enkripcijski algoritmi, mogu primijeniti u industriji i standardizaciji u oblasti sigurnosti prijenosa i zaštite sadržaja u BiH.*

Ključne riječi: *Zaštita podataka, kriptografija, simetrični blok algoritmi, Advanced Encryption Standard (AES), Rijndael*

Abstract: *Paper presents Advanced Encryption Standard (AES), US government standard for protection of electronic data. Short review of symmetric block ciphers is given. AES selection finalists (MARS, RC6, Rijndael, Serpent and Twofish) are described. Selected Rijndael algorithm is presented in more detail. Comparative test results of different characteristics of AES finalists and other industry algorithms are presented. Possible applications of AES, as well as other encryption algorithms, in industry and standardization in the area of data transfer security and content protection in B&H are reviewed.*

Key words: *Data security, cryptography, symmetric block ciphers, Advanced Encryption Standard (AES), Rijndael*

1. UVOD

Sigurnost podataka, kao i njihova cjelovitost i autentičnost, su preduslovi sigurnih komunikacija. Tajnost podataka koji se prenose elektronskim putem od suštinske je važnosti za razvoj elektronskog poslovanja. Ova tajnost obezbjeđuje se šifriranjem podataka primjenom enkripcijskih algoritma. Razvoj računarske tehnologije i nova matematička otkrića čine da nekada sigurni algoritmi to prestaju biti, ali istovremeno omogućavaju razvoj novih algoritama. U maju 2002. godine Napredni enkripcijski standard (AES) po prijedlogu Nacionalnog instituta za standarde i tehnologiju SAD (NIST) postao je zvanični algoritam za enkripciju dokumenata u federalnim organima SAD [1]. Pošto je dosadašnji Standard za enkripciju podataka (DES) istih institucija bio *de facto* industrijski standard u svijetu, realno je očekivati je da će sada AES preuzeti tu ulogu.

2. ISTORIJAT

Standard za enkripciju podataka (DES) razvijen je sedamdesetih godina ovog vijeka od strane IBM-a. Nacionalni biro za standarde SAD, sadašnji NIST, prihvatio je DES i on je postao zvanični standard vlade SAD 1977 godine [2]. DES je simetrični sistem kriptografije sa privatnim ključem. DES šifrira podatke u blokovima od 64 bita koristeći 56 bitni ključ. DES je testiran tokom godina i smatra se pouzadnim algoritmom. Njegova slabost je u dužini bloka podataka koji šifrira, a još više u dužini ključa. Za današnje pojmove, mala dužina ključa učinila je DES neotpornim na napade u kojima se isprobava svaka moguća kombinacija (brute force) tako da je u junu 1997. DES razbijen koordiniranim naporom koji je uključivao veliki broj računara povezanih preko Interneta.

Trostruki DES (3-DES) koji je trenutno važeći SAD standard paralelno sa AES-om je na neki način riješio problem sa dužinom DES ključa primjenjujući DES algoritam tri puta. U ovom procesu se ulazni podaci prvo enkriptuju, zatim dekriptuju, a onda ponovo enkriptuju, koristeći različiti ključ svaki put ili u drugoj verziji isti ključ za prvi i treći korak, a različit za drugi. Ovaj algoritam je mnogo sigurniji, ali je osjetno sporiji u odnosu na neke novije algoritme koji rade sa istom (64 bita) dužinom bloka podataka. ([2]).

Devedesetih godina se pojavilo još nekoliko izvrsnih simetričnih blok algoritama. Nabrojaću samo najvažnije:

International Data Encryption Algorithm (IDEA) razvijen od strane Švajcarskog ETH 1992 godine. Ovaj algoritam se smatra jednim od naboljih poznatih algoritama. Šifriraju se blokovi od 64 bita podataka sa 128 bitnim ključem. Za sada se ne zna niti za jedan uspješan pokušaj razbijanja ovog algoritma. Algoritam je dizajniran tako da njegova implementacija, bilo u softveru ili hardveru, bude brza i da ne bude skupa. Korist se kao jedan od algoritama za PGP (Pretty Good Privacy) skup programa za šifriranje.

Blowfish je algoritam koji je razvio Bruce Schneier 1993 godine. Blokovi dužine 64 bita šifriraju se ključem promijenjljive dužine do 448 bita. Ovaj algoritam je izrazito brz, jednostavne strukture i malih memorijskih zahtjeva. Nije tako detaljno kriptološki analiziran kao što je slučaj sa DES-om i IDEA-om. Koristi substitucione matrice (S-box) zavisne od slučajnog ključa. Primjenjuje se u velikom broju aplikacija.

RC5 je razvio Ron Rivers 1994 za kompaniju RSA Data Security Inc. Dužina bloka podataka može biti 32, 64 ili 128 bita. Ključ može bit dužine i do 2040 bita. Ovo daje veliku fleksibilnost pri izboru željene brzine i sigurnosti. Jednostavne je strukture i malih memorijskih zahtjeva. Koristi se u aplikacijama kompanije RSA Security.

I pored neospornih kvaliteta gore navedenih algoritama niti jedan od njih nije preovladao kao *de facto* industrijski standard. Neki od razloga za to leže i u ograničenjima na njihovo korištenje, kao što su izvozna ograničenja za algoritme razvijene u SAD, kao i troškovima licenciranja.

3. IZBOR AES-a

Pošto je postalo očigledno da DES više nije dovoljno siguran standard, a da trostruki DES nije algoritam sa potrebnim karakteristikama za savremenu primjenu, NIST je 1997. pokrenuo proceduru za izbor Naprednog enkripcijskog standarda (AES). Pri definisanju zahtjeva koje traženi algoritam mora ispunjavati otklonjeni su neki problemi i prigovori koji su postojali prilikom izbora DES-a. Prije svega ovo se odnosi na to da su prijave mogle stizati iz cijelog svijeta, da ne smije biti ograničenja na izvoz ili licenciranja algoritma i da algoritam mora biti javno dostupan. Algoritam je trebao biti simetrična blok enkripcija sa podržanom dužinom bloka od 128 bita i podržanim ključevima dužine 128, 192 i 256 bita. Kriteriji za izbor su bili sigurnost, efikasnost (brzina) pri softverskim i hardverskim implementacijama, memorijski zahtjevi i fleksibilnost [3].

Od 15 prijavljenih algoritama 1999. godine izabrano je pet finalista [4]. Radi kompletnosti rada dajem karakteristike svakog od finalista.

3.1. MARS

MARS je algoritam koji je razvio IBM. Algoritam ima nekoliko nivoa: dodavanje ključa kao preizbjeljivanje, osam iteracija miješanja unaprijed bez ključa, osam iteracija miješanja unaprijed sa ključem, osam iteracija miješanja unazad sa ključem, osam iteracija miješanja unazad bez ključa i oduzimanje ključa kao postizbjeljivanje. Iteracije sa ključem, njih šesnaest, se nazivaju kriptografskim jezgrom. Iteracije bez ključa koriste dvije 8x32 bitne substitucione matrice, dodavanje i XOR operaciju. Pored ovih elemenata iteracije sa ključem koriste 32 bitno množenje sa ključem, rotacije zavisne od podataka i dodavanje ključa. I miješanje i iteracije sa ključem su Feistel iteracije u kojima se jedna četvrtina bloka podataka koristi za promjenu ostale tri četvrtine bloka. Algoritam ima veliku marginu sigurnosti. Performanse su vrlo dobre na 32 bitnim platformama, a odlične na platformama koje podržavaju rotaciju i množenje 32 bitnih varijabli. Podržava ključeve i veće dužine od zahtijevanih 256 bita. Lošije strane su moguća degradacija performansi na neodgovarjućim platformama kao i kompleksnost algoritma. [4], [5],[6]

3.2. RC6

RC6 je algoritam koji je razvio Ronald Rivest iz RSA Security u saradnji sa Matt Robshaw, Ray Sidney i Yiqun Lisa Yin. Algoritam je parametrizirana familija šifratora koji u osnovi koristi Feistel strukturu. Iterativna funkcija obavlja rotaciju variabli koja je regulisana kvadratičnom funkcijom podataka. Svaka iteracija takođe uključuje 32 bitnu modularnu multiplikaciju, dodavanje, XOR operaciju i dodavanje ključa. Dodavanje ključa se takođe koristi za pre- i postizbjeljivanje. Nastao je na osnovi ranije pomenutog RC5. RC6 je vrlo jednostavan i njegova struktura je raltivno poznata i testirana kroz RC5. Vrlo je brz na 32 bitnim platformama. Algoritam je iznimno fleksibilan po pitanju veličine ključa, bloka i broja rundi pri enkripciji i dekripciji. Lošije strane su relativno niska margina sigurnosti, opadajuće performanse na neodgovarajućim platformama, te nedovoljna pogodnost za implementaciju na pametnim karticama (smart card) [4],[5],[7].

3.3. Rijndael

Autori Rijndael-a su Joan Daemen i Vincent Rijmen iz Belgije. Algoritam je substituciono permutaciona mreža sa 10, 12 ili 14 iteracija u zavisnosti od dužine ključa. Blok podataka koji se procesira dijeli se u niz bajta i sve operacije šifriranja su bazirane na bajtima. Iteraciona funkcija ima četiri nivoa. U prvom se 8x8 substituciona matrica primijenjuje na svaki bajt. Drugi i treći su nivoi linearnog miješanja u kojima se redovi matrice pomjeraju a kolone miješaju. Na četvrtom nivou nad bajtima podključeva i matrice podataka obavlja se XOR operacija. U posljednjoj iteraciji izostavlja se miješanje kolona. Nastao je na osnovama algoritma Square od istih autora. Algoritam ima dobru sigurnosnu marginu. Izvrsne performanse na svim platformama. Mali memorijski zahtijevi čine ga vrlo pogodnim za implementaciju na pametnim karticama. Mogućnost korištenja i većih blokova i većih ključeva od zahtijevanih čini ga dodatno fleksibilnim. Lošije strane bi mogle biti relativno nova i nedovoljno ispitana struktura i nešto lošija performansa sa većim ključevima. [4],[5],[8]

3.4. Serpent

Serpent su dizajnirali Ross Anderson, Eli Biham and Lars Knudsen. Struktura algoritma je substiciono permutaciona mreža koja se sastoji od 32 iteracije. Algoritam takođe specificira nekriptografske inicijalne i završne permutacije koje omogućavaju alternativnu implementaciju zvanu ”bitslice” način rada. Iterativna funkcija se sastoji od tri nivoa: XOR operacija sa ključem, 32 paralelne primjene jedne od osam posebnih 4x4 substitucionih matrica i linearne transformacije. U posljednjoj iteraciji drugi nivo XOR operacije sa ključem zamjenjuje linearnu transformaciju. Osnova na kojoj je razvijen je algoritam Bitslice. Serpent ima sveukupno najbolje sigurnosne karakteristike. Pogodan je za implementaciju na pametnim karticama. Lošija strana mu je mala brzina. [4],[5],[9]

3.5. Twofish

Twofish je razvijen u kompaniji Counterpane Internet Security od strane Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall i Niels Ferguson. Algoritam je Feistel mreža sa 16 iteracija. Feistel struktura je nešto modificirana korištenjem jednobitne rotacije. Iterativna funkcija radi sa 32 bitnim riječima sa četiri o ključu zavisne 8x8 substitucione matrice, nakon čega slijedi fiksna 4x4 separabilna matrica maksimalne udaljenosti po GF(2^8), pseudo-Hadamard transformacija i dodavanje ključa. Zasnovan je na ranije spomenutom Blowfish. Algoritam ima veliku marginu sigurnosti. Twofish je voma brz na svim platformama. Mali meorijski zahtijevi čine ga pogodnim za pametne kartice i druge memorijski restriktivne sredine. Određena podešavanja omogućavaju podešavanje odnosa brzine i zahtijevanog memorijskog prostora. Lošija strana je ukupna kompleksnost dizajna. [4],[5],[10]

U oktobru 2000. godine NIST je objavio odluku da je izabrao Rijndael kao prijedlog za novi AES.[5]

4. RIJNDAEL ALGORITAM

Rijndael je iterativni blok šifrarni algoritam kod koga dužina bloka podataka koji se šifrira i dužina ključa koji se koristi za šifriranje mogu nezavisno jedna od druge biti 128, 192 ili 256 bita. Ovdje dajem kratak opis algoritma. Detaljniji opis i specifikacije dostupni su navednoj literaturi [11],[1].

Rutina za šifriranje podataka radi na ulaznim blokovima (in) koristeći prošireni ključ (w) i kao izlaz daje šifrirani blok (out) iste dužine kao i ulazni blok. Pseudo kod za ovu rutinu je:

```
Cipher(byte in[4*Nb],
       byte out[4*Nb],
       word w[Nb*(Nr+1)])
begin

  byte state[4,Nb]

  state = in

  AddRoundKey(state, w[0, Nb-1])

  for round = 1 step 1 to Nr–1
    SubBytes(state)
    ShiftRows(state)
    MixColumns(state)
    AddRoundKey(state,
          w[round*Nb, (round+1)*Nb-1])
  end for

  SubBytes(state)
  ShiftRows(state)
  AddRoundKey(state,
        w[Nr*Nb, (Nr+1)*Nb-1])

out = state

end
```

Gdje je:
`Nb` – dužina bloka / (4 * 8) (za blok od 128 bita Nb=4)
`Nr` – broj iteracija, zavisan od duzine ključa (ključ 128 bita, Nr = 10, ključ 192, Nr = 12, ključ 256 Nr = 14).

`SubBytes()` transformacija je nelinearna substititucija bajta koja nezavisno radi na svakom bajtu matrice `state` koristeći substitucionu matricu (S-box). Tokom `ShiftRows()`transformacije bajti u tri zadnja reda matrice `state` se ciklički pomjeraju za različit broj bajta. Prvi red se ne pomjera. `MixColumns()`

transformacija množi svaku kolonu matrice state sa fiksnim polinomom

$C(x) = \{03\}*x^3 + \{01\}*x^2 + \{01\}*x + \{02\}$

Tokom `AddRoundKey()` transformacije prošireni ključ za tu iteraciju se dodaje matrici `state` jednostavnom bit XOR operacijom.

Rijndael algoritam izvršava rutinu za proširenje ključa nad zadatim ključem (key). Rutina za generisanje proširenog ključa (w) kreira ukupno Nb (Nr + 1) riječi; algoritam zahtijeva inicijalni skup od Nb riječi i svaka od Nr iteracija zahtijeva skup od Nb riječi. Pseudo kod za ovu rutinu je:

```
KeyExpansion(byte key[4*Nk],
             word w[Nb*(Nr+1)],
             Nk)
begin
  word temp

  i = 0

  while (i < Nk)
      w[i] = word(key[4*i],
                  key[4*i+1],
                  key[4*i+2],
                  key[4*i+3])
      i = i+1
  end while

  i = Nk

  while (i < Nb * (Nr+1)]
      temp = w[i-1]
      if (i mod Nk = 0)
        temp =
            SubWord(RotWord(temp))
            xor Rcon[i/Nk]
      else if (Nk > 6 and
               i mod Nk = 4)
        temp = SubWord(temp)
      end if
      w[i] = w[i-Nk] xor temp
      i = i + 1
  end while

end
```

Iz pseudo koda se vidi da da se prvih Nk riječi proširenog ključa populiše sa izvornim ključem. Svaka slijedeća riječ jednaka je XOR prethodne riječi i riječi Nk pozicija ranije. Za riječi koje se nalaze na poziciji koja je umnožak Nk prvo se obavlja transformacija koja se sastoji od cikličkog pomjeranja bajta u riječi (`RotWord()`) i primjene substitucione matrice na svaki od bajta u riječi, a nakon toga slijedi XOR sa iteracionom konstantom `Rcon`.

Dešifriranje se obavlja inverziom transformacija koje su se koristile pri šifriranju i njihovim primjenjivanjem obrnutim redom. Pseudo kod slijedi:

```
InvCipher( byte in[4*Nb],
           byte out[4*Nb],
           word w[Nb*(Nr+1)])
begin
  byte state[4,Nb]

  state = in

  AddRoundKey(state,
              w[Nr*Nb, (Nr+1)*Nb-1])

  for round = Nr-1 step -1 downto 1
      InvShiftRows(state)
      InvSubBytes(state)
      AddRoundKey(state,
          w[round*Nb, (round+1)*Nb-1])
      InvMixColumns(state)
  end for

  InvShiftRows(state)
  InvSubBytes(state)+
  AddRoundKey(state, w[0, Nb-1])

  out = state
end
```

Transformacija `InvShiftRows()` je inverzna u odnosu na `ShiftRows()`. Bajti u posljednja tri reda matrice `state` se ciklički pomjeraju za različit broj bajta. Prvi red se ne pomjera. `InvSubBytes()` je inverzna transformaciji substitucije bajtova u kojoj se inverzna substitucione matrice primjenjuje na svaki bajt matrice `state`. `InvMixColumns()` je inverzna transformacija u odnosu na `MixColumns()`. `AddRoundKey()` koja je opisana pri opisu šifriranja je svoja vlastita inverzija pošto uključuje samo XOR operaciju.

Snaga ovog algoritma leži u činjenici da je vrlo teško dobiti međurezultat u nekoj iteraciji n iz iteracije n+1 bez ključa za tu iteraciju. U svakoj iteraciji se obavljaju operacije ne-linearizacije (stara se da ne bude nikakve linerne korelacije između ulaza i izlaza u iteraciju), linearnog miješanja (garantuje veliku difuziju tokom više iteracija) i dodavanja ključa (XOR). Ove tri operacije obezbjeđuju otpornost na poznate tipove napada: linerana i diferencijala kriptoanaliza, poznati (known-key) i povezani (related-key) ključ napad, Square napad, interpolacioni napad, slabi ključevi. Rijndael se pokazao kao K-siguran što znači da ne postoji nikakav način otkrivanja ključa osim pretraživanja svih kombinacija, da nema poznatih simetričnih osobina u mapiranju iteracija, da nema slabih ključeva, da nema vezanih ključeva koji bi imali veliki broj zajedničkih proširenih ključeva.

5. UPOREDBA ALGORITAMA

Tokom procedure izbora AES-a, a i po njenom završetku, svi ponuđeni algoritmi, a pogotovo finalisti, su bili

temeljito testirani i poređeni. Opšti zaključak je bio da su svi finalisti izvrsni, a razlika u procjeni najboljeg algoritma najviše je bilo zbog razlika u stepenu važnosti koji se daje kriterijima kao što su sigurnost, brzina, memorijski prostor i fleksibilnost.

Pošto je prvi kriteriji za izbor AES bila sigurnost svi finalisti više nego zadovoljavaju ovaj kriteriji. Pošto je većina algoritama relativno nova nisu mogli biti podvrgnuti pravom testu realnih situacija, ali po sadašnjim znanjima iz oblasti kriptografije može se uspostaviti poredak po kome je najsigurniji Serpent, a zatim ga slijede jednako plasirani MARS i Twofish, pri čemu plasman Rijndael-a zavisi od dužine ključa, ali za osnovnu dužinu od 128 bita, on nešto zaostaje, dok je RC6 na posljednjem mjestu [6],[5].

Rezultati testova programskih implementacija pokazali su sličan poredak za ANSI C i Java implementacije. Rezultati ANSI C analize su pokazali određene varijacije u zavisnosti od korištene hardverske platforme (PC, SUN, SGI), ali se u prosjeku može reći da su se RC6 i Rijndael pokazali kao najbrži, MARS nešto sporijim, dok su Twofish, a pogotovo Serpent zaostajali na svim platformama [12]. Java implementacije su se pokazale dva do tri puta sporijm od optimiziranih C implementacija[13]. Java implementacije su takođe pokazale da stariji algoritmi DES, a Triple DES još i više, daleko zaostaju za AES finalistima. IDEA je približne brzine kao najsporiji finalista Serpent [13]. Određene razlike u rezultatima pojavile su se i u zavisnosti od korištene verzije Java Development Kit (JDK 1.1.6 ili JDK 1.3), ali su se opet na prvom mjestu smjenjivali RC6 i Rijndael, nakon čega su slijedili MARS, Twofish, pa Serpent [14].

Testovi hardverskih implemenentacija na rekonigurabilnom hardveru pokazali su različite rezultate od softverskih testova, zapravo čak pokazali da nema korelacije između hardverskih i softverskih performansi, pa se je Serpent, najsporiji algoritam u softveru, pokazao najbržim. Twofish i RC6 se mogu implementirati kompaktno na maloj površini uz osrednju brzinu u odnosu na ostale. Serpent i Rijndael garantuju vrlo veliku brzinu, ali po cijeni reltivno velike površine u odnosu na prethodna dva algoritma. MARS se plasirao na posljednje mjesto kao najsporiji algoritam koji zahtijeva relativno veliku površinu za implementaciju. Iteresantana je činjenica iako svih pet AES finalista ima veću brzinu izvršavanja na FPGA od Triple DES, komparativan odnos brzine i površine sa Triple DES ima samo Twofish, dok ostali algoritmi zaostaju [15]. Rezultati su takođe pokazali superiornost hardverskih u odnosu na programske implementacije po brzini i to sa faktorom od 4 – 20 [16].

Implementacija na pametnim karticama, zbog njihovog ograničenog memorijskog prostora ima posebne zahtjeve. Na manjim pametnim karticama od 256 bajta koje su trenutno najraširenije smisla ima implementirati samo Rijndael, Serpent i Twofish [17]. Na kvalitetnijim pametnim karticama Rijndael se pokazao najboljim. Slijede ga RC6 i Twofish, dok MARS i Serpent na ovim karticama zaostaju. Stari algoritmi, DES i Triple DES, se zbog malih memorijskih potreba još uvijek mogu natjecati sa AES finalistima na pametnim karticama[18].

Zaključak koji se može izvući iz rezultata testiranja je da je Rijndael dobar izbor visoko plasiran u svim kategorijama. Važna informacija je da za neke posebne namjene i implementacije postoje adekvatniji algoritimi , ta da u svakom konkretnom slučaju izbora algoritma treba dobro razmotriti sve potrebe i vrijednost kriterija. Cilj ove uporedbe i bio je da pruži podatke i ukaže na literaturu koja bi mogla pomoći pri izboru.

6. ZAKLJUČAK

Prikazan je Napredni enkripcijski standard (AES) sa algortimom Rijndael i detaljnim karakteristikama. Predstavljna su ostala četiri finalista izbora za AES (MARS, RC6, Serpent i Twofish). Dati su uporedni rezultati različitih testova ovih algoritama i algoritma koji su od ranije u upotrebi. Na osnovu izloženog može se reći da su svi finalisti izvrsnih karakteristika i da mogu biti upotrebljeni u aplikacijama u kojima njihove najbolje strane dolaze do izražaja. Sa druge strane izabrani AES bi na osnovu istorijskih pokazatelja trebao biti industrijski *de facto* standard za narednu deceniju. Za očekivati je da Triple DES kao standard u odlasku jedno vrijeme koegzistira sa AES-om kao novim standardom, ali da polako bude potiskivan u novim aplikacijama.

Važna karakteristika AES-a je njegova javna dostupnost bez ikakvih licenci i izvoznih ograničenja. Ovo ga čini pogodnim za implementaciju u novim aplikacijama. U trenutnom stanju razvoja BiH telekomunikacija kada ubrzano uvodimo nove tehnologije, pogodan je trenutak da se AES, kao i ostali predstavljeni algoritmi, ozbiljno razmotre. U novim aplikacijama elektronskog poslovanja kao što je Internet bankarstvo, koje se trenutno uvode, kao i drugim primjenama šifriranja podataka, korisno bi bilo gledati u budućnost i odmah koristiti nove algoritme. Pri usvajanju novih standarda treba uzeti u obzir iskustva stečena tokom petogodišnjeg svjetskog konkursa i procesa izbora AES. Cilj ovog rada je da bude jedan korak u tom pravcu.

LITERATURA

[1] Federal Information Processing Standards - Publication 197

[2] Federal Information Processing Standards - Publication 46 – 3

[3] National Institute of Standards and Technology - ”Request For Candidate Algorithm Nominations For The Advanced Encryption Standard (AES) ”, 12. Sep. 1997

[4] James Nechvatal, Elaine Barker, Donna Dodson, Morris Dworkin, James Foti, Edward Roback - “Status Report On The First Round Of The Development Of The Advanced Encryption Standard”, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, 9. Avg. 1999.

[5] James Nechvatal, Elaine Barker, Lawrence Bassham, William Burr, Morris Dworkin, James Foti, Edward Roback - “Report on the Development of the Advanced Encryption Standard (AES)”, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, 2. Okt. 2000.

[6] Don Coppersmith, Rosario Gennaro, Shai Halevi, Charanjit Jutla, Stephen M. Matyas Jr., Mohammad Peyravian, David Safford, Nevenko Zunic - “IBM Comments”, Third AES Conference, 13 Apr. 2000

[7] Ronald L. Rivest, M.J.B. Robshaw, Yiqun Lisa Lin - “RC6 as AES”, Third AES Conference, 13 Apr. 2000

[8] Joan Daemen, Vincent Rijmen - “Rijndael for AES”, Third AES Conference, 13 Apr. 2000

[9] Ross Anderson, Eli Biham and Lars Knudsen - “The Case for Serpent”, Third AES Conference, 24 Mar. 2000

[10] Bruce Schneier, John Kelsey y Doug Whiting, David Wagner, Niels Ferguson - “Comments on Twofish as an AES Candidate”, Third AES Conference, 24 Mar. 2000

[11] Joan Daemen, Vincent Rijmen - “The Rijndael Block Cipher”, AES proposal

[12] Lawrence E. Bassham III - “Efficiency Testing of ANSI C Implementations of Round 2 Candidate Algorithms for the Advanced Encryption Standard”, Third AES Conference, 27. Apr. 2000.

[13] Andreas Sterbenz, Peter Lipp - “Performance of the AES Candidate Algorithms in Java”, Institute for Applied Information Processing and Communications Graz, University of Technology, Third AES Conference, 24 Mar. 2000

[14] Jim Dray - “NIST Performance Analysis of the Final Round Java™ AES Candidates ”, Computer Security Division, The National Institute of standards and Technology, University of Technology, Third AES Conference, 15 Mar. 2000

[15] Kris Gaj and Pawel Chodowiec - “Comparison of the hardware performance of the AES candidates using reconfigurable hardware”, George Mason University, Third AES Conference, 15 Mar. 2000

[16] Andreas Dandalis, Viktor K. Prasanna, and Jose D. P. Rolim- “ A Comparative Study of Performance of AES Final Candidates Using FPGAs ”, Department of Electrical Engineering-Systems, University of Southern California, Third AES Conference, 15 Mar. 2000

[17] Bruce Schneier, Doug Whiting - “ A Performance Comparison of Five AES Finalist ”, Third AES Conference, 15 Mar. 2000

[18] Fumihiko Sano, Masanobu Koike, Shinichi Kawamura, Masue Shiba- “Performance Evaluation of AES Finalists on the High-End Smart Card”, Third AES Conference, 15 Mar. 2000