بهبود پروتکل AODV جهت مقابله با حملات کرم‌چاله در شبکه‌های اقتضایی

فرید محمدی[1] و حسین قرایی[2*]

[1] دانشگاه تهران، پردیس بین الملل کیش، تهران، ایران

[2] پژوهشگاه ارتباطات و فناوری اطلاعات، تهران، ایران

چکیده

حملهٔ کرم چاله یک نوع حمله فعال است که در لایهٔ سوم شبکه از شبکه‌های اقتضایی رخ می‌دهد. در این حمله مهاجمان با متقاعدکردن گره فرستنده برای ارسال اطلاعات از یک مسیر جعلی که کوتاه‌تر و سریع‌تر از مسیر عادی به نظر می‌رسد، سعی دارند ارسال بسته‌ها از تونل ایجادشده انجام شود تا بتوانند، حملات تحلیل ترافیک، انکار سرویس، رهاکردن بسته‌ها و یا جلورانی انتخابی را انجام دهند. هر پروتکلی که از مقیاس کم‌ترین تأخیر و کم‌ترین تعداد گام برای مسیریابی استفاده کند، در برابر این حمله آسیب‌پذیر است. در این مقاله یک راه‌کار جدید برای مقابله با حملات کرم چاله ارائه خواهد شد. در راه‌حل پیشنهادشده هر گره دارای یک وزن است و مجموع وزن‌ها در شبکه برابر صد خواهد بود. هرگاه گره‌ای قصد ارسال ترافیک به گره دیگر را داشته باشد، در بستهٔ RREQ[1] حداقل وزن درخواستی برای ایجاد ارتباط را بیان می‌کند. گره فرستنده با توجه به اهمیت داده‌هایی که ارسال خواهد کرد، مشخص می‌کند که مجموع وزن گره‌های شرکت‌کننده، در فرایند کشف مسیر، باید چقدر باشد. روش پیشنهادی MAODV نام‌گذاری خواهد شد. روش ذکرشده به‌صورت نرم‌افزاری بوده و با توجه به این‌که از روش رمزنگاری استفاده نمی‌کند، پیش‌بینی می‌شود سربار کمتری نسبت به سایر روش‌ها داشته باشد؛ و همچنین به‌علت عدم استفاده از الگوریتم‌های سخت، توان گره‌ها که اتفاقاً محدود است، کمتر صرف محاسبات خواهد شد.کارایی الگوریتم پیشنهادی در محیط ns-2 نشان داده خواهد شد.

واژگان کلیدی: شبکهٔ MANET، حملهٔ کرم چاله، پروتکل مسیریابی AODV، NS-2

Modified AODV Routing Protocol in Order to Defend Wormhole Attacks

Farid Mohammadi[1] & Hossein Gharaee[2]*

[1]Kish International Campus, Tehran University, Tehran, IRAN

[2]IRAN Telecom Research Center (ITRC), Tehran, IRAN

Abstract

Mobile Ad hoc Networks (MANET) are vulnerable to both active and passive attacks. The wormhole attack is one of the most severe security attacks in wireless ad hoc networks, an attack that can be mounted on a wide range of wireless network protocols without compromising any cryptographic quantity or network node. In Wormhole attacks, one malicious node tunnels packets from its location to the other malicious node. Such wormhole attacks result in a false route with fewer. If the source chooses this fake route, malicious nodes have the option of sniff, modify, selectively forward packets or them. Existing solution defends wormhole attacks, such as SECTOR, Packet Leashes, DelPHI, directional antenna. These solutions require special hardware or strict synchronized clocks or cause message overhead, or generate false-positive alarms. A novel approach MAODV: Modified AODV is proposed to defend wormhole attacks, launched in AODV. The proposed approach is based on weight per hop. Each node in network has its own weight, given

[1]RREQ

* نویسندهٔ عهده‌دار مکاتبات

[DOI: 10.18869/acadpub.jsdp.13.4.109]

by administration due to trusty power capability. Sum of weight will not be exceeded from 100. Whenever a source node wants to send a traffic to destination, puts its minimum weight in RREQ packet to constitute the route. The destination node is selected in the route that its weight is close to destination announcement weight. Since no special hardware and no encryption techniques are used, it is likely to have less overhead and delay, compared to other techniques.

The proposed wormhole defend mechanism is discussed in detail. Our proposed system does not require any synchronized clocks or special hardware to defend wormhole attacks. In our proposed system some parameters will be added to AODV routing protocol and make it more secure against wormhole attacks. We will name this new protocol as MAODV. In the first place, there is a master node in network, which weighs 100 (weighs of whole network). Whenever a node attends to enter the network, sends a join message to nearest neighbor. After receiving the message, master node will share its weights with the node requester, and sends the weight to this node requester. This process and weight sharing will be repeated after any requests to join a network, and total weight of network is not exceeded from 100. In our proposed method, each path which is created between source and destination, has a particular weight and this weight equals to intermediate node weights being added to each other. In MAODV whenever a source node wants to send RREQ packet, it adds the minimum weight to constitute route. After receiving RREQ packets, each intermediate node increases its weight beside increasing hop count. Each intermediate node does the same action, as far as destination node receives, RREQ packet among the received RREQ, one of them will be selected which its weight is the same as minimum requested weight by source, or slightly more than that. For instance, consider fig 1 which has 14 nodes. Assuming the node weights are equal for each node and its 7. As mentioned, the weight of whole network is tantamount to 100. Example 1: consider fig. 1 in which node A sends RREQ to node B. At first, node A checks its cache table to see whether there is a route between A and B, or not. If the answer is positive, it starts to send data. If the answer is negative, it sets up RREQ as follow: <A,B,1,7.25,[]> which means: A: source, B: destination, 1: hop count, 7: constitute path weight, 25: request weight, []: intermediate nodes. Each node which receives RREQ will check if it is the destination or not. If it wasn't: 1. Increase hop count, 2. puts its weight to constitute path weight, 3. Adds its address as an intermediate node. And then broadcasts RREQ packet to the neighbors. In this example node A sends RREQ to X and C, which are legitimate neighbor of A. When X receives the packet, modifies it as: <A,B, 2,(4,25,[X]> and forwards it to its neighbors on the other hand node. C modifies packet as: <A,B,2,(4,25,[C]> and forwards it to its neighbor D. This action will be repeated until B gets two RREQ - <A,B,4,28,25,[C,D,E]> and <A,B,7,25,48,[X,U,V,W, Z,Y> - among the received RREQ, B will be selected which its weight is the same as minimum requested weight by A, or slightly more than that, so the first route will be chosen by B. node B setup RREP packet as <A,B,1,4,25,7, [E,D,C]> which means: A: source, B: destination, 1: back path weight, 4: hop count, 25: request weight, 7: constitute path weight, [E,D,C]: intermediate nodes.

The effectiveness of the propose mechanism is evaluated using ns2 network simulator. The simulator's outcome demonstrates that PDR in MAODV rose by 5% up to 8% in presence of two malicious nodes, compared to PDR in AODV routing protocol. The average delay point to point in MAODV is more than AODV, but on the other hand, it is less than SAODV due to not using encryption.

Keywords: MANET, Wormhole attacks, AODV, NS2

۱-مقدّمه

شبکه‌های MANET به مجموعه شبکه‌های متحرک اطلاق می‌شود که بدون داشتن زیرساخت ثابت و داشتن خاصیت تحرک گره‌ها، با یکدیگر ارتباط برقرار می‌کنند. این گره‌ها همانند گوشی‌های بی‌سیم، لپ‌تاپ و دستیار دیجیتال دارای برد انتقال محدود هستند؛ از این‌رو، هر گره توانایی ارتباط مستقیم با گره دیگر و جلورانی پیام‌ها به همسایه‌ها را تا زمانی که بسته‌ها به گره‌های مقصد برسد دارد. برقراری امنیت در چنین شبکه‌هایی به بزرگترین چالش تبدیل شده است.

ماهیت باز شبکه‌های بی‌سیم کار را برای مهاجمان برای بررسی ترافیک شبکه و یا دخالت در آن آسان کرده است. گره‌های بداندیش می‌توانند حملات فعال و یا غیر فعال را به شبکه وارد کنند. در حملات غیر فعال یک گرهٔ بداندیش فقط محتوای بسته‌ها و ترافیک را شنود می‌کند؛ درحالی‌که در حملات فعال، هدف مهاجم خنثی‌کردن و یا حذف ویژگی‌های امنیتی، رهاکردن و یا تغییر بسته‌های قانونی است[4]. حمله کرم چاله یک حمله فعال است که در لایهٔ سوم شبکه رخ می‌دهد. دراین‌حمله، حمله‌کننده بسته‌ها و یا بیت‌ها را در یک محل شبکه ذخیره کرده و آن‌ها را از طریق یک تونل به محل دیگر منتقل می‌کند. تونل کرم‌چاله به‌طورمعمول بین دو گرهٔ بدرفتار[1] که به‌طورعمومی در فاصله دور از هم قرار دارند و با هم برای ایجاد یک مسیر جعلی با فاصله یک گام تبانی کرده باشند، تأسیس می‌شود. مهاجمان

[DOI: 10.18869/acadpub.jsdp.13.4.109]

در این حمله، مسیر کوتاه‌تر را معرفی می‌کنند و منجر به مسیریابی بسته‌ها از این مسیر جعلی می‌شوند. پس از برقراری موفق تونل کرم‌چاله، نظیرهای کرم‌چاله، بسته‌های داده را می‌توانند تغییر دهند، آن‌ها را رها کنند، حملات انکارسرویس را انجام دهند و یا بسته‌ها را به‌صورت دلخواه جلورانی و یا رها کنند [11]. پروتکل‌های مسیریابی بر مبنای نیاز همانند AODV و DSR در هنگام فرایند کشف مسیر در خطر حمله کرم‌چاله قرار دارند. بدین معنی که گره‌های بداندیش گره مبدأ را متقاعد می‌کنند، مسیری با تعداد گام کمتر و یا سریع‌تر برای ایجاد ترافیک به گره مقصد وجود دارد. درحالی‌که در واقعیت چنین مسیری وجود ندارد. ارزیابی تأثیرات حمله کرم‌چاله نشان می‌دهد، پارامترهای مختلف شبکه، همانند خروجی شبکه، میانگین تأخیر انتها به انتها و نرخ ازدست‌دادن بسته‌ها تحت تأثیر تونل کرم‌چاله در شبکه قرار می‌گیرد [1]. بنابراین در این مقاله سعی خواهد شد، با اعمال سازوکارهایی پروتکل مسیریابی AODV که یکی از مهم‌ترین پروتکل‌های مسیریابی بر مبنای نیاز است, نسبت به حمله کرم چاله مقاوم شود. در ادامه این مقاله، در بخش ۲ حمله کرم چاله و انواع مختلف آن بررسی، در بخش ۳ کارهای مرتبط معرفی خواهد شد؛ در بخش ۴ روش پیشنهادی مطرح و در بخش ۵ نتایج شبیه‌سازی نمایش و درنهایت بخش ۶ مقاله نتیجه‌گیری بیان می‌شود.

۲- سبک‌های مختلف حمله کرم چاله

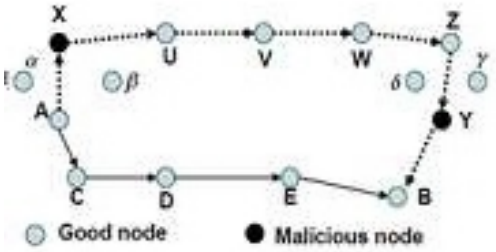
بر اساس تکنیک‌های مورد استفاده برای انجام حمله کرم چاله، سبک‌های این حمله به‌صورت زیر تقسیم بندی شده است[6]:

۲-۱- کرم چاله با استفاده از کپسوله‌کردن

زمانی که گره مبدأ، بسته RREQ را پخش می‌کند، یک گرهٔ بداندیش که یک قسمت از شبکه است، بستهٔ RREQ را دریافت می‌کند. گرهٔ بداندیش بسته را به گره‌ای که از قبل با هم تبانی کرده‌اند و این گره در یک فاصلهٔ دور نسبت به گرهٔ بداندیش و نزدیک به گرهٔ مقصد است، تونل می‌زند و بعد RREQ را دوباره پخش همگانی می‌کند. همسایه‌های گرهٔ تبانی‌کنندهٔ دوم، RREQ را دریافت کرده و تمامی تقاضاهای مجاز بعدی را که ممکن است، بعداً از مسیرهای چندگامی مجاز برسد حذف می‌کنند. نتیجه به این صورت خواهد بود که مسیر بین مبدأ و مقصد از طریق دو گرهٔ تبانی‌کننده عبور خواهد کرد که می‌توان گفت تشکیل یک کرم چاله را بین مبدأ و مقصد داده‌اند. این موضوع به گره‌ها، اجازه کشف مسیرهای مجازی را که بیش از دو گام فاصله دارند، نخواهد داد. شکل (۱) را در نظر بگیرید، جایی که A گره مبدأ وB گره مقصد است، هر دو با حضور دو گرهٔ بداندیش X و Y برای یافتن کوتاه‌ترین مسیر میان یکدیگر تلاش می‌کنند. گره A بسته RREQ را پخش می‌کند، X بسته RREQ را دریافت کرده و آن را به مقصد Y با توجه به مسیری که میان X و Y وجود دارد، یعنی(U-V-W-Z) بازگشایی می‌کند. گرهٔ Y بسته را کپسوله و آن را دوباره پخش می‌کند که به گره B می‌رسد. با توجه به کپسوله‌کردن بسته، مقدار گام در طول پیمایش (U-V-W-Z) اضافه نخواهد شد. به‌طور هم‌زمان RREQ مسیر خود از A به B را از طریق (C-D-E) طی خواهد کرد. اکنون گرهٔ B دو مسیر را پیش خواهد داشت، اولی مسیری با طول چهار گام خواهد بود (A-C-D-E-B)و مسیر دوم که به نظر مسیری با طول سه گام است (A-X-Y-B). گرهٔ B مسیر دوم را با توجه به این‌که کوتاه‌تر به‌نظر می‌رسد، انتخاب خواهد کرد؛ درحالی‌که درواقع این مسیر با طول گام هفت است. هر پروتکل مسیریابی که از مقیاس کوتاه‌ترین مسیر برای انتخاب بهترین مسیر استفاده می‌کند، در برابر این روش حملهٔ کرم چاله آسیب‌پذیر است [2].

۲-۲- کرم چاله با استفاده از کانال خارج از باند

این مدل برای حمله, از کانال out-of-band استفاده می‌کند. این حمله با استفاده از کانال پهنای باند قوی بین گره‌های بداندیش انجام می‌شود. این مدل از حمله نیازمند ظرفیت سخت‌افزاری ویژه است.

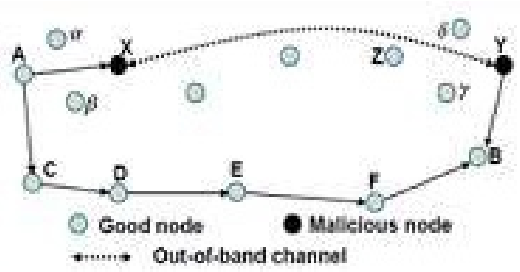


شکل-۱): کرم‌چاله با استفاده از کپسوله کردن
(Figure-1): Wormhole using encapsulation method

شکل (۲) را در نظر بگیرید، جایی که گرهA ، RREQ را به گرهٔ B ارسال می‌کند و گره‌های X و Y گره‌های بداندیش هستند که دارای کانال خارج از باند میان یکدیگر می‌باشند. گرهٔ X بسته RREQ را به Y که همسایه مجاز B است، تونل می‌زند. گرهٔ Y بسته را به همسایه‌های خود که شامل B نیز است، پخش می‌کند. گرهٔ B ، دو RREQ را دریافت می‌کند

[DOI: 10.18869/acadpub.jsdp.13.4.109]

نخستین مسیر شامل (A-X-Y-B) و دومی (A-C-D-E-F-B) است. اولین مسیر، هم کوتاه‌تر و هم سریع‌تر از مسیر دوم است و به‌همین دلیل توسط B انتخاب می‌شود.



(شکل ـ ۲): کرم چاله با استفاده از کانال خارج از باند
(Figure-2): Wormhole using out of bannd channel method

۲-۳- کرم چاله با استفاده از توان بالای ارسال

در این شیوه، وقتی یک گره بداندیش بسته RREQ را دریافت می‌کند، تقاضا را به یک سطح توان بالا که این توانایی در بین گره‌های دیگر شبکه موجود نیست، پخش می‌کند. هر گره که پخش توان بالا را می‌شنود، دوباره آن را به سمت مقصد، پخش همگانی می‌کند. با استفاده از این روش، شانس گرهٔ بداندیش برای این‌که در مسیر ایجادشده میان مبدأ و مقصد باشد، افزایش خواهد یافت.

۲-۴-کرم چاله با استفاده از بازپخش‌کردن بسته[1]

در این شیوه گرهٔ بداندیش بسته‌ها را بین دو گره دور ارسال می‌کند تا آن‌ها را متقاعد کند که با یکدیگر همسایه هستند. این شیوه می‌تواند با حضور یک گرهٔ بداندیش نیز انجام شود. هرچند با وجود چند گرهٔ بداندیش می‌توان فهرست همسایه گره قربانی را به‌صورت چندگامی گسترش داد. این عمل توسط گره نفوذگر X که داخل محدوده انتقال دو گره مجاز A و B قرار دارد، انجام می‌شود، جایی که A وB خود در داخل محدوده انتقال یکدیگر نیستند. گره نفوذگر X فقط کنترل ترافیک میان A وB (و برعکس) را تونل می‌زند و این کار را بدون تغییر فرض‌شده در پروتکل مسیریابی انجام می‌دهد. برای مثال بدون واردکردن نشانی به‌عنوان منبع در سرآمد بسته‌ها، و به این ترتیب گره X به‌صورت مجازی نامرئی خواهد بود. گره X پس از آن می‌تواند بسته‌های تونل‌شده را حذف و یا هر زمان که خواست این ارتباط را قطع کند. درواقع یک ارتباط خارج از قلمرو میان A-B به‌صورت مصنوعی توسط گره نفوذگر X با کنترل کرم چاله پیام‌ها میان A-B می‌تواند شکل گیرد[13].

[1] Packet relay

۲-۵- کرم چاله با استفاده از انحراف پروتکل[2]

برخی پروتکل‌های مسیریابی مانند ARAN، مسیر با کوتاه‌ترین تأخیر را نسبت به کم‌ترین تعداد گام، در اولویت قرار می‌دهند. در طی تقاضای ارسال مسیر، گره‌ها به‌طورمعمول منتظر یک مقدار تصادفی از زمان قبل از ارسال می‌باشند. این عمل با توجه به این واقعیت که ارسال تقاضا با انتشار همگانی به پایان می‌رسد رخ می‌دهد، بنابراین کاهش‌دادن تصادم لایه MAC[3] امر مهمی به‌شمار می‌رود. یک گره بداندیش به‌راحتی می‌تواند حمله کرم چاله را با موافقت نکردن پروتکل و انتشار همگانی بدون منتظرماندن[4] برای مقدار تصادفی زمان، انجام دهد. هدف از انجام این کار به این شکل خواهد بود که بسته درخواست ارسال، به‌عنوان نخستین بسته به مقصد برسد، این عامل احتمال این را که مسیر بین مبدأ و مقصد شامل گرهٔ بداندیش باشد، افزایش می‌دهد [10].

(جدول- ۱): روش‌های مختلف حمله کرم‌چاله
(Table- 1): Wormhole attack methods

نام روش	حداقل گره‌های تبانی کننده	نیازهای خاص
کپسوله کردن بسته	دو	ــــــــــ
کانال خارج از باند	دو	ارتباط خارج از باند
توان بالای ارسال	یک	منبع با انرژی زیاد
باز پخش بسته‌ها	یک	ــــــــــ
انحراف پروتکل	یک	ــــــــــ

۳-پژوهش‌های پیشین

۳-۱-روش مهار بسته‌ها

سازوکار مهار بسته‌ها یک سازوکار کلی جهت تشخیص و دفاع از حملات کرم چاله است. یک مهار، یک‌سری اطلاعات است که به یک بسته اضافه می‌شود [5] تا بیشینهٔ فاصله انتقال بسته را محدود و مشخص کند. مهارها جهت محافظت در برابر حملات کرم چاله در انتقال‌های بی‌سیم طراحی شده‌اند. مهار بسته‌ها به دو صورت جغرافیایی و زمانی انجام می‌شود. در مهار جغرافیایی هر گره موقعیت جغرافیایی خود

[2] Protocol Deviation
[3] Message Authentication Code
[4] Backing off

را با استفاده از GPS[1] در شبکه پیدا می‌کند. همچنین، تمامی گره‌ها دارای کلاک هماهنگی ضعیف[2] در حد میلی ثانیه هستند. هر گره قبل از ارسال بسته، موقعیت خود (p_s) و زمان ارسال بسته (t_s) را به بسته اضافه می‌کند. گره گیرنده با دریافت بسته و مقایسه p_s با موقعیت خودش (p_r) و مقایسه t_s با زمان دریافت بسته (t_r) و با درنظرگرفتن بیشینهٔ تفاوت کلاک هماهنگی بین فرستنده و گیرنده برابر $\pm\Delta$ و سرعت نور برابر v و بیشینهٔ خطا در اطلاعات محل برابر δ، فاصله بین فرستنده و گیرنده (d_{sr}) را با فرمول (۱) محاسبه می‌کند:

$$d_{sr} \le \| p_s - p_r \| + 2v.(t_r - t_s + \Delta) + \delta \qquad (۱)$$

گیرنده با محاسبه d_{sr} از این مسافت جهت کشف و شناسایی رخ داد و یا عدم رخ داد حمله کرم چاله استفاده می‌کند. در مهار زمانی تمامی گره‌ها نیاز به یک کلاک هماهنگی دقیق[3] در حد نانو ثانیه دارند. هنگام استفاده از مهار زمانی گره فرستنده، زمان ارسال بسته (t_s) و همچنین زمان انقضای بسته (t_e) را که بعد از چه مدت بسته رها خواهد شد طبق فرمول (۲) محاسبه کرده و به بسته اضافه می‌کند. (c سرعت انتشار سیگنال در شبکه بی‌سیم و کلاک هماهنگی Δ نشان‌دهندهٔ بیشینهٔ تفاوت بین دو گره بوده و مقدار آن باید برای تمامی گره‌های شبکه واضح بوده و در حدود چند میلی ثانیه باشد.) و گیرنده می‌تواند با استفاده از زمان ارسال بسته (t_s) توسط گره فرستنده و زمان دریافت بسته (t_r) و سرعت نور از وجود مسیر قانونی مطلع شود. اگر زمان رسیدن بسته از زمان محاسبه شده توسط گیرنده زیادتر باشد، گیرنده بستهٔ بالا را رها می‌کند.

$$t_e = t_s + L/c - \Delta \qquad (۲)$$

۳-۲- روش DelPHI[4]

اندازه تأخیر برای گام توسط [8] ارائه شد. در DelPHI تلاش‌ها به‌منظور یافتن مسیرهای ازبین‌رفته میان فرستنده و گیرنده انجام می‌شود. سپس میزان تأخیر و طول هر مسیر محاسبه شده و میانگین تأخیر به‌ازای گام، در راستای هر مسیر تعیین می‌شود. این مقادیر برای مشخص‌کردن حمله کرم چاله به کار می‌رود. مسیری که شامل پیوند کرم چاله است، دارای مقدار تأخیر بیشتری برای هر گام خواهد بود.

[1] Global Positioning system
[2] Loosely Synchronized Clock
[3] Tightly Synchronized Clock
[4] Delay per Hop Indicator

این سازوکار می‌تواند حمله کرم چاله را شناسایی کند، گرچه توانایی شناسایی محل دقیق کرم چاله را ندارد.

۳-۳- روش‌های مبتنی بر RTT[5]

در [9] گره فرستنده با تشخیص میزان تأخیر مسیرهای مختلف تا مقصد، حمله کرم چاله را شناسایی می‌کند. شماره گام[6] و اطلاعات تأخیر مسیرهای جداشده، جمع‌آوری و مقدار مقدار تأخیر برای هر گام به‌عنوان نشان‌گر حمله به کار گرفته می‌شود. در حالت عادی مقدار تأخیر بسته که در یک گام منتشر می‌شود، در طول مسیر برای هر گام مشابه است، ولی در هنگام حمله کرم چاله این مقدار تأخیر بدون توجیه با توجه به حضور گره‌های بدرفتار در طول مسیر بالا خواهد بود. بنابراین اگر مسیری تأخیر بالا به‌ازای هر گام داشته باشد، در معرض کرم چاله قرار خواهد گرفت. مشکل این روش این است که، شناسایی، تنها توسط زمان انتقال می‌تواند منجر به تولید زیاد نرخ خطای مثبت شود. یعنی ممکن است تأخیر ایجادشده بنا به ازدحام بسته‌ها در مواردی خاص رخ دهد، نه این‌که لزوماً حمله اتفاق افتاده باشد.

۳-۴- روش مبتنی بر چالش/ پاسخ[7]

در [3] هدف استفاده از سیستم چالش/ پاسخ برای به کمینه رساندن تأخیرهای ممکن بدون استفاده از ابزار CPU است. با استفاده از الگوریتم محدودکنندهٔ فاصله، فاصله میان دو همسایه با ارسال یک بیت چالش محاسبه شده و مشخص می‌شود که آیا فاصلهٔ محاسبه‌شده در محدودهٔ انتقال ممکن وجود دارد یا خیر. از آنجایی که این الگوریتم، فاصله در هر گام را محاسبه می‌کند، نمی‌تواند راه حلی برای مقابله با حملات کرم چاله آشکار داشته باشد. این روش به یک سخت‌افزار ویژه نیاز دارد تا بتواند به یک بیت چالش به‌صورت آنی پاسخ دهد.

۳-۵- روش LiteWorp

این روش [7] پروتکلی جهت کشف حمله کرم چاله در شبکه‌های ایستا ارائه کرد که به آن LiteWorp گفته می‌شود. هنگامی که LiteWorp اجرا می‌شود، گره‌ها اطلاع کاملی از مسیرهای با دو گام با توجه به همسایه‌های خود

[5] Round Trip Time
[6] Hop number
[7] Challenge/Response

[DOI: 10.18869/acadpub.jsdp.13.4.109]

به‌دست می‌آورند. درحالی‌که در پروتکل‌های مسیریابی استاندارد شبکه‌های اقتضایی، گره‌ها به‌طورمعمول همسایه‌های یک قدمی خود را بررسی می‌کنند؛ ولی در پروتکل LiteWorp، آن‌ها شناختی در مورد همسایهٔ،گره همسایه خود را دارند، و با این کار می‌توانند از مزیت اطلاعات همسایه با دو گام، به‌جای یک گام بهره‌مند باشند. این اطلاعات ، برای آشکارکردن حمله کرم‌چاله مورد استفاده می‌تواند قرار گیرد. بعد از احراز هویت، گره‌ها پیام‌ها را از گره‌ای که خود را به‌عنوان همسایه ثبت نکرده باشد، نخواهند پذیرفت. همچنین گره‌ها رفتار همسایه‌ها را به‌منظور این‌که آیا بسته‌های داده به‌درستی توسط همسایه ارسال می‌شود، نظارت می‌کنند، (روشی که به Watchdog) معروف است. در LiteWorp گره‌ها نه تنها تأیید می‌کنند تمامی بسته‌ها از همسایه‌های خود به‌درستی ارسال می‌شوند، بلکه اطمینان حاصل می‌کنند که هیچ گره‌ای بسته‌ای را که دریافت نکرده ارسال نکند.

۳-۶- روش شناسایی انتها به انتها

در [12] روشی جهت شناسایی حمله کرم چاله ارائه شده که به **EDWA**[1] معروف است. این روش بر مبنای کم‌ترین تعداد گام بنا نهاده شده است. در EDWA هر گره با استفاده از سامانهٔ موقعیت‌یاب، جایگاه خود را در شبکه پیدا می‌کند، و از تخمین مسافت اقلیدسی جهت تخمین کوتاه‌ترین مسیر استفاده می‌کند. در مرحلهٔ ردیابی، فرستنده کوتاه‌ترین مسیر تا گیرنده را با تخمین شمارش گام محاسبه می‌کند و آن را با مقدار تخمینی موجود در بسته RREP مقایسه می‌کند. اگر این مقدار کمتر از مقدار تخمین شده باشد، مسیر بالا را یک مسیر کرم چاله پیش‌بینی می‌کند. در مرحلهٔ آخر گره فرستنده از کوتاه‌ترین مسیر، از میان مسیرهای قانونی شروع به ارسال داده می‌کند.

(جدول- ۲): مقایسه روش‌های مختلف مقابله با کرم چاله
(Table- 2): Comparison between Prevention of Wormhole attack

روش	توضیحات
مهار بسته (جغرافیایی)	نیاز به GPS برای هر گره، نیاز به همگام‌سازی زمانی (ms)، محدودیت‌های فناوری GPS
مهار بسته (زمانی)	نیاز به همگام‌سازی زمانی قوی (ns)، غیرقابل پیاده‌سازی
انتها به انتها	نیاز به همگام‌سازی زمانی ضعیف (ms)، نیاز به GPS
آنتن‌های جهتی	نیاز به آنتن‌های جهتی برای تمامی گره‌ها
LiteWorp	بدون نیاز به سخت‌افزار اضافی، فقط قابل استفاده در شبکه‌های ایستا
چالش/ پاسخ	نیاز به سخت‌افزار برای ارسال پاسخ آنی، عدم کارایی مقابل کرم‌چاله آشکار

۴- مدل پیشنهادی

راه‌کار ارائه‌شده برای این نوع حمله، یک روش نرم‌افزاری و بدون نیاز به سخت‌افزار اضافی و کلاک هماهنگی است. در روش پیشنهادی یک‌سری سازوکارهای امنیتی بر روی پروتکل AODV اضافه شده است و باعث ایجاد امنیت بیشتر در برابر حمله کرم چاله خواهد شد. این پروتکل جدید[2] MAODV نام‌گذاری خواهد شد. در ابتدای راه‌اندازی شبکه، یک گره به‌عنوان گره مادر در شبکه وجود دارد که وزن آن برابر ۱۰۰ (وزن کل شبکه) خواهد بود. هنگامی که گره‌ای قصد ورود به شبکه را داشته باشد، به نزدیکترین گره پیام عضویت ارسال می‌کند. گره مادر پس از دریافت پیام عضویت گره جدید وزن خود را با درخواست کننده تقسیم می‌کند و به گره مذبور می‌فرستد. این فرایند و تقسیم وزن با حضور هر گره جدید در شبکه تکرار می‌شود، تا وزن کل شبکه در عدد ۱۰۰ باقی بماند.

فرایند کشف مسیر–در پروتکل مطرح‌شده هر مسیر که بین گره‌های مبدأ و مقصد ایجاد می‌شود، دارای یک وزن بوده و وزن هر مسیر از جمع وزن‌های گره‌های میانی به‌دست می‌آید. همانند پروتکل AODV، برای کشف مسیر از بسته‌های RREP و RREQ استفاده می‌شود. این بسته‌ها ساختاری متفاوت با بسته‌های پروتکل AODV دارند. در این پروتکل گره مبدأ هنگام ارسال بسته RREQ ، وزن مسیر درخواستی را اعلام می‌کند و هر یک از گره‌های میانی هنگام ارسال بسته RREQ به سمت مقصد ، علاوه‌بر این‌که مقدار hopCount را افزایش می‌دهند، وزن خود را نیز به وزن مسیر ایجادشده اضافه می‌کنند. این روند آن‌قدر تکرار می‌شود تا بستهٔ مذبور به مقصد برسد. برای طراحی بسته‌های RREQ پروتکل MAODV فیلدهای PathReq ، Patweight به ساختار بسته‌های RREQ پروتکل AODV اضافه شده است.

[1] End-to-End Detection Wormhole Attack

[2] Modified AODV

[DOI: 10.18869/acadpub.jsdp.13.4.109]

فیلد PathReq حاوی وزن مسیری است که توسط گره مبدأ درخواست می‌شود. فیلد Patweight حاوی وزن مسیر ایجادشده است. وزن مسیر ایجادشده از مجموع وزن‌های گره‌های شرکت‌کننده در فرایند کشف مسیر به‌دست می‌آید. این فیلد به‌عنوان وزن بسته RREQ نامیده می‌شود. بسته RREP نیز همانند بسته RREQ دارای فیلد PathReq است.

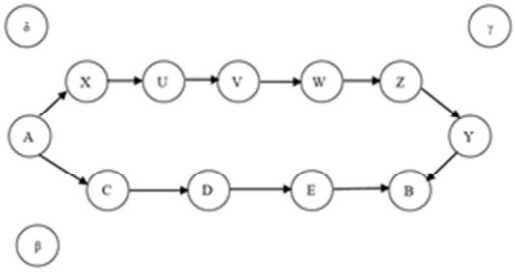
بسته RREP حاوی فیلدی به نام Bakpatweight است. هنگام ارسال بسته RREP از طرف مقصد، مقدار آن برابر وزن گره مقصد خواهد بود. هنگامی که گره‌های میانی بسته RREP را دریافت می‌کنند، مقدار آن را به اندازهٔ وزن خود افزایش می‌دهند و بسته را به سمت مبدأ ارسال می‌کنند. این فیلد به گره‌های میانی کمک می‌کند تا متوجه شوند در چه فاصله وزنی از گره مقصد قرار دارند. هنگامی‌که بسته RREP به مبدأ می‌رسد ، مقدار آن برابر وزن مسیر ایجادشده خواهد بود. فیلد Bakpatweight حاوی وزن مسیر بازگشتی است. این فیلد به‌عنوان وزن بسته RREP نامیده می شود.

با توجه به این‌که مسیرهای ایجادشده ممکن است از نظر فاصله و وزن بهینه نباشند، باید این امکان در آنها وجود داشته باشد تا بتوان آن‌ها را بهینه کرد. برای بهینه‌کردن مسیرها جدول Cache را تغییر می‌دهیم. با رسیدن یک RREP جدول Cache مورد جستجو قرار می‌گیرد. اگر این RREP جدید باشد (یعنی در جدول وجود نداشته باشد) به جدول اضافه خواهد شد در غیر این صورت سه حالت زیر می‌تواند اتفاق بیافتد: حالت نخست این‌که وزن RREP رسیده از وزن مسیر موجود در Cache بیشتر باشد. در این صورت جدول تغییری نخواهد کرد. حالت دوم زمانی است که وزن RREP رسیده مساوی وزن مسیر موجود در Cache باشد. در این‌صورت دو حالت ممکن است اتفاق بیافتد. حالت نخست زمانی رخ می‌دهد که فیلد hopcount، RREP از hopcount موجود در Cache کوچک‌تر باشد، در این صورت RREP جدید هم وزن مسیر قدیم بوده با این تفاوت که از لحاظ فاصله کوتاه‌تر است. درنتیجه جدول Cache با توجه به اطلاعات موجود در RREP تغییر خواهد کرد. حالت دوم زمانی است که فیلد hopcount، RREP از hopcount موجود در Cache بزرگ‌تر یا مساوی باشد در این‌صورت جدول Cache تغییر نخواهد کرد. حالت سوم زمانی است که وزن RREP رسیده از وزن موجود در Cache کوچک‌تر باشد. در این‌صورت نیز جدول Cache با توجه به اطلاعات موجود در RREP تغییر خواهد کرد.

در این بخش مثال‌های مختلفی از پروتکل MAODV مطرح خواهد شد تا عملکرد آن بهتر معلوم شود. همان‌طور که بیان شد، وزن کل شبکه برابر ۱۰۰ است. در مثال ذکرشده فرض کردیم، وزن به‌طور تقریبی به‌طور مساوی بین گره‌ها تقسیم شده است. با توجه به اینکه چهارده گره در شبکه فرضی وجود دارند، وزن هر گره برابر هفت در نظر گرفته شده است.

مثال ۱: برای شکل (۳) سناریوی نیز را در نظر بگیرید. فرض کنید گره A بخواهد اطلاعاتی را به گره B ارسال کند.

مرحله نخست : ابتدا این گره جدول Cache خود را مورد جستجو قرار می‌دهد تا ببیند آیا مسیری برای رسیدن به گره B وجود دارد یا نه. در صورتی‌که پاسخ مثبت باشد بلافاصله شروع به ارسال داده می‌کند. در غیر این‌صورت بسته RREQ را به‌صورت <A,B,1,7,25,[]> تنظیم می‌کند، بدین معنی که : **A** : مبدأ ارسال ترافیک. **B:** مقصد مورد نظر. **۱:** تعدادگام **۷:** وزن مسیر ایجادشده (یا وزن بسته RREQ) که در ابتدا وزن مبدأ است. **۲۵** : وزن مسیر درخواستی توسط مبدأ. [X,U,V,W,Z,Y]: گره‌های میانی شرکت کننده در فرایند مسیریابی. گره A آن‌را به تمامی همسایه‌های خود ارسال می‌کند. **مرحله دوم :** هر کدام از همسایه‌ها که بسته بالا را دریافت کنند، بلافاصله بررسی می‌کند که آیا مقصد بسته است یا خیر (با توجه به فیلد دوم). درصورتی‌که مقصد بسته نباشد، بسته RREQ رسیده را بدین شکل تغییر می‌دهند: ۱. مقدار hopcount را یک واحد افزایش می‌دهند.۲. وزن خود را به وزن بسته اضافه می‌کنند. ۳. نشانی خود را به فهرست گره‌های میانی اضافه می‌کنند.



(شکل-۳): چگونگی اتصال گره‌ها به یکدیگر
(Figure-3): Node connection to each other

پس از انجام مراحل بالا بسته RREQ به تمامی همسایه‌ها ارسال می‌شود. با توجه به مثال بالا گره A بسته RREQ را به همسایه‌های یک قدمی خود یعنی، X و C ارسال می‌کند. X با دریافت بسته RREQ و با توجه به مطالب گفته شده، آن‌را به‌صورت <A,B,2,14,25,[X]>

[DOI: 10.18869/acadpub.jsdp.13.4.109]

تغییر می‌دهد؛ سپس بسته RREQ را به تمامی همسایه‌های خود که در این مثال فقط U است ارسال می‌کند. در سوی دیگر گره C با دریافت بسته از گره A آن‌را به‌صورت <A,B,2,14,25[C]> تغییر می‌دهد و به همسایه خود، یعنی گره D ارسال می‌کند. مرحله دوم آن‌قدر تکرار می‌شود تا بسته به مقصد برسد. با توجه به شکل (۳) و انجام مراحل بالا دو بسته به شکل‌های <A,B,4,28,25[C,D,E]> و <A,B,7,25,48[X,U,V,W,Z,Y]> به گره B می‌رسند.

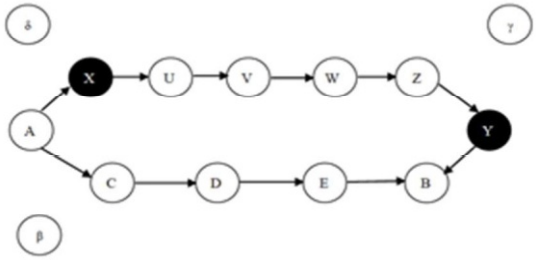
مرحله سوم : گره مقصد از بین RREQ های رسیده RREQ را قبول می‌کند که نخست این‌که وزن بسته آن از وزن مسیر درخواست‌شده زیادتر باشد و دوم این‌که اختلاف کمی داشته باشد. باتوجه به مطالب گفته شده گره B از بین دو درخواست رسیده به درخواست <A,B,4,28,25[C,D,E]> پاسخ می‌دهد؛ زیرا وزن این بسته به وزن درخواست‌شده نزدیک‌تر است.

مرحله چهارم : گره مقصد پس از انتخاب RREQ، اقدام به ارسال بسته می‌کند. با توجه به مثال بالا گره B، بسته RREP را به‌صورت<A,B,1,4,25,7 [E,D,C]> تنظیم می‌کند، که:

A: مبدأ. **B**: مقصد. **۱** : تعداد گام‌های مسیر بازگشت. **۴** : تعدادگام‌های مسیر رفت. **۲۵** : وزن درخواست‌شده توسط مبدأ. **۷** : وزن مسیر بازگشت (یا وزن بسته RREP) که در ابتدا وزن مقصد است. **[E, D, C]**: گره‌هایی که بسته RREP باید از آن‌ها عبور کند. پس از تنظیم بسته RREP ، گره B آن‌را تنها به گره E ارسال می‌کند.

مرحله پنجم: هر کدام از گره‌ها که بسته RREP را دریافت کند، ابتدا جدول Cache خود را بر اساس اطلاعات مورد نیاز تکمیل می‌کند و سپس تغییراتی را به‌صورت زیر روی بسته RREP اعمال می‌کند: ۱. مقدار hopcount مسیر بازگشت را یک واحد افزایش می دهد. ۲. مقدار hopcount مسیر رفت را یک واحد کاهش می دهد. ۳. وزن خود را به وزن بسته اضافه می‌کند. پس از انجام مراحل بالا ، بستهٔ RREP به گرهٔ بعدی ارسال می‌شود. مرحلهٔ پنجم آن‌قدر تکرار می‌شود تا بسته RREP به مبدأ برسد. با توجه به مثال بالا و انجام مراحل گفته شده، بسته RREP به‌صورت <A,B,4,1,25,28,[D,C,B]> به مبدأ می‌رسد. پس از رسیدن بسته به مبدأ ، مبدأ شروع به ارسال داده می‌کند.

مثال ۲: در این مثال عملکرد پروتکل MAODV در مقابل حملات کرم چاله بررسی خواهد شد. برای این منظور فرض می‌کنیم گره A می‌خواهد ترافیکی را به گره B در حضور دو گره بداندیش X و Y که سعی دارند، روند مسیریابی را به‌گونه‌ای تغییر دهند تا در هنگام ارسال داده‌ها بخشی از مسیر باشند، ارسال کند. ابتدا گره A بسته RREQ به‌صورت <A,B,1,7,25,[]> تنظیم کرده و آن‌را به تمامی همسایه‌های خود ارسال می‌کند. این بسته در گره بدخواه X دریافت و به‌صورت <A,B,2,14,25,[x]> به همسایه‌های آن می‌رسد. با ادامه مراحل فرایند کشف مسیر بسته‌ای به‌صورت <A,B,6,42,35,[X,U,V,W,Z]> به گره بدخواه Y می‌رسد. این گره با پردازش گره‌های میانی در می‌یابد که گره بدخواه X نیز در بین گره‌های شرکت کننده در کشف مسیر است. به همین دلیل بسته رسیده را به‌صورت <A,B,3,42,25,[X,Y]> به همسایههای خود ارسال می کند. در این ارسال تمامی گره‌های میانی بین گره‌های بدخواه حذف شده و علاوه‌بر این‌که مقدار hopcount افزایش نیافته است، بلکه از مقدار آن به تعداد گره‌های حذف‌شده کاسته می‌شود[۱۴]. با انجام کارهای بالا دو بسته به شکل‌های <A,B,4,28,25[C,D,E]> و <A,B,3,49,[X,Y]> به گره B می‌رسند. هرچند بسته <A,B,3,49,[X,Y]> از لحاظ گام کوچک‌تر است، ولی در گره B انتخاب نمی‌شود، زیرا در پروتکل MAODV تعداد گام ملاک انتخابRREQ نیست؛ و عامل دوم این‌که نسبت به بسته دیگر اختلاف وزن بسته به وزن مسیر درخواست‌شده بیشتر است.



(شکل- ۴): گره‌های X و Y گره‌های بداندیش هستند.
(Figure-4): Malicious X and Y nodes

در مثال ذکرشده اگر از پروتکل‌های DSR و یا AODV استفاده کنیم، مسیر دوم یعنی <A,B,3,49,[X,Y]> به‌علت این‌که دارای تعداد گام کمتری است، انتخاب می‌شود؛ و درواقع مسیر انتخابی شامل تونل کرم چاله خواهد بود. سؤال

[۱] به این‌کار به اصطلاح کپسوله‌کردن می‌گویند.

[DOI: 10.18869/acadpub.jsdp.13.4.109]

قابل طرح این است که گره Y چرا وزن بسته را همانند hopcount کاهش (یا افزایش) نمی‌دهد تا توسط مقصد انتخاب شود. پاسخ می‌تواند بدین شکل باشد که: چون گره E هیچ‌گونه اطلاعاتی در مورد محل گره مقصد و وزن گره‌های میانی بین آن‌ها ندارد، هر گونه کاهش (یا افزایش) نمی‌تواند دلیلی برای انتخاب بسته باشد؛ زیرا اگر مقدار وزن بسته را کاهش دهد، ممکن است که به هیچ وجه بسته مذبور انتخاب نشود و یا اگر وزن بسته را افزایش دهد ممکن است در رقابت با سایر بسته‌ها از دور خارج شود. در نتیجه انتخاب بسته به‌ازای افزایش یا کاهش وزن آن در مقصد امری تصادفی خواهد بود، درحالی‌که در پروتکل‌هایی مثل DSR و AODV اگر گره‌های بدرفتار هر چقدر مقدار hopcount را بیشتر کاهش دهند، امکان انتخاب‌شدن آنها نیز بیشتر خواهد شد؛ در صورتی‌که در پروتکل MAODV این‌گونه نیست. و عامل دوم این‌که، با توجه به توزیع به‌طور تقریبی یکنواخت وزن میان گره‌ها، گره‌های بداندیش توانایی کاهش یا افزایش قابل توجه وزن را نخواهند داشت.

۵- پیاده‌سازی و نتایج شبیه‌سازی

برای نشان دادن کارایی راه‌کار ارائه‌شده، آن را روی پروتکل AODV اعمال کرده و از محیط ns-2 بدین منظور استفاده خواهد شد. همان‌طور که پیش‌تر اشاره شد، پروتکل مسیریابی AODV یک پروتکل مسیریابی بر مبنای نیاز است. در این پروتکل مسیرها تنها زمانی کشف می‌شوند که مبدأ اقدام به برقراری ارتباط با گره دیگری را انجام دهد. زمانی که یک گره بخواهد با گره دیگری ارتباط برقرار کند، باید فرایند کشف مسیر را در شبکه فراخوانی کند. در این حالت قبل از برقراری ارتباط، تأخیر قابل توجهی مشاهده می‌شود. جدول (۳) پارامترهای استفاده‌شده در محیط شبیه‌سازی را نشان می‌دهد.

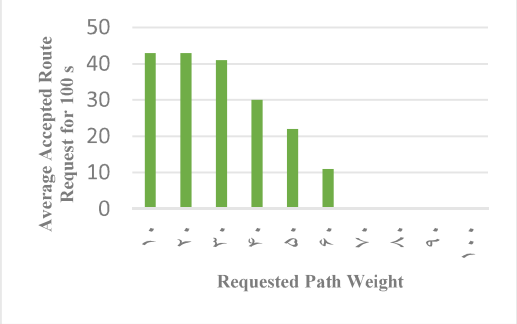
تعداد RREQهای پذیرفته‌شده در مقصد

هدف از این پارامتر این است که ببینیم، چه تعداد از مسیرهایی که توسط مبدأ با وزن‌های مختلف درخواست می‌شود، در مقصد مورد پذیرش قرار می‌گیرند. در این بخش شبیه‌سازی با ده گره در مدت زمان صد ثانیه انجام خواهد شد.

نتایج این شبیه‌سازی نشان می‌دهد که هرچه وزن درخواستی توسط مبدأ افزایش یابد تعداد RREQ پذیرفته‌شده به‌ازای آن درخواست در مقصد کاهش می‌یابد. برای مثال اگر وزن درخواستی توسط مبدأ شصت باشد، ده بسته RREQ در مقصد مورد پذیرش قرار خواهد گرفت. همان‌طور که دیده می‌شود، تعداد RREQهای پذیرفته‌شده به‌ازای وزن‌های بالای شصت، به‌طور تقریبی صفر است. یعنی اگر مبدأ مسیری با وزن هفتاد را درخواست کند، چنین مسیری ایجاد نخواهد شد. با توجه به این‌که وزن صد بین گره‌ها توزیع می‌شود، نتایج این شبیه‌سازی را می‌توان برای تعداد گره‌های بیشتر بسط داد.

(جدول-۳): پارامترهای شبیه‌سازی
(Table- 3): Simulation Parameters

پروتکل استفاده شده در شبیه سازی	MAODV
زمان شبیه سازی	۱۰۰۰S
ابعاد شبیه سازی	۱۰۰۰*۱۰۰۰
تعداد گره‌ها	۱۰-۵۰
محدوده انتقال بی‌سیم	۲۵۰ متر
نرخ انتقال داده	۴Mb/s
اندازه بسته‌ها	۵۱۲
سرعت گره‌ها	۱-۱۰S
نوع تحرک گره‌ها	تصادفی
نوع ترافیک	CBR



(شکل- ۵): تعداد RREQهای پذیرفته‌شده به‌ازای ۱۰ گره
(Figure- 5): Number of acceptance RREQ nodes

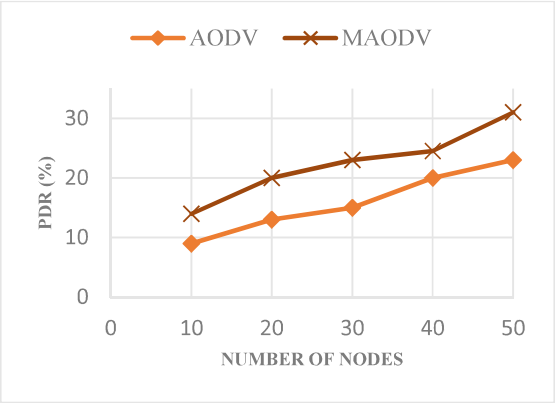
میانگین نرخ بسته‌های رسیده

نرخ بسته‌های رسیده که دراصطلاح با PDR[1] نمایش داده می‌شود، برای تعریف تعداد بسته‌های داده‌ای است، که یک گره فرستاده و گره مقصد آن‌ها را صحیح دریافت کرده است. میانگین نرخ بسته‌های رسیده عبارت از نرخ بسته‌های رسیده به یک مقصد از تمامی گره‌های شبکه است. شکل (۶) میزان بسته‌های رسیده به مقصد را در ترافیک UDP با تعداد ده الی پنجاه گره، با حضور دو گرهٔ بداندیش که سعی دارند با تونل‌زدن مسیر بین منبع و مقصد، در فرایند مسیریابی شرکت کنند؛ و با حداقل وزن‌های مختلف بین ده

[1] Packet Delivery Ratio

[DOI: 10.18869/acadpub.jsdp.13.4.109]

تا سی را نمایش خواهد داد. همان‌طور که در شکل مشخص است، میزان بسته‌های رسیده در پروتکل AODV برابر ۹ الی ۲۳ درصد و در پروتکل MAODV به‌طور تقریبی برابر۱۴ الی ۳۱ درصد است.



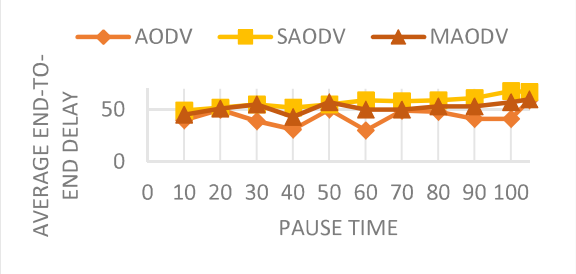
(شکل-۶): نمودار نرخ بسته‌های رسیده
(Figure- 6): Arrival packet rate

میانگین تأخیر End-to-End

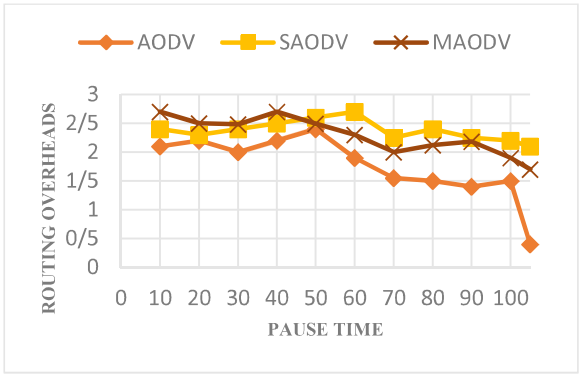
تأخیر انتهابه‌انتهای یک بسته به زمان رسیدن بسته از مبدأ تا مقصد گفته می‌شود. میانگین تأخیر انتهابه‌انتها برابر متوسط زمان رسیدن بسته‌ها از مبدأ تا مقصد در شبکه است. در این سناریو تعداد گره‌ها پنجاه و وزن درخواستی برای تشکیل مسیر پنج تا بیست در نظر گرفته شده است. شکل (۷) الگوریتم پیشنهادی را با پروتکل AODV وSAODV مقایسه می‌کند. همان‌طور که مشاهده می‌شود، میزان تأخیر برای رسیدن بسته‌ها در پروتکل MAODV بیشتر از پروتکل AODV و کمتر از پروتکل SAODV است؛ زیرا در پروتکل استاندارد AODV مسیر، با کم‌ترین تعداد گام انتخاب می‌شود؛ و در پروتکل SAODV به‌علت فرایند رمزنگاری تأخیر قابل توجهی مشاهده می‌شود؛ درحالی‌که در پروتکل فرض‌شده، هر مسیر کوتاهی الزاماً امن نبوده و باید کمینه وزن مورد نیاز را تأمین کند. با اجرای مداوم پروتکل و انتخاب وزن بهینه مقدار تأخیر پروتکل MAODV به AODV استاندارد بسیار نزدیک می‌شود.

میزان سربار مسیریابی

در این بخش سربار مسیریابی پروتکل MAODV بر اساس بایت مورد ارزیابی قرار خواهد گرفت؛ و نتیجه آن با پروتکل AODV و SAODV[1] مقایسه خواهد شد. شبیه‌سازی با پنجاه گره و وزن‌های درخواستی مختلف پنج تا بیست در نظر گرفته شده است. در ابتدای اجرای شبیه‌سازی به دلیل اینکه گره‌ها از وزن مناسب برای یافتن مقصد مورد نظر اطلاعات کمتری دارند، سربار مسیریابی بیشتری خواهیم داشت؛ اما با گذشت زمان اجرا و روزآمدکردن جداول route cache، میزان سربار مسیریابی بهبود می‌یابد و نزدیک به پروتکل استاندارد AODV می‌شود. عدم استفاده از روش رمزنگاری، موجب مشاهدهٔ سربار کمتری نسبت به SAODV می‌شود.



(شکل-۷): نمودار میانگین تأخیر نقطه به نقطه
(Figure- 7): Average delay point to point



(شکل-۸):نمودار سربار مسیریابی بر حسب بایت
(Figure- 8): Routing overhead bytes

همان‌طورکه جدول (۴) نشان می‌دهد، روش مهار جغرافیایی به یک سازوکار تصدیق پخش همگانی کلی نیاز دارد که باعث افزایش حجم محاسبات و سربار خواهد شد. دستیابی به اطلاعات محل قرارگیری گره‌ها در شبکه نیز میزان سربار شبکه را افزایش می‌دهد. روش مهار زمانی نیازمند یک همگام‌سازی زمانی قوی ns است، که فراهم‌کردن آن در شبکه بسیار دشوار است. روش EDWA بر مبنای کوتاه‌ترین مسیر بین فرستنده و گیرنده، بنا نهاده شده است، که برای محاسبهٔ فاصله، نیاز به GPS دارد. روش RTT زمان تأخیر برای هر گام را به‌عنوان نشان‌گر حمله معرفی می‌کند؛ با توجه به این‌که ممکن است در مواردی ازدحام بسته‌ها منجر به تأخیر شود، این روش منجر به تولید زیاد نرخ خطای مثبت خواهد بود. روش چالش/ پاسخ از الگوریتم محدودکنندهٔ فاصله، فاصله میان دو گره را بررسی می‌کند، که آیا این دو گره درحقیقت با یک‌دیگر همسایه هستند یا خیر. این روش نیازمند سخت‌افزار ویژه است که به یک بیت

[1] Secure AODV

[DOI: 10.18869/acadpub.jsdp.13.4.109]

چالش فرستاده‌شده به‌صورت آنی پاسخ دهد. روش LiteWorp به هیچ سخت‌افزار اضافی نیاز ندارد؛ ولی این روش تنها در شبکه‌های ایستا کاربرد دارد و برای شبکه‌های متحرک کارآمد نخواهد بود. در راه‌کار پیشنهادی که بدون نیاز به موقعیت‌سنج، هماهنگ‌کننده زمانی، سخت‌افزار اضافی و بدون استفاده از تکنیک‌های رمزنگاری است، با توزیع وزن به گره‌ها و بیان حداقل وزن درخواستی برای ایجاد ترافیک، پروتکل AODV در برابر حمله کرم‌چاله مقاوم‌تر شده است. جدول (۴) راه‌کار پیشنهادی را با روش‌های دیگر مقایسه خواهد کرد.

(جدول-۴): مقایسه روش‌های مختلف مقابله با حمله کرم‌چاله
(Table-4): Comparison between Prevention of Wormhole attack

متد	نیاز به موقعیت سنج	نیاز به همگام-سازی زمان	نیازهای خاص	کاربرد
مهار جغرافیایی	دارد	ضعیف (میلی-ثانیه)	ندارد	شناسایی و جلوگیری کرم‌چاله
مهار زمانی	ندارد	قوی (نانوثانیه)	ندارد	شناسایی و جلوگیری کرم‌چاله
EDWA	دارد	ندارد	ندارد	شناسایی کرم‌چاله
RTT	ندارد	ندارد	ندارد	جلوگیری کرم‌چاله
LiteWorp	ندارد	ندارد	ندارد	جلوگیری در شبکه‌های ایستا
چالش/ پاسخ	ندارد	ندارد	سخت‌افزاری برای پاسخ آنی	جلوگیری کرم‌چاله
MAODV	ندارد	ندارد	ندارد	جلوگیری کرم‌چاله

۶- نتیجه‌گیری

حملهٔ کرم چاله یکی از سخت‌ترین حملات **DOS** است که در لایهٔ شبکه رخ می‌دهد. این حمله می‌تواند روی مسیریابی داده‌ها و تراکم داده‌ها تأثیرگذار باشد. در این حمله مهاجمان با ایجاد یک تونل بین گره‌های بدانندیش که از قبل با یکدیگر تبانی کرده‌اند، گرهٔ فرستنده را تحریک می‌کنند که مسیری کوتاه‌تر و سریع‌تر برای ارسال داده‌ها، نسبت به مسیر عادی وجود دارد، که درواقع این یک مسیر جعلی است. تمامی پروتکل‌هایی که از کوتاه‌ترین مسیر و یا سریع‌ترین بسته رسیده به مقصد استفاده می‌کنند، در برابر این حمله آسیب‌پذیر خواهند بود. در راه‌کار ارائه‌شده که بدون نیاز به سخت‌افزار خاص و همگام‌سازی زمان است، هر گره دارای یک وزن است، که مجموع وزن‌ها در شبکه برابر صد خواهد بود. هرگاه فرستنده قصد ارسال ترافیک به مقصد را داشته باشد، در کنار بسته RREQ در قسمت حداقل وزن، مقدار وزن درخواستی خود را وارد می‌کند که با توجه به اهمیت و ارزش بستهٔ ارسالی می‌تواند تعیین کند، حداقل چند گره در فرایند کشف مسیر باید حضور داشته باشند. در آینده سعی خواهد شد درخواست وزن بهینه مسیر توسط مبدأ انجام گیرد. مقدار وزن درخواستی رمز خواهد شد که این مقدار فقط توسط گره مقصد قابل بازگشایی خواهد بود و گره بداندیش که در فاصلهٔ یک‌قدمی مقصد قرار دارد، نخواهد توانست وزن را تغییر دهد؛ زیرا هیچ ایده‌ای در مورد وزن پیشنهادی نخواهد داشت. همچنین در آینده راه‌کار پیشنهادی روی پروتکل‌های مسیریابی که در برابر کرم چاله آسیب‌پذیر هستند از جمله DSR, ARAN و DSDV اعمال‌شده و نتایج به‌دست‌آمده با راه‌کار پیشنهادی مقایسه خواهد شد.

۷- مراجع 7-Refrences

[1] A. F. S. D. Vandana, "Evaluation of Impact of wormhole Attack on AODV," *International journal of advanced Networking and Applications*, vol. 4, no. 4, pp. 1652-1656, 2013.

[2] A. F. S. D. Vandana, "Wormhole attack Detection Using Hop Latency and Adjoining Node Analysis in MANET," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 3, 2013.

[3] B. H. Capkun, "SECTOR: Secure Tracking of Node encounters in Multihop Wireless Networks," *First ACM Workshop on Security of Ad-hoc and Sensor Networks*, pp. 21-32, 2003.

[4] B. Patel, "A survey on detecting wormhole attack in MANET", *Int. Journal of Engineering Research and Applications*, pp.653-656, 2014.

[DOI: 10.18869/acadpub.jsdp.13.4.109]

[5] J. Perrig, "Wormhole Attacks in Wireless Networks," *IEEE Journal on Selected Areas in Communications,* vol. 24, pp. 370-380, 2006.

[6] K. H. N.S Raote, "Approaches toward Mitigating Wormhole Attack in Wireless Ad-hoc Network," *International Journal of Advanced Engineering Sciences and Technologies*

[7] *(IJAEST),* vol. 2, no. 2, pp. 172-175, 2011.

[8] I. Khalil, and K. S. Bagchi, "A Lightweight Countermeasure for the wormhole Attack in Multihop Wireless Networks," In *Proceedings of DSN,* 2005.

[9] L. Wong, "DELPHI: Wormhole Detection Mechanism for adhoc Wireless Networks," in *Proceeding of International Symposium on Wireless Pervasive Computing*, 2006, pp. 6-11.

[10] L. X. H. Phuong Van Tran, "An efficient Mechanism to Detect Wormhole Attacks in Wireless ad-hoc Networks," *IEEE Consumer Communications and Networking Conference,* 2007.

[11] V. Mahajan, M, Natu, and A, Sethi "Analysis of wormhole Intrusion Attacks in MANETS," in *IEEE Military Communications Conference,* 2008.

[12] V. G. Supriya Tayal, "A Survey of Attacks on MANET Routing Protocols," *International Journal of Innovative Research in Science, Engineering and Technology,* vol. 2, no. 6, pp. 2280-2286, 2013.

[13] W. Wang, "EDWA: End-to-End detection of wormhole attack in wireless adhoc networks," in *Computer Software and Applications Conference*, 2007. *31st Annual International, IEEE*, 2007.

[14] Y. Zhiwei, and J. Zhongyuan, "A Survey on the Evolution of Risk Evaluation for Information Syst-ems Security," in *International Conference on Future Electrical Power and Energy System*, 2012, vol. 17, pp. 1288–1294.

فرید محمدی تحصیلات خود را در مقطع کارشناسی مهندسی کامپیوتر در دانشکده مهندسی برق کامپیوتر دانشگاه زنجان در سال ۱۳۹۰ در دانشگاه زنجان و در رشته فناوری اطلاعات و مدرک کارشناسی ارشد را در رشته امنیت اطلاعات در دانشگاه بین-الملل تهران در سال ۱۳۹۲ اخذ کرد. از ایشان یک مقاله در در هفتمین همایش بین‌المللی ارتباطات در سال ۱۳۹۲ به چاپ رسیده است.

نشانی رایانامه ایشان عبارت است از:
Fa.mohammadi1988@yahoo.com

حسین قرائی تحصیلات خود را در مقطع کارشناسی مهندسی برق-الکترونیک در دانشکده مهندسی برق دانشگاه خواجه نصیرالدین طوسی در سال ۱۳۷۷ به اتمام رساند. مقاطع کارشناسی ارشد و دکتری مهندسی برق - الکترونیک در دانشکده مهندسی برق دانشگاه تربیت مدرس را به ترتیب در سال‌های ۱۳۷۹ و ۱۳۸۸ تکمیل کرد. از سال ۱۳۸۱ تاکنون عضو هیئت علمی مرکز تحقیقات مخابرات است و زمینه‌های پژوهشی مورد علاقه ایشان طراحی مدارات VLSI دیجیتال، آنالوگ و سیگنال مختلط، پردازش سیگنال‌های دیجیتال، سامانه‌های تشخیص و پیش‌گیری از نفوذ و مرکز عملیات امنیت و اجزای آن است.

نشانی رایانامه ایشان عبارت است از:
gharaee@itrc.ac.ir

[DOI: 10.18869/acadpub.jsdp.13.4.109]