Sicherheit in WSNs – Node Replication Angriffe

Johannes Schlicker
Betreuerin: Corinna Schmitt
Seminar Innovative Sensorknoten: Betrieb, Netze und Anwendungen SS2011
Lehrstuhl Netzarchitekturen und Netzdienste, Lehrstuhl Betriebssysteme und Systemarchitektur
Fakultät für Informatik, Technische Universität München
Email: schlicke@in.tum.de

KURZFASSUNG

In dieser Ausarbeitung wird beschrieben, wie Angriffe auf drahtlose Sensornetzwerke abgewehrt werden können. Es gibt verschiedene Verfahren, solche Angriffe zu erkennen. Im Folgenden wird erklärt, wie „Node Replication Attacks“ ablaufen und sowohl auf verbreitete Verfahren wie „Randomized Multicast“ [1] und „Line-Selected Multicast“ [1], aber auch detaillierter auf ein neues Verfahren eingegangen [2]. Dieses neue Verfahren versucht die Nachteile und Einschränkungen, wie hohen Rechenaufwand und immer noch vorhandene Sicherheitslücken, etablierter Verfahren zu verringern.

Schlüsselworte

Sensornetzwerk, Angriff, Node Replication Attack, Sicherheitsprotokoll

1. EINLEITUNG

Drahtlose Sensornetzwerke (WSNs – Wireless Sensor Networks) werden heute hauptsächlich zur Überwachung von eingegrenzten Gebieten eingesetzt. Der Vorteil der Verwendung von Sensornetzwetzen ist die Skalierbarkeit und einfache Erweiterbarkeit. Neue Sensorknoten (oder auch nur Knoten genannt) können ohne Änderungen an der Konfiguration des Netzwerks als „neue Generation“ hinzugefügt werden. Ein wunder Punkt vieler Protokolle zur Abwehr von Angriffen ist der Vorgang der Erweiterung des Netzwerks, da nicht sichergestellt werden kann, dass keine feindlichen Knoten ins Netzwerk aufgenommen werden [1]. Feindliche Knoten sind im Falle von Node Replication Angriffen Kopien von Sensorknoten, die nach erfolgreichem Einklinken ins Sensornetz, Informationen auslesen oder sogar Schaden anrichten können.

2. ANGRIFFE AUF SENSORNETZE

Bei der hier behandelten Bedrohung für Sensornetzwerke – den Node Replication Attacks – gibt es verschiedene Angriffspunkte, die im Folgenden kurz erläutert werden. Abbildung 1 zeigt schematisch den Überblick über ein Sensornetzwerk, welches durch neue Knoten erweitert wurde. Ein Knoten wurde kompromittiert und könnte sowohl ein Klon eines bereits vorhanden, als auch eines neu eingesetzten Knotens sein kann. Aus diesem Szenario ergeben sich zwei Angriffspunkte: Klonen eines vorhandenen Knotens und Klonen eines neuen Knotens [1][2]. Beide Vorgehen müssen von einem Sicherheitsprotokoll abgefangen werden können.

Um Abwehrmechanismen verstehen zu können, muss zunächst erklärt werden, wie Angriffe ablaufen.

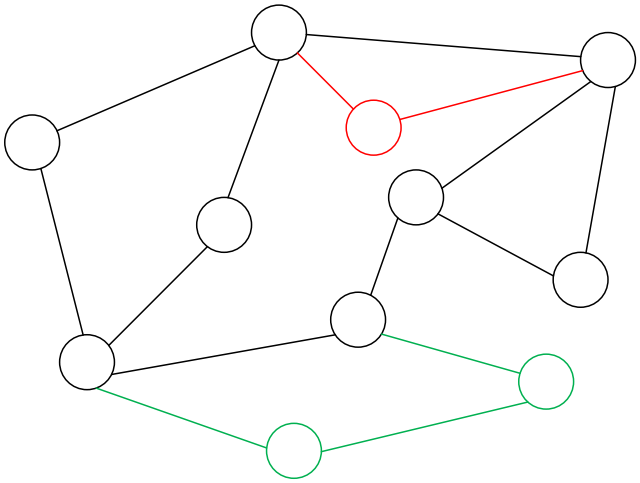

Abbildung 1: Aufbau eines Sensornetzwerks nach dem Hinzufügen neuer Knoten

2.1 Ablauf eines Angriffs

Der erste Schritt bei einem Node Replication Attack ist das Klonen eines Sensorknotens. Dabei werden sowohl Kommunikationsprotokolle, als auch eventuell vorhandene Sicherheitsmechanismen und -zertifikate kopiert [2]. Zusätzlich ist der bösartige Knoten aber natürlich mit Schadsoftware bestückt. Eine wichtige – wenn auch offensichtliche – Eigenschaft duplizierter Knoten ist, dass die Signatur dieses Knotens doppelt im Sensornetz vorhanden ist. Die Signatur ist meistens ein Schlüssel, der zur Authentifizierung gegenüber dem Sensornetz dient.

Des Weiteren ist in jedem Sensornetzwerk ein Protokoll vorhanden, welches die Erweiterung eines Netzwerkes definiert. Ein kompromittierter Knoten, muss diesen Protokolldurchlauf aber nicht zwingend durchführen, da er vorgeben kann, bereits Mitglied einer vorher eingesetzten Generation zu sein. Individuelle Schlüssel, die für die Kommunikation verwendet werden, müssen in diesem Fall anderweitig besorgt werden.

Ist ein neuer (noch nicht im Netzwerk registrierter) Knoten der Angriffspunkt, muss ein Mechanismus durchlaufen werden, der den geklonten Knoten als legitim eingesetzten Knoten identifiziert.

Das technische Ziel eines jeden Angriffs auf ein Sensornetzwerk ist das erlangen eines Schlüssels, der für die Kommunikation mit anderen Knoten unerlässlich ist. Dieser Schlüssel dient dem Ver- und Entschlüsseln zu sendender oder empfangener Nachrichten.

doi: 10.2313/NET-2011-07-1_13

Welches Schlüsselprotokoll verwendet wird, soll hier nicht genauer beschrieben werden. Es sind jedoch sowohl symmetrische als auch asymmetrische Verfahren denkbar.

2.2 Annahmen

Um Angriffe abwehren zu können, müssen zusätzlich Annahmen getroffen werden. Es wird beim Entwickeln eines Protokolls davon ausgegangen, dass nur ein bestimmter Prozentsatz von Knoten kompromittiert wird. Wird die Kontrolle über jeden einzelnen Knoten übernommen, können Sicherheitsmechanismen einfach ausgeschaltet werden [1]. Außerdem wird die Annahme getroffen, dass feindliche Knoten den Protokollen weiterhin folgen und Informationen ausschließlich von kompromittierten Knoten lesen können [1]. Des Weiteren hat jeder Knoten, der einmal eingesetzt wurde eine feste, unveränderbare Position.

3. ZENTRALISIERTE PROTOKOLLE

Bisherige Ansätze waren meist zentralisiert. Das bedeutet, dass ein zentraler Punkt für die Sicherheit des Sensornetzes verantwortlich war. Auf diese Weise kann relativ einfach sichergestellt werden, dass Angriffe erkannt werden. Eine zentrale Station vergleicht die Signaturen aller Knoten und kann nach Erkennen einer doppelt vorhandenen ID das Sensornetz sofort warnen. Dies hat jedoch mehrere entscheidende Nachteile.

Zum einen entsteht ein hoher Kommunikationsaufwand, da alle Knoten im Netzwerk Informationen zu einem zentralen Knoten weiterleiten müssen. Basierend auf der Tatsache, dass nicht alle Knoten direkt miteinander verbunden sind, müssen Sensoren nicht nur ihre eigenen Daten versenden, sondern zusätzlich auch die von weiter entfernten Sensoren weiterleiten. Ausgehend von einer durchschnittlichen Pfadlänge von $O(\sqrt{n})$ zur Base Station, wobei n die Anzahl der Knoten ist, müssen $O(n\sqrt{n})$ Übertragungen vorgenommen werden, bis alle Informationen versendet wurden [1]. Diesem Kommunikationsaufwand ist ein großes Sensornetzwerk nicht gewachsen, da die begrenzten Energieressourcen vor Allem bei Knoten, die sehr viel senden müssen, sehr schnell erschöpft sind.

Ein weiterer Nachteil ist, dass die Base Station zum „Single Point of Failure“ wird. Das bedeutet, dass ein Ausfall oder die Kompromittierung dieser Base Station das komplette Sensornetzwerk lahmlegen und somit zu einem beliebten Angriffsziel werden kann [1].

Abgesehen von diesen Nachteilen, besteht zusätzlich die Gefahr, dass verteilte Angriffe nicht erkannt werden. Das bedeutet, dass ein Angriff, der auf mehrere Knoten gleichzeitig abzielt und somit möglicherweise einen Teil eines Netzwerkes abtrennt, nicht entdeckt werden kann, da ein kompromittierter Verbindungsknoten Informationen verfälscht weitergibt.

4. DEZENTRALISIERTE PROTOKOLLE

In diesem Abschnitt werden verschiedene Protokolle erklärt, die die beiden in Abschnitt 2 beschriebenen Angriffspunkte schließen sollen und den Annahmen aus Abschnitt 2.2 folgen. Außerdem werden die Nachteile zentralisierter Ansätze aus Abschnitt 3 minimiert.

Grundsätzlich sind dezentralisierte Protokolle nicht an Base Stations gebunden, was offensichtlich den „Single Point of Failure“ entfernt.

In den folgenden Abschnitten wird zunächst die grundlegende Idee dezentralisierte Protokolle erläutert, um danach zwei etablierte Ansätze – das Randomized Multicast Protokoll und das Line-Selected Multicast Protokoll – genauer zu erläutern. Schließlich wird noch ein neuer Ansatz beschrieben, der den Energiebedarf bei gleicher Sicherheit nochmals reduziert.

4.1 Idee

Die Idee einer dezentralen Lösung ist, dass an die Position geknüpfte Signaturen der einzelnen Knoten nicht an eine zentrale Station übertragen werden, sondern an andere gleichwertige Knoten im Netzwerk. Wenn jeder Knoten über jeden anderen Bescheid weiß, können – unter der Voraussetzung, dass die Broadcasts jeden Sensor erreichen – alle doppelten Positionsangaben erkannt werden. Allerdings ist dabei der Kommunikationsaufwand im Bereich von $O(n^2)$ noch höher, als bei der zentralisierten Herangehensweise [1].

Um diesen Ansatz zu verbessern, werden die Positions- und Signaturnachrichten nur an eine bestimmte Gruppe von Knoten gesendet. Die Knoten werden nach einem Algorithmus basierend auf der Signatur ausgewählt, um sicherzustellen, dass geklonte Knoten ihre Positionsinformation an die gleiche Gruppe wie auch der kompromittierte Sensor weiterleiten. Das birgt allerdings wieder eine entscheidende Sicherheitslücke: Kennt der Angreifer die Signatur eines Knotens und den Algorithmus, mit dem die sogenannten „Witness Nodes“ ausgewählt werden, kann er versuchen auch alle diese Witness Nodes zu übernehmen, um einen neuen Sensor unbemerkt einzuschleusen [1].

Das Randomized Multicast Protokoll und das Line-Selected Multicast Protokoll gehen daher noch einen Schritt weiter und setzen als Witness Nodes nicht direkte Nachbarknoten ein und randomisieren außerdem die Auswahl.

4.2 Das Randomized Multicast Protokoll

Das Randomized Multicast Protokoll setzte voraus, dass jeder Knoten im Sensornetz seine Position kennt und, dass jeder einzelne Sensor Signaturen erstellen kann, die ihn eindeutig identifizieren [1][2].

Die Idee des Protokolls ist das Ausnutzen des sogenannten „Geburtstags-Paradoxon“ [3]. Ein kurzes Beispiel erklärt dieses Problem am besten: „Sind mindestens 23 Personen in einem Raum, so ist die Wahrscheinlichkeit, dass zwei oder mehr am selben Tag im Jahr Geburtstag haben, größer als 50%.“ Bei 50 Personen liegt die Wahrscheinlichkeit sogar bei über 97%. Die Tatsache dieser relativ geringen Menge an Personen wird hier auf das Sensornetz übertragen, wobei sich hier natürlich die Frage stellt, wie viele zufällige Witness Nodes von Nöten sind, um mit hoher Wahrscheinlichkeit einen Node Replication Angriff zu erkennen.

doi: 10.2313/NET-2011-07-1_13

4.2.1 Beschreibung des Protokolls

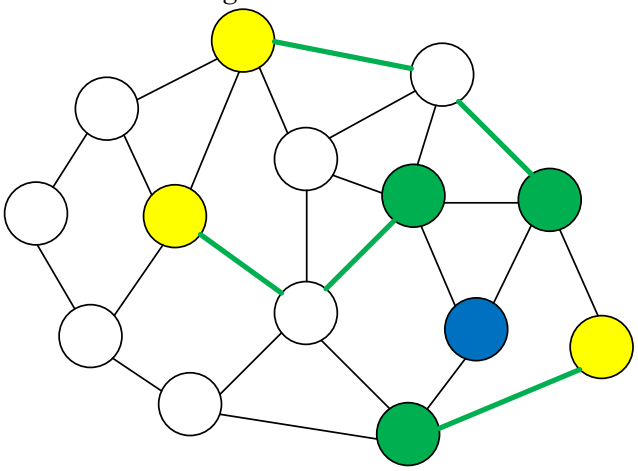

Abbildung 2: Randomized Multicast Protokoll

Die angesprochene Erweiterung der dezentralisierten Idee besteht darin, dass Knoten nicht direkte Nachbarn als Witness Nodes wählen, sondern zufällige Knoten aus dem Sensornetz. In diesem Fall trifft die Wahl der zufälligen Knoten aber nicht der Knoten, dessen Authentizität überprüft werden soll, sondern dessen direkte Nachbarn. In Abbildung 2 ist dies schematisch dargestellt. Der blaue Knoten sendet seine Signatur und seine Position an alle direkten Nachbarn, die grün dargestellt sind. Diese Nachbarn suchen dann zufällig eine bestimmte Anzahl an geographischen Punkten im Netzwerk (hier im Beispiel ist die Anzahl nur 1). Durch geographisches Routing werden dann die Knoten, die diesen Punkten am nächsten liegen, als Zeugenknoten ausgewählt und die Signatur/Position-Kombination an diese weitergeleitet. Am Beispiel des Geburtstagsparadoxon kann man erkennen, dass eine relativ geringe Anzahl an Zeugenknoten ausreicht, um sicherzustellen, dass – trotz der zufälligen Auswahl der Zeugenknoten – mit sehr hoher Wahrscheinlichkeit eine doppelt vorhandene Signatur mit unterschiedlichen Positionsangaben entdeckt wird.

Ein weiterer Punkt, der im Protokoll beachtet werden muss, ist die Integrität und vor Allem Authentizität der Nachrichten, die versendet werden. Um diese zu gewährleisten liegt dem Sensornetz ein Signaturverfahren zugrunde. Jeder Knoten kann seine Identität mit einem privaten Schlüssel garantieren. Die Nachrichten die versendet werden, haben folgendes Format:

$$\langle ID, l, \{H(ID, l)\}_{KP} \rangle$$

Dabei ist *ID* die Signatur des sendenden Knotens, *l* die Positionsangabe und $\{H(ID, l)\}_{KP}$ ein Hashwert über den Nachrichteninhalt, der mit dem privaten Schlüssel *KP* des Knotens verschlüsselt ist. Dieser Hashwert dient dabei der Sicherstellung der Integrität der Nachricht. Ist die Verschlüsselung des Hashwertes falsch, so schlägt die Integritätsprüfung fehl und die Nachricht wird verworfen [1].

4.2.2 Sicherheit und Effizienz

Das Maß an Sicherheit des Protokolls ist gleichzusetzen mit der Wahrscheinlichkeit, dass erkannt wird, wenn mindestens zwei Knoten mit der gleichen Signatur zwei verschiedene Positionen haben. Ohne hier genauer auf die Herleitung einzugehen stößt man bei der Analyse des Problems auf folgende Formel, die genau diese Wahrscheinlichkeit angibt:

$$P \geq 1 - e^{\frac{-p^2 d^2 g^2 L(L-1)}{n \quad 2}}$$

Dabei ist n die Anzahl der Knoten im Sensornetz, p die Wahrscheinlichkeit, dass ein Nachbarknoten Positionsinformationen doppelt weiterleitet, g die Anzahl der Zeugenknoten, d der Grad jedes einzelnen Knotens (also die durchschnittliche Entfernung) und L die Anzahl der Klone des kompromittierten Knotens [1][3].

Spielt man dieses Szenario mit realistische Zahlen durch, so kommt man mit $n = 10000$, $g = 100$, $d = 20$ und $p = 0{,}05$ auf eine Wahrscheinlichkeit von 63%, wenn ein Knoten einmal geklont wurde, bei nur zwei Klonen liegt die Wahrscheinlichkeit, dass mindestens einer erkannt wird schon bei über 95%.

Problematisch bei diesem Protokoll ist allerdings wieder der Kommunkationsaufwand von $O(n^2)$. In der Realität wird allerdings nicht immer eine perfekte Erkennung gefordert. Oftmals reicht es aus, wenn nicht jeder einzelne Klon erkannt wird, sondern das System erst bei mehreren Klonen alarmiert wird. Dafür kann beispielsweise die Anzahl der Zeugenknoten reduziert werden [1]. Durch weitere kleinere Optimierungen kann die Anzahl der Nachrichten pro Knoten auf $O(\sqrt{n}\, p\, g)$ reduziert werden.

4.3 Das Line-Selected Multicast Protokoll

Das Line-Selected Multicast Protokoll ist eine Erweiterung des Randomized Multicast Protokolls [2]. Ziel dieser Erweiterung ist es, den Kommunikationsaufwand weiter zu reduzieren, da dieser hohe Energiekosten erzeugt und somit die Lebensdauer eines Sensornetzwerks beträchtlich mindert.
Wie man in Abbildung 2 sehen kann, kreuzen die Nachrichten der Nachbarknoten mehrere unbeteiligte Knoten (grüne Pfade). Das hier beschriebene Protokoll nutzt die Tatsache aus, dass einzelne Sensoren im Netzwerk sowohl als Router, als auch als Sensor fungieren und somit die Nachrichten, die sie in ihrer Funktion als Router eigentlich nur weiterleiten, auch speichern können [1]. Somit ist die Signatur-/Positionsinformation eines Knotens nicht nur auf den Zeugenknoten, sondern auch auf jedem Knoten auf dem Weg zu den Zeugenknoten vorhanden, ohne den Kommunikationsaufwand dabei erhöht zu haben.

4.3.1 Beschreibung des Protokolls

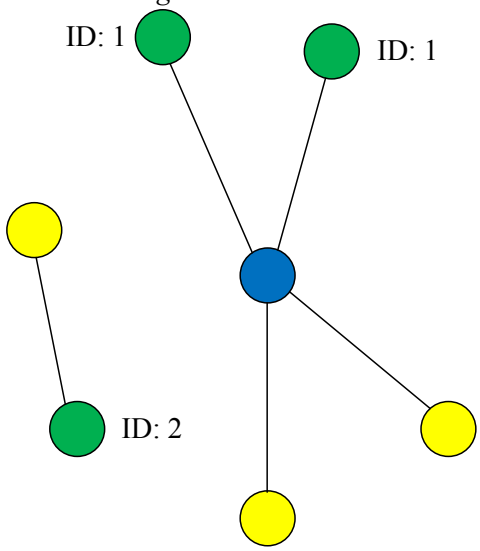

Abbildung 3: Line Selected Multicast Protokoll

doi: 10.2313/NET-2011-07-1_13

Das in Abbildung 3 (zur Vereinfachung wurden die Nachbarknoten nicht mit in die Zeichnung aufgenommen) dargestellte Szenario ist, dass die oberen beiden grünen Knoten doppelt vorhanden sind. Einer dieser beiden Knoten muss also ein bösartiger Klon sein. Bei Anwendungen des Randomized Multicast Protokolls würden Signatur- und Positionsinformationen zu jeweils einem der beiden grünen Knoten nur bei den beiden unteren gelben Knoten vorliegen. Dass die Knoten doppelt vorliegen würde man also nicht erkennen.

Man kann aber erkennen, dass sich die Pfade über die die Nachrichten verschickt wurden, bei dem blauen Knoten kreuzen. Wird hier das Line-Selected Multicast Protokoll verwendet, ist der Ablauf folgender: Die Nachricht des linken Knotens (mit ID 1) wird an den rechten Zeugenknoten (gelb) versendet und auf jedem Knoten, der auf dem Pfad dorthin liegt zwischengespeichert. Sendet der rechte Knoten (mit ID 1) eine Nachricht mit gleicher Signatur aber anderen Positionsinformationen, so muss diese Nachricht nicht bis zum Zeugenknoten weitergeleitet werden, da die Kollision (also das Auftreten verschiedener Positionsinformationen zu gleichen IDs) bereits beim blauen Knoten erkannt wird.

4.3.2 Sicherheit und Effizienz

Um die beiden Protokolle genauer vergleichen zu können, wird auch hier kurz auf die Fehlererkennungswahrscheinlichkeit und den Kommunikationsaufwand eingegangen.

Beim Line-Selected Multicast Protokoll kommt wiederum die Lösung eines anderen Problems zum Einsatz. Sylvesters sogenanntes „Four-Point Problem“ besagt, dass die Wahrscheinlichkeit, dass vier zufällig in einem Kreis gewählte Punkte eine konvexe Hülle bilden, $\frac{35}{12\pi^2}$ ist. Ausgehend von dieser Wahrscheinlichkeit lässt sich berechnen, wie wahrscheinlich eine Kreuzung von Pfaden und somit auch wie wahrscheinlich ein früheres Erkennen einer Kollision ist.

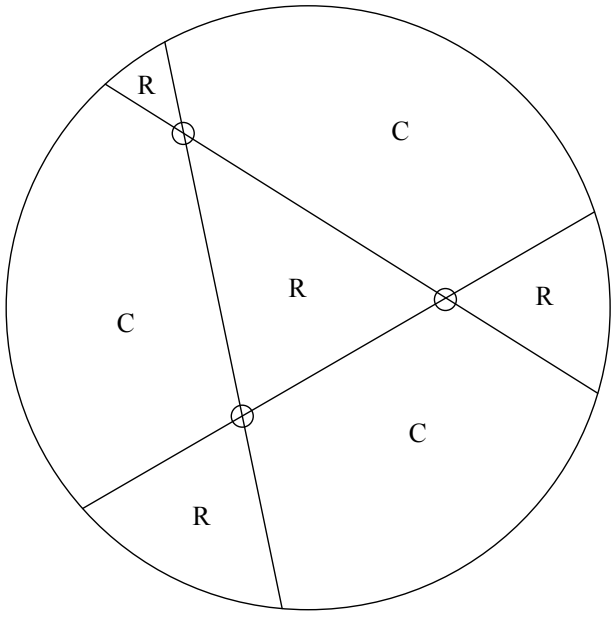

Abbildung 4: Das "Four-Point Problem"

Wählt man beispielsweise drei beliebige Punkte in einer kreisförmigen Fläche, berechnet sich die Wahrscheinlichkeit, dass ein vierter Punkt eine Kreuzung verursacht wie folgt [1][4]:

$$P = \frac{1}{3}\left(1 - \frac{35}{12\pi^2}\right) \approx 0{,}235$$

Das Ergebnis der Klammer ist dabei die Wahrscheinlichkeit, dass der vierte gesetzte Punkt eine Form mit einer Mulde produziert (im Gegensatz zu einer konvexen Hülle). Der vierte Punkt muss dabei in eine in Abbildung 4 mit „R“ beschriftete Region fallen. Um ausgehend vom Punkt oben links eine Kreuzung zu erreichen, muss der vierte Punkt als in den unteren mit „C“ beschrifteten Bereich fallen [1].

Im Falle eines Sensornetzes ist die Annahme allerdings komplizierter. Man geht davon aus, dass alle Pfade mehr oder weniger radial von der Mitte des Kreises nach außen laufen [1]. Diese Annahme beruht auf der Tatsache, dass die Pfade von einem Knoten – oder besser – dessen Nachbarn ausgehen. Durch Simulationen kann man berechnen, dass die Wahrscheinlichkeit, dass sich mindestens zwei Pfade überschneiden bei nur fünf Pfaden bei ungefähr 95% liegt [1/4].

Ausgehend davon, dass jeder Pfad eine Länge von $O(\sqrt{n})$ hat, liegt der Kommunkationsaufwand nur bei $O(n\sqrt{n})$, was eine wesentliche Verbesserung darstellt.

5. FORSCHUNG VON BEKARA UND LAURENT-MAKNAVICIUS

Um den Kommunikationsaufwand der vorigen Protokolle zu minimieren haben Chakib Bekara und Maryline Laurent-Maknavicius ein neues Protokoll entwickelt. Dieses Protokoll basiert auf einer anderen Idee. Deshalb gibt es zusätzliche Annahmen, die es sicherzustellen gilt.

So existiert eine zentrale Base Station, die nicht kompromittiert werden kann. Diese Annahme ist insofern denkbar, da die Basisstation nur zum Einsetzen neuer Generationen verwendet wird und in dieser Zeit überwacht und ansonsten abgeschaltet werden kann. Diese Basisstation setzte einzelne Knoten in fortlaufenden Generationen ein. Außerdem besitzt jeder einzelne Knoten Kenntnis über die Nummer der höchsten Generation.

Des Weiteren verwendet das Protokoll ein symmetrisches Verschlüsselungsverfahren. Sendet ein Knoten mit einem validen Schlüssel, gilt er als nicht kompromittiert. Die Zeit, die für den Schlüsselaustausch mit den direkten Nachbarn aufgebracht wird liegt unter T_{est}. Ein Angreifer braucht allerdings $T_{comp} > T_{est}$, um einen Knoten zu kompromittieren [2]. Diese Annahme ist realistisch, da zusätzlich zu der Zeit, die der Schlüsselaustausch in Anspruch nimmt, noch der Klonvorgang vorgenommen werden muss.

Zuletzt muss sichergestellt sein, dass die internen Uhren aller Knoten mit der Basisstation synchron sind. Dazu wird ein authentifizierter „Beacon“ verwendet [2] – eine kleine Nachricht, die in regelmäßigen Abständen die Uhren abgleicht.

5.1 Aufbau und Erweiterung des Netzwerks

Wie bereits erwähnt werden neue Sensoren in Gruppen eingesetzt. Eine neue Gruppe wird als Generation bezeichnet. Der erste Schritt, den jeder Knoten ausführt, ist ein Schlüsselaustausch mit seinen direkten Nachbarknoten. Erhält ein Sensor also eine Anfrage zum Schlüsselaustausch, geht dieser davon aus, muss die Anfrage von einem Knoten einer Generation größer oder gleich der eigenen Generation stammen, da nur neue Knoten Schlüssel anfragen. Ein erstes Aussortieren maligner Knoten kann also bereits hier getroffen werden. Denn wenn ein bereits etablierter Knoten geklont wird und einen Schlüssel anfragt, stammt die

doi: 10.2313/NET-2011-07-1_13

Anfrage von einer alten Generation und kann als bösartig identifiziert werden. Auf die Kenntnis über die höchste Generation wird später in Abschnitt 5.3 eingegangen [2].

5.2 Beschreibung des Protokolls

5.2.1 Schritt 1 – Hello Nachricht

Wenn ein Knoten *u* eingesetzt wird, sendet dieser zunächste eine „Hello“ Nachricht an alle direkten Nachbarn:

$$\langle Hello, j, Id_u, N_u \rangle$$

N ist dabei ein zufällig steigender Wert, der garantiert, dass keine bereits versendeten Nachrichten wiederverwendet werden können. *j* beschreibt die Generation des sendenden Knotens. *Id* ist eine eindeutige Signatur.

5.2.2 Schritt 2 – Schlüsselgenerierung

Jeder Empfänger einer solchen *Hello* Nachricht führt nun folgende Überprüfung durch:

Zunächst wird überprüft, ob $j = i + 1$ ist. *i* ist dabei die höchste bisher eingesetzte Generation. Schlägt diese Überprüfung fehl, wird die Anfrage verworfen, da der anfragende Knoten bereits eingesetzt ist.

Wenn der überprüfende Knoten zu einer Generation größer als *i* gehört, berechnet er K_{VU} mit Hilfe einer Funktion, die aus dem Hash Wert von $i + 1$ konkateniert mit der *Id* einen Schlüssel berechnet. Dieser Schlüssel wird dann wie folgt verpackt zurückgesendet.

$$\langle z, Id_v, N_v, MAC_{K_{VU}}(z, Id_v, N_v, N_u) \rangle$$

Ist jedoch $j = z = i + 1$ und der von *v* gesetzte Timer, der auf T_{est} gesetzt ist, noch nicht abgelaufen, wird wie im eben beschriebenen Fall vorgegangen. Dieser Fall deckt den Schlüsselaustausch zwischen zwei Knoten ab, die beide der neuesten Generation angehören. Ist der Timer abgelaufen wird die Anfrage verworfen, da davon ausgegangen wird, dass der anfragende Knoten kompromittiert wurde.

5.2.3 Schritt 3 – Schlüsselbestätigung

Im dritten Schritt bestätigt Knoten *u* den Empfang des Schlüssels und kann danach am normalen Netzwerkverkehr teilnehmen. Dazu wird zunächst K_{UV} berechnet. Die Berechnung verwendet dabei die gleiche Funktion wie der Partnerknoten, übergibt der Hashfunktion aber diesmal die Generationsnummer des verifizierenden Knotens konkateniert mit dessen Signatur Id_v. Folgende Nachricht wird anschließend zurückgeschickt [2]:

$$\langle ok, MAC_{K_{UV}}(ok, N_v) \rangle$$

Beide Knoten verifizieren dabei die Authentizität des jeweils anderen Knotens mit Hilfe der Schlüsselberechnungen. Stimmen die Ergebnisse nicht überein liegt ein Fehler vor.

Die paarweise authentifizierten Knoten sind nun sicher legal Teilnehmer und können sich mit den paarweise generierten Schlüsseln Nachrichten senden.

5.3 Berechnen der höchsten Generation

Bei der Verifizierung, ob ein neuer Knoten der höchsten Generation angehört kommt wieder die Basisstation zum Einsatz. Diese setzt dabei ein statisches Verfahren ein, das nach jeder Zeitspanne *T* neue Generationen einsetzt. Dabei wird beim Einsetzen der ersten Generation die Zeit T_1 auf 0 gesetzt. Für jede weitere Generation wird der Timer sukzessive erhöht. Jeder Knoten kennt die aktuelle Zeit, die Zeit T_i, zu der er eingesetzt wurde und die Zeitspanne *T*. Er kann damit verifizieren, dass ein Knoten aus einer neuen Generation stammt, indem er folgende Überprüfung durchführt [2]:

$$0 < t_{current} - (i - 1) * T < T$$

$t_{current}$ ist dabei die aktuelle Zeit und *T* eine von der Basisstation festgesetzte und im Sensornetz publizierte Zeitspanne nach welcher neue Generationen eingesetzt werden. *i* bezeichnet wieder die neueste Generation.

5.4 Sicherheit

Das vorgestellte Protokoll hat allerdings in manchen Fällen noch Probleme, Angreifer zu erkennen.

Der erste Fall ist: Ein neu eingesetzter Knoten ist ein Klon eines neu eingesetzten Knotens. Wenn der Knoten also noch keinen Schlüsselaustausch vorgenommen hat, aber schon geklont ist, läuft der Austausch natürlich innerhalb der erlaubten Zeitspanne T_{est} ab. Abhilfe schafft hierbei die Einführung einer neuen Zeitspanne T_{max}, mit $T_{est} < T_{max} < T_{comp}$. T_{max} ist dabei die maximal zulässige Zeit nach dem Einsetzen einer neuen Generation, in welcher ein Schlüsselaustausch stattfinden darf. Der Schlüsselaustausch darf als nicht nur maximal T_{est} dauern, sondern muss innerhalb eines Zeitraums T_{max} durchgeführt worden sein.

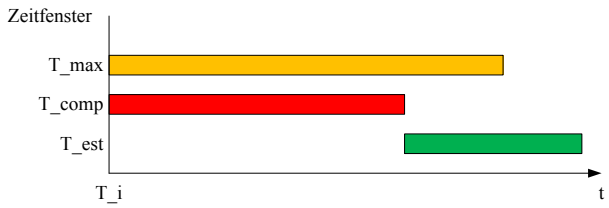

Abbildung 5: Kompromittierung eines neuen, noch nicht eingesetzten Knotens

Abbildung 5 verdeutlicht diesen Fall. T_i ist die Zeit zu der eine neue Generation *i* eingesetzt wurde. Ab diesem Zeitpunkt kann ein Angreifer aktiv werden, da vorher die neuen Knoten noch nicht existieren. Der zweite Balken visualisiert T_{comp}. Der erste Balken T_{max} und der unterste schließlich T_{est}. Ohne T_{max} könnte ein geklonter Knoten bereits vor dem Versuch eines Schlüsselaustausches geklont worden sein und den Austausch legitim vornehmen. Mit T_{max} würde der Gesamtvorgang (Klonen und Schlüssel austauschen) allerdings die maximale Zeitspanne überschreiten und verworfen werden [2]. Diese Lösung kann aber auch dazu führen, dass „langsame“ authentische Knoten verworfen werden.

Ein weiterer schwer zu identifizierender Fall ist, wenn ein alter Knoten geklont wurde und sich Zugang zum Netzwerk über einen neuen Knoten verschaffen will, der legitim dem restlichen Netzwerk beitritt. Der geklonte Knoten hat bis zu diesem Zeitpunkt keine andere Verbindung zum Sensornetz. Um dieses Problem zu lösen müsste man erkennen, dass ein Nachbarknoten eines neuen Knotens existiert, zu dem aber sonst keine sichere Verbindung besteht. Träte so ein Fall auf, könnte man den geklonten Knoten als bösartig erkennen und verwerfen [2].

5.5 Effizienz

Ausgehend davon, dass wie bisher jeder Knoten mit maximal $\sqrt{n}$ weiteren Knoten verbunden ist liegt sowohl der Kommunikationsaufwand, als auch der Speicheraufwand im Bereich von $\sqrt{n}$.

doi: 10.2313/NET-2011-07-1_13

6. ZUSAMMENFASSUNG

Diese Arbeit hat sich mit verschiedenen Methoden der Abwehr von Node Replication Angriffen auseinandergesetzt. Dabei hat sich herausgestellt, dass zentralisierte Herangehensweisen einen „Single Point of Failure“ haben, dessen Kompromittierung das ganze Netzwerk lahmlegen kann. Dezentralisierte Protokolle haben dieses Problem nicht, allerdings ist der Speicher-, Rechen- und Kommunikationsaufwand zur Abwehr von Angriffen um ein vielfaches größer.

Es wurden drei Protokolle vorgestellt, die sich im Maße der Sicherheit und des Aufwands unterscheiden. Vergleicht man die drei genauer beschriebenen Protokolle in Bezug auf die aufwändigsten Operationen Kommunikation und Speichernutzung, so kommt man auf folgendes Ergebnis:

Tabelle 1: Vergleich der Protokolle

	Kommunikation	Speicher
Randomized Multicast	$O(n^2)$	$O(\sqrt{n})$
Line-Selected Multicast	$O(n\sqrt{n})$	$O(\sqrt{n})$
Verbesserung Abschnitt 5	$O(\sqrt{n})$	$O(\sqrt{n})$

Im Vergleich zu zentralisierten Protokollen unterscheiden sich dezentrale Ansätze vor Allem im Speicheraufwand, da einzelne Sensoren keinen Speicher für Node Replication Erkennung belegen müssen.

Man kann nicht sagen, welches Protokoll allgemein das Beste ist. Vielmehr muss beim Einsatz die gewünschte Sicherheit und die Größe des Netzwerks in Betracht gezogen werden, da vor Allem bei dezentralisierten Sicherheitsmechanismen der Aufwand stark von der Anzahl der teilnehmenden Knoten abhängt.

Was sich jedoch zeigt ist, dass die Sicherheit einen sehr engen Zusammenhang zu Kommunikations- und Speicheraufwand hat. Je ressourcenintensiver ein Protokoll ist, desto sicherer scheint es auch zu sein. Das in Abschnitt 5 vorgestellte Protokoll erkauft sich ein hohes Maß an Sicherheit allerdings durch häufigeres Auftreten von Falscherkennungen legitim eingesetzten Sensoren.

7. REFERENZEN

[1] Parno, B., Perrig, A. und Gligor, V. 1993. Distributed Detection of Node Replication Attacks in Sensor Networks

[2] Bekara, C., Laurent-Maknavicius, M. Defending Against Nodes Replication Attacks on Wireless Sensor Networks

[3] Kirchner s., Geschichte des Geburtstagsparadoxon, https://groups.google.com/group/de.sci.mathematik/msg/6ffd85fbcb15647a?hl=de&&pli=1, 2005

[4] Weisstein, E. W. "Sylvester's Four-Point Problem.", http://mathworld.wolfram.com/SylvestersFour-PointProblem.html

doi: 10.2313/NET-2011-07-1_13