REVISTA COLOMBIANA DE COMPUTACIÓN
Volumen 11, número 1
Págs. 27 - 40

Anycast Anónimo Basado en Firmas de Grupo

Juan Camilo Corena * Jaime Andrés Posada †
Fecha de Recibido: 11/01/2010 Fecha de Aprobación: 01/03/2010

Resumen

Presentamos en este artículo un esquema criptográfico para la implementación de grupos *Anycast* dentro de un sistema de enrutamiento anónimo con n miembros, que ya cuente con una llave pública por cada uno de ellos y una llave simétrica compartida por cada pareja de los mismos. Nuestro esquema permite la comprobación de la recepción por cualquiera de los destinatarios legítimos de un mensaje y la emisión de recibos firmados digitalmente al emisor sin necesidad de generar llaves de firmado para cada uno de los 2^n posibles grupos *Anycast* en la red. En total se usarán η llaves simétricas para firmas de los recibos y $\frac{n(n-1)}{2}$ llaves simétricas para proteger las comunicaciones entre miembros. Dentro de las principales características de nuestro sistema se encuentran: *i)* El balanceo de cargas en redes anónimas, mientras se protege a los generadores de las peticiones de ataques de *spoofing* por parte de impostores. *ii)* Obligar a los usuarios a utilizar el sistema de balanceo de cargas, al estos no tener acceso a las direcciones de la red, pero si a las llaves de firmado, con lo cual un atacante debe realizar un ataque a escala global de la red para atacar a un grupo *Anycast* específico. Posibles aplicaciones del sistema incluyen tolerancia a fallas en servicios de nombres y replicación de datos.

Palabras clave: *Firmas de grupo, anycast, redes anónimas.*

Abstract

We present in this article a cryptographic scheme to implement anycast groups in an n member anonymous network where each participant owns a public key and there is a symmetric key for every pair of nodes. Our system allows the clients to receive a digitally signed receipt from any of the legitimate receivers of a message without generating keys for each one of the 2^n possible anycast groups within the network. In total we will use η assymetric keys to sign receipts and $\frac{n(n-1)}{2}$ symmetric keys to protect communications between every two members. The main characteristics of our system are: *i)*Load balancing in anonymous networks along with spoofing protection to clients. *ii)*The load balancing system is mandatory and cannot be bypassed since clients do not have the network level addresses of a given anycast group, that is why in order to attack a particualr anycast group a network scale attack must be performed. Possible aplications for our system include fault tolerance, name services and data replication.

Keywords: *Group signatrues, anycast, anonymous networks.*

* Politécnico Grancolombiano, Calle 57 #3-00 este Bloque I, Bogotá-Colombia, investigacion@juancamilocorena.com
† Politécnico Grancolombiano, Calle 57 #3-00 este Bloque I, Bogotá-Colombia japosada@poligran.edu.co

1. Introducción

Presentamos en este artículo una aplicación novedosa de las firmas de grupo [2] para implementar *Anycast* resistente a *spoofing* en sistemas de enrutamiento anónimo sin revelar la identidad del miembro del grupo *Anycast* que atiende la solicitud realizada por el usuario. Nuestro sistema requiere de un menor número de llaves criptográficas que el esquema trivial en el cual hay una llave por cada grupo Anycast posible. Por ejemplo, dada una red de tamaño 10, el esquema trivial necesita en el peor de los casos 1024 llaves para los posibles grupos *anycast*, mientras que nuestro esquema requiere de 10 llaves. Si se desean comunicaciones seguras en ambos esquemas, se requieren 45 llaves adicionales.

El esquema consta de dos fases. En la primera, el nodo que envía el mensaje debe plantear un reto para comprobar que solo los destinatarios legítimos del mensaje sean capaces de leerlo. En la segunda fase, un receptor válido del mensaje debe emitir un recibo de confirmación sin revelar su identidad. Para generar este recibo, dependiendo de las necesidades de comunicación de los nodos, se usan firmas de grupo descritas en [7] para protocolos que requieran estado, mientras que en el caso de protocolos sin estado su usan firmas de anillo descritas originalmente en [11].

Se conoce de otros protocolos para *Anycast* anónimo que emplean *Multicast* como en [6]. Durante la investigación no se encontró un sistema que haga uso de firmas de grupo para la emisión de recibos. Finalmente, antes de presentar el esquema propuesto se describe en detalle el problema a tratar y se explica detalladamente el esquema de firmas de anillo [11]. Lo anterior en vista que nuestro estudio se enfoca en protocolos sin estado al ser estos los más comunes en sistemas que usan Anycast. Se termina la presentación con algunos detalles de implementación y rendimiento.

2. Descripción del Problema

Se cuenta con una red anónima A compuesta por los nodos $\{a_1,a_2,...a_n\}$, un nodo emisor a_e, $1 \leq e \leq n$ que conoce al conjunto $R = \{a_{i1},a_{i2},...a_{ik}\} \subseteq A$ de posibles receptores (todos ellos equivalentes) del mensaje m. El sistema a proponer debe garantizar que algún $a_{ij} \in R$ recibió el mensaje

m, un nodo $a_x \notin R$ no puede conocer a $a_\in$ y por último que no existe $a_x \notin R$, $a_x \neq a_\in$ que conozca a $a_\in$ o los miembros de R a partir de este mensaje. En la figura 1 se ilustra esta situación.

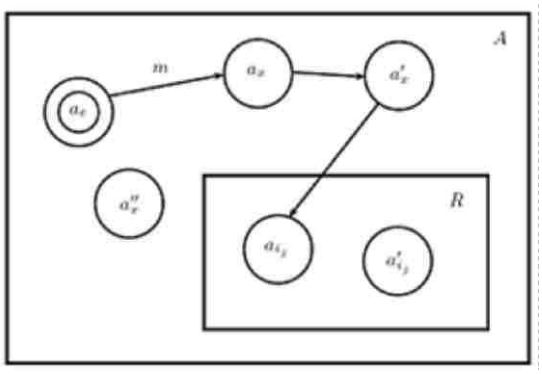

Fig. 1. Ejemplo de envío del mensaje por parte de un nodo

Para satisfacer los anteriores requerimientos se propone un reto criptográfico para la validación de pertenencia al conjunto R que cumpla las siguientes características:

1. Si el nodo a_{ij} soluciona el reto, entonces conoce la identidad de $a_\in$
2. Un $a_x \notin R$ no puede solucionar el reto.

Una vez solucionado el reto por parte de un nodo $a_{ik} \in R$ dicho nodo debe emitir una respuesta con las siguientes condiciones:

1. Solamente miembros de R pueden generar una respuesta con firma digital válida.
2. No es posible para el emisor $a_\in$ comprobar cual de los miembros de R generó la firma.

3 Firmas de Grupo

Los firmas de grupo permiten a un miembro (sin que se revele su identidad a menos que se cumplan ciertas condiciones) firmar a nombre de un grupo. Dicho concepto se le atribuye a Chaum [2]. En particular, describimos a continuación dos tipos de firma de grupo. La primera se conoce como firma de anillo en la que cualquier persona sin necesidad de autorización central, puede generar un grupo y firmar a nombre de este. La identidad del firmante está protegida bajo cualquier condición, mientras que en el segundo tipo de firma, conocidas como linkable

democratic group signatures, cualquier miembro del grupo puede identificar al firmante, pero miembros externos no lo pueden hacer.

3.1 Linkable Democratic Group Signatures

En [7] se construye un esquema para firmas de grupo en el cual la identidad del firmante se protege de miembros no pertenecientes al grupo, pero cualquier miembro del grupo puede conocer la identidad del firmante. Como propiedad adicional, miembros externos al grupo pueden saber si dos firmas fueron realizadas por el mismo firmante pero sin conocer la identidad de este. Esta última propiedad es de vital importancia para la implementación de protocolos con estado en nuestra red anónima.

La construcción se basa en la dificultad de calcular logaritmos discretos y en pruebas de cero conocimiento no interactivas. Durante la etapa de inicialización se generan la llaves públicas y privadas a usar y pseudónimos que etiquetan a cada miembro del grupo; estos pseudónimos constan de dos números $(g^k, (g^x)^k)$, donde x es la llave privada del nodo al que le corresponde el pseudónimo, g^x es la llave pública, k es un número secreto conocido solo por los miembros del grupo y g es un generador del subgrupo cíclico donde se llevan a cabo los cálculos.

La firma consta de un componente aleatorio, un pseudónimo, una función de una sola vía y una prueba de cero conocimiento no interactiva sobre la igualdad de dos logaritmos discretos propuesta en [5]. Para verificar la validez de una firma se comprueba que el pseudónimo pertenece al grupo y luego se aplica una comprobación de la prueba de cero conocimiento. Para lograr identificar al que firmó, los nodos del grupo pueden resolver el logaritmo discreto $(g^x)^k$ que aparece en la firma al calcular $1/k$ en el subgrupo en el que se realizan los cálculos, lo que revela la identidad del firmante. Usando este mismo pseudónimo, un nodo fuera del grupo puede identificar que dos firmas corresponden al mismo firmante comparando los pseudónimos asociados a esta.

3.2 Firmas de Anillo

En [11] se propone un esquema de firma que describimos a continuación: Supongamos que existen r posibles firmantes $A_1, A_2, \ldots A_i$ que conforman lo que se denomina *anillo*, y que el individuo $A_1, A_2, \ldots A_i$ del anillo. Suponemos que todos los miembros del anillo usan RSA (ver [10]) como su esquema de cifrado. Cada miembro A_i del anillo tiene entonces una llave pública $Pi = (n_i, e_i)$ que especifica una permutación de una sola vía f_i definida en Z_{ni} de la siguiente forma

$$f_i(x) = x^{ci} \pmod{n_i}$$

donde se supone que solamente el individuo A_i conoce la forma de calcular eficientemente la permutación inversa F_i^{-1} dada por:

$$f_i^{-1}(y) = y^{di} \pmod{n^i}$$

donde

$$d_i = e_i^{-1} \pmod{\varnothing(n_i)}$$

3.2.1 Extensión a un Dominio Común

Las permutaciones f_i deben ser primero extendidas a un dominio común $\{0, 1\}^b$ *donde* 2^b es mayor que todos los módulos n_i Para ello se define la extensión g de una permutación de una sola vía f de la siguiente forma:

Para cada entrada m de b bits, se encuentran números no negativos q y r de forma tal que $m = qn + r$ y $O \leq r < n$. En este caso.

$$g(m) = \begin{cases} qn + f(r) \text{ si } (q+1)n < 2^b \\ m \quad \text{en caso contrario} \end{cases}$$

3.2.2 Cifrado Simétrico

Se supone la existencia de un algoritmo de cifrado simétrico *SE* tal que para cada llave k de longitud l la función SE_k es una permutación del conjunto $\{0, 1\}^b$

3.2.3 Función de Hash

Se supone la existencia de una función de hash de una sola vía h resistente a colisiones que asigna a entradas arbitrarias cadenas de longitud l, que son usadas como llaves para *SE*.

3.2.4 Función Combinadora

Se construye una familia de funciones combinadoras

$$C_{k,v}(y_1, y_2, \ldots, y_r)$$

que toman como entrada una llave k, un valor inicial v, y valores arbitrarios $y_1, y_2, \ldots, y_r$ *en* $\{0, 1\}^b$. Cada una de estas funciones usa a Se_k para calcularse, y produce como resultado un valor $z \in \{0, 1\}^b$ tal que, al fijar k y v, se tienen las siguientes propiedades:

1. Permutación en cada entrada: Para cada $s \in \{1, 2, \ldots, r\}$, y para cualquier valor fijo de las otras entradas y_i, $i \neq s$, la función $C_{k,v}$ es uno a uno de y_s en z.

2. Solucionable eficientemente para una entrada: Para cada $s \in \{1, 2, ...,r\}$, dado $z \in \{0, 1\}^b$, y valores para todas las entradas y_i, $i \neq s$, es posible encontrar y_s tal que $C_{k,v}(y_1, y_2, ..., y_r) = z$.

3. Impracticabilidad de solución para todas las entradas sin las trampas: Dados *k, v* y *z,* no es práctico para un adversario resolver la ecuación

$$C_{k,v}(g_1(x_1), g_2(x_2), ..., g_r(x_r)) = z$$

para $(x_1, x_2, ..., x_r)$ si el adversario no puede invertir las trampas $(g_1, g_2, ..., g_r)$.

Una función con dichas propiedades es la siguiente:

$$C_{k,v}(y_1, y_2, ..., y_r) =$$
$$SE_K(y_r \oplus SE_K(y_{r-1} \oplus SE_K(...SE_K(y_1 \oplus v) ...)))$$

donde $\oplus$ *denota XOR sobre el conjunto* $\{0, 1\}^b$. Con los anteriores ingredientes se procede a continuación a describir el esquema de *firma de anillo.*

3.2.5 Descripción del Esquema de Firma

Fase I: Generación de la firma.
Dado un mensaje *m* a ser firmado, una secuencia de llaves públicas $(p_1, p_2, ..., p_r)$ y una llave S_s, el firmante A_s calcula la la firma de anillo como sigue:

1. Determinar la llave simétrica: El firmante calcula la llave simétrica *k* del mensaje *m* usando la función hash: $k = h(m)$.

2. Escoger un valor de pegamento: El firmante escoge al azar un valor $v \in \{0, 1\}^b$

3. Escoger valores de x_i: El firmante escoge al azar valores de $x_i \in \{0, 1\}^b$, $i \neq s$ para los otros miembros del anillo, y calcula $y_i = g_i(x_i)$, $i \neq s$.

4. Solucionar para y_s: El firmante soluciona para y_s la ecuación de anillo

$$C_{k,v}(y_1, y_2, ..., y_r) = v$$

5. Invertir y_s: El firmante usa su llave privada para calcular

$$x_s = g_s^{-1}(y_s)$$

6. Emitir la firma de anillo: El firmante emite la firma

$$(p_1, p_2, \ldots, p_r; v; x_1, x_2, \ldots, x_r)$$

Fase II: Verificación de la firma.
Un verificador puede establecer la validez de una firma

$$(p_1, p_2, \ldots, p_r; v; x_1, x_2, \ldots, x_r)$$

sobre el mensaje *m* de la siguiente forma:

1. Aplicar las trampas: El verificador calcula $y_i = g_i(x_i)$.

2. Calcular *k*: El verificador calcula la llave simétrica k del mensaje m usando la función hash: $k = h(m)$.

3. Verificar la ecuación de anillo: Por último el verificador comprueba que los y_i satisfacen la ecuación de anillo

$$C_{k,v}(y_1, y_2, \ldots, y_r) = v$$

Si dicha ecuación se satisface, el verificador acepta la firma. En caso contrario la rechaza.

4 Enrutamiento anónimo

Los sistemas de enrutamiento anónimo moderno se basan en el diseño Mix-Net de Chaum [1]. Se trata de un sistema de correo electrónico que busca mantener tanto la privacidad de la información entre las partes involucradas, así como la identidad de las partes en si mismas. Requisito que se ha hecho indispensable para todos los sistemas que lo sucedieron. La idea central de las *Mix-nets* consiste en generar un camino aleatorio mediante el uso de llaves públicas para hacer imposible el rastreo del correo electrónico. A continuación se presenta un sistema basado en esta idea: *Crowds*. El motivo de su inclusión en este artículo obedece a que nuestra propuesta está estrechamente ligada a él. Consultar [9] para mas detalles al respecto.

4.1 Crowds

Crowds es un sistema que busca garantizar el anonimato de las partes en las comunicaciones al incorporar a los miembros de la red en una

multitud (De ahí el nombre *crowds* en inglés). Los nodos del sistema reciben el nombre de *jondos* y trabajan de la siguiente manera:

Cuando un *jondo* desea enviar una petición a un servidor, elige un *jondo* al azar y envía la petición. El *jondo* receptor con una probabilidad *Pf* enviará de nuevo el mensaje a un tercer *jondo*, y con probabilidad (1 - *Pf*) lo enviará al servidor. La comunicación entre cada uno de los nodos se realiza de manera cifrada y la respuesta se envía en texto plano.

Para medir el anonimato de nuestro esquema, recurrimos al estudio de Pfitzmann and Köhntopp [8], donde anonimato se describe como el estado de no ser identificable dentro de un conjunto de sujetos, conocido como el conjunto de anonimato. En vista que el sistema propuesto funcionará de manera análoga a un *crowd* el siguiente análisis presente en [9] puede hacerse.

Decimos que un emisor es probablemente inocente si desde el punto de vista del atacante, la probabilidad de ser el verdadero emisor del mensaje es menor a la probabilidad de no serlo. Se tiene entonces (ver [9] Teorema 5.2) que si PF > ½ y adicionalmente

$$n \geq \frac{Pf}{Pf - \frac{1}{2}} (c + 1)$$

entonces se garantiza la inocencia probable del emisor del mensaje ante *c* atacantes, donde *n* el número de elementos del crowd y *Pf* la probabilidad de reenvío por parte de un *jondo* a otro.

En nuestro caso los miembros de *A* ue no están en *R* se comportarán como los miembros de un *crowd* donde

$$Pf = 1 - \frac{|R|}{|A|-2}$$

es la probabilidad de que el mensaje no llegue al conjunto *R* y por tanto la probabilidad de reenvío del *crowd*. Por ejemplo, si $|A| = 100$ y $|R| = 20$, entonces $n = 80$, $Pf = \frac{39}{49}$, y por tanto para garantizar la inocencia probable de un emisor, debe haber a lo más $c = 28$ nodos comprometidos. Esto ofrece seguridad razonable ante los ataques que pretendan identificar al verdadero emisor del mensaje.

5 Esquema Propuesto

Un esquema tradicional de firmas digitales necesitaría 2^n llaves para la

totalidad de posibles subconjuntos $R \subseteq A$ lo que hace irrealizable el sistema tanto en términos de generación de llaves como en distribución de las mismas inclusive para conjuntos pequeños de nodos. Para evitar la generación de llaves por cada uno de los subconjuntos $R \subseteq A$ describimos el siguiente reto criptográfico y luego el esquema basado en firmas de grupo discutidas en la sección 3.

5. 1 Reto Criptográfico

Sea $k_{i,j}$ una llave simétrica conocida únicamente por a_i y a_j. Sea (P_α , S_α) una pareja de llaves, pública y privada respectivamente y sea σ (S_α , m) una función de firma digital con las siguientes propiedades:

- Una firma generada a partir de S_α y un mensaje m se debe poder verificar usando P_α
- Es computacionalmente imposible generar una firma válida para alguien que no conoce S_α . Es decir, con el poder de computo actual tardaría bastantes años resolver el anterior problema.

Sea *SE* la función de cifrado simétrico. Definimos el siguiente reto criptográfico: dados $\{e, j\} \subset \{1, 2, ..., n\}, e \neq j$ se calcula

$$c_j = SE_{Ke,j} (e, \eta, \tau, m, \sigma(s_c (e, \eta, \tau)))$$

donde η es un *nonce* para agregar entropía al mensaje, τ un *timestamp* para garantizar que el mensaje sea nuevo y evitar ataques de repetición. Este reto se eligió basado en los requisitos planteados en la sección 2 dado que solamente a_j puede descifrar c_j , y es posible verificar la autenticidad de la firma emitida por a_c al usar p_c.

5.2 Procedimiento

A continuación se describen los pasos requeridos para el envío de mensajes en la red *A* donde los pasos que se siguen son los mismos en el caso de protocolos con estado y sin estado a excepción del tipo de firma a usar. La elección de la última depende de la necesidad de comunicarse siempre con el mismo nodo dentro del grupo anycast. En protocolos con estado esto es necesario ya que los otros nodos del grupo no conocen la información necesaria para procesar la petición siguiente, a menos que se les informe mediante un mensaje entre nodos del grupo o que el mensaje contenga la información de las interacciones anteriores. La garantía de que las peticiones están siendo procesadas por el mismo nodo

del grupo se obtiene empleando linkable democratic group signatures, en caso de que esto no sea necesario se pueden usar firmas en anillo.

1. Calcular c_j: a_c conociendo el conjunto R dado por $\{a_{i_1}, a_{i_2}, ..., a_{i_k}\}$ genera c_j para cada $a_{i_j} \in R$.

2. Transmisión de firmas: a_c elige un computador al azar $a_x, x \neq e$ y transmite

$$(c_1, c_2, ..., c_k)$$

3. Verificar pertenencia: Para cada reto c_j y cada $y, 1 \leq y \leq n, a_x$ intenta resolver dicho reto al calcular

$$SE^{-1}_{K_{ys}} (cj) = (e', \eta', \tau, m', \lambda)$$

e intenta verificar la validez de la firma al calcular

$$\sigma^{-1} (P_{e'}, (e', \eta', \tau\ 'm'))$$

con lo cual pueden ocurrir los siguientes casos:
a. Si $\sigma^{-1} (P_{e'},(e', \eta', \tau', m')) = \lambda$ entonces a $a_x \in R$, y por tanto es uno de los destinatarios legítimos del mensaje en cuyo caso debe procesar el mensaje m. Una vez terminado el procesamiento, a_x debe retornar

$$(E_{pe} (\mu , \iota); SE_{\mu O} (\lambda , p, GS))$$

donde μ es una llave simétrica aleatoria usada exclusivamente para la transmisión de la respuesta, ι es un *nonce* para agregar entropía a la llave, p es la respuesta al mensaje m y GS es alguno de los dos tipos de firma de grupo para R del mensaje (λ,p) Los datos firmados por GS garantizan que el nodo que está emitiendo el mensaje conoce λ pero ningún agente externo conoce la identidad de dicho nodo.

b. Si $\sigma^{-1} (P_{e'}, (e', \eta', \tau\ 'm')) \neq \lambda$ entonces $a_w, w \neq x$ y repetir el paso 2.

5.3 Requisitos de una Implementación

En esta sección mostramos un escenario que garantiza la seguridad en la implementación de las dos aplicaciones propuestas inicialmente. Suponemos que todos los nodos cuentan con una dirección a nivel de capa de red d_j que los identifica de manera única y que todos los nodos están conectados entre sí. Adicionalmente, las llaves $k_{i\ j}$ han sido distribuidas y las parejas (P_i, S_i) se encuentran almacenadas en cada uno

de los nodos a_j. Por último, dada d_j no es posible encontrar su llave p_j correspondiente, ni viceversa; esto con motivo de mantener los grupos *Anycast* independientes del direccionamiento de la red.

Un esquema de creación de grupos *Anycast* implica dar a conocer las llaves públicas de los nodos participantes al conjunto de nodos emisores de mensajes. Lo anterior se puede lograr en forma personal entre usuarios y proveedores o mediante un tercero de confianza que certifique que este grupo se encuentra capacitado para operar. En un escenario real los usuarios del sistema tienen fuentes confiables de información como la página web del proveedor del servicio. Algunas de las ventajas que tiene el esquema son:

1. Balanceo de cargas obligatorio: Al definir los grupos que prestan los servicios en términos de las llaves criptográficas, se evita que los nodos empleen las direcciones d_j para enviar mensajes de manera directa, con lo cual se obliga a que los nodos elijan distintos servidores dentro de R como lo corroboran los datos experimentales expuestos en la sección 6.

2. Resistencia al Spoofing: Una vez creado y notificado a los usuarios un grupo *Anycast*, el sistema es resistente a ataques de *spoofing* en varios niveles: como primera medida un nodo por fuera del grupo no puede conocer el contenido de la petición ya que carece de la llave para procesarla, como segunda medida en caso de conocer el contenido de la petición no puede generar una firma válida por lo cual el emisor rechazaría esta respuesta y además tendría conocimiento de que está siendo atacado.

6 Detalles de Implementación y Simulación

Para el modelamiento del rendimiento del algoritmo en términos de la velocidad en que se establecen las conexiones se usará una distribución geométrica, la cual brinda la distribución X del número de experimentos de Bernoulli necesarios para obtener un éxito. Si la probabilidad de éxito es P, entonces la probabilidad de que el

$$P(p,k) = p\ (1 - p)^{k-1}$$

El éxito en el modelo *Anycast* se refiere a que cualquiera de los nodos en R reciba el mensaje. En este caso P (p, k) es precisamente la probabilidad de que después de k pasos un miembro de R ha recibido el mensaje, donde

$$P = \frac{|R|}{|A|-2}$$

Lo anterior ya que un nodo intermedio no puede reenviar el mensaje al nodo anterior ni a el mismo. En el caso del nodo emisor Lo anterior ya que un nodo intermedio no puede reenviar el mensaje al nodo anterior ni a el mismo. En el caso del nodo emisor *e,* se escoge al azar un nodo *g* al cual no se le emitirá el mensaje. Esto con el fin de tener probabilidades iguales durante todo el modelamiento. El valor esperado de la distribución usada es $E(x) = \frac{1}{P}$, el cual representa en nuestro caso el número promedio de conexiones intermedias necesarias para que un nodo de *R* reciba el mensaje emitido por *e.* Se sigue que dicho número promedio de conexiones no depende del tamaño de la red *A.* Dos redes de distinto tamaño pueden tener iguales probabilidades de éxito en la transmisión de un mensaje.

Para la implementación de prueba del esquema se usó jdk 1.6 de Java, como algoritmo simétrico AES en modo CBC con llaves de 128 bits y *padding* PKCS5, como función de hash se usó SHA2, y por último el algoritmo asimétrico usado fué RSA con llaves de 1024 bits.

Para corroborar el anterior modelo teórico, se realizaron 1000 simulaciones por cada tamaño de grupo, dentro de una red con tamaño 100. En el cuadro 1 se detallan los resultados de la simulación. En la tabla superior se incluyen las probabilidades incrementadas cada *20%* y se incluyen el número de saltos tanto esperados como obtenidos experimentalmente. En el cuadro 2 se incluye el tamaño del grupo *Anycast* y la desviación estándar del número de conexiones atendidas por cada uno de los miembros del grupo durante las 1000 simulaciones. Cabe anotar que el resultado experimental de esta simulación se acerca en alto grado con los resultados teóricos, confirmando así la validez del modelo empleado para modelar las velocidades de conexión. Por otro lado las desviaciones estándar de la tabla derecha son bastante pequeñas en relación al tamaño del grupo confirmándose así que el sistema logra balancear las conexiones del sistema.

Probabilidad	Saltos esperados	Saltos obtenidos
0.11	9.091	8.889
0.31	3.226	3.266
0.51	1.961	1.951
0.71	1.408	1.409
0.91	1.099	1.108

Cuadro 1: Resultados experimentales 1

Tamaño R	Desviación
11	6.374
31	6.365
51	3.436
71	3.808
91	3.069

Cuadro 2: Resultados experimentales 2

7 Conclusiones

Se expone un esquema en el que los servicios *Anycast* se pueden desplegar sin incrementar el número de llaves criptográficas que usa la red. Las simulaciones corroboran el rendimiento teórico del sistema contra el real. Dichas simulaciones también confirman que el sistema es capaz de balancear la carga transaccional de un servicio determinado ofrecido por un grupo *Anycast*. Desde el punto de vista de viabilidad computacional, el sistema emplea primitivas criptográficas estándares, como lo son la creación de llaves y el intercambio de estas por parte de los distintos nodos. Solucionar ecuaciones de anillo y logaritmos discretos conociendo los exponentes, no presenta complejidad computacional más elevada que otros protocolos criptográficos usados en la práctica. Por último, intentar resolver el reto criptográfico toma alrededor de 50ms para un grupo de tamaño 10 en un computador de escritorio, con lo cual podemos afirmar que el esquema propuesto es viable computacionalmente.

Referencias

[1] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, February 1981.

[2] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology - Eurocrypt '91*, Lecture Notes in Computer Science. Springer-Verlag, 1991.

[3] J. Daemen and V. Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer-Verlag, 2002.

[4] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644-654, November 1976.

[5] E. Goh and S. Jarecki. A signature scheme as secure as the diffie-hellman problem. In *Advances in Cryptology - Eurocrypt'03, Lecture Notes in Computer Science*, pages 401-415. Springer-Verlag, 2003.

[6] A. Kostin. An anycasting protocol for anonymous access to a group of contents-equivalent servers in a distributed system. In *Advances in Information Systems*, volume 4243. Springer-Verlag, 2006.

[7] M. Manulis, A. Sadeghi, and J. Schwenk. Linkable democratic group signatures. In *Information Security Practice and Experience*, Lecture Notes in Computer Science. Springer-Verlag, 2006.

[8] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability and pseudonymity - a proposal for terminology. In *Privacy Enhancing Technologies*, Lecture Notes in Computer Science. Springer-Verlag, 2001.

[9] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *Communications of the ACM*, 42(2):32-48, 1999.

[10] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120-126, 1978.

[11] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Proceedings of the 7TH International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, pages 554-567. Springer-Verlag, 2001.