ROMÂNIA

(19) OFICIUL DE STAT
PENTRU INVENŢII ŞI MĂRCI
Bucureşti

(11) **RO 132127 A2**
(51) **Int.Cl.**
H04L 12/26 (2006.01)

(12) # CERERE DE BREVET DE INVENŢIE

(21) Nr. cerere: **a 2016 00116**

(22) Data de depozit: **15/02/2016**

(41) Data publicării cererii:
30/08/2017 BOPI nr. **8/2017**

(71) Solicitant:
• **IXIA**, *A CALIFORNIA CORPORATION, 26601 WEST AGOURA ROAD, CALABASAS, CALIFORNIA, US*

(72) Inventatori:
• **NISTOR MARIUS PAVEL**, *STR.MITROPOLITUL VARLAAM NR.88, AP.4, SECTOR 1, BUCUREŞTI, B, RO;*
• **CONSTANTINESCU MIHAIL FLORIN**, *STR.ION TUCULESCU NR.36, BL.21A, AP.11, SECTOR 3, BUCUREŞTI, B, RO*

(74) Mandatar:
RATZA ŞI RATZA SRL, *B-DUL A.I. CUZA, NR. 52-54, SECTOR 1, BUCUREŞTI*

(54) # METODE, SISTEM ŞI SUPORT INFORMATIC PENTRU REZILIENŢA CONEXIUNII SISTEMULUI DE TESTARE

(57) **Rezumat:**

Invenţia se referă la un sistem, o metodă şi un suport informatic pentru testarea rezistenţei conexiunii unui sistem. Sistemul de testare a unui dispozitiv de reţea (**DUT**), conform invenţiei, cuprinde: un dispozitiv de testare a echipamentelor de reţea, ce include cel puţin un procesor; un client şi un server, implementaţi de către dispozitivul de testare a echipamentelor de reţea, şi configuraţi să stabilească o conexiune printr-un protocol de control al transmisiei (**TCP**) între client şi server, şi prin intermediul dispozitivului de reţea (**DUT**), în care clientul şi serverul sunt implementaţi pe unul sau mai multe procesoare; un controler de testare implementat de către dispozitivul de testare a echipamentelor de reţea, şi configurat să execute un script de testare pentru a testa dispozitivul de reţea (**DUT**) printr-un schimb de secvenţe de pachete de date prin conexiunea (**TCP**) dintre client şi server; şi un sistem de reziliență, ce include o porţiune pe partea de client şi o porţiune pe partea de server, implementate de către dispozitivul de testare a echipamentelor de reţea, şi configurat să stocheze, pentru fiecare pachet de date schimbat între client şi server prin conexiune (**TCP**), un identificator al secvenţei de client pentru pachetul de date, şi un identificator al secvenţei de server pentru pachetul de date şi, ca răspuns la detectarea unei erori pe conexiune (**TCP**), să sincronizeze clientul şi serverul pentru un pachet de date anterior, schimbat înainte de eroare, folosind identificatorii secvenţei de client şi identificatorii secvenţei de server, şi să reia scriptul de testare la un pachet de date ce urmează după ultimul pachet de date schimbat înainte de eroare, în secvenţa de pachete de date.

Revendicări: 21
Figuri: 6

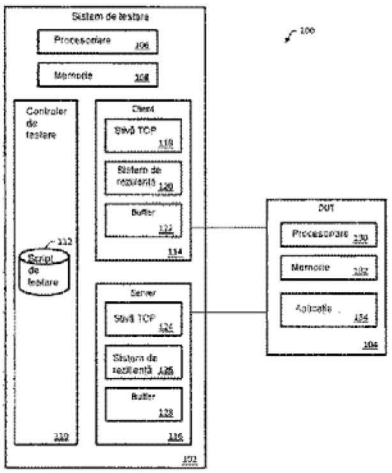

Fig. 1

RO 132127 A2

82

METODE, SISTEME ȘI SUPORT INFORMATIC PENTRU REZILIENȚA CONEXIUNII SISTEMULUI DE TESTARE

Domeniul tehnic

Prezenta invenție se referă în general la sisteme de testare pentru comunicații. Mai exact, prezenta invenție se referă la metode, sisteme și suport informatic pentru reziliența conexiunii sistemului de testare.

Stadiul anterior al tehnicii

Sistemele de testare rețea pot să măsoare și să testeze diverse aspecte ale rețelelor de comunicații de date, cum ar fi performanța rețelei și situația modului de lucru. Sistemele de testare de rețea pot fi folosite pentru a detecta și rezolva problemele de rețea, îmbunătățind performanța rețelei și experiența utilizatorului. Unele sisteme de testare de rețea funcționează prin executarea script-urilor de testare pentru a transmite informații unei rețele de comunicații de date la un punct final și pentru a primi informații la un alt punct final. Datele primite pot fi comparate cu datele transmise pentru a determina unele aspecte ale performanței rețelei, cum ar fi dacă un anumit dispozitiv de rețea funcționează în conformitate cu specificațiile de performanță ale dispozitivului de rețea.

În unele sisteme de testare de rețea, atunci când nu este posibilă o conexiune la rețea în timpul unui script de testare, acesta din urmă se termină cu o eroare de comunicație în rețea. Acest lucru poate fi neplăcut atunci când se dorește testarea anumitor dispozitive, de exemplu, în cazul unui producător de dispozitive de rețea care se așteaptă ca un dispozitiv de rețea să piardă conexiunile atunci când se execută teste de performanță și dorește ca testul să continue în ciuda conexiuni pierdute pentru a observa performanța pe termen lung a dispozitivului. În aceste cazuri, script-ul de testare trebuie să fie re-executat de la început. Având în vedere aceste dificultăți, sunt necesare metode, sisteme și suport informatic pentru reziliența conexiunii sistemului de testare.

Expunerea pe scurt a invenţiei

Invenţia prezentată în această lucrare se referă la metode, sisteme şi suport informatic pentru rezilienţa conexiunii sistemului de testare. În unele exemple de realizare, o metodă de testarea unui dispozitiv de reţea supus testării (DUT) include schimbul de secvenţe de pachete de date peste conexiunea protocolului de control al transmisiei de date (TCP), între un client şi un server şi prin intermediul dispozitivului DUT de reţea. Metoda include stocarea identificatorilor de secvenţă a clientului şi a identificatorilor de secvenţă a serverului. De asemenea, metoda include sincronizarea, ca răspuns la detectarea unei erori pe conexiunea TCP, a clientului şi serverului pentru cel puţin un pachet de date schimbat înainte de eroare folosind identificatorii de secvenţă client şi identificatorii de secvenţă server şi rezumarea unui script de testare la un pachet de date următor, după ce pachetul de date s-a schimbat înaintea erorii in secvenţa de pachete de date.

Prezenta invenţie poate fi implementată în hardware, software, firmware, sau combinaţii de hardware, software şi/sau firmware. In unele exemple de realizare, prezenta invenţie poate fi implementată cu ajutorul unui suport informatic non-tranzitoriu ce stochează instrucţiuni executabile de calculator, care atunci când sunt executate de către unul sau mai multe procesoare ale unui calculator acestea fac ca acel calculator să execute operaţii. Suportul informatic adecvat pentru implementarea invenţiei prezentat în această lucrare include un suport informatic non-tranzitoriu, cum ar fi dispozitivele de memorie disc, dispozitivele de memorie cip, dispozitivele logice programabile, memoria cu acces aleator (RAM), memoria de citire (ROM), memoria optică de citire/scriere, memoria de stocare, memoria magnetică de citire/scriere, memoria flash şi circuitele integrate specifice aplicaţiei. În plus, un suport informatic care implementează prezenta invenţie poate fi localizat pe un singur dispozitiv sau pe o platformă de calcul sau poate fi distribuit pe mai multe dispozitive sau platforme de calcul.

Descrierea pe scurt a desenelor

Figura 1 este o schemă bloc a unui exemplu de mediu pentru comunicaţii care include un dispozitiv de testare cu echipament de reţea configurat pentru a testa un dispozitiv de reţea supus testării (DUT);

Figura 2 este o organigramă de mesaje a unui exemplu de schimb de mesaje, care ilustrează reziliența conexiunii sistemului de testare;

Figura 3 este o organigramă a unui exemplu de metodă pentru primirea pachetelor de date utilizând sistemul de reziliență;

Figura 4 este o organigramă a unui exemplu de metodă pentru transmiterea pachetelor de date utilizând sistemul de reziliență;

Figura 5 este o organigramă a unui exemplu de metodă pentru inițierea unei conexiuni folosind sistemul de reziliență;

Figura 6 este o organigramă a unui exemplu de metodă pentru a accepta o conexiune utilizând sistemul de reziliență.

Descriere detaliată

Figura 1 este o schemă bloc a unui exemplu de mediu pentru comunicații **100** care include un dispozitiv de testare cu echipament de rețea **102** configurat pentru a testa un dispozitiv de rețea supus testării (DUT) **104**. Dispozitivul de testare cu echipament de rețea **102** poate fi definit ca unul sau mai multe dispozitive fizice care transmit pachete de date de testare la DUT **104** și care monitorizează răspunsul lui DUT **104**.

Dispozitivul de testare cu echipament de rețea **102** include unul sau mai multe procesoare **106** și memoria **108**. Memoria **108** poate stoca instrucțiuni executabile pentru procesoarele **106** care, atunci când sunt executate de către aceste procesoare, instrucțiunile pot determina procesoarele **106** să efectueze operațiuni pentru testarea DUT **104**. Instrucțiunile pot include software-ul care este încărcat în memorie cu acces aleator (RAM) și executat de către procesoarele **106**.

Dispozitivul de testare cu echipament de rețea **102** include un controler de testare **110** implementat utilizând procesoarele **106** și memoria **108**, pentru executarea unuia sau mai multor script-uri de testare selectate dintr-o bază de date de script-uri de testare **112**. Un script de testare specifică o secvență de pachete de date care urmează să fie schimbate cu o rețea de comunicații de date. Un pachet de date poate fi, de exemplu, un pachet de date prin Internet Protocol (IP) sau un număr corespunzător de pachete IP sau alte pachete. Un script de testare poate specifica diverse alte date pentru punerea în aplicare a unui test, de exemplu, faptul că un controler de testare **110** poate

folosi condițiile de eroare pentru a determina daca testul s-a încheiat cu succes sau nu.

Dispozitivul de testare cu echipament de rețea **102** include un client **114** și un server **116** care sunt implementate folosind procesoarele **106** și memoria **108**. Clientul **114** și serverul **116** pot fi implementate ca, de exemplu, două procese software separate executate pe un sistem din același calculator, sau ca două unități hardware separate de conexiuni de date separate sub controlul controlerului de testare **110**.Controlerul de testare **110** utilizează clientul **114** și serverul **116** pentru a face schimb de secvențe de pachete de date prin intermediul unei conexiuni de rețea pentru comunicații de date prin intermediul rețelei **104** DUT.

Clientul **114** include o stivă **118** cu un protocol de control al transmisiei de date (TCP), un sistem de reziliență client **120**, și o memorie tampon **122**. Stiva TCP **118** este configurată pentru a stabili o conexiune TCP cu serverul **116** și pentru a face schimb de mesaje TCP cu serverul **116**. Sistemul de reziliență client **120** este configurat pentru a implementa o reziliență a conexiunii sistemului de testare prin stocarea identificatorilor de secvență client. Memoria tampon **122** este configurată pentru a stoca mesaje trimise și primite și identificatorii de secvență client pentru mesajele trimise și primite.

Serverul **116** include o stivă TCP **124**, un sistem de reziliență server **126** și o memorie tampon **128**. Stiva TCP **124** este configurată pentru a stabili o conexiune TCP cu clientul **114** și pentru a face schimb de mesaje TCP cu acesta. Sistemul de reziliență server **126** este configurat pentru a implementa rezilienta conexiunii sistemului de testare prin stocarea identificatorilor de secvență server. Memoria tampon **128** este configurată pentru a stoca mesaje trimise și primite și identificatori de secvență client pentru mesajele trimise și primite.

DUT **104** de rețea poate include unul sau mai multe procesoare **130** și memoria **132**. Memoria **132** poate stoca instrucțiuni executabile pentru procesoarele **130** care, atunci când sunt executate de către aceste procesoare, determină ca procesoarele **130** să efectueze operațiuni pentru executarea unei aplicații **134**. Instrucțiunile pot include software-ul care este încărcat în memoria cu acces aleator (RAM) și executat de către procesoarele **130**.

DUT **104** de rețea este configurată pentru a primi, procesa și transmite pachete de date. DUT de rețea **104** poate fi, de exemplu, un router wireless, un firewall sau o altă versiune a adresei de rețea (NAT). DUT de rețea **104** poate include hardware specializat pentru efectuarea de operațiuni de rețea, de exemplu, sub controlul aplicației **134**. DUT de rețea **104** ajunge la diferite stări ale dispozitivului, în funcție de mesajele care sunt prelucrate de către DUT de rețea **104**. De exemplu, DUT de rețea **104** poate stoca un anumit număr de adrese de rețea într-un tabel de adrese de rețea sau să fie stocate sub o anumită sarcină în raport cu o sarcină specificată de DUT de rețea **104**.

În timpul funcționării, controlerul de testare **110** execută un script de testare la DUT **104** de rețea de testare schimbând secvențe de pachete de date printr-o conexiune TCP între clientul **114** și serverul **116**. Script-ul de testare este configurat, de exemplu, la valoarea unui administrator de sistem sau unui designer al dispozitivului de rețea ce selectează mesajele corespunzătoare în script-ul de testare, astfel încât schimbul de secvențe de pachete de date de rețea să permită ca DUT de rețea **104** să ajungă la diferite stări ale dispozitivului. Controlerul de testare **110** monitorizează răspunsul DUT **104** de rețea la diferite stări ale dispozitivului pentru a stabili dacă testul a fost terminat cu succes sau nu.

În timp ce clientul **114** și serverul **116** fac schimb de secvențe de pachete de date, sistemul de reziliență client **120** înmagazinează identificatorii de secvențe care referă la numerele de secvență transmise (SSN) pentru fiecare pachet de date, transmise de către clientul **114** și la numerele de secvență primite (RSN) pentru fiecare pachet de date primit de către clientul **114**. Sistemul de reziliență client **120** incrementează SSN-urile și RSN-urile, astfel încât fiecare pachet de date transmis are un SSN unic de client și fiecare pachet de date primit are un RSN unic de client. RSN-urile și SSN-urile pot fi numere de strat a aplicației ce diferă de numerele de secvență TCP.

Similar, sistemul de reziliență a serverului **126** stochează identificatorii de secvență de server, inclusiv SSN-uri pentru fiecare pachet de date trimis de serverul **116** și RSN-uri pentru fiecare pachet de date primit de serverul **116**. Sistemul de reziliență a serverului **126** incrementează SSN-urile și RSN-urile, astfel încât fiecare pachet de date trimis are un SSN unic de server și fiecare pachet de date primit are un RSN unic de server. În general, sistemele de reziliență de clientul **120** și de serveri **126** vor

inițializa primul SSN al clientului **114** și primul RSN al serverului **116** la o valoare inițială identică, ca de exemplu, 0 sau 1, iar primul RSN al clientului **114** și primul SSN al serverului **116** se vor inițializa la aceeași valoare inițială. Astfel, stocarea identificatorilor de secvență nu include înlocuirea identificatorilor de secvență pe conexiunea TCP și nu include alterarea sarcinilor de pachete de date.

În timp ce execută un script de testare, dispozitivul de testare a echipamentului de rețea **102** poate detecta o eroare la conexiunea TCP, de exemplu, prin primirea unui mesaj de eroare de la DUT de rețea **104**. De exemplu, DUT **104** poate să nu reușească să funcționeze conform specificațiilor date din cauza unei sarcini de pe DUT de rețea **104** care depășește o sarcină specificată. Clientul **114**, serverul **116** sau chiar ambele pot detecta eroarea pe conexiunea TCP.

Ca răspuns la detectarea erorii pe conexiunea TCP, sistemul de reziliență a clientului **120** și sistemul de reziliență a serverului **126** sincronizează clientul **114** și serverul **116** la un ultim pachet de date schimbat înainte de eroare folosind identificatorii de secvență de client și identificatori de secvență de server. Controlerul de testare **110** execută, pe scurt, script-ul de testare curent la următorul pachet de date după ce ultimul pachet de date a fost trimis înainte de eroare în secvența de pachete de date.

De exemplu, sincronizarea clientului **114** și serverului **116** poate include schimbul de cel puțin un prim RSN de la clientul **114** la serverul **116** și un al doilea RSN de la serverul **116** la clientul **114**. Fiecare client **114** și server **116** transmite RSN-ul la ultimul pachet de date primit cu succes. Serverul **116** utilizează primul RSN și cel puțin un SSN pentru a determina ultimul pachet de date schimbat înainte de eroare, de exemplu, dacă primul SSN este SSN-ul de server corespunzător primului RSN de la clientul **114**. Clientul **114** folosește al doilea RSN și cel puțin un al doilea SSN pentru a determina ultimul pachet de date schimbat înainte de eroare, de exemplu, în cazul în care al doilea SSN este SSN-ul de client care corespunde celui de al doilea RSN de la serverul **116**.

În acest mod, clientul **114** și serverul **116** pot interschimba orice pachete de date care au fost retrase ca urmare a erorii. Mai mult decât atât, este posibil ca sincronizând clientul **114** și serverul **116** să nu se ajungă la perturbarea DUT **104** de rețea din starea dispozitivului, astfel încât DUT de rețea **104** să se afle la momentul erorii. Așa că, menționatul controler de testare **110** poate relua script-ul de testare, fără a fi nevoie

să reia fiecare procedură de la început, iar controlerul de testare **110** poate testa DUT de rețea **104**, cu condițiile care au ajutat la recepționarea erorii.

În unele exemple, sistemul de reziliență client **120** și sistemul de reziliență server **126** corelează clientul **114** și serverul **116** pentru a identifica care script de testare a fost executat la momentul erorii, astfel încât acel script de testare să poate fi reluat la următorul pachet de date. Corelarea clientului **114** și serverului **116** poate include transmiterea unui identificator de corelare care identifică script-ul de testare executabil. De exemplu, corelarea clientului **114** și serverului **116** poate include stabilirea faptului că un port de destinație al conexiunii TCP este partajat între mai multe conexiuni și, ca răspuns, înlocuirea unui identificator de corelare de la clientul **114** la serverul **116** identifică script-ul de testare executabil. Identificatorii de corelare pot fi stocați cu script-uri de testare în baza de date de script-uri de testare **112** și pot fi furnizați de clientul **114** și serverul **116**, dacă este cazul.

În timpul funcționării, sistemele de reziliență server și client **120** și **126** pot recupera conexiuni TCP în timpul rulării, menținând în același timp starea de testare și pe clientul **114** și pe serverul **116**. Sistemele de reziliență client și server **120** și **126** nu trebuie să modifice sarcinile utile ale traficului de testare, astfel că nu este nevoie de timp suplimentar pentru a reseta un test și pentru a reproduce problemele care produc erorile, acest lucru putând fi greu sau imposibil de realizat, în unele cazuri, chiar și prin simpla reluare a testului. Sistemele de reziliență client și server **120** și **126** pot menține condițiile de lucru în timp ce se recuperează conexiunile TCP și permit ca monitorizarea DUT **104** de rețea să aibă un comportament specific condițiilor nefuncționale, iar sistemele de reziliență client și server **120** și **126** să poată oferi funcționalitate, în cazul ignorării eșecurilor rețelei tranzitorii, astfel că încă să poată determina rezultate de testare exacte pentru testele de performanță de lungă durată.

Figura 2 este o organigramă de mesaje **200** a unui exemplu de schimb de mesaje care ilustrează reziliența conexiunii sistemului de testare. În acest exemplu, clientul **114** și serverul **116** au stabilit o conexiune TCP prin intermediul unei conexiuni de rețea de comunicații de date prin intermediul DUT **104** astfel că se execută un script de testare.

Într-un prim schimb de mesaje **202**, clientul **114** transmite un mesaj la serverul **116**. Clientul **114** stochează un SSN de 1 pentru acel mesaj și serverul **116** stochează un RSN de 1 pentru acel mesaj. Serverul **116** transmite apoi un mesaj la clientul **114**.

Clientul **114** stochează un RSN de 1 pentru acel mesaj și serverul **116** stochează un SSN de 1 pentru acel mesaj. Serverul **116** transmite un alt mesaj, iar clientul **114** stochează un RSN de 2 pentru acel mesaj și serverul **116** stochează un SSN de 2 pentru acel mesaj. Apoi, clientul **114** transmite un alt mesaj, iar clientul **114** stochează un SSN de 2 pentru acel mesaj și serverul **116** stochează un RSN de 2 pentru acel mesaj.

Apoi DUT de rețea **104** transmite un mesaj de eroare la serverul **116**. În acest timp, clientul **114** nu primește încă un mesaj de eroare și se continuă cu scriptul de testare. Clientul **114** transmite un prim mesaj și înregistrează un SSN de 3 pentru mesaj, dar serverul **116** nu primește mesajul, deoarece DUT de rețea **104** se află într-o stare de eroare. Clientul **114** transmite un alt mesaj și înregistrează un SSN de 4 pentru acel mesaj, dar serverul **116**, de asemenea, nu primește acel mesaj. DUT de rețea **104** transmite apoi un mesaj de eroare către clientul **114**.

Ca răspuns la detectarea erorii, clientul **114** și serverul **116** încep sincronizarea pentru a determina ultimul mesaj trimis înainte de eroare și corelarea pentru a se asigura că clientul **114** și serverul **116** sunt pe același script de testare. Într-un al doilea schimb de mesaje **204** pentru corelare, clientul **114** transmite un mesaj de conexiune TCP urmat de un mesaj cu un identificator de corelare. Serverul **116** răspunde cu un mesaj de corelare deja construit.

Într-un al treilea schimb de mesaje **206** ce ajută la sincronizare, clientul **114** transmite un RSN de sincronizare de 2 la serverul **116**. Clientul 114 transmite un RSN de 2, deoarece 2 este cel mai mare RSN stocat înainte de eroare. Serverul **116** transmite un RSN de sincronizare 2 la clientul **114**. Serverul **116** transmite un RSN de 2, deoarece 2 este cel mai mare RSN stocat înainte de eroare.

Clientul **114** determină, apoi, faptul că mesajul trimis cu un SSN de 2 a fost ultimul mesaj trimis înainte de eroare și deci retrimite mesajul cu un SSN de 3. Serverul **116** primește mesajul și reia stocarea de identificatorilor de secvențe, prin stocarea unui RSN de 3 pentru mesaj. Serverul **116** stochează un RSN de 3 prin incrementarea RSN-ului de sincronizare, pe care îl trimite la clientul **114**. Clientul **114**, apoi, retrimite de asemenea și mesajul cu un SSN de 4, iar serverul **116** stochează un RSN de 4 pentru acel mesaj. La acest moment, sincronizarea este completă.

Într-un al patrulea schimb de mesaje **208**, clientul **114** și serverul **116** reiau script-ul de testare ca și în cazul în care eroarea nu ar fi apărut. Clientul **114** trimite un mesaj și înregistrează un SSN de 5 pentru acel mesaj, iar serverul **116** recepționează mesajul și înregistrează un RSN de 5 pentru mesaj. Clientul **114** trimite un alt mesaj și înregistrează un SSN de 5 pentru mesaj, iar serverul **116** recepționează mesajul și înregistrează un RSN de 6 pentru mesaj.

Figura 3 este o organigramă a unui exemplu a metodei **302** pentru primirea pachetelor de date folosind sistemul de reziliență. Metoda **302** este realizată de către oricare dintre clientul **114** sau serverul **116**. În scopul ilustrării, metoda **302** va fi descrisă cu referire la un sistem care execută metoda **302** cu o pereche. În cazul în care sistemul este clientul **114**, atunci perechea este serverul **116**; în cazul în care sistemul este serverul **116**, atunci perechea este clientul **114**.

Sistemul execută o acțiune de recepționarea TCP pentru a transfera k bytes (**304**). În cazul în care transferul se face cu succes (**306**), sistemul incrementează un RSN curent (**308**). Sistemul stochează pachetul primit asociat cu RSN-ul incrementat (**310**).

În cazul în care transferul nu se face cu succes (**306**), atunci sistemul execută procedura acceptată TCP (**314**), în cazul în care sistemul este serverul (**312**), iar sistemul execută procedura de conectare TCP (**316**), în cazul în care sistemul este client (**312**). Apoi, sistemul trimite un mesaj SYNC la recenzie cu RSN-ul ultimului pachet primit (**318**), iar sistemul recepționează un mesaj SYNC la pereche cu RSN-ul ultimului pachet primit (**320**).

Sistemul compară RSN-ul primit de la pereche cu ultimul SSN a pachetelor din lista sistemului de pachete trimise (**322**). Sistemul retrimite pachetele din lista sistemului de pachete trimise având un SSN mai mare decât RSN-ul primit de la pereche (**324**). Sistemul primește, din nou, pachetul de k octeți (**326**). În acest mod, sistemul determină care pachete nu au fost primite de la pereche și trimite aceste pachete din nou, astfel încât sistemul să fie sincronizat cu perechea, la sfârșitul fazei de sincronizare.

Figura 4 este o organigramă a unui exemplu a metodei **402** pentru transmiterea pachetelor de date folosind sistemul de reziliență. Metoda **402** este realizată de către oricare dintre clientul **114** sau serverul **116**. În scopul ilustrării, metoda **402** va fi

descrisă cu referire la un sistem care execută metoda **402** cu o pereche. În cazul în care sistemul este clientul **114**, atunci perechea este serverul **116**; în cazul în care sistemul este serverul **116**, atunci perechea este clientul **114**.

Sistemul execută o acțiune de trimitere TCP pentru a transfera k bytes (**404**). În cazul în care transferul se face cu succes (**406**), sistemul incrementează SSN-ul curent (**408**). Sistemul stochează pachetul trimis asociat cu SSN-ul incrementat (**410**).

În cazul în care transferul nu se face cu succes (**406**), atunci sistemul execută procedura acceptată a TCP-ului (**414**), în cazul în care sistemul este server (**412**) și sistemul execută procedura de conectare a TCP-ului (**416**), în cazul în care sistemul este client (**412**). Apoi, sistemul trimite un mesaj SYNC la pereche cu RSN-ul ultimului pachet primit (**418**) și sistemul recepționează un mesaj SYNC de la pereche cu RSN-ul ultimului pachet primit (**420**).

Sistemul compară RSN-ul primit de la pereche cu ultimul SSN a pachetelor din lista sistemului de pachete trimise (**422**). Sistemul retrimite pachetele din lista sistemului de pachete trimise cu un SSN mai mare decât RSN-ul primit de la pereche (**424**). În acest mod, sistemul determină care pachetele nu au fost primite de la pereche și trimite aceste pachete din nou, astfel încât sistemul să si fie sincronizat cu perechea, la sfârșitul fazei de sincronizare.

Figura 5 este o organigramă a unui exemplu a metodei **502** pentru inițierea unei conexiuni cu ajutorul sistemului de reziliență. Metoda **502** este realizată de către clientul **114**. Clientul **114** execută operația de conectare a TCP-ului pe stivă (**504**). În cazul în care conexiunea se realizează cu succes (**506**), clientul **114** trimite un identificator de corelare la serverul **116**, astfel încât serverul **116** să poată identifica în mod corect script-ul de testare care urmează să fie executat (**508**). În cazul în care conexiunea nu este realizată cu succes (**506**), operația de conectare poate fi repetată până când acesta este realizată cu succes sau se ajunge la o altă condiție de sfârșit corespunzătoare, de exemplu, după ce se ajunge la un anumit număr de încercări.

Figura 6 este o organigramă a unui exemplu a metodei **602** pentru a accepta o conexiune utilizând sistemul de reziliență. Metoda **602** este efectuată de către serverul **116**. Serverul **116** execută operația de acceptare a TCP-ului pe stivă (**604**). În cazul în care conexiunea se realizează cu succes (**606**), serverul **116** primește un identificator

de corelare pentru a fi utilizată în identificarea script-ului de testare care urmează să fie executat (**608**). Serverul **116** nu îi este necesar identificatorului de corelare în situațiile în care script-ul poate fi identificat folosind alte informații, de exemplu, portul de destinație, dar în situațiile în care portul de destinație este partajat între mai multe conexiuni, serverul **116** poate utiliza identificatorul de corelare.

În cazul în care identificatorul de corelare corespunde cu script-ul de executare curent (**610**), de exemplu, script-ul care rulează în prezent pe un anumit fir de comunicație pentru serverul **116**, atunci serverul **116** poate relua pur și simplu script-ul. În caz contrar, serverul **116** adaugă perechea (identificatorul de corelare, stiva) într-o amestecătură globală (**612**), iar serverul **116** așteaptă ca un alt fir de comunicație să adauge identificatorul de corelare și stiva pentru script-ul de executare (**614**). În acest mod, serverul **116** poate evita afectarea straturilor superioare de software sau logica aplicației în timp ce se recuperează dintr-o eroare.

Prin urmare, în timp ce metodele, sistemele și suportul informatic pentru reziliența conexiunii sistemului de testare au fost descrise aici cu referire la exemplele de realizare specifice, la caracteristicile și exemplele de realizare ilustrative, se va aprecia faptul că utilitatea invenției nu este astfel limitată, ci se extinde și cuprinde numeroase alte variații, modificări și aplicații alternative concrete, așa cum se vor sugera persoanelor de specialitate în domeniul invenției prezentate, pe baza descrierii din acest document.

Diferite combinații și subcombinații ale structurilor și caracteristicilor descrise aici sunt complementate și vor fi evidente pentru o persoană de specialitate având cunoștințele necesare pentru înțelegerea acestei descrieri. Oricare dintre diferitele caracteristici și elemente așa cum sunt descrise aici pot fi combinate cu unul sau mai multe alte caracteristici și elemente dacă nu se indică contrariul în prezentul document. În mod corespunzător, invenția revendicată în continuare intenționează să fie construită și interpretată pe larg, ca incluzând toate aceste variații, modificări și aplicații alternative concrete, în domeniul său, inclusiv echivalente ale revendicărilor.

Se înțelege că diferitele detalii ale invenției descrise pot fi modificate fără a ne îndepărta de la scopul subiectului dezvăluit în document. În plus, descrierea de mai sus are scopul de a ilustra, și nu de a limita.

REVENDICĂRI

1. Sistem de testare a unui dispozitiv de rețea (DUT) supus testării, care conține:
   - un dispozitiv de testare echipamente de rețea care include cel puțin un procesor;
   - un client și un server implementați de către dispozitivul de testare echipament de rețea și configurați pentru a stabili o conexiune prin protocolul de control al transmisiei (TCP), între client și server și prin intermediul DUT de rețea, în care clientul și serverul sunt implementați pe unul sau mai multe procesoare;
   - un controler de testare implementat de către dispozitivul de testare echipament de rețea și configurat pentru a executa un script de testare pentru a testa DUT de rețea prin schimbul unei secvențe de pachete de date prin conexiunea TCP între client și server; și
   - un sistem de reziliență care include o porțiune pe partea de client și o porțiune pe partea de server implementate de către dispozitivul de testare echipament de rețea și care este configurat pentru a stoca, pentru fiecare pachet de date schimbate între client și server prin conexiunea TCP, un identificator de secvență de client pentru pachetul de date și un identificator de secvență de server pentru pachetul de date, și, ca răspuns la detectarea unei erori pe conexiunea TCP, pentru a sincroniza clientul și serverul pentru un pachet de date anterior schimbate înainte de eroare folosind identificatorii de secvență de client și identificatorii de secvență de server și pentru a relua script-ul de test la un pachet de date următor ultimului pachet de date schimbate înainte de eroare în secvența de pachete de date.
2. Sistem, conform revendicării 1, **caracterizat prin aceea că** sistemul de reziliență este configurat pentru a stoca un prim număr de secvență transmis (SSN) pentru un prim pachet de date trimis și un al doilea SSN pentru un al doilea pachet de date transmis, trimis primul pachet de date transmis, și pentru a determina al doilea SSN prin incrementarea primului SSN.
3. Sistem conform revendicării 2, **caracterizat prin aceea că** sistemul de reziliență este configurat pentru a stoca un prim număr de secvență (RSN) recepționat la primul pachet de date recepționat și un al doilea RSN pentru un

al doilea pachet de date recepționat după primul pachet de date primit, și pentru a determina al doilea RSN prin incrementarea primului RSN.

4. Sistem, conform revendicării 3, **caracterizat prin aceea că** operațiile cuprind inițializarea primului SSN a clientului și primului RSN a serverului la o aceiași valoare inițială și inițializarea primului RSN a clientului și primului SSN a serverului la aceeași inițială valoare.
5. Sistem, conform revendicării 1, **caracterizat prin aceea că** sistemul de reziliență este configurat pentru a stoca un număr unic de secvență (RSN) pentru fiecare dintre o multitudine de pachete de date recepționate și un număr unic de secvență (SSN) pentru fiecare dintre o multitudine pachete de date trimise, și în care sincronizarea clientului și serverului cuprinde schimbul de cel puțin un prim RSN de la client la server și un al doilea RSN de la server la client.
6. Sistem, conform revendicării 5, **caracterizat prin aceea că** primul RSN indică un ultim pachet de date primite de către client, iar al doilea RSN indică un ultim pachet de date primit de către server, și în care serverul este configurat să utilizeze primul RSN și cel puțin o primul SSN pentru a determina ultimul pachet de date schimbat înainte de eroare, și în care clientul este configurat să utilizeze doilea RSN și cel puțin un al doilea SSN pentru a determina ultimul pachet de date schimbat înainte de eroare.
7. Sistem, conform revendicării 1, **caracterizat prin aceea că** scriptul de test este configurat astfel încât schimbarea secvenței de pachete de date determină DUT de rețea să ajungă la o stare a dispozitivului de înainte de sau din același timp cu eroarea pe conexiunea TCP, și în care sincronizarea clientului și serverului la ultimul pachet de date schimbat înainte de eroare nu perturbă DUT de rețea de la starea dispozitivului special înainte de reluarea script-ului de testare la următorul pachet de date.
8. Sistem, conform revendicării 1, **caracterizat prin aceea că** identificatorul de secvență client și identificatorul de secvență server conțin numere de secvență pentru strat de aplicații distincte de numerele de secvență TCP.
9. Sistem, conform revendicării 1, **caracterizat prin aceea că** sistemul de reziliență este configurat pentru a corela, ca răspuns la detectarea erorii pe conexiunea TCP, clientul și serverul pentru a identifica script-ul de testare, pentru reluarea script-ului de testare la următorul pachet de date.

69

10. Sistem, conform revendicării 9, **caracterizat prin aceea că** menționata corelare a clientului și serverul cuprinde determinarea că un port de destinație a conexiunii TCP este partajat între o multitudine de conexiuni și schimbul unui identificator de corelare de la client la server pentru a identifica script-ul de test.
11. Metodă pentru testarea unui dispozitiv de rețea supus testării (DUT), care constă în:
    - executarea unui script de testare pentru a testa DUT de rețea prin schimbul unei secvențe de pachete de date într-o conexiune sub protocolul de control al transmisiei (TCP) între un client și un server și prin DUT de rețea, în care clientul și serverul sunt implementate pe unul sau mai multe procesoare;
    - stocarea, pentru fiecare pachet de date schimbate între client și server prin conexiunea TCP, a unui identificator de secvență de client pentru pachetul de date și a unui identificator de secvență de server pentru pachetul de date;
    - și, ca răspuns la detectarea unei erori pe conexiunea TCP, sincronizarea clientului și serverului pentru un ultim pachet de date schimbat înainte de eroare folosind identificatorii de secvență de client și identificatorii de secvență de server și reluarea script-ului de testare la un pachet de date următor după ultimul pachet de date schimbat înainte de eroare în secvența de pachete de date.
12. Metodă, conform revendicării 11, **caracterizată prin aceea că**, mai constă în stocarea unui prim număr de secvență transmis (SSN) pentru un prim pachet de date trimis și un al doilea SSN pentru un al doilea pachet de date transmis, trimis la primul pachet de date transmis, și determinarea celui al doilea SSN prin incrementarea primului SSN.
13. Metodă conform, revendicării 12, **caracterizată prin aceea că** mai constă în stocarea unui prim număr de secvență (RSN) recepționat la primul pachet de date recepționat și a unui al doilea RSN pentru un al doilea pachet de date recepționat după primul pachet de date primit, și determinarea celui de al doilea RSN prin incrementarea primului RSN.
14. Metodă, conform revendicării 13, **caracterizată prin aceea că** mai constă în inițializarea primului SSN a clientului și a primului RSN a serverului la o aceiași valoare inițială și inițializarea primului RSN a clientului și primului SSN a serverului la aceeași inițială valoare.

15. Metodă, conform revendicării 11, **caracterizată prin aceea că** mai constă în stocarea unui număr unic de secvență (RSN) pentru fiecare dintre o multitudine de pachete de date recepționate și a unui număr unic de secvență (SSN) pentru fiecare dintre o multitudine pachete de date trimise, și în care sincronizarea clientului și serverului cuprinde schimbul de cel puțin un prim RSN de la client la server și un al doilea RSN de la server la client.
16. Metodă conform revendicării 15, **caracterizată prin aceea că** primul RSN indică un ultim pachet de date primite de către client, iar al doilea RSN indică un ultim pachet de date primit de către server, și în care serverul este configurat să utilizeze primul RSN și cel puțin o primul SSN pentru a determina ultimul pachet de date schimbat înainte de eroare, și în care clientul este configurat să utilizeze doilea RSN și cel puțin un al doilea SSN pentru a determina ultimul pachet de date schimbat înainte de eroare.
17. Metodă, conform revendicării 11, **caracterizată prin aceea că** scriptul de test este configurat astfel încât schimbarea secvenței de pachete de date determină DUT de rețea să ajungă la o stare a dispozitivului de înainte de sau din același timp cu eroarea pe conexiunea TCP, și în care sincronizarea clientului și serverului la ultimul pachet de date schimbat înainte de eroare nu perturbă DUT de rețea de la starea dispozitivului special înainte de reluarea script-ului de testare la următorul pachet de date.
18. Metodă, conform revendicării 11, **caracterizată prin aceea că** identificatorul de secvență client și identificatorul de secvență server conțin numere de secvență pentru strat de aplicații distincte de numerele de secvență TCP.
19. Metodă, conform revendicării 11, **caracterizată prin aceea că** mai constă în corelarea, ca răspuns la detectarea erorii pe conexiunea TCP, clientului și serverului pentru a identifica script-ul de testare, pentru reluarea script-ului de testare la următorul pachet de date.
20. Metodă, conform revendicării 19, **caracterizată prin aceea că** menționata corelare a clientului și serverul constă în determinarea că un port de destinație a conexiunii TCP este partajat între o multitudine de conexiuni și schimbul unui identificator de corelare de la client la server pentru a identifica script-ul de test.
21. Unul sau mai multe suporturi informatice non-tranzitorii ce stochează instrucțiuni executabile de calculator, care atunci când sunt executate de către

unul sau mai multe procesoare ale unui calculator facă acel calculator să execute operații care constau în:

- executarea unui script de testare pentru a testa DUT de rețea prin schimbul unei secvențe de pachete de date într-o conexiune sub protocolul de control al transmisiei (TCP) între un client și un server și prin DUT de rețea, în care clientul și serverul sunt implementate pe unul sau mai multe procesoare;
- stocarea, pentru fiecare pachet de date schimbate între client și server prin conexiunea TCP, a unui identificator de secvență de client pentru pachetul de date și a unui identificator de secvență de server pentru pachetul de date;
- și, ca răspuns la detectarea unei erori pe conexiunea TCP, sincronizarea clientului și serverului pentru un ultim pachet de date schimbat înainte de eroare folosind identificatorii de secvență de client și identificatorii de secvență de server și reluarea script-ului de testare la un pachet de date următor după ultimul pachet de date schimbat înainte de eroare în secvența de pachete de date.

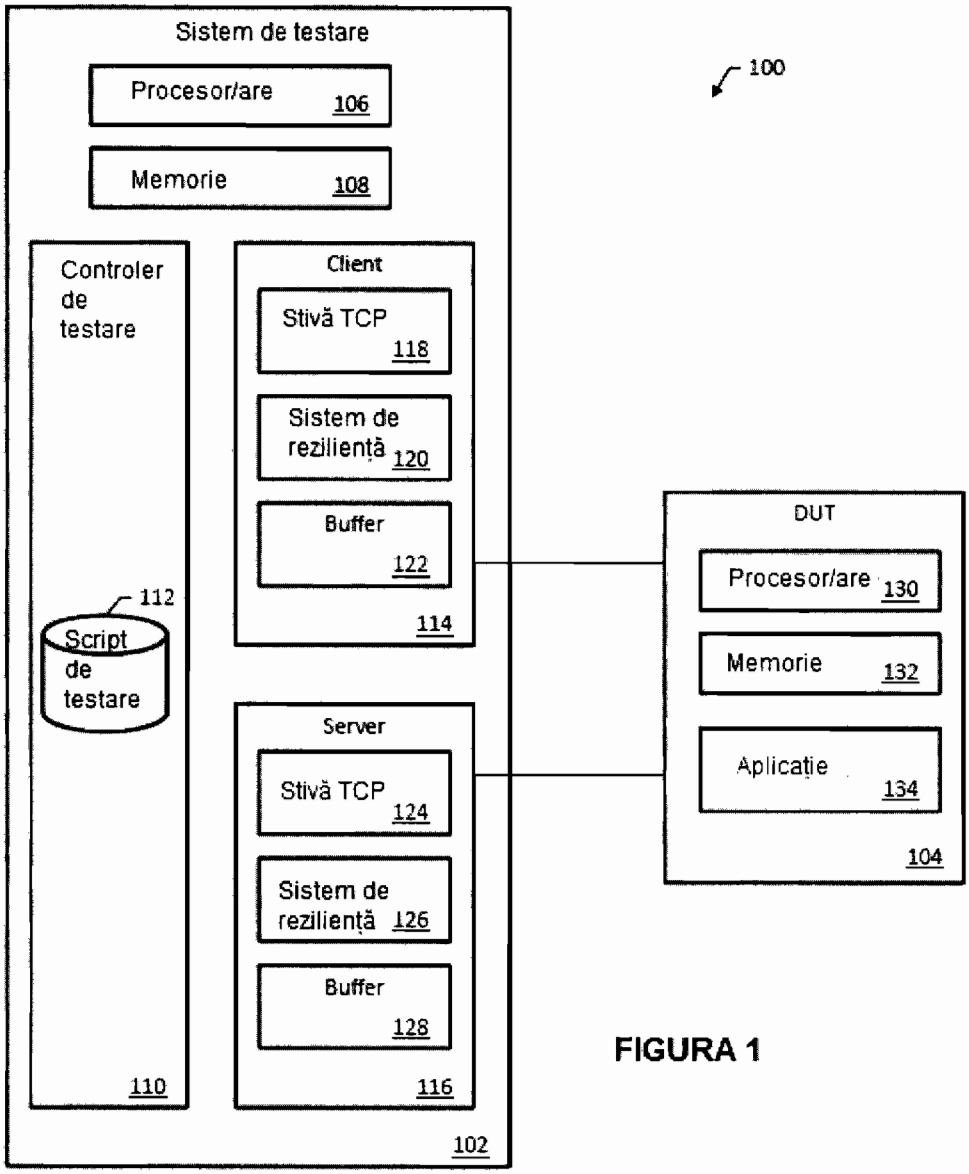
Sistem de testare
Procesor/are 106
Memorie 108
Controler de testare
112
Script de testare
110
Client
Stivă TCP 118
Sistem de reziliență 120
Buffer 122
114
Server
Stivă TCP 124
Sistem de reziliență 126
Buffer 128
116
102
100
DUT
Procesor/are 130
Memorie 132
Aplicație 134
104

FIGURA 1

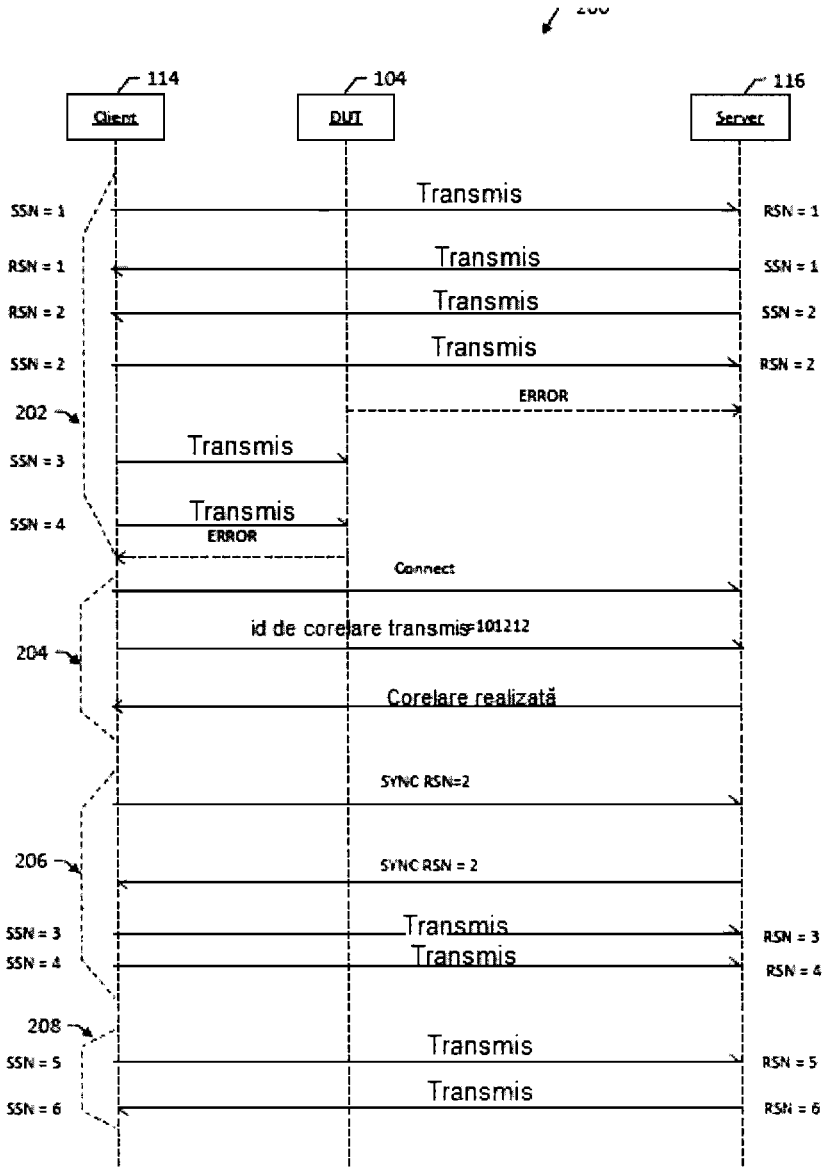
114
Client
104
DUT
116
Server
Transmis
SSN = 1
RSN = 1
Transmis
RSN = 1
SSN = 1
Transmis
RSN = 2
SSN = 2
Transmis
SSN = 2
RSN = 2
ERROR
202
Transmis
SSN = 3
Transmis
SSN = 4
ERROR
Connect
id de corelare transmis=101212
204
Corelare realizată
SYNC RSN=2
206
SYNC RSN = 2
SSN = 3
Transmis
RSN = 3
SSN = 4
Transmis
RSN = 4
208
Transmis
SSN = 5
RSN = 5
Transmis
SSN = 6
RSN = 6

FIGURA 2

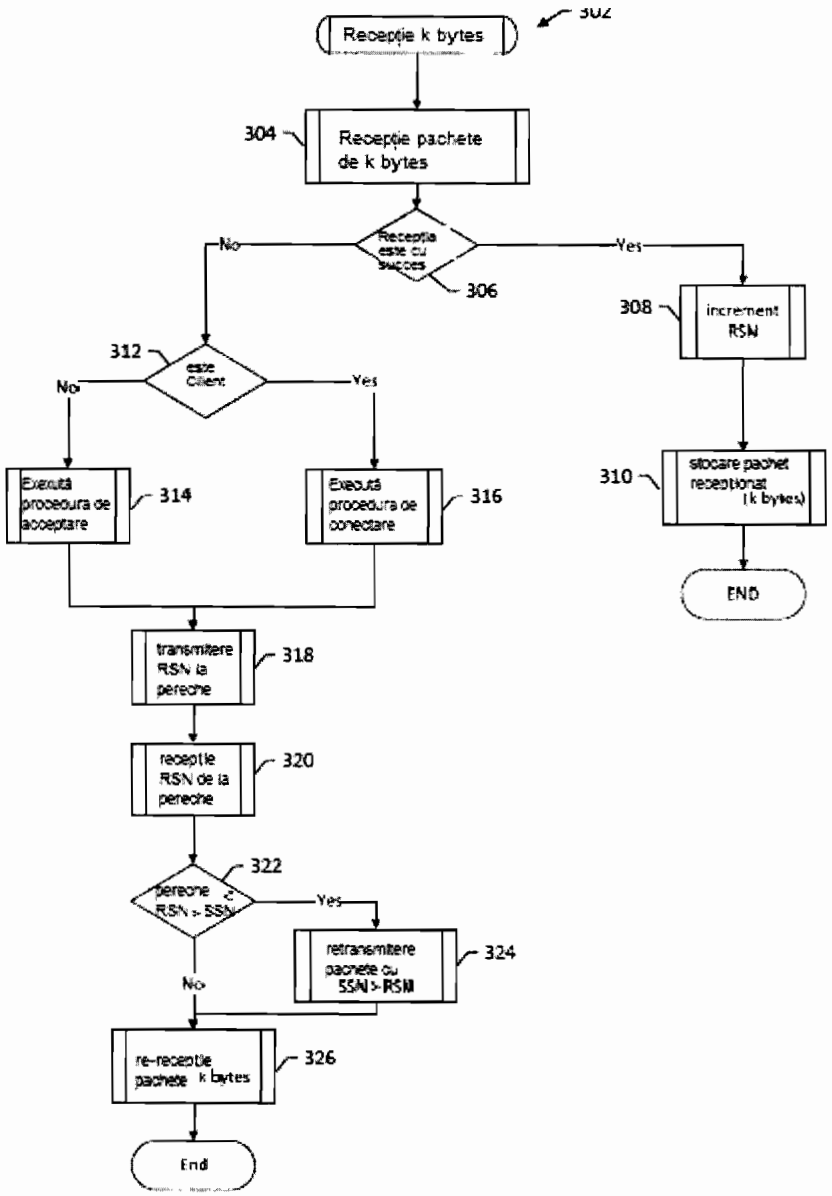
302
Recepţie k bytes
304
Recepţie pachete de k bytes
Recepţia este cu succes
306
No
Yes
308
increment RSN
310
stocare pachet receptionat (k bytes)
END
312
este Client
No
Yes
Execută procedura de acceptare
314
Execută procedura de conectare
316
transmitere RSN la pereche
318
receptie RSN de la pereche
320
322
pereche RSN > SSN
Yes
No
retransmitere pachete cu SSN > RSN
324
re-receptie pachete k bytes
326
End

FIGURA 3

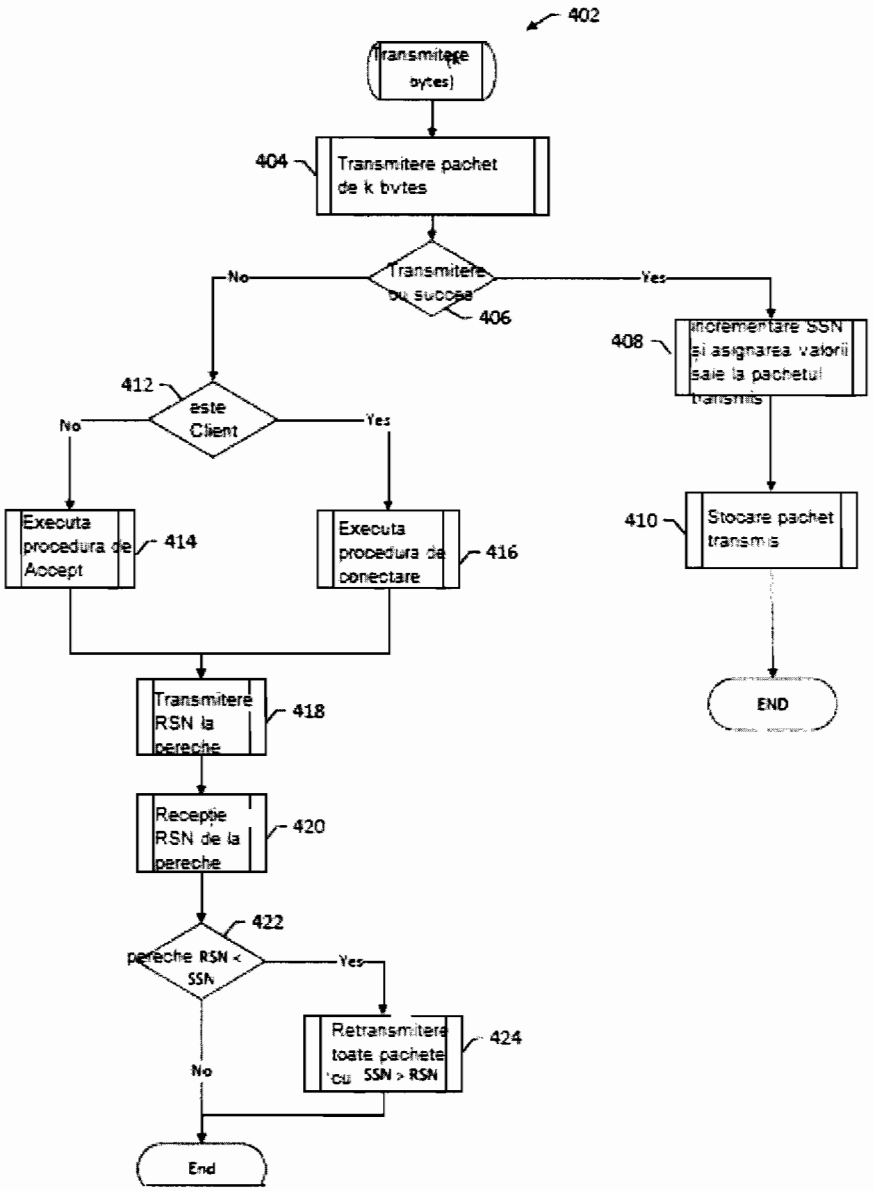
402
Transmitere (k bytes)
404
Transmitere pachet de k bytes
Transmitere cu succes
406
No
Yes
408
incrementare SSN si asignarea valorii sale la pachetul transmis
412
este Client
No
Yes
Executa procedura de Accept
414
Executa procedura de conectare
416
410
Stocare pachet transmis
END
Transmitere RSN la pereche
418
Receptie RSN de la pereche
420
422
pereche RSN < SSN
Yes
No
Retransmitere toate pachete cu SSN > RSN
424
End

FIGURA 4

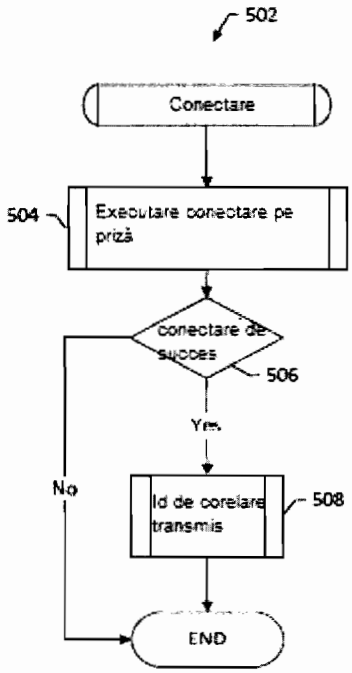
502
Conectare
504
Executare conectare pe priză
conectare de succes
506
Yes
No
Id de corelare transmis
508
END

FIGURA 5

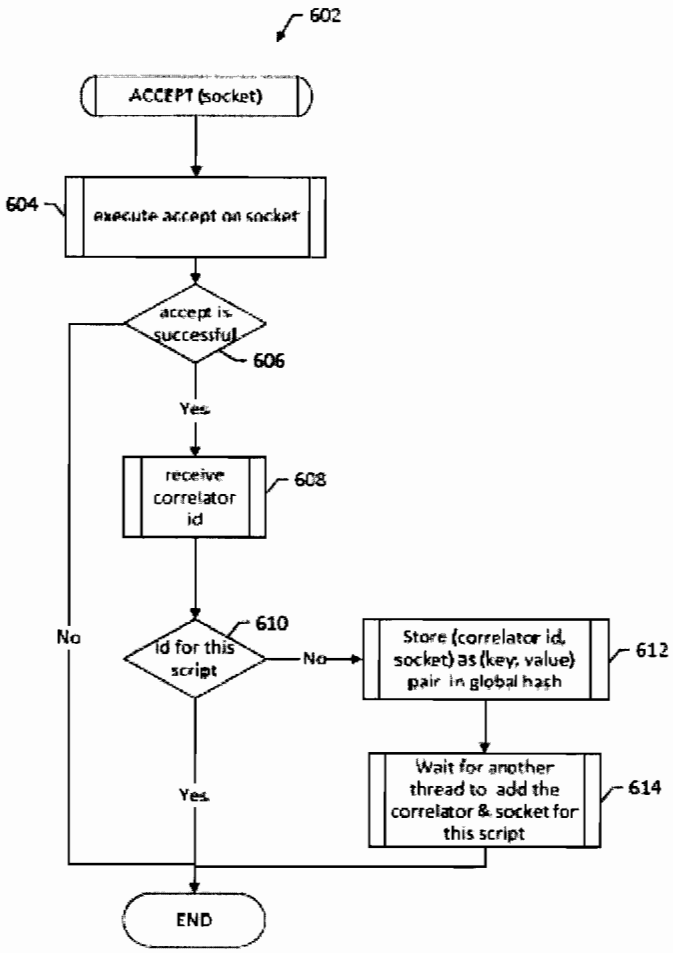
602
ACCEPT (socket)
604
execute accept on socket
accept is successful
606
Yes
receive correlator id
608
610
Id for this script
No
Store (correlator id, socket) as (key, value) pair in global hash
612
Wait for another thread to add the correlator & socket for this script
614
Yes
No
END

FIGURA 6