ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.10, October 2009, pp.2705−2720
doi: 10.3724/SP.J.1001.2009.03488

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种无线传感器网络分布式安全成簇协议*

余 磊+, 李建中, 骆吉洲

(哈尔滨工业大学 计算机科学与技术学院,黑龙江 哈尔滨 150001)

Distributed Secure Clustering Protocol in Wireless Sensor Networks

YU Lei+, LI Jian-Zhong, LUO Ji-Zhou

(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)

+ Corresponding author: E-mail: yulei2008@hit.edu.cn, http://www.cs.hit.edu.cn

Yu L, Li JZ, Luo JZ. Distributed secure clustering protocol in wireless sensor networks. *Journal of Software*, 2009,20(10):2705−2720. http://www.jos.org.cn/1000-9825/3488.htm

Abstract: The cluster-based hierarchical topology control has been widely studied and applied in wireless sensor networks. But because of the open nature and limited resource of the sensor network, clustering protocols are vulnerable to the misuse and disruption attacks from the adversaries. As a result, the security of the clustering protocols is a basic requirement for its wide application. A distributed secure clustering protocol is proposed, in which the secure network initialization, random number broadcast from the base station and one-way hash chain are used to achieve the resiliency against possible attacks including node personating, cluster-head occupying, malicious cluster-member recruiting and multiple cluster-membership attacks. The security and cost of the protocol are evaluated and the results show the resiliency and efficiency of the proposed protocol.

Key words: wireless sensor networks; security; clustering protocol; one-way hash chain; broadcast authentication

摘 要: 分簇的层次型拓扑控制方式在无线传感器网络中得到广泛研究和应用.然而,由于传感器网络本身所具有的开放性和资源有限的特点,攻击者可以很容易对成簇协议实施有效的误用和破坏.因此,保证成簇协议安全性是其实际广泛应用的基本前提.针对成簇协议所面临的各种安全威胁,提出了一种分布式安全成簇协议,通过网络安全初始化、可信基站的随机数广播和单向密钥链技术来有效地抵御节点伪装和簇首占据攻击、簇成员恶意征募攻击和多重簇成员身份攻击.对协议的安全性和开销进行了广泛和深入的分析,证明了协议的安全性和有效性.

关键词: 无线传感器网络;安全;成簇协议;单向哈希链;广播认证

中图法分类号: TP393 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant No.60533110 (国家自然科学基金); the National Basic Research Program of China under Grant No.2006CB303000 (国家重点基础研究发展计划(973)); the Program for New Century Excellent Talents in University of China under Grant No.NCET-05-0333 (新世纪优秀人才支持计划); the Heilongjiang Province Scientific and Technological Special Fund for Young Scholars of China under Grant No.QC06C033 (黑龙江省青年科技专项资金)

Received 2008-07-07; Revised 2008-10-06; Accepted 2008-10-17; Published online 2009-05-07

1 引 言

无线传感器网络是由大量的计算能力、通信能力、电源能量均有限的传感器节点组成的无线网络.节点被部署到观测区域,感知环境物理信号并以多跳传输的方式向基站发送感知数据.传感器网络正在民用和军事领域得到广泛的应用,例如环境监测、对象跟踪、森林防火、战场态势监控等.

为了节省能量,提高网络通信效率和大规模网络下的可扩展性,无线传感器网络一般以成簇的拓扑组织方式工作完成数据融合、查询、路由、广播等工作.层次性的簇拓扑结构有很多优点,例如:由簇首节点负责簇内数据融合,减少了数据通信量;分簇使得拓扑结构有利于分布式算法的应用,适合大规模部署的网络;由于大部分节点仅与簇首通信且在簇首控制下完成睡眠和通信调度,所以显著地节省了能量,延长了整个网络的生存时间.目前,研究人员针对簇拓扑组织下的通信能量有效性、负载均衡等网络性能优化问题提出了多种有效的成簇算法[1−4],但是这些工作的成簇算法或协议设计没有考虑到传感器网络面临的安全问题.而安全性在无线传感器网络的实际应用中至关重要,这是因为,网络部署环境的开放性和节点的资源有限性使得传感器网络面临着各种严重的攻击威胁.对于成簇这种具有普遍意义和能量有效性的拓扑控制方式,外部攻击和内部攻击都能破坏成簇协议的正常运行而使得基于簇拓扑结构的网络功能失效或者被攻击者误用.因此,协议安全性是成簇拓扑控制实际应用的基本前提.

1.1 成簇协议面临的安全威胁

传感器网络面临的各种威胁可以分为两类:一类是外部攻击,即攻击者通过物理破坏节点硬件,干扰、阻塞、窃听无线信道,向信道中插入伪造数据包等外在方式攻击网络;另一类是内部攻击,即攻击者通过俘获节点,读取节点存储信息,重编程控制节点实施数据包的伪造和篡改、丢包和选择转发、数据窃取等攻击来干扰破坏或误用网络服务.对外部的物理破坏和信道阻塞没有有效的安全机制来抵御它们,但这种蛮力攻击的目的仅是破坏网络正常运行且较容易被发现.本文主要考虑针对成簇协议的内部攻击,即攻击者通过恶意节点实施攻击,目的主要不在于蛮力破坏网络正常运行,而在于伪造数据聚集结果、截获数据或者路由丢包等,而且它很难被直接检测出来,因此也称其为 Stealthy attack.通过这些攻击手段,攻击者可以对成簇协议造成如下 4 类主要的安全威胁:

(1) 节点伪装攻击:如果网络中没有有效的节点认证机制,外部攻击者可以任意伪装成某簇首节点或簇成员节点参与到协议运行过程中,并可以获得簇内控制权、发送错误数据等等.这种攻击属于外部攻击.基于对称密钥的消息认证机制即可有效防止这种攻击.

(2) 簇首占据攻击:恶意节点可在一轮或连续多轮分布式成簇协议中通过伪造状态信息使自己或者其他某个节点成为簇首节点,从而可以在网络运行过程中收集簇内节点数据、干扰数据聚集结果或者破坏数据包路由.由于这种攻击以及下述第(3)类、第(4)类攻击均由内部恶意节点发起,因此,基于对称密钥的消息认证机制不能解决这些问题.

(3) 簇成员恶意征募攻击:恶意节点试图通过这类攻击使更多节点加入其所在的簇以获取这部分节点的数据,或者使其攻击影响的范围扩大以获得最大收益.攻击者通过 3 种手段实现这类攻击:

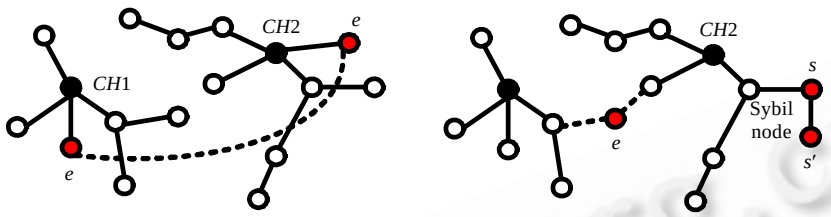
a. Hello flood 攻击[5]:恶意节点可以增加无线电发射功率以向更远距离广播簇首通知消息,从而招募更多的节点作为自己的簇内成员.

b. 簇首通知消息篡改攻击:对于 d-跳成簇协议(d>1),簇首通知消息在 d-跳内广播,消息的跳数(hop)字段每跳加 1 后转发,直到为 d 时停止广播.对于恶意节点来说,它可以减小或者不增加 Hop 字段的值以使通知消息广播到更远距离,从而有更多的节点加入到簇内并成为自己的子孙节点.攻击者可以通过这种攻击干扰或者监听更多节点与簇首之间的通信.

c. 虫洞攻击(wormhole attack)[6]:攻击者通过在传感器网络内相距较远的两点以有线或者无线高速链路形式建立带外连接(out-of-band connection),使得在这两点的传感器节点相互可以直接通信从而误认为彼此是一跳邻居节点.在成簇协议运行过程中,攻击者通过这种攻击方式可以使得簇首征募

不在原 d-跳范围内的远处节点作为簇成员,从而扩大簇规模和获取更大范围的数据,以获取更大的攻击效果.

(4) 多重簇成员身份攻击:如果恶意节点可以拥有多个簇成员身份,则可以通过在多个簇内提供错误的感知数据或者一个簇内提供多份错误数据来干扰大范围内的数据聚集结果,使数据聚集结果和正确结果之间的偏离最大化.这种攻击的实现手段可以有虫洞攻击[6]、节点复制攻击(node replication attack)[7]和女巫攻击(sybil attack)[8].图 1(a)表示恶意节点 e 通过 wormhole 或者 node replication attack 分别成为两个簇的成员节点;图 1(b)中恶意节点 e 通过邻居节点加入了本地的两个簇,而恶意节点 s 通过 sybil attack 引入虚假子节点 s'可以增加簇内伪造数据的比例,从而有效地偏离数据聚集的正确结果.



(a) (b)

Fig.1 Effect of multiple cluster-membership attack, d=3

图 1 多重簇成员身份攻击效果示例图,d=3

1.2 本文工作

本文研究了无线传感器网络成簇所面临的安全问题,并针对成簇协议面临的各种安全威胁,包括节点伪装攻击、簇首占据攻击、簇成员恶意征募攻击、多重簇成员身份攻击,提出了一种分布式 d-跳安全成簇协议.本文在成簇协议设计中通过基站的随机数广播使可信第三方参与到簇首选举的过程中,保证了簇首选择结果的强制随机性和可验证性,同时又避免了集中式成簇带来的不可扩展性;通过在网络初始化阶段节点建立 d-跳邻居列表和到 d-跳范围内每个节点的近似最短路径跳数来抵御簇成员恶意征募攻击和多重簇成员身份攻击;以单向密钥链技术为基础建立有效的簇首身份认证机制,实现 d-跳邻居节点对簇首通知消息和簇首身份的认证.

本文第 2 节简要回顾无线传感器网络安全成簇方面的相关工作,并分析当前已有研究工作的不足.第 3 节给出问题定义,包括本文协议基于的系统模型,协议的设计目标.第 4 节给出分布式安全成簇协议的细节描述.第 5 节分析协议的性能、安全性和开销.第 6 节进行总结.

Table 1 Notations in this paper

表 1 本文使用的记号

u,v	Low-Case letters represent principals or node ID	$MAC(K,M)$	Message authentication code of message M using a symmetric key K
N_u	Set of node u's one-hop neighbors	$E(K,M)$	Encrypting message M with key K
N_u^d	Set of node u's d-hop neighbors	$len(M)$	Bit length of message M
K_u	Individual key of node u	BS	Base station
K_{uv}	Pairwise key between node u and v	L	Length of key chain L+1
$S1\|S2$	Conjunction of string $S1$ and $S2$		

2 相关工作

已有的成簇协议设计[1−4]主要考虑成簇性能问题,而没有考虑安全性,使得它们不能抵御上述攻击.尽管这些成簇协议可以通过建立节点成对密钥实现消息认证来避免节点伪装攻击,但是,这些协议中算法固有的性质使得它们面对其他几种威胁时依然是脆弱的.根据簇首选择方式的不同,这些成簇协议基本上分为 3 种:集中式,即由基站选择簇首节点,并将结果向全网广播;簇首自举,即不需要周围节点参与,节点独立决定自己是否充当

簇首;基于信息交换的分布式簇首选择,即通过节点之间的信息交换,协商决定哪个节点充当簇首.集中式算法尽管可以通过基站安全选择来避免簇首占据攻击,但却面临着可扩展性的问题,不适用于大规模网络.在后两种成簇方式中,恶意节点很容易成为簇首并且占据簇首位置.在基于概率自举的成簇协议,如 LEACH(low-energy adaptive clustering hierarchy)协议[1]和 *k*-跳成簇协议[2]中,由于簇首选择的独立性,恶意节点可以随意充当簇首.尽管 LEACH 协议要求在连续 *N*/*k* 轮中当过簇首的节点不能继续充当簇首,但是协议没有对此要求进行检查以防止违反规则的恶意节点.而在基于信息交换的分布式簇首选择,如在 HEED(hybrid energy-efficient distributed clustering)[3]和 ACE(algorithm for cluster establishment)[4]中,恶意节点可以通过伪造自己的状态信息,如簇内通信开销或者忠实节点数目,来欺骗周围的邻居节点以达到自己成为簇首节点的目的.同时,攻击者对所有这些协议均可以有效地实施簇成员恶意征募和多重簇成员身份攻击并达到攻击效果.

由于已有成簇协议存在着安全威胁,安全成簇便成为无线传感器网络中的一个重要问题.对此,研究人员提出了多种安全成簇机制[9−12]在不同程度上保证了安全性.

Ferreira1 等人在 LEACH 协议[1]基础上扩展了以抵御外部攻击者的集中式安全机制[11]来完成安全成簇.每个簇首除了向成员节点广播簇首通知消息之外,还向基站发送该消息;基站根据与每个节点共享的密钥认证簇首节点通知消息,所有认证正确的簇首组成簇首列表,使用μTESLA 协议向网络内广播;网络中接收到该列表的普通节点认证该广播消息后,选择信号能量选择最大的簇首节点加入该簇.这里的安全机制是一种集中式策略,需要每个簇首与基站的通信,而且只是针对外部攻击者防止节点伪装和消息伪造攻击,也没有解决成员节点加入簇首的注册消息认证问题.Oliveria 等人在文献[12]中扩展了 Ferreiral 的工作,其中引入了随机密钥分布机制,使成员节点与簇首节点以一定概率共享密钥来完成簇首的认证,与簇首没有共享密钥的节点成为孤立节点或者直接与基站通信.然而,这里存在与文献[11]同样的问题.

Sun 等人提出了一种基于图上团划分的分布式安全成簇算法[10].该成簇算法不同于一般 Leader-First 成簇算法,不是以簇首选择作为成簇的第 1 步,而是首先通过邻居列表交换节点间划分成互不相交的团,以团作簇之后,再根据团内节点联合产生的随机数来随机选择簇首节点.最后,通过团内成员关系一致性检验保证算法结果的正确性并抵御恶意节点可能的攻击.如果恶意节点伪造邻居列表或者试图成为多个团的成员,则会导致团内成员关系的不一致性而被检测出来;同时,团内联合的随机数生成决定了簇首选择的可靠随机性.相比 Leader-First,该成簇协议更具安全性,避免了在 Leader-First 的协议中恶意节点可以征募大量的合法节点,恶意扩大簇范围,始终保留自己的簇首位置等安全威胁.但是,文献[7]采用了公钥密码体系实现簇成员关系一致性检验,带来很大的能量开销,不适用于传感器网络;同时,以团为簇的方案不能扩展到多跳簇结构,损失了成簇在广播、路由和数据融合上带来的能量有效性.

Liu 在文献[9]中主要针对 wormhole attack[6]和 node replication attack[7]对成簇协议造成的威胁,提出了相应的安全成簇方案.其基本依据在于,在正常网络环境中,节点的邻居节点数目和两个节点之间共享的邻居节点数目有界.一个节点通过 wormhole 检测技术删除邻居中 wormhole 引入的虚假邻居节点,之后选择与自己共享邻居数大于正常下界并且优先级高于自己的邻居节点作为候选上层节点(candidate uplink node),再从中选择共享邻居数最大的作为自己的上层节点(uplink node),请求与它加入同一个簇.在请求上层节点时,节点需要周围的邻居节点给该上层节点提供确认信息.基站通过收集到的节点邻居列表检查一个节点周围的邻居数是否超过上限,若是,则认为该节点受到攻击,撤销该节点.文献[6]的主要问题在于,安全机制依赖的预分配优先级成簇仅能用于网络部署后一次成簇,不能支持负载均衡所必需的周期性簇首重新选择和簇结构的重组织,同时也使得成簇无法避免簇头占据攻击.

由此可见,目前的安全成簇工作没有系统地针对第 1.1 节中指出的所有 4 种安全威胁给出解决方案,也没有在成簇协议的性能和安全性之间做出合理的权衡,如文献[9]没有考虑协议的周期性运行以达到负载均衡;文献[10]中限制成簇的形式仅为一跳的团结构,损失了数据聚集等网络操作带来的能量有效性;文献[11]中的集中式方法不具有可扩展性.对此,本文提出的分布式成簇协议能够在保证性能的同时抵御这 4 种安全威胁.

3 问题定义

3.1 系统模型

(1) 网络模型

不失一般性,设无线传感器网络由 N 个资源受限的静态传感器节点组成.节点随机、均匀地部署在监测区域.传感器节点具有相同的计算、通信能力和能量资源.每个节点具有唯一的 ID.传感器节点通过安全的时间同步服务[13]达到宽松的时间同步.节点的感知数据由一个固定基站收集,基站具有更大的存储、计算能力以及充足的能量供应.网络链路具有双向性.节点使用 CSMA/CA 通信以减小信道冲突.同时,本文假设网络具有失效节点检测机制.通过每个节点周期性的心跳信号,周围邻居节点可以发现失效节点并通知基站.

(2) 攻击模型

攻击者可以在无线链路上侦听数据传输、注入数据包和重放旧数据包,也能够俘获传感器节点,获得它的所有信息,包括密钥信息,并且能够完全控制该节点实施包括消息丢弃、篡改和伪造等的各种内部攻击,也就是说,攻击者可以针对成簇协议实施在第 1.1 节中指出的 4 类攻击.本文假定基站是安全的,不能被攻击者俘获.

(3) 安全假设

每个传感器节点都与基站共享唯一的密钥,称为节点密钥.节点密钥在节点部署之前载入节点.每个节点在部署之后,可以通过已有的机制,如 LEAP(localized encryption and authentication protocol)[14]、二元对称多项式方案[15,16]与每个邻居节点之间建立共享成对密钥(pair-wise key).基站使用安全的机制来完成广播消息认证(如 μTESLA[17]).网络中,每个节点通过认证机制检验基站广播消息的合法性和完整性.节点使用安全的一跳广播源认证机制(如 LEAP[14])完成一跳消息广播和认证.同时,假定在传感器网络部署之后初始启动阶段是安全的,也就是说,在这个阶段,攻击者不能俘获和破解任何传感器节点.因为攻击者不知道确切的部署地点以及破解节点需要耗费一定的时间,所以这种假设是合理的且被广泛采用[14,18].

3.2 问题描述

由于 d-跳成簇在已有成簇协议中具有一般性,因此本文针对 d-跳成簇的安全问题提出了相应的分布式安全方案,以抵御第 1.1 节中列举的各种攻击.但考虑到成簇协议所构造的簇拓扑服务于网络的数据融合、路由等基础网络功能,在解决安全问题的同时,协议不能够损失簇拓扑性能上的有效性.因此,本文协议设计目标需要在满足安全性的同时也能达到成簇性能目标.

协议在安全目标上要求:

(1) 恶意节点不能以比诚实节点更大的概率使自己成为一轮或多轮的簇首节点,并且每一轮决策必须具有强制的随机性.也就是说,恶意节点也不能预知和控制下一轮的选择结果,从而抵御簇首占据攻击;
(2) 需要保证簇首选择结果的可验证性;
(3) 能够有效抵御簇成员恶意征募攻击;
(4) 能够有效抵御多重簇成员身份攻击.

协议在性能目标上要求:

(1) 安全成簇协议必须能够达到一般成簇协议的性能要求且不损失成簇拓扑带来的有效性,即需要保证簇首选择的公平性和簇结构的均匀性,达到网络的负载均衡,而且能够保证网络簇拓扑带来的能量有效性以节省簇内通信和簇首与基站通信的能量开销;
(2) 安全成簇协议的通信开销合理,在大规模网络下具有可扩展性.

4 分布式安全成簇协议

4.1 基本协议描述

本文的分布式安全成簇协议设计基于文献[2]中的 d-跳成簇协议.该协议分为两个阶段:簇首选择阶段和成

簇阶段.考虑到负载均衡,成簇协议按轮周期性运行来选择新的簇首和建簇.假设 T_{cp} 为成簇协议完成网络簇拓扑结构建立所需要的最大运行时间.在簇首选择阶段,网络中每个节点以概率 p_r 选择自己成为簇首(自举簇首)并且向 d-跳范围内节点广播一个簇首通知消息 AD_u={*ADV_Clusterhead*,*u*,*Hop*},其中包括消息类型 *ADV_Clusterhead*、节点 ID *u*.*Hop* 表示跳数计数,初始为 0,在消息到达每个转发节点时加 1,计数到 *d*,则节点不再广播该消息.在成簇阶段,任何没有成为自举簇首的节点 *u* 如果在有限时间间隔 *t* 内接收到一个或多个不同的簇首通知消息,则选择加入距离自己跳数最近的一个簇首 *CH*(如果跳数均相同,则随机选择),把广播 *CH* 簇首通知消息的邻居节点 *w* 作为上层父节点,通过 *w* 向簇首 *CH* 发送包括成员注册请求消息 $Register_u$={*ADV_Join*,*u*,*CH*}.如果在时间间隔 *t* 内 *u* 没有接收到任何通知消息,则自己便成为簇首节点(非自举簇首).这里,时间 *t* 要大于消息从簇首广播到 *d*-跳范围内所有节点所需要的最大时间.这里,*d* 和 p_r 是基本的协议参数.

4.2 网络安全初始化

在安全的初始启动阶段,作为安全成簇协议的基础,网络初始化过程用于建立安全成簇所需的基本安全信息,包括完成密钥建立以及带跳数更新的 *d*-跳邻居发现.在完成安全初始化之后,每个节点上存储 *d*-跳邻居列表 N_u^d,与每个 *d*-跳邻居之间的成对密钥和近似最短路径跳数.

在节点部署之前,每个节点 *u* 预载入仅与基站共享的节点密钥 K_u 和邻居列表认证密钥 K_{Nu}.在传感器网络部署之后,节点执行 *d*-跳邻举发现过程,并与每个邻居节点建立成对密钥.节点间成对密钥用于两者之间的消息加密和认证,以抵御外部攻击者.节点完成邻居成对密钥建立之后删除用于密钥建立的初始信息,不能再次生成与其他节点间的成对密钥.例如,在基于 LEAP[14]的密钥建立方案中,相邻节点 *u*,*v* 根据节点 ID *u*<*v* 由全局主密钥 K_I 可以得到成对密钥 $K_{uv}=f_{f_{K_I}(v)}(u)$,在节点完成与所有邻居成对密钥建立之后删除 K_I.

在 *d*-跳邻居发现过程中,每个节点 *u* 在 *d*-跳范围内广播 Beacon 消息 NM_u={*ADV_discovery*,*u*,*Hop*},其中,*Hop* 字段为消息跳数计数,初始为 0.接收到 NM_u 的节点 *v* 将 *u* 加入自己的 *d*-跳邻居列表 N_v^d.对接收到的一份或多份冗余 NM_u,*v* 选择 *Hop* 值最小的 NM_u,将其 *Hop* 字段加 1 并更新 $Hop_{u,v}$=*Hop*,如果 *Hop*<*d*,则继续向邻居广播,然后 *v* 进入等待状态.如果 *v* 在等待状态时又侦听到 *Hop*+1<$Hop_{u,v}$ 的 NM_u,则记录该消息并将 *Hop* 字段加 1,重复上述过程;否则,*v* 不作任何动作直到计时器超时.这里,计时器时长可以设为网络初始化阶段时长.对于来自不同节点的 Beacon 消息,接收节点 *v* 都执行相同的过程.图 2 是通知消息处理过程的自动机表示.通过 Beacon 消息广播每个节点 *v* 可以获得 *d*-跳范围内节点集合 $N_v^d=\{u \mid Hop_{u,v} \le d\}$,$Hop_{u,v}$ 表示由这种带跳数更新的广播重传过程所得到的节点 *u*,*v* 之间的近似最短路径跳数.由于传感器网络静态部署,对每个节点 *u*, N_u^d 是稳定的.这里,N_u^d 即为所有可能簇成员的集合.

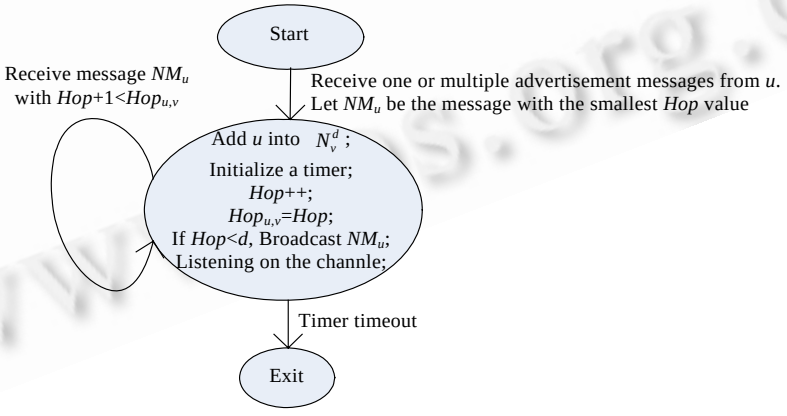


Fig.2 Processing procedure of NM_u on *v*

图 2 节点 *v* 的 NM_u 通知消息处理过程

通过邻居发现过程,节点得到一跳邻居列表 *ID_list* 后,使用 K_{Nu} 计算消息认证码得到:

$$AuthID\_list=\{ID\_list,u,MAC(K_{Nu},ID\_list\,|\,u)\}.$$

AuthID_list 用于后文第 4.5 节所述的非自举簇首认证.在存储 *AuthID_list* 之后,节点 u 删除 K_{Nu}.邻居列表认证码使得当节点在不知道加密密钥的情况下,不能向基站伪造一跳邻居,保证了确实性.

4.3 安全的概率簇首选举

在成簇协议每轮运行开始之前,由基站使用μTESLA 广播认证机制向网内广播一个随机数 $R_{\neg B}$.f 是一个部署前预载入节点的密码学上安全的伪随机函数,将输入变量映射到值域[0,1].

对于每个节点 u,基于自己的节点密钥 K_u 计算:

$$f_{K_u}(R_B\,|\,u)<p_r,0<p_r<1 \tag{1}$$

当不等式成立时,节点选择自己作为本轮簇首.安全伪随机函数 f 的使用,使得攻击者在不知道节点密钥的情况下无法在成簇协议运行之前确定哪个节点将被选为簇首.基站随机数 R_B 的选择每一轮都不同,使得上面的 $f_{K_u}(R_B\,|\,u)$ 不等式在每一轮计算结果是随机的,保证了簇首选择的随机性,性能上达到了负载均衡,安全性上使得攻击者在簇首选择阶段猜中正确的簇首的机会更小,同时也可以避免一个恶意节点长期占有簇首位置.

可见,每一轮每个传感器节点以概率 p_r 选作簇首节点.p_r 的选择决定了网络中自举簇数目.网络中有 n 个节点,那么平均一轮自举簇首数目 $n_{cs}=n\times p_r$.

4.4 安全的簇首通知消息广播

协议的簇首通知消息广播过程利用节点通知消息广播所得到的近似最短路径跳数和 d-跳邻居列表来抵御簇成员恶意征募攻击:当节点 v 接收到簇首通知消息 AD_u 时,将该消息跳数 *Hop* 加 1 后检查两个安全条件:(1) $u\in N_v^d$ 和(2) $Hop\geq Hop_{u,v}$,如果两者中任何一个不能满足,则忽略该消息,否则,继续广播转发过程.如果 v 在广播之前同时接收到满足安全条件的多个冗余的簇首通知消息,则 v 记录 *Hop* 最小的消息.不同于在初始化阶段的带跳数更新的消息广播过程,对同一个簇首通知消息每个节点仅广播一次,忽略后来的重复消息.AD_u 以及在后面协议描述中的其他类型消息均包含基站随机数 $R_{\neg B}$,目的是为了防止消息重放攻击.本文默认节点对接收到的消息都需要检查消息所包含的 $R_{\neg B}$ 与基站本轮广播值是否一致,若不一致,则忽略此消息.

通过对簇首通知消息的安全条件检查,可以有效地防止簇成员恶意征募攻击.在 Hello flood 攻击中,尽管恶意节点增大了消息广播的范围,但是根据安全条件 1,对于不在 d-跳邻居列表范围内的簇首通知消息,节点给予忽略处理,避免了 Hello flood 和 wormhole 攻击造成的影响.在安全条件 2 的限制下,恶意节点减小 Hop 字段最多也不能使其小于接收节点到广播源的最短跳数,保证了消息广播限制在 N_u^d 范围内.对于消息丢弃攻击以及增大 Hop 字段的消息篡改攻击,广播洪泛自身的鲁棒性极大地消减了攻击的效果,原因在于:由于广播洪泛过程中的中间节点会侦听到多个由不同邻居转发的冗余广播消息,即使一个恶意节点丢弃消息,仍可以从其他邻居节点接收到;同理,即使一个恶意节点恶意增加 Hop 字段,d-跳范围内节点依然有机会从其他邻居节点上接收到广播消息,并且会选择跳数最小的消息转发,不会造成簇规模被恶意缩减.

4.5 簇首身份认证

在簇首通知消息广播过程中,攻击者或者可以伪造簇首通知消息以假冒簇首节点,或者通过消息广播消耗 d-跳范围内的节点能量,因此在广播过程中,节点必须对簇首通知的合法性验证后再进行广播转发.同时,基站也需要具有对簇首身份的认证能力,以验证声称来自簇首的数据聚集结果或者事件报告,防止恶意节点伪装成簇首节点发送错误报告.根据基本成簇协议,簇首分为两种类型:自举簇首和非自举簇首.由于两种类型的簇首选择基于两种不同的原则,并且非自举簇首不需要有簇成员加入,因此本节分情况讨论簇首身份认证问题.

4.5.1 自举簇首身份认证

根据第 4.3 节中所述的簇首选举方式,一个节点 u 具有合法的自举簇首身份当且仅当不等式 $f_{K_u}(R_B\,|\,u)<p_r$ 成立.因此,要验证节点 u 的自举簇首身份合法性,验证方需要根据 R_B 和 K_u 计算不等式(1),如果不等式成立,则认

为 u 是合法的自举簇首,否则认证失败.然而,由于 K_u 是节点 u 仅与基站共享的秘密密钥,它的公开使得攻击者可以确定每一轮成簇节点 u 是否为自举簇头,违反了本文提出的第一个安全目标.因此,本文提出了一种基于单向密钥链的簇首身份认证技术.根据验证方的不同,这里将自举簇首身份认证分为本地认证和基站认证来讨论:本地认证是指 d-跳范围内的节点对簇首通知消息广播源其簇首身份合法性的验证;基站认证是指基站对数据聚集结果或事件报告的来源其簇首身份合法性的验证.

(1) 本地簇首身份认证

在网络初始化阶段,每个节点使用单向散列函数 H 计算长度为 L+1 的单向密钥链 *Chain*_0.

$$Chain\_0=K_u^{0,0} \xleftarrow{H} K_u^{0,1} \xleftarrow{H} ... \xleftarrow{H} K_u^{0,L}.$$

这里, $K_u^{0,i}=H(K_u^{0,i+1})=H^2(K_u^{0,i+2})=H^k(K_u^{0,i+k})$, $K_u^{0,L}=H(K_u|0)$, $K_u^{0,L}$ 称为初始密钥, $K_u^{0,0}$ 称为初始密钥链承诺.然后,每个节点向 d 跳范围内的所有节点广播 $K_u^{0,0}$.由于同是在网络初始化阶段,这个过程可以与第 4.2 节中节点 Beacon 消息广播过程合并起来,以节省通信开销.节点对接收到的每个 d-跳邻居 u 发送的初始密钥链承诺以三元组$\langle u, K_u^{0,0},(0,0)\rangle$形式记录.

为了实现簇首身份认证,节点概率自举过程和簇首通知消息需加以变动.节点 u 在每轮概率自举计算中,使用密钥链中当前密钥 $K_u^{0,i}$ (i>0)替换不等式(1)的节点密钥K_u,其他不变.如果计算结果不等式(1)成立,则 u 从密钥链中删除 $K_u^{0,i}$,这也表示在下一轮协议中使用的当前密钥为 $K_u^{0,i+1}$;如果不等式(1)不成立,则下一轮计算中当前密钥仍为 $K_u^{0,i}$.节点 u 成为自举簇首后,广播簇首通知消息 $AD_u=\{ADV\_Clusterhead, u, R_B, K_u^{0,i},(0,i),Hop\}$.$(0,i)$ 表示当前的密钥链序号 0 和当前密钥序号 i.当 u 的 d-跳范围内邻居节点接收到 AD_u 时,在检查满足第 4.4 节中的安全条件后,计算 $H(K_u^{0,i})$ 是否等于上一次从 u 接收到的簇首通知消息中的 $K_u^{0,i-1}$,如果相等,则说明 AD_u 中 $K_u^{0,i}$ 是 u 的当前密钥,并使用该密钥验证 u 的簇首身份合法性,即计算 $f_{K_u^{0,i}}(R_B|u)$ 是否小于 p_r.如果验证结果为真,则记录该消息,将保存的密钥链承诺$\langle u, K_u^{0,i-1},(0,i-1)\rangle$更新为$\langle u, K_u^{0,i},(0,i)\rangle$并继续簇首通知消息广播,否则,忽略该簇首通知消息.

令 i 为 u 的当前密钥序号.当 i=L−1 时,簇首节点 u 计算一个同样长 L+1 的新密钥链 *Chain*_1 以接替将要耗尽的 *Chain*_0,其中,初始密钥为 $K_u^{1,L}=H(K_u|1)$.u 在簇首通知消息中除了包含原消息内容,还包括对应 *Chain*_1 的初始密钥链承诺 $K_u^{1,0}$,其中 $AD'_u=\{ADV\_Clusterhead, u, R_B, K_u^{0,L-1},(0,L-1), Hop, E(K_u^{0,L}, K_u^{1,0}|NEWCHAIN)\}$.$d$-跳邻居记录下 $E(K_u^{0,L}, K_u^{1,0}|NEWCHAIN)$.在 i=L 时,u 簇首通知消息中包含 $K_u^{0,L}$,因此,d-跳邻居使用 $K_u^{0,L}$ 解密 *Chain*_1 初始密钥链承诺 $K_u^{1,0}$,这里通过检查字符串 *NEWCHAIN* 来检验消息可靠性,以防止消息篡改攻击.

尽管由于广播洪泛的鲁棒性,丢包发生的可能性很小,但是一旦发生消息丢失,对 d-跳内邻居 v 存储的 u 密钥链承诺$\langle u, K_u^{0,k},(0,k)\rangle$和簇首节点 u 当前密钥为 $K_u^{0,i}$,k<i−1.根据密钥链本身的性质,节点可以计算 $H^{i-k}(K_u^{0,i})$ 是否等于 $K_u^{0,k}$.同样,由于消息丢失,v 也有可能发现自己没有簇首通知消息中对应密钥链序号的初始密钥链承诺,则它可以向簇首节点发送请求消息,簇首节点将对应的初始密钥链发送给该节点;或者它也可以向周围一跳邻居节点发送请求消息,拥有该密钥链承诺的邻居节点将其返回给该节点.这种情况下,为了保证安全性,可以执行如下策略:如果从多个邻居收到密钥链承诺,可以采用多数投票原则选择更多节点一致的密钥链承诺,提高可靠性;如果仅从一个邻居收到回复,则按第 1 种方式向簇首直接请求.

如上,主要讨论了节点从 *Chain*_0 到 *Chain*_1 的使用过程,更一般的过程以此类推,见后文第 4.7 节协议的总体算法描述.

(2) 基站簇首身份认证

在已知每个节点密钥、协议参数 P_r、随机数 R_B、密钥链长度 L+1 的基础上,基站在可以计算出每个节点密钥链并对每个节点检验不等式(1)后,获知任何一轮的所有自举簇首节点.因此,基站可以准确维护每个节点充当自举簇首的计数,跟踪每个节点的密钥链更新和当前密钥.设节点 u 当前自举计数为 C_u,则可知 u 密钥链使用计数 l=⌊C_u/L⌋,当前使用的密钥链即为 *Chain_l*,其初始密钥即为 $K_u^{l,L}=H(K_u|l)$.节点 u 本轮概率自举所使用的

当前密钥即为 *Chain_l* 中的 $K_u^{l,(C_u-l\times L)}$.基站根据初始密钥计算出 $K_u^{l,(C_u-l\times L)}$,代入不等式(1)确定 *u* 是否是本轮的簇首节点,若是,则更新 *u* 的自举计数和密钥链信息.基站对网络中每个节点作上述计算,可以获得本轮的自举簇首列表.当基站在接收到声称来自自举簇首的数据报告时,除了根据节点密钥检验消息认证码以外,还检查节点 ID 是否在本轮的自举簇首列表中,如果有任何一项检查失败,则忽略报告.

4.5.2 非自举簇首身份认证

非自举簇首不需要征募簇成员,因此不需要向其他节点证明身份,只需要处理基站认证的问题.根据非自举簇首选择原则,其身份验证可以转化为验证不在任何自举簇首的 *d*-跳范围内的问题.对于非自举簇首 *v*,根据选择原则,其一跳邻居节点状态必然属于两种状态之一:或者是任何簇首通知消息的边界节点,或者也是非自举簇首.簇首通知消息的边界节点是指由于 *d*-跳限制到达自己的簇首通知消息都没有继续转发的节点.因此,当 *v* 向基站证明身份时,需要周围一跳邻居节点的状态确认.具体过程如下:

非自举簇首 *v* 广播确认请求消息 $REQUEST_v$ 给所有一跳邻居节点 $w\in N_v$.

$$REQUEST_v=\{ADV\_Nonselfelected\text{-}clusterhead,v,R_B\}.$$

如果 *w* 确认自己属于上述两种状态之一且 *v* 在一跳邻居列表中,则返回包含请求确认信息 $ACK_{wv}=MAC(K_w,REQUEST_v)$的回应消息:

$$REPLY_{wv}=\{ACK\_Bordernode,ACK_{wv},MAC(K_{wv},ACK_{wv})\}.$$

当 *v* 收到所有邻居的确认消息、验证消息认证码后,将一跳邻居列表 *AuthID_list* 和包含所有邻居请求确认信息的 $MAC\_list=\{ACK_{wv}\mid w\in N_v\}$ 作为自己的非自举簇首身份证明,并构造证明消息:

$$CERTIFICATE_v=\{P_v,MAC(K_v,P_v)\},$$

其中,$P_v=\{ADV\_Nonselfelected\text{-}clusterhead,v,R_B,AuthID\_list,MAC\_list\}$.当 *v* 需要向基站进行非自举簇首身份认证时,将证明消息 $CERTIFICATE_v$ 发送给基站.基站验证消息和 *AuthID_list*,再根据每个邻居的节点密钥,检验 *MAC_list* 中的消息认证码.如果验证结果均为真,则基站承认 *v* 非自举簇首身份的合法性,并记录下 *AuthID_list*.由于证明消息中 *AuthID_list* 是不变量,如果节点 *v* 曾经当过非自举簇首,那么在下次身份证明中可以省略该部分,以进一步节省能量.如果基站发现 *v* 身份证明中没有 *AuthID_list*,但也没有关于节点 *v* 的 *AuthID_list* 的历史记录,则认为认证失败.

如果基站发现在非自举簇首身份证明 *MAC_list* 中有错误认证码,则说明在该节点不是合法的非自举簇首,或者一跳邻居内有恶意节点提供了错误的确认消息.关于认证失败后的恶意节点发现检测处理,超出了本文的范围,这里不再展开.但考虑到如果有节点失效不能提供自己的确认消息,则基站会发现对应的 *MAC_list* 有缺失.在这种情况下,由于本文假设网络通过周期性检测发现失效节点,因此,如果缺失认证码的节点属于失效节点且其他检验均正确,则非自举簇首的身份证明依然有效.

4.6 安全成簇

在成簇阶段,非簇首的恶意节点可以发动多重簇成员身份攻击,从而污染多个簇内聚集数据,扩大攻击范围或者使数据聚集结果与正确结果的偏离最大化.针对 wormhole,node replication 和 sybil 攻击,协议通过在安全初始阶段建立一跳和 *d*-跳邻居列表来抵御这几种攻击:簇首节点 *u* 通过检查注册节点是否属于 N_u^d,若是,则将其加入成员列表,不是,则丢弃注册请求.但是,针对本身处于多个簇范围内的恶意节点,这种手段无法防止其加入周围的多个簇.文献[5]的邻居一次承诺技术可以在一定程度上抵御这种攻击,即上层父节点需要得到该节点周围邻居的承诺才转发它的注册请求,否则忽略,并且周围邻居节点仅提供一次承诺来保证节点只能加入一个簇中.但是,这种要求每个邻居节点返回其承诺信息的方法会带来很大的通信负载.本文采用简单的节点侦听技术,对于非簇首节点 *u*,成簇协议要求其仅能加入一个簇,也即仅向一个簇首发送注册请求消息:

$$REGISTER_u=\{ADV\_Join,u,CH,R_B,MAC(K_{uCH},ADV\_Join|u|CH|R_B)\}.$$

u 使用一跳广播源认证机制[14]广播该消息.周围一跳的邻居节点均可以侦听并认证该注册请求.如果消息认证有效,属于 *CH* 簇的邻居节点向上转发该注册请求;其他邻居节点则记录 *u* 的注册请求.如果节点 *u* 在此之

后又向属于其他簇 CH'的邻居节点发送注册请求消息,对已侦听到 $REGISTER_u$ 的诚实节点来说,则忽略该请求不予转发.通过这种邻居侦听,使得恶意节点不能通过诚实的邻居节点而成为多个簇的成员.对于合谋攻击,也就是说,一个恶意节点 e 可以通过一个不诚实的上层节点 w 加入 w 所在的簇,w 不作上述检查就进行转发注册请求,它的影响不作考虑.这是因为合谋攻击要求在要加入的簇内已有被俘获的节点作为共谋者,其开销和影响与攻击者直接在多个簇内俘获节点以污染多个簇数据的攻击开销和影响等价.

4.7 协议的算法描述

在上述几节讨论的基础上,这里给出分布式安全协议的形式化描述.协议在每个节点上按轮运行.在第 1 次协议开始之前网络完成安全初始化.在每轮协议运行之前,每个节点接收到一个基站广播的随机数 R_B.为了清晰起见,算法中没有表示出在消息丢失情况下密钥链的同步处理.

分布式安全成簇协议(设在节点 u 上运行).

Input:节点 u 当前的密钥链 $Chain\_j$ 上的密钥 $K_u^{j,i}$, R_B;

Output:节点 u 的状态 $Mystate$、所在簇的簇首标识 CH、簇成员集合 CMS.

1: $VCHS=\varnothing$; //VCH 表示通过验证的簇首集合
2: $CMS=\varnothing$; //CMS 表示簇成员集合
3: Initialize a $Timer1$ with a timeout T_{cp};
4: if $f_{K_u^{j,i}}(R_B \mid u) < p_r$ {
5: $Mystate=Selfelected\_Clusterhead$; $CH=u$; //Select myself as a cluster head
6: if $i=L-1$
7: Broadcast $AD'_u = \{ADV\_Clusterhead, u, R_B, K_u^{j,i}, (j,i), Hop, E(K_u^{j,L}, K_u^{j+1,0} \mid NEWCHAIN)\}$;
8: else Broadcast $AD_u = \{ADV\_Clusterhead, u, R_B, K_u^{j,i}, (j,i), Hop\}$;
9: while ($Timer1$ not timeout) {
10: On receiving a register message
$REGISTER_v=\{ADV\_Join, v, u, R_B, MAC(K_{vu}, ADV\_Join|v|u|R_B)\}$
11: if $v \in N_u^d$ Add v into CMS;
12: }
13: }
14: else {
15: Initialize a $Timer2$ with a timeout t;
16: On Receiving a advertisement message AD_x or multiple copy of that
$AD_x = \{ADV\_Clusterhead, x, R_B, K_x^{m,n}, (m,n), Hop\{, E(K_x^{m,L}, K_x^{m+1,0} \mid NEWCHAIN)\}\}, n \le L$
17: {
18: Select a AD_x with the minimum Hop value and discard other copies;
19: Hop++;
20: if $x \in N_u^d$ and $Hop \ge Hop_{x,u}$ {
21: if $H(K_x^{m,n}) == K_x^{m,n-1}$ and $f_{K_x^{m,n}}(R_B \mid x) < p_r$ {
22: Add x into $VCHS$;
23: Broadcast AD_x;
24: if $n==L-1$
25: Store $E(K_x^{m,L}, K_x^{m+1,0} \mid NEWCHAIN)$;
26: if $n==L$ {

```
27:                                  Decrypt  E(K_x^{m,L}, K_x^{m+1,0} | NEWCHAIN) ;
28:                                  if NEWCHAIN exists, Store ⟨x, K_x^{m+1,0} ,(m+1,0)⟩;
29:                              }
30:                          }
31:                      }
32:                  }
33:                  On Timer2 Timeout {
34:                      if VCHS=∅ {
35:                          Mystate=NonSelfelected_Clusterhead;CH=u;   //Select myself as a cluster head
36:                          Broadcast  REQUEST_u = {ADV_Nonselfelected-clusterhead, u, R_B} ;
37:                          For each w∈N_u, Receive REPLY_wu;
38:                          P_u = {ADV_Nonselfelected-clusterhead, u, R_B, AuthID_list, MAC_list} ;
39:                          Send  CERTIFICATE_u = {P_u, MAC(K_u, P_u)} to BS;
40:                      }
41:                      else {
42:                          Select the closet cluster head CH_closet in VCHS;
43:                          Mystate=Cluster_Member;CH=CH_closet;          //Join the cluster of CH
44:                          Send a register message REGISTER_u={ADV_Join,u,CH,R_B,MAC} to CH;
45:                      }
46:                  }
47:                  while (Timer1 not timeout) {
48:                      On receiving a request message  REQUEST_v = {ADV_Nonselfelected-clusterhead, v, R_B}
49:                          if v∈N_u and (u is a border-node or a nonselfelected clusterhead)
50:                              Send  REPLY_uv = {ACK_Bordernode, ACK_uv, MAC(K_uv, ACK_uv)} ;
51:                  }
52:              }
```

5 协议评价

5.1 分析假设和协议参数优化

假设通信半径为 R 的传感器节点部署在边长为 $2a$ 的正方形区域,节点分布服从参数为 λ 的二维平面上的齐次空间泊松过程,同时,假设基站位于正方形区域的中心.在此假设条件下,这里列出本节分析所用到的各个参数和文献[2]中基本 d-跳成簇协议分析过程的相关推导结果:

(1) 平均网络中节点数目 n,$n=4\lambda a^2$.

(2) 簇首节点和非簇首节点的分布服从参数分别为 $\lambda_1=p_r\lambda$ 和 $\lambda_0=(1-p_r)\lambda$ 的两个独立的齐次空间泊松过程 $PP1$ 和 $PP0$.

(3) 平均一个簇的簇内成员数目 n_m, $n_m=\dfrac{\lambda_0}{\lambda_1}=\dfrac{1-p_r}{p_r}$.

(4) 平均一个簇中所有簇内成员到簇首的总跳数 hop, $hop=\dfrac{\lambda_0}{2R\lambda_1^{3/2}}$.

(5) 平均一个节点 d-跳范围内邻居数目 n_d,$n_d=\pi\lambda d^2R^2$;对一跳邻居数 n_1,$n_1=\pi\lambda R^2$.

(6) 簇首节点到基站的平均跳数 hop_B, $hop_B=\dfrac{0.765a}{R}$.

由上述分析假设和基本结果,Bandyopadhyay[2]讨论了基本 d-跳成簇协议中每个节点选择成为簇首的概率 p_r 和簇内成员距离簇首的最大跳数 d 这两个协议参数的优化选择:通过建立网络节点通过簇首向基站传输数据的能耗方程,可以求出使得能耗最小的 p_r 的优化解 p_1;同时,考虑到非自举簇首仅一个节点成簇,数据没有聚集直接发送给基站,因此,其个数越多,簇结构的能量有效性越差.以非自举簇首尽可能地少为目标,在给定概率要求,即要求一个节点不在任何簇首 d-跳范围内的概率为 α 的基础上,d 值选取 $d_1=\left\lceil \frac{1}{R}\sqrt{\frac{-0.917\ln(\alpha/7)}{p_1\lambda}} \right\rceil$.$\alpha$ 的取值也决定了网络中非自举簇首节点的数目.α 是一个很小的值以尽量最小化非自举簇首,保证网络成簇有效性.在给定 α 时,平均一轮非自举簇首数目 $n_{cn}=n\times\alpha$.

本文的分布式安全成簇协议建立在基本 d-跳成簇协议基础上,除了由于安全性需要引入的额外通信开销以外,保留了基本协议的成簇性质,因此,可以通过 p_1,d_1 的优化协议参数选择而达到第 3.2 节提出的第一个性能目标.

5.2 协议安全性分析

引理 1. 在一轮成簇协议运行中,一个恶意节点 e 成功地成为一个自举簇首的概率 P_{es} 仅为 p_r.

证明:令 $K_e^{j,i}$ 为 e 概率自举计算中应使用的当前密钥,$K_e^{j,i-1}$ 为 e 的 d-跳范围内的邻居节点中存储的当前密钥链承诺,有 $H(K_e^{j,i})=K_e^{j,i-1}$.恶意节点 e 在一轮协议里充当自举簇首当且仅当下述条件两者之一成立:

A. 计算不等式(1)成立,即 $f_{K_e^{j,i}}(R_B\,|\,u)<p_r$.这种情况下,$e$ 是协议中的合法簇首;

B. 计算不等式(1)不成立,即 $f_{K_e^{j,i}}(R_B\,|\,u)\geq p_r$ 但 e 成功欺骗 d-跳范围内的邻居节点自己是合法簇首.e 成功欺骗是指在 e 的簇首通知消息中伪造概率自举计算中所使用的当前密钥 K_e' 满足 $f_{K_e'}(R_B\,|\,u)<p_r$ 且 $H(K_e')=K_e^{j,i-1}$.显然,$K_e'\neq K_e^{j,i}$.由于 H 是安全的单向哈希函数,满足第二原象稳固(second preimage resistant),因此认为成功欺骗的概率基本为 0.

由于 R_B 是随机数,因此对任何一个节点不等式(1)成立的概率为 p_r.因此有

$$P_{es}=P(A \text{ is true})+P(B \text{ is true})=P(A \text{ is true})=p_r.$$ □

定理 1. 不考虑合谋攻击,在一轮成簇协议运行中,一个恶意节点 e 成功成为一个簇首的概率 P_{ec} 为 $\alpha+p_r$.

证明:令 $P_{en}=P(e$ 为非自举簇首$)$,e 成功成为非自举簇首或者是由于不在任何簇首节点的 d-跳范围内成为合法的非自举簇首,或者在某簇范围内但通过邻居节点的俘获得到了所有一跳邻居的确认而成功地向基站伪造证明.这里由于不考虑合谋攻击,即后一种情况,那么 P_{en} 就是前一种情况发生的概率,根据第 5.1 节 $P_{en}=\alpha$,则有

$$P_{ec}=P_{en}+P_{es}=\alpha+p_r.$$ □

定理 1 说明,即使是恶意节点,它在一轮中充当簇首的概率与其他所有正常节点一样,不能以更大的概率获得簇首位置.对合谋攻击来说,即使恶意节点 e 通过合谋攻击获得非自举簇首的身份证明,由于没有任何簇成员,e 的影响仅限于自身,所以这种攻击除了使攻击者需要俘获多余的节点之外,却不能带来更大的攻击效果.

定理 2. 每一轮自举簇头选择结果具有强制随机性,即恶意节点不能预知和控制下一轮的选择结果.

证明:由第 4.3 节不等式(1)可知,每一轮成簇的簇首选择结果仅由基站广播的随机数 R_B 和节点当前密钥所决定,因此有:(1) 假设基站作为可信第三方,由于基站每轮广播安全伪随机数 R_B 不可预测,则关于不等式(1)的伪随机函数计算结果每轮也均不可预测,所以在未获知下一轮 R_B 的情况下,恶意节点不能预先确定节点下一轮的簇首身份;(2) 因为 R_B 由可信基站每轮唯一确定,概率 p_r 作为固定参数不变,协议又通过采用单向密钥链认证技术使恶意节点不能任意伪造其当前密钥,所以恶意节点无法通过伪造参数输入来控制不等式(1)的计算结果——即其在每一轮的选择结果.作为特例,恶意节点可以在密钥链更新时任意伪造新密钥链的初始密钥链承诺,但是这并不影响定理 2 的正确性,因为初始密钥链承诺发布之时,与对应新密钥链一起使用作为不等式(1)输入的 R_B 未知且随机,保证了选择结果的不可控性. □

定理 1 和定理 2 说明,分布式安全协议能够有效地达到抵御簇首占据攻击的目的,实现第一个安全目标.

对节点伪装攻击,除了传统的基于对称密钥的节点认证机制以外,协议分别通过基于密钥链技术的簇首身份认证实现 *d*-跳范围内的自举簇首认证;通过邻居确认技术实现了面向基站的非自举簇首认证;对以 *Hello flood* 和消息篡改为攻击手段的簇成员恶意征募攻击,协议基于初始化建立的邻居列表来保证簇内节点均是在簇首 *d*-跳范围内的节点;对多重簇成员身份攻击,通过邻居侦听的方法保证恶意节点不能通过诚实节点加入多个簇.因此,分布式安全成簇协议有效地达到了第 3.2 节所述的安全目标.

通过上述分析可以看到,相比于已有的安全成簇方案,本文提出的协议在达到安全目标的同时满足了成簇的性能要求——规模可扩展性和负载均衡.表 2 列出了本文与已有方案之间的比较结果.

Table 2 Comparison between the existing secure clustering protocols and ours

表 2 已有安全成簇协议和本文工作的对比

Property	Protocol				
	Ours	Liu[9]	Sun[10]	Ferreira1[11]	Oliveria[12]
Resiliency against node personating attack	Yes	Yes	Yes (public key)	No	Yes
Resiliency against cluster-head holding attack	Yes	Yes	Yes	Yes	Yes
Resiliency against malicious recruiting attack	Yes	Yes	Yes	No	No
Resiliency against multiple cluster-membership attack	Yes	Yes	Yes	No	No
Scalability	Yes (distributed/d-hop)	Yes	No (one hop clique)	No (centralized)	No (centralized)
Load balance (support cluster reorganization)	Yes	No	Yes	Yes	Yes

5.3 协议开销分析

针对 *d*-跳成簇协议的安全问题,类似于文献[11,12]的集中式算法,也可以实现本文分布式安全成簇协议达到所要达到的安全性,其中的不同之处在于每一轮簇首节点由基站决定,基站将簇首列表消息使用μTESLA 机制广播到网络中,以及集中式协议的簇首通知消息广播过程中基于单向密钥链的认证技术仅用于消息广播认证.不在簇首列表中的节点对接收到的任何簇首通知消息通过检查消息源是否在簇首列表中来验证簇首的合法身份.这种集中式 *d*-跳安全成簇协议的簇首选择由可信第三方决定,可以有效地抵御簇首占据攻击.其他安全目标在集中式协议中也可以通过网络安全初始化来实现.因此,本节以该集中式 *d*-跳成簇协议作为基准,详细分析分布式安全成簇协议的开销和有效性.

为了定量分析协议开销,不失一般性,本节作如下合理设定:节点 ID 长 $\log_2 n$bit.安全协议中所使用的密钥长度 L_k 为 64bit.随机数 R_B、消息认证码长度也均为 64bit.密钥链序号和当前密钥序号为 16bit 整数.成簇跳数 d<256,因此,跳数字段长 8bit.成簇协议运行轮数 n_r 等于整个网络生存期时间除以成簇协议运行间隔.根据前述分析,一轮协议平均自举簇首数目 $n_{cs}=n\times p_r$,平均非自举簇首数目 $n_{cn}=n\times\alpha$.

5.3.1 存储开销分析

在完成网络初始化后,节点需要存储 *d*-跳邻居列表和与邻居共享的成对密钥.在协议运行期间,每个节点存储一个长度为 L+1 密钥链和所有 *d*-跳邻居节点的密钥链承诺.因此每个节点上比特存储开销 $C_{storage}$ 为

$$C_{storage}=n_d\times(\log_2 n+L_k)+(L+1)\times L_k+n_d\times L_k=n_d\times(\log_2 n+128)+64\times(L+1) \tag{2}$$

由此可见,协议存储开销随 *d*-跳邻居数目和密钥链长度的增加而增加.而在集中式协议运行过程中,每个节点还需要额外存储由基站广播的簇头列表,因此假设两种协议下的簇首节点数目相同,则集中式与分布式协议存储开销之差为

$$\Delta C_{storage}=(n_{cs}+n_{cn})\times\log_2 n=n\times(p_r+\alpha)\times\log_2 n \tag{3}$$

5.3.2 计算开销分析

在计算开销分析中,假设计算单向哈希函数为 *H*,伪随机函数为 *f*,消息认证码计算,加密解密计算开销均为常数,给出复杂度分析结果.

在基站随机数广播过程中,每个节点需要认证广播消息,计算开销为$\Theta(1)$.整个网络的开销则为$\Theta(n)$.

在每一轮中,每个节点成为自举簇首的概率为 p_r,则在整个网络生存期内每个节点成为自举簇首的轮数期望为 $p_r n_r$.而每一轮簇首节点需要消耗长 L+1 密钥链中的一个当前密钥,那么平均需要密钥链的数目为 $p_r n_r/L$,

而每个密钥链产生需要计算 L+1 次 H,则在网络生存期内,一个节点需要计算 H 的次数即为 $p_r n_r(L+1)/L$,可近似为 $p_r n_r$.整个网络的计算开销则为 $\Theta(nn_r p_r)$.

在一轮协议运行中,自举簇首 d 跳范围内节点需要计算一次 H 和 f 以完成簇首身份认证,因此,一个自举簇首引入的计算开销为 $\Theta(n_d)$,所有自举簇首引入的计算开销为 $\Theta(n_{cs}n_d)=\Theta(nn_d p_r)$.一个非自举簇首需要周围一跳邻居节点完成消息认证计算.因此,它引入的计算开销为 $\Theta(n_1)$,则所有非自举簇首引入的计算开销为 $\Theta(n_{cn}n_1)=\Theta(nn_1\alpha)$.在成员注册请求过程中,成员节点计算消息认证码,簇首节点认证所有簇内成员注册消息,其计算开销为 $\Theta(n_m)$,则所有簇的成员注册过程引入的计算开销为 $\Theta(nn_m p_r)$.每一轮这些过程所引入的网络总计算开销为

$$\Theta(nn_d p_r + nn_1\alpha + nn_m p_r).$$

由此,在整个网络生存期内,网络计算开销为 $CP_{network\_inlife}=\Theta(n_r(n+nn_d p_r+nn_1\alpha+nn_m p_r+np_r))$, 代入第 5.1 节基本结果中的参数值,化简可得 $CP_{network\_inlife}=\Theta(nn_r(2+n_d p_r+n_1\alpha))$,进一步可得网络生存期内平均每个节点计算开销为 $CP_{node\_inlife}=\Theta(n_r(2+n_d p_r+n_1\alpha))$,从而平均每一轮每个节点的计算开销为

$$CP_{node\_inround}=\Theta(2+n_d p_r+n_1\alpha) \tag{4}$$

由上式可以看到,$n_d p_r$ 是 d-跳范围内自举簇首平均数目,$n_1\alpha$ 是一跳范围内非自举簇首平均数目.每个节点的计算开销与它们相关,这是因为节点需要对 d-跳邻居中的每个自举簇首通知消息进行一次认证,对一跳邻居中的非自举簇首需要提供确认证明.常数部分表示节点用于广播认证和产生密钥链的计算开销.

对集中式算法而言,计算开销也包括广播认证和密钥链生成开销,而不同之处在于,因为集中式算法簇首选择在基站上完成,所以不需要非自举簇首提供确认证明.依然假设两种协议下的簇首节点数目相同,则集中式的节点计算开销为 $CP_{node\_center}=\Theta(2+n_d(p_r+\alpha))$,它与分布式协议的计算开销之差为

$$\Delta CP=CP_{node\_center}-CP_{node\_inround}=\Theta((n_d-n_1)\alpha) \tag{5}$$

5.3.3 通信开销分析

分布式安全成簇协议通信开销包括 4 部分:每轮基站的随机数广播开销、自举簇首通知消息广播开销、非自举簇首认证通信开销和簇成员注册请求消息开销.

(1) 协议消息长度

结合前述设定和在协议描述中所给出的各类消息格式,可以得出不同类型的消息长度.其中,因为协议包含 5 种不同的消息类型,所以消息类型字段占 3 个比特.

a.1. $len(AD_u)=3+\log_2 n+64+64+2\times16+8=171+\log_2 n$

a.2. $len(AD'_u)=len(AD_u)+64+64=299+\log_2 n$

b. $len(REQUEST_u)=3+\log_2 n+64+(64)=131+\log_2 n$,这里还附加考虑了 64bit 的一跳广播消息认证信息.

c. $len(REPLY_{uv})=3+64+64=131$

d. $len(CERTIFICATE_u)=len(P_u)+64=67+\log_2 n+n_1\times(\log_2 n+64)$

e. $len(REGISTER_u)=3+2\times\log_2 n+64+64+(64)=195+2\log_2 n$

(2) 协议通信开销分析

在基站使用 μTESLA 机制广播随机数过程中,网络中所有节点广播一次消息,消息中包括随机数和消息认证码,因此,$C_{R_B}=(64+64)\times n=128n$.

根据在计算开销中的分析,网络生存期内一个节点 u 平均需要密钥链数为 $p_r\times n_r/L$,故需要广播带密钥链更新信息的 AD'_u 次数也为 $p_r\times n_r/L$,而广播 AD_u 的次数则为 $p_r\times n_r\times(1-1/L)$.因为节点的每个 d-跳邻居至多广播一次簇首通知消息,所以每个节点整个网络生存期所引入的平均簇首通知消息开销为

$$C_{node\_AD}=n_d\times(len(AD'_u)\times p_r\times n_r/L+len(AD_u)\times p_r\times n_r\times(1-1/L)).$$

那么,整个网络平均一轮簇首通知消息广播开销 $C_{AD\_inround}$ 为

$$C_{AD\_inround}=C_{node\_AD}\times n/n_r=n_d\times n\times p_r\times(171+\log_2 n+128/L).$$

非自举簇首认证机制需要非自举簇首节点与所有一跳邻居通信,并向基站发送身份证明消息.由此,一个非自举簇首 u 引入的开销为 $len(REQUEST_u)+n_1\times len(REPLY_{uv})+hop_{uB}\times len(CERTIFICATE_u)$,$hop_{uB}$ 为 u 到基站的跳

数.由此可得平均一轮协议中所有非自举簇首所引入的认证通信开销为

$$C_{certificate\_inround}=E[n_{cn}]\times(len(REQUEST_u)+n_1\times len(REPLY_{uv})+hop_B\times len(CERTIFICATE_u)).$$

平均每个簇的成员节点向簇首发送注册请求消息的通信开销为 $hop\times len(REGISTER_u)$, 则平均一轮网络注册请求消息开销为 $C_{Register\_inround}=n\times p_r\times hop\times len(REGISTER_u)$.

分布式安全成簇协议平均每轮整个网络通信开销即为

$$C_{protocol\_inround}=C_{R_B}+C_{AD\_inround}+C_{certificate\_inround}+C_{Register\_inround}.$$

那么,代入第 5.1 节参数取值,可以得到平均每轮每个节点的通信开销为

$$\begin{aligned}C_{node\_inround}&=C_{protocol\_inround}/n\\&=128+n_d\times p_r\times\left(171+\log_2 n+\frac{128}{L}\right)+n_1\times\alpha\times\left(131+\frac{48.96a}{R}+\frac{0.765a}{R}\times\log_2 n\right)+\\&\quad\alpha\times\left(131+\frac{51.255a}{R}+\frac{0.765a}{R}\log_2 n\right)+p_r\times\frac{\lambda_0}{2R\lambda_1^{3/2}}\times(195+2\log_2 n)\end{aligned}\tag{6}$$

由公式(4)可以看到,通信开销与 L 成反比关系,这是因为 L 越小意味着越频繁的密钥链更替,从而引入更多的通信开销用于密钥链承诺广播.而 L 越大意味着存储开销 $C_{storage}$ 越大,所以 L 的选择应该在满足存储限制的情况下尽可能地大.

(3) 协议通信开销对比

集中式与分布式协议两者在通信开销上相比,其主要差别是由不同的基站广播消息开销引入的,其他消息开销基本相同,包括簇首通知消息、注册请求消息.由于α决定分布式协议中非自举簇首数目很小,因此,这里没有考虑非自举簇首的认证消息开销.在集中式成簇协议中,簇首列表广播消息包括平均 $n\times p_r$ 个节点 ID、64bit 消息认证码和一个用于防止消息重放攻击的随机数 R_B,它与本文分布式协议每轮节点平均通信开销之差即为基站广播消息开销之差:

$$\Delta C=C_{node\_center}-C_{node\_inround}=n\times p_r\times\log_2 n+64+64-128=n\times p_r\times\log_2 n\tag{7}$$

通过上述分析对比,根据公式(3)、公式(5)、公式(7)在 n 较大的大规模网络下分布式安全成簇协议比集中式方案进一步节省了存储开销、计算开销和通信开销,达到了第 3.2 节的第 2 个性能目标.值得注意的是,在公式(5),实际上,由于α往往很小,因此集中式和分布式协议两者之间在计算开销上的差别也相对很小,而在存储开销和通信开销上,两者之差与网络规模 n 成正比,体现了分布式协议的有效性.

6 结束语

本文针对无线传感器网络成簇协议的安全问题提出了一种分布式安全 d-跳成簇协议.本文系统地识别了作为无线传感器网络基本拓扑控制方法的成簇协议面临的各种安全问题,包括节点伪装和簇首占据攻击、簇成员恶意征募攻击和多重簇成员身份攻击,并针对这些安全问题提出了基于 d-跳成簇的分布式安全成簇协议,其中包括安全的概率簇首选择、安全的簇首通知消息广播、簇首身份认证和安全成簇.协议通过安全伪随机函数、单向密钥链技术以及网络安全初始化达到抵御各种攻击的安全目标.最后证明了协议的安全性,并分析了安全协议的通信、计算及存储开销,证明了协议的有效性.

References:

[1] Heinzelman WR, Chandrakasan AP, Balakrishnan H. An application-specific protocol architecture for wireless microsensor networks. IEEE Trans. on Wireless Communications, 2002,1(4):660−670.

[2] Bandyopadhyay S, Coyle EJ. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In: Proc. of the IEEE INFOCOM. San Francisco: IEEE Computer Society, 2003. 1713−1723.

[3] Younis O, Fahmy S. Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach. In: Proc. of the IEEE INFOCOM. Hong Kong: IEEE Computer Society, 2004. 629−640.

[4] Chan H, Perrig A. ACE: An emergent algorithm for highly uniform cluster formation. In: Proc. of the 1st European Workshop on Wireless Sensor Networks. LNCS 2920, Berlin: Springer-Verlag, 2004. 154−171.

[5] Karlof C, Wagner D. Secure routing in sensor networks: Attacks and countermeasures. Ad Hoc Networks, 2003,1(1):293−315.

[6] Hu Y, Perrig A, Johnson D. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In: Proc. of the IEEE INFOCOM. San Francisco: IEEE Computer Society, 2003. 1976−1986.

[7] Parno B, Perrig A, Gligor V. Distributed detection of node replication attacks in sensor networks. In: Proc. of the IEEE Symp. on Security and Privacy. Oakland: IEEE Computer Society, 2005. 49−63.

[8] Newsome J, Shi E, Song D, Perrig A. The sybil attack in sensor networks: Analysis & defenses. In: Proc. of the 3rd Int'l Symp. on Information Processing in Sensor Networks (IPSN). Berkeley: ACM Press, 2004. 259−268.

[9] Liu DG. Resilient cluster formation for sensor networks. In: Proc. of the Int'l Conf. on Distributed Computing Systems (ICDCS). Toronto: IEEE Computer Society, 2007. 40−48.

[10] Sun K, Peng P, Ning P. Secure distributed cluster formation in wireless sensor networks. In: Proc. of the 22nd Annual Computer Security Applications Conf. (ACSAC). Shanghai: Springer-Verlag, 2006. 131−140.

[11] Ferreira AC, Vilaca MA, Oliveira LB, Habib E, Wong HC, Loureiro AA. On the security of cluster-based communication protocols for wireless sensor networks. In: Proc. of the 4th IEEE Int'l Conf. on Networking. LNCS 3420, Reunion Island: Springer-Verlag, 2005. 449−458.

[12] Oliveria L, Wong H, Bern M, Dahab R, Lourerio F. SecLEACH-A random key distribution solution for securing clustered sensor networks. In: Proc. of the 5th IEEE Int'l Symp. on Network Computing and Applications. Cambridge: IEEE Computer Society, 2006. 145−154.

[13] Sun K, Ning P, Wang C, Liu A, Zhou Y. TinySeRSync: Secure and resilient time synchronization in wireless sensor networks. In: Proc. of the ACM Conf. on Computer and Communications Security. Alexandria: ACM Press, 2006. 264−277.

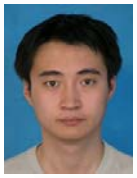
[14] Zhu S, Setia S, Jajodia S. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In: Proc. of the 10th ACM Conf. on Computer and Communications Security. Washington: ACM Press, 2003. 62−72.

[15] Liu D, Ning P. Establishing pairwise keys in distributed sensor networks. In: Proc. of the ACM CCS. Washington: ACM Press, 2003. 52−61.

[16] Blundo C, Santis AD, Herzberg A, Kutten S, Vaccaro U, Yung M. Perfectly-Secure key distribution for dynamic conferences. In: Advances in Cryptology—CRYPTO'92. Santa Barbara: Springer-Verlag, 1993. 471−486.

[17] Perrig A, Szewczyk R, Tygar J, Wen V, Culler D. SPINS: Security protocols for sensor networks. ACM Wireless Network, 2002, 8(5):521−534.

[18] Yang H, Ye F, Yuan Y, Lu S, Arbaugh W. Toward resilient security in wireless sensor networks. In: Proc. of the ACM Mobihoc. Urbana-Champaign: ACM Press, 2005. 34−45.



余磊(1982－),男,安徽六安人,博士生,主要研究领域为无线传感器网络,网络安全.

骆吉洲(1975－),男,博士,讲师,主要研究领域为数据安全,压缩数据库技术.

李建中(1950－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为数据库系统实现技术,数据仓库,半结构化数据,传感器网络,压缩数据库技术,Web 数据集成,数据挖掘,计算生物学.