Increasing Structured P2P Protocol Resilience to Localized Attacks

Zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation von Daniel Germanus, M.Sc. aus Frankfurt am Main
Tag der Einreichung: 23.4.2015, Tag der Prüfung: 10.06.2015
Darmstadt 2015 — D 17

1. Gutachten: Prof. Neeraj Suri, Ph.D.
2. Gutachten: Prof. Dr. Thorsten Strufe

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Fachbereich Informatik
DEEDS Group

Increasing Structured P2P Protocol Resilience to Localized Attacks

Genehmigte Dissertation von Daniel Germanus, M.Sc. aus Frankfurt am Main

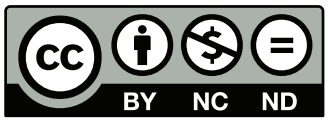
1. Gutachten: Prof. Neeraj Suri, Ph.D.
2. Gutachten: Prof. Dr. Thorsten Strufe

Tag der Einreichung: 23.4.2015
Tag der Prüfung: 10.06.2015

Darmstadt 2015 — D 17

Bitte zitieren Sie dieses Dokument als:
URN: urn:nbn:de:tuda-tuprints-45825
URL: http://tuprints.ulb.tu-darmstadt.de/4582

Dieses Dokument wird bereitgestellt von tuprints,
E-Publishing-Service der TU Darmstadt
http://tuprints.ulb.tu-darmstadt.de
tuprints@ulb.tu-darmstadt.de



Erklärung zur Dissertation

Hiermit versichere ich, die vorliegende Dissertation ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 15. Juni 2015

(D. Germanus)

Abstract

The Peer-to-Peer (P2P) computing model has been applied to many application fields over the last decade. P2P protocols made their way from infamous – and frequently illicit – file sharing applications towards serious applications, e.g., in entertainment, audio/video conferencing, or critical applications like smart grid, Car-2-Car communication, or Machine-to-Machine communication. Some of the reasons for that are P2P's decentralized design that inherently provides for fault tolerance to non-malicious faults. However, the base P2P scalability and decentralization requirements often result in design choices that negatively impact their robustness to varied security threats. A prominent vulnerability are Eclipse attacks (EA) that aim at information hiding and consequently perturb a P2P overlay's reliable service delivery. This dissertation provides the necessary background to understand the different types and inherent complexity of EAs, the susceptibility of many P2P protocols to EAs, and a mitigation technique for the localized EA variant. The applicability of the proposed mitigation technique has been validated experimentally and shows for a wide range of system parameters and application scenarios good mitigation rates reaching up to 100%.

Zusammenfassung

Peer-to-Peer (P2P) Computing hat sich in der letzten Dekade für vielerlei Anwendungsgebiete erfolgreich bewährt. Ursprünglich in der janusköpfigen Filesharing-Szene sehr beliebt, hat P2P Einzug in die Unterhaltungsmedien-Verbreitung, Internet-Telefonie, Videokonferenzen, aber auch in kritische Anwendungsgebiete wie die des Smart Grid, Car-2-Car Kommunikation, oder Machine-to-Machine Kommunikation gehalten. Das grundlegend verteilte Design vieler P2P-Protokolle spiegelt sich u.a. in hervorragender Fehlertoleranz im Hinblick auf zufällige Fehler wider. Weitere P2P-Eigenschaften, wie z.B. Skalierbarkeit und Dezentralisierung haben oftmals Design-Entscheidungen zur Folge, die sich negativ auf die Widerstandsfähigkeit gegen Angriffe auswirken. Ein bekanntes Beispiel aus dem Bereich der Angriffe gegen P2P-Systeme ist die Eclipse Attacke (EA). EAn zielen darauf ab, gewisse Teile eines P2P-Netzwerks auszublenden, d.h., es wird Teilnehmern im P2P-System erschwert oder unmöglich gemacht, auf gewisse Services zuzugreifen. Diese Dissertation widmet sich im Detail unterschiedlichen EA-Varianten und zeigt weitverbreitete P2P-Design-Eigenschaften auf, die zur Verwundbarkeit führen. Des Weiteren schlagen wir eine neue Technik vor, mit der P2P-Protokolle gegen sogenannte lokale EAn resistent werden. Wir haben die Anwendung dieser Technik experimentell für eine große Anzahl Parameter validiert und konnten die lokalen EA-Angriffe in bis zu 100% der Fälle verhindern.

Acknowledgements

The road towards one's Ph.D. is bumpy and paved with accomplishments and setbacks even-handedly. Support in professional and private life is indispensable for this ride.

First of all, I want to thank my supervisor and mentor Prof. Neeraj Suri, head of the DEEDS group, from whom I have learned more than the craft of research. He provided me the flexibility and freedom to create my very own research frame. Interactions with him were always characterized by his wealth of experience paired with his holistic view and unbent attitude.

Furthermore, I want to thank Dr. Abdelmajid Khelil, who was working as a postdoc at the DEEDS group at the time when I joined. He was my daily research sparring partner, is coauthor of various papers, and I have learned a lot from him during many INSPIRE EC project meetings.

I want to thank all my coauthors, especially Hatem Ismail, Stefanie Roos, and Robert Langenberg for the precious discussions we had.

I also appreciate the endless support of Sabine and Ute, my office inmate Thorsten, and the many Ph.D. students and postdocs that constituted the DEEDS group during my time.

Finally, I also want to thank my parents, my brother's family, and my friends for supporting me.

Contents

List of Figures

List of Tables

1 Introduction

Over the last two decades Peer-to-Peer (P2P) computing made its way from infamous – and mostly illicit – file sharing applications towards a diverse set of application domains. These include instant messaging, video conferencing, web technologies, streaming media, JXTA, or VoIP communication to name just a few of them. Furthermore, a diverse set of critical applications is considering P2P as an enabling technology as well, such as critical information infrastructures, Car-2-Car communication, or the more generic case of machine-to-machine (M2M) communication [STFG13, CDF+14]. M2M is essential for the Internet of Things (IoT) or the Internet of Everything (IoE) [Cis15] which provide more application scenarios, for example home automation, pervasive health monitoring through wearable devices, or renewable energy utilization through the Smart Grid. A growth by 45% for mobile, M2M-enabled devices is expected for the timespan from 2014 to 2019, totaling in 11.5 billion devices by 2019 (outranking the projected world population of 7.6 billion at that time) [Cis15].

M2M communication infrastructure requirements correspond very good with P2P protocol features and we believe that various M2M systems will be built upon P2P architectures [STFG13, CDF+14]. Widely appreciated P2P benefits – not only for the field of M2M – are decentralization, collaborative computing, virtual overlay topologies independent of the physical network topology, information aggregation and dispersal, fault tolerance, and scalability into the millions of participating peers (and beyond). Prominent P2P protocol use cases are data dissemination and data discovery. The first one provides data to a multitude of consuming peers, as it is the case for streaming media or file sharing. The second one, i.e., data discovery, provides fast access to a large amount of comparably small data sets. Usually, these two use cases are combined in a single system, in other words, no data dissemination comes without data discovery.

Several widely applied design choices in P2P protocols foster scalability and fault tolerance. For example, peers have only partial knowledge about the overlay network, this is a choice to underpin scalability as knowing all other present peers of the overlay would quickly overstrain maintenance efforts. Moreover, their communication is redundantly designed to overcome failing peers and ensure delivery.

A dependable and secure system is clearly desirable, especially when those systems take responsibilities in autonomous car communication or health monitoring. While a dependable system overcomes random failures in hard- and software with common responses like redundancy and replication techniques, these responses are naturally insufficient to deal with an adversary that is attacking the system. Adversaries focus on specific system weaknesses and once they have identified them, they might modify the system, and potentially sneak in their own malicious resources to finally put the overall system at stake.

The so called *Eclipse Attack* (EA) [SCDR04] is in the focus of this thesis. It exploits fundamental P2P protocol features to launch an attack comparable to a combination of man-in-the-middle, routing table poisoning, and distributed denial-of-service attacks, which are individually well known from client/server architectures. As a consequence, attacked peers are hindered in providing their correct service to the rest of the overlay. In more detail, we analyze EAs with substantial focus on localized EA variants [SNDW06, SENB07, CCFD13], i.e., those attacking a specific subset of peers in an overlay. Peers ought to be *symmetric*, i.e., they should provide both, client and server functionality at once. Also, peers have to provide identical software interfaces for exchangeability reasons. While

peers are considered symmetric in terms of service interfaces, the data maintained by peers is highly asymmetric and the amount of data copies in an overlay is rather small compared to the overall amount of peers. Hence, an attacker might only harm a limited set of peers and therefore launches a localized EA against that set. We expect that the importance of localized EA variants grows with the overall amount of connected devices in the overlay network, since non-localized variants turn out to be unprofitable in large systems due to their high cost. At next, we provide a technical problem description for the susceptibility to localized EAs.

1.1 Problem Context

EAs have been discussed for more than a decade [SCDR04] and over the last years localized attacks [SNDW06, SENB07, CCFD13] became the focus of attention. Although several P2P protocol design choices were originally intended to provide scalable, decentralized, and fault-tolerant systems, they can be exploited as a weakness in order to launch EAs. We mainly account the following two reasons for that:

1. *Partial knowledge* - each peer has only partial knowledge of the overlay network due to the routing table structure and its capacity. This allows for scalability and a moderate maintenance overhead. As a consequence, peers do rely on other peers when requesting routing table entries.

2. *Deterministic behavior* - to promote timeliness and scalability, several fundamental P2P protocol operations are subject to a deterministic behavior. Consequently, attackers are able to identify hotspots for attacks with very high chance.

We narrow the technical focus of this thesis to the localized EA susceptibility of *structured P2P protocols*, which are also referred to as *distributed hash tables* (DHT); a protocol class that is usually applied for data discovery services. In particular, a manifestation of the two previously described weakness reasons is the *lookup mechanism* in DHTs which happens to be a frequently used security exploit. Lookups are required to resolve contact information of other peers prior to message exchanges. Maliciously acting peers that intercept such lookup messages are consequently able to inject malicious information in the overlay network and thereby launch an EA. Moreover, lookups unveil a deterministic behavior which manifests itself in convergence towards a destination peer. In other words, queries are sent on each lookup algorithm iteration to peers closer to the destination peer. Consequently, specific hotspots in the overlay graph can be discovered which allow for highly efficient lookup message interception. We will refer to localized EAs that exploit such hotspots as topology-aware localized Eclipse attacks (taLEA).

In this thesis, we seek for a taLEA assessment and mitigation. The latter should retain P2P protocol features to allow its application for many scenarios.

1.2 Research Questions and Contributions

Given the high efficiency of taLEAs and P2P's dispersion in various important application domains, we have formulated the following two research questions that steer the research presented in this thesis:

- **(R1)** - What is the impact of taLEAs?

- **(R2)** - Is there a taLEA mitigation technique that does not sacrifice P2P's benefits?

Given the lack of a systematic discourse or models covering security issues in P2P systems, the thesis developed the background contribution **(C0)** of a detailed taxonomic security survey. Moreover, we address the previous two research questions by three substantial technical contributions to the P2P security research community.

- **(C0)** - P2P taxonomic security survey.
- **(C1)** - Assessment of taLEA impact.
- **(C2)** - taLEA mitigation with divergent lookups using random walks.
- **(C3)** - P2P address space slicing strategy to improve divergent lookup efficiency over random walks.

The **contributions** in this thesis provide answers to our previously stated research questions. Our first technical contribution **(C1)** discusses the taLEA, which is central part of our investigations, in the context of three different P2P protocols, i.e., Chord [SMLN+03], Pastry [RD01], and Kademlia [MM02]. We highlight the severity and efficiency of taLEAs and furthermore come up with baseline measurements. These allow to compare the reliability with and without our mitigation technique which is presented as second contribution **(C2)**. The technique proposes our novel *divergent* lookup approach. This approach lacks deterministic behavior as opposed to the lookup mechanisms that is typically found in structured P2P protocols. Therefore, it is harder to intercept divergent lookup calls as opposed to convergent ones. In contrast to several mitigations for P2P attacks, our approach of divergent lookups supports scalable, fully decentralized, and anonymous operation. Also, our approach does not require additional infrastructures to certify the benignity of lookup results. Contribution **(C3)** provides an optimization over the random walk approach proposed in **(C2)** through a novel technique called P2P address space slicing. We have validated divergent lookups in a comprehensive, realistic simulation case study and were able to mitigate taLEAs on average for 95% to 100% of the cases.

1.3 Publications

This thesis is based upon the following articles:

- **Daniel Germanus**, Hatem Ismail, and Neeraj Suri, *PASS: An Address Space Slicing Framework for P2P Attack Mitigation*, submitted to the 34th IEEE Symposium on Reliable Distributed Systems (SRDS), 2015
- **Daniel Germanus** and Neeraj Suri, *Security Aspects of Peer-to-Peer Protocols*, submitted to ACM Computing Surveys (CSUR), 2015
- **Daniel Germanus**, Stefanie Roos, Thorsten Strufe, and Neeraj Suri, *Mitigating Eclipse Attacks in Peer-to-Peer Networks*, in Proceedings of the IEEE Conference on Communications and Network Security, San Francisco, CA, USA, pp. 400–408, 2014
- **Daniel Germanus**, Robert Langenberg, Abdelmajid Khelil, and Neeraj Suri, *Susceptibility Analysis of Structured P2P Systems to Localized Eclipse Attacks*, in Proceedings of the 31st IEEE Symposium on Reliable Distributed Systems (SRDS), Irvine, CA, USA, pp. 11–20, 2012

Besides the previously listed articles, the author has published articles related to critical information infrastructure protection and how P2P protocols may increasing the resilience of Smart Grid applications. These articles are related to but not in the focus of this thesis. The author's full publication record can be found in Chapter 7.

1.4 Thesis Structure

We provide at first a detailed overview of P2P security aspects, attacks, and mitigation techniques (Chapter 2) as contribution (C0). Then, we further detail a structured P2P protocol system model alongside with our attack model (Chapter 3). Subsequently, a baseline evaluation is conducted using heuristics and experiments in Chapter 4 which corresponds to our first technical contribution. Afterwards, we present the mitigation technique in Chapter 5 as our second and third technical contribution. The thesis concludes with a summary, conclusion, and future work provided in Chapter 6.

2 P2P Security Survey

This chapter refers to contribution (C0) of this thesis.

2.1 Introduction

Peer-to-Peer (P2P) protocols are a specific variant of distributed systems. Their popularity is driven by characteristic P2P features of scalability, decentralization, and cost. Scalability implies that no changes to the protocol design are demanded with increasing amounts of peers. Whereas a client/server computing architecture demands increasing back-end resources with increasing numbers of requests, this is not the case for P2P due to its by-design decentralized architecture. Furthermore, the decentralized P2P system designs promote inherent resilience against individual peer failures. The peer population itself represents the service provisioning infrastructure of the system. Thereby, potential service consumers are required to partake in resource provisioning making the need for dedicated datacenters void. Over the past decade, a multitude of P2P protocols have emerged. Regardless of their specifics, they usually combine the following five principles: (i) symmetry of interfaces as peers take coincident duties of servers and clients, (ii) resilience to perturbations in the underlying network substrate and to peer failures, (iii) data and service survivability through replication schemes, (iv) usage of peer resources at the network's edge, imposing potentially low infrastructure costs and fostering scalability as well as decentralization, and (v) address variance of resource provisioning among peers.

These five principles make P2P a vital foundation for a diverse set of applications. Originally, P2P was (in)famous for filesharing, but nowadays it can be found in social networks, multimedia content distribution, online games, Internet telephony services, instant messaging, the Internet of Things, Car-to-Car communication, supervisory control and data acquisition (SCADA) systems, and wide area monitoring systems (WAMS), e.g., in the context of the power grid or the Smart Grid. Some of the previously mentioned application fields are either time-critical, safety-critical, or both. They demand dependable and secure operation to prevent hazards that may result in losses of life or equipment. Hence, researchers and engineers should factor in three important aspects related to P2P security:

1. P2P attacks and mitigation techniques may significantly differ from those found in the client/server system model.
2. Many P2P attacks affect a variety of P2P protocols as a consequence of common design choices.
3. Mitigation design should not impose loss of P2P benefits.

2.1.1 Contributions

On the above introduction, the goals of this chapter are:

1. A comprehensive P2P system and application model.

2. A comprehensive detailing of P2P protocol security issues, “attacks”, and mitigation techniques conducted from P2P classifying viewpoints to simplify understanding the pool of related work.

3. A taxonomy analysis of the *Eclipse attack* (EA) susceptibility as an example case to study.

2.1.2 Survey Structure

An overview on related surveys which discuss P2P security and dependability aspects is given in Section 2.2. Subsequently, fundamental P2P features are described alongside with a generic P2P protocol system model in Section 2.3. The security notion used throughout this thesis is provided in Section 2.4. Section 2.5 provides a detailed overview on typical P2P attacks and their mitigation algorithms, our proposed EA taxonomy, and a detailed discussion of EA mitigation techniques. In all sections, we use the style of discussing prominent works in the subject over an introduction followed by a retrospective discussion as applicable.

2.2 Related Surveys

Many surveys in the field of P2P security and related categories have been published. However, we found that these only address a limited subset of the prominent aspects which are listed in Table 2.1. We aim to close this gap with this chapter. Table 2.2 lists the relevant selected work surveyed as a basis for our aim.

As it has partially been targeted in related work, we discuss security, attacks and dependability, but also further aspects that include mitigation techniques, taxonomic and classification schemes. Moreover, the scope and protocol class coverage of existing surveys is of importance for us in order to ascertain whether the article develops a system and/or application centric view for structured, unstructured, or hybrid P2P protocols.

2.2.1 Security Overview

Among the mostly discussed P2P protocol security mechanisms are authentication mechanisms, secure storage, and secure routing. These three mechanisms allow the implementation of various downstream mechanisms. We now link the different survey articles according to the mechanisms they discuss. Also, the subsequent discussion of attacks refers back to selected security mechanisms. Authentication mechanisms [RHB08, ATS04, VdPVA10] help in maintaining a benign peer population and provide the technical basis for downstream mechanisms such as secure admission, secure storage or secure routing. Secure storage is vital not only for data centric applications in order to prevent attackers from illicit data modifications [DM04, UPS11, Wal03, ATS04]. In a broader sense, illicit data modification in online games is considered as cheating [Kwo09, YK13]. Secure routing enables communicating peers to identify message senders as well as message authenticity [UPS11, Wal03, ATS04, RHB08].

The aforementioned security mechanisms increase the resilience of P2P systems against various attacks. Some of these mechanisms are robust up to a critical mass of colluding malicious peers. Unfortunately, some of these require cryptography and the identification of peers. These requirements may interfere with application requirements like anonymity, heterogeneity, or resource frugality.

Table 2.1: Considered aspects for the related survey discussion.

Aspect	Abbreviation	Description
Security	SEC	Substantial focus on P2P security weaknesses and threats. Articles with a parenthesized checkmark indicate that a generic security consideration is of marginal importance to this survey.
Attack(s)	ATK	Additionally to SEC, articles discuss vulnerabilities and how they can be exploited through specific attacks.
Dependability	DEP	Substantial focus on P2P dependability.
Mitigation	MIT	Complementary to SEC or DEP includes a mitigation or countermeasure discussion.
Taxonomy	TAX	Technical concepts are segregated and discussed in multiple dimensions to reduce complexity.
Classification	CLS	Technical concepts are classified for complexity reduction.
Terminology	TERM	Inconsistent terminology usage across articles is unified or aligned.
Scope	SCP	Scope limitation to protocol (Pro), application (App), or middleware (MW) context (✗ indicates no scope limitation)
Protocol class	PRO	Which protocol classes are part of the discussion (S: structured, U: unstructured, H: hybrid)

Table 2.2: Aspect coverage of related surveys.

Citation	SEC	ATK	DEP	MIT	TAX	CLS	TRM	SCP	PRO
[Wal03]	✓	✓	✗	✓	✗	✗	✗	Pro	S
[ATS04]	✓	✓	✗	✓	✗	✓	✗	✗	S,U
[DM04]	✓	✗	✓	✗	✗	✓	✗	✗	S,U
[LCP+05]	(✓)	✓	✗	✗	✓	✗	✗	✗	S,U
[RHB08]	✓	✗	✗	✗	✓	✗	✗	✗	S,U,H
[Kwo09]	✓	✓	✗	✓	✗	✓	✓	App	S,U,H
[VdPVA10]	✓	✓	✗	✓	✗	✓	✗	✗	S,U,H
[UPS11]	✓	✓	✗	✓	✗	✗	✗	Pro	S
[Pas12]	(✓)	✗	✓	✗	✗	✗	✗	✗	S,U,H
[GIP+13]	✗	✗	✗	✗	✗	✓	✗	✗	S,U,H
[KT13]	✗	✗	✓	✗	✗	✓	✗	MW	S,U
[KKHY13]	(✓)	✗	✗	✗	✗	✓	✗	✗	S,U,H
[LMJV13]	(✓)	✗	✓	✗	✗	✓	✗	✗	S,U,H
[YK13]	✓	✗	✓	✓	✗	✓	✗	App	S,U,H
This paper	✓	✓	(✓)	✓	✓	✓	✓	✗	S,U,H

2.2.2 Attack Overview

Attacks against P2P systems usually show an impact in terms of the system's availability, integrity, or confidentiality. Several of the attacks are known from other system architectures, such as client/server, others are completely new or composed of various attacks. We highlight now the different attacks along with the corresponding articles that provide a discussion of them.

Denial of service attacks degrade or prevent a system from correct service delivery [DM04, VdPVA10]. The more sophisticated Sybil attack [UPS11, VdPVA10, Kwo09] can be used as a potential precursor for an Eclipse attack [UPS11, VdPVA10].

If either secure storage, secure routing, or authentication mechanisms cannot be provided, a set of attacks including omission, content forgery, content pollution, censorship, or routing table poisoning may be the consequence [VdPVA10, Kwo09].

Churn denotes the effect of joining and leaving peers in an overlay. Churn attacks consider artificially induced churn with potentially high rates to cause bandwidth consumption due to overlay maintenance. This leads in the worst case to denial of service or its degradation [Kwo09].

Different cheating attack strategies exist for massive multiplayer online games (MMOG) which are built on top of a P2P system architecture [Kwo09, YK13].

The adversarial collusion of malicious peers is a key factor to launch the aforementioned attacks with a significant impact. In many cases, inherent P2P design choices which foster scalability and fault tolerance are exploited.

2.2.3 Dependability Overview

Usually, P2P protocols are resilient to dynamic behavior such as random peer failures. Scalability is also provided by many P2P protocols, i.e., efficient overlay operation is maintained while the peer population grows. In this subsection, these dependability strengthening features are linked to the articles that discuss them.

Replication is a common technique to achieve fault tolerance for data storage [DM04, YK13, Pas12]. Robust content discovery, search algorithms, and lookup mechanisms address fault tolerance [DM04, KT13, Pas12]. Also, consistency and persistence address fault tolerance and are dedicatedly discussed in [YK13].

Scalability is an essential feature for large-scale dependable systems, especially when their design targets openness to new participants and the amount of participants is unknown during design and deploy time. To this end, overlay maintenance, searches and lookups are required to be efficient [DM04, LMJV13, KT13, YK13].

Furthermore, robust routing mechanisms address overlay resilience to failing peers; this can be achieved, for example, by independent paths, multiple overlay dimensions [DM04], or robust topology creation [LMJV13].

Those features are well established and provide for good robustness against non-malicious faults. Partly, they provide basic attack resilience but only up to a critical partition size of colluding malicious peers.

2.2.4 Mitigation Overview

Following this synopsis on common P2P security and attack issues, we now present an overview of mitigation techniques. These address either protocol design weaknesses or specific attacks.

The hardening of protocol extensions has been proposed to replace features which reveal design weaknesses [UPS11]. Furthermore, partitioning resilient topologies, robust routing, and monitoring schemes to check for routing table consistency and peer protocol compliance exist [VdPVA10, Wal03]. Incentive mechanisms to penalize misbehaving peers or so called free-riders have been proposed [ATS04, Kwo09, YK13]. Moreover, cheaters in MMOG can be detected, prevented and penalized [YK13].

While these mitigation techniques increase the resilience to specific attacks, they naturally come at a cost. Computation cycles and bandwidth consumption are required to support mitigation, also some techniques require to access external infrastructures or to append additional hardware resources to the overlay.

2.2.5 Taxonomy Overview

In the context of security related P2P surveys, existing taxonomies address P2P protocol comparisons [LCP+05, RHB08] as well as resource and data structures [RHB08]. While this supports the selection process of a suitable P2P design scheme for given requirements, those taxonomies fail to address security considerations.

A taxonomy allows us to segregate a technical concept across specified dimensions. Eventually, dimensions can be augmented with numerical scales to allow for quantitative comparisons of different problem instances. This allows to compare different specific instances – attacks in our case – with each other. This also helps to conduct mitigation planning based on attack severity estimations.

2.2.6 Classification and Terminology Overview

Often, P2P protocols are classified in a scheme to compare across application domains and features such as scalability, fault tolerance, or security [DM04, ATS04].

Moreover, a security goal oriented protocol classification based on a data/control centric application differentiation is given in [VdPVA10].

A classification of resource discovery mechanisms is provided in [LMJV13], and a traffic classification scheme either on packet or flow level is discussed in [GIP+13].

Furthermore, group management mechanisms are classified in [KKHY13] based on generic admission/maintenance, indexing, and resource discovery aspects.

Cheating categories in MMOGs are classified in [YK13], closely related reputation management techniques are presented in [ATS04, Kwo09] as well as incentive mechanisms like payments or auctions [Kwo09].

A unified terminology for game theoretic modeling of data provisioning and retrieval in overlay networks is given in [Kwo09].

Overall, a classification allows for aspect oriented comparisons, e.g., to outline different articles that use a specific hardening or mitigation technique. Similarly, a unified terminology is helpful. For instance, when different groups of authors have published work about the same or a closely related topic but use differing or even contradicting terminology. A simple approach to achieve terminology unification is to put different articles on top of a common system model, and then subsequently discuss notions alongside with a mapping between different terminology uses if necessary.

2.2.7 Summary & Research Gap

Table 2.2 provides an overview of related survey articles and their aspect coverage. We criticize that multiple presented aspects should be discussed in an interlinked manner, e.g., articles about security issues that may need to consider dependability aspects lack insights about protocol inherent defense mechanisms and how they cater for resilience as well. Furthermore, surveys usually discuss a multitude (or few groups) of closely related articles, in such cases classification, terminology, or taxonomy schemes are strongly recommended and unfortunately missing in many places.

We aim to close such gaps of incompletely covered technical aspects. To this end, we provide a classifying overview on P2P security weaknesses, attacks, and mitigations. The classification reveals relationships between different weakness and attack classes in regard to the affected security goals. Moreover, we provide a taxonomy for the class of Eclipse Attacks (EA). We have picked EA as it affects all three security goals (i.e., availability, integrity, confidentiality), is a combination of various attacks, and multiple variants are possible.

From a methodological viewpoint, we want to stimulate security researchers to pursue a holistic approach in order to provide a full coverage of related technical aspects, highlight coherences/commonalities/differences in specific fields, and also show new directions.

2.3 P2P Systems

The P2P computing paradigm may refer to the networking or the computational/storage aspect of a distributed system. P2P exploits resources at the edge of the network, i.e., there are no dedicated data centers but participants are instead required to share their resources.

This fosters a design with no or few data warehouses and enables low profit or non-commercial organizations an architectural stepping stone to develop large-scale applications. For example, large file downloads in the open source community are nowadays offered through P2P distribution as well to financially distribute their infrastructure cost which is driven by direct downloads. Other examples are contemporary instant messaging and video telephony applications, most of them are – except their authentication and billing services – based upon fully decentralized P2P technology. Moreover, distribution and decentralization improve the data survivability and provide a basis for fault tolerance in terms of both the communication and computation capabilities. Another common P2P feature is scalability, which refers to maintaining growing and shrinking systems without requiring inherent changes to the system's design.

In the following, we provide an overview of different P2P systems categories.

2.3.1 P2P Application Categories

We consider two leading dimensions for the classification of P2P based applications, namely *intent* and *extent*. The intent defines the major goal of the application, i.e., data dissemination or data discovery. The extent, on the other hand, describes the amount and emergence of data the application is dealing with. The two main extent classes are control-centric applications and data-centric applications. We provide an overview of the different combinations of intents and extents in Table 2.3. Data dissemination focuses on fast provision of contents to a large amount of consuming peers. In contrast to dissemination, data discovery is addressing the persistent and survivable storage of data or pointers to data. For example, control-centric data discovery applications include P2P messaging with small message sizes in aperiodic intervals. On the other hand, control-centric data dissemination

applications exchange messages, e.g., using a periodic scheme with lower messaging frequencies. Quite different, data-centric data discovery applications show an aperiodic messaging behavior with comparably small message overhead opposed to data-centric data dissemination applications. We provide further examples in Table 2.3.

Table 2.3: P2P application dimensions.

Intent	Extent	Example
data dissemination	control-centric	Crititical information infrastructures, e.g., SmartGrid
data dissemination	data-centric	streaming media, e.g. P2PTV
data discovery	control-centric	maintenance, e.g., JXTA searching/resolver services
data discovery	data-centric	indexing in DHTs, e.g. BitTorrent

2.3.2 P2P Protocol Categories

The two major P2P protocol designs are unstructured and structured protocols, and they correlate with the application categories that have been introduced in the previous section, i.e., unstructured protocols are mostly suitable for data dissemination, whereas structured ones are usually applied for data discovery. Hybrid protocol designs combine aspects from unstructured and structured ones within one system.

Moreover, hierarchical systems partly contradict the P2P principle that all peers are *equal* in the sense of service provision. These systems can be considered layered, e.g., for two layers as a composite overlay consisting of front-end and back-end peers.

The next subsection provides a generic technical overview on P2P protocols and subsequent subsections detail the aforementioned P2P protocol categories in regard of the overlay topology, resources discovery, and message passing.

Generic P2P Protocol Description

The basic building blocks for a P2P overlay network are join/leave operations, maintenance, and message exchange among peers. The P2P application layer usually builds its implementation upon these few basic features.

In order to be part of an overlay network, a *join* (or *bootstrapping*) operation is required. Explicitly joining the overlay allows peers to (i) announce their presence and (ii) initially receive information about other peers on the overlay and global configuration parameters. Peers maintain a specific *identifier* which is valid for the overlay network in order to decouple from lower layer identifier schemes such as an IP address.

Typically, several protocols implement a *leave* operation which allows peers to announce their intention to quit the overlay network ahead in time of the actual disconnection. This allows peers in the overlay to reassign resources, possibly copy data from the leaving peer and to remove its contact information from routing tables.

Contact information about peers is usally stored in *contact lists* (also *neighbor lists* or *routing tables*) on each peer. Contact information consists of a tuple of the overlay identifier or *key* and the corresponding underlay network address, e.g., IP address and port number. Overlay identifiers are often

a 128 or 160 bit string. They may be arbitrarily chosen or computed from a feature of the peer, e.g., IP address, MAC address, inventory or serial number.

The *topology* of a P2P overlay network is defined as a directed graph $D = (P, E)$ where peers are represented by the graph's vertices $p \in P$. Moreover, p's contact information about other peers $o \in P \setminus \{p\}$ is represented by edges $(p, o) \in E$ on the topology graph.

Recurrently running tasks are part of the *maintenance* operation. This includes the activity to ping peers from the contact list in order to check for their liveliness and the removal of unresponsive peers' contact information. Also, peers might propagate contact information of known peers to specific peer subsets on the overlay such as replica groups.

Moreover, peers require a message passing mechanism for any interaction among them. Therefore, peers implement a *send* and a *receive* operation. The following two subsections present the different technical approaches for structured and unstructured protocols.

2.3.3 Unstructured P2P Protocols

Representatives of the unstructured P2P protocol class [Rip01, CRB+03, GKSS13] are mainly used for data dissemination applications. Their topology is usually embedded within the physical underlay network topology and unveils often tree or mesh like subgraphs which allow for low latency message exchange, e.g., to address timeliness requirements of data dissemination applications. Tree topologies can be found, e.g., in single source streaming media data dissemination with various consumers as leave nodes. Meshes are the more generic case, e.g., in applications with multiple sources and sinks.

Unstructured P2P protocols usually search for resources (i.e., peers and data) and, in contrast to structured protocols, do not use an addressing scheme. Peers nevertheless maintain an identifier to allow independence of the underlay network address. Resources are discovered using search algorithms on the overlay graph. Examples for search algorithms are breadth-first search, depth-first search, random walks, or expanding ring searches. These can be varied according to application requirements.

Message passing may be direct, i.e., using an underlay network connection between two peers, but this usually requires that peers know each other upfront. In case the destination peer for a message is unknown, it may be piggybacked with a resource discovery operation.

Peers maintain lists with contact information about other peers. Hence messaging works efficiently and the network does not suffocate from search messages. The efficiency of such lists depends on the peers' liveliness. Therefore, stored peers are pinged periodically and removed in case no ping reply is received.

2.3.4 Structured P2P Protocols

Structured P2P protocols [SMLN+03, RD01, ZHS+04, MM02] are mostly applied for data discovery applications. Their topology graphs usually show small-world properties, i.e., there exists a path between any two peers with a relatively small amount of edges. Structured topologies appear as ring structures with crosslinks, which form a basis for scalability and efficient operations like resource discovery and message passing. Some protocols unveil more exotic topologies, e.g., butterfly graphs [MNR02], or a multi-torus [RFH+01].

In structured P2P protocols, pointers to resources (peers or data) are stored in a distributed data structure called *distributed hash table*. The overlay's *address space* is usually an integer scale in the range of $[0, \ldots, 2^w - 1]$ with key length w being 128 or 160 in general. Usually, a *distance function*

$d(a, b)$ is defined which allows distance computations between any two identifiers a and b in the address space. Distance computations are crucial for the lookup mechanism and data storage responsibilities. The distance function and its properties differ among protocol implementations. Data discovery is realized by computing the key of an easy-to-grasp resource identifier like a given name and subsequently requesting that key and the data associated to this key from one of the responsible peers.

Messages – e.g. to request the data for a given key – are exchanged in most structured protocols directly, i.e., using an underlay network connection between two peers. If peers do not know each other, no direct connection can be set up and the destination peer needs to be resolved at first. To this end, an overlay lookup mechanism exists, which aims to steadily decrease the address space distance towards the destination on each iteration of the lookup algorithm until the identifier could be resolved. This design approach turns out to be very efficient and promotes scalability. Once the lookup has retrieved the destination's underlay network address, messages can be exchanged. Lookup variants include iterative or recursive algorithms as well as parallelized queries to a set of several closest peers each time.

Routing tables usually store $k \cdot w$ entries with k being a protocol specific constant. Moreover, for the i-th portion of k entries with $i \in [0 \dots w]$, the peer stores contact information of peers that share i common prefix bits of the peers' key. In other words, routing tables usually provide more storage for closer peers than more distant ones. Also, routing tables should keep only information about live and reachable peers, which is why peers are periodically pinged. In structured protocols, maintenance is more expensive as the topological structure needs to be retained, e.g., newly joined peers have to be put in appropriate peer's routing tables and leaving/unresponsive peers have to be replaced by live ones in many peers' routing tables.

2.3.5 Hybrid P2P Protocols

Hybrid variants of P2P protocols integrate unstructured and structured aspects, as their intent is focusing on data discovery and data dissemination. A prominent hybrid protocol example is BitTorrent [Bit08]. It was originally an unstructured protocol but has been extended with structured P2P features to provide a fully decentralized data discovery mechanism. Consequently, BitTorrent could abandon the concept of so called tracker servers in favor of the newly integrated decentralized data discovery features to improve the system's availability.

2.3.6 Hierarchical P2P Protocols

Usually, peers in a P2P system are considered *equal* in terms of the service they provide. Yet, for some application cases, it turned out that a hierarchical P2P design can be advantageous. In hierarchical designs, peers are categorized. Categories can be based on their bandwidth, latency, storage, or computation cycles provisioning. Also, to address churn effects, the (expected) peer online time may function as a categorization measure. Usually, the category with fewer peers represents the back-end part of the hierarchical system, whereas the multitude of peers act as front-end peers which process service requests at the first level and only forward requests to the back-end, in case they cannot fulfill the service request in the first place. This design has proven successful, for example, in the eDonkey file sharing system.

2.4 On the Attackability of P2P-based Distributed Systems

We now describe security goals that are well established in the literature and discuss the assets of a P2P system. After introducing these basic notions, we discuss different attack categories and relate them to the affected security goals and assets. This helps the reader to define security goal requirements for the system assets in specific application cases, assess risks stemming from attacks, and to plan mitigations.

We emphasize that our focus is limited on attacks *against* P2P systems and do not consider attacks that are prepared or conducted *using* P2P systems in order to harm non-P2P systems.

2.4.1 Security Goals

We refer to the established security notion of [ALRL04] for availability, integrity and confidentiality as defined below. Whenever a definition refers to authentication, we assume that peers are implicitly authenticated after joining the overlay network. P2P protocols may be extended using admission control systems or are completely open for arbitrary peers. The three security goals are defined as follows:

1. **Availability:** readiness for correct service of authenticated peers.
2. **Integrity:** absence of unauthorized system alterations.
3. **Confidentiality:** absence of unauthorized disclosure of information.

These fundamental goals help to classify the target and impact of P2P attacks.

2.4.2 Assets

Assets are abstractions from technical concepts of the system which have to be protected in regard of the previously defined security goals, in order to maintain a secure system. We consider two types of assets in P2P systems, namely:

1. *Operations* (Op) accessible through the service interface of the P2P protocol. These assets are accessible on network level.
2. *Data structures* (DS), e.g., data stored in a peer's routing table or resources that are shared with other peers of the overlay network. These assets may be accessible at network level or locally on the peer's host machine.

We will refer to these two assets (Op and DS) in the following subsection where we discuss different P2P attacks. Table 2.4 provides a comprehensive overview that relates attacks to security goals and assets.

2.4.3 Attacks

We now present the different attacks that are specific to P2P systems. Besides the (distributed) denial of service attacks, which are well known from client/server system architectures and apply to P2P as

well, most attacks exploit fundamental P2P features such as decentralization and the partial view on the system of each peer. Consequently, attacks aim at tricking other peers by provision of incorrect data, e.g., to mislead peers in terms of routing. Moreover, attackers could take an advantage in terms of own or others' resource provisioning, overcome limitations in voting systems or games, or selectively hide information in the overlay.

- *Denial of service attacks* (DoS) [ALRL04], *distributed denial of service attacks* (DDoS), or *disruption attacks* [WZR08] are well known from system architectures other than P2P, e.g., client/server computing. In the case of P2P architectures, the attacker aims to decrease the overlay network's service availability by excessively sending messages to a specific set of peers and thereby negatively affect the Op asset. This could affect the join/leave mechanism, or arbitrary other service aspects, e.g., put/get operations in a DHT. For example, benign peers may be impaired by an excessive maintenance workload. Moreover, DoS and DDoS attacks have a negative impact on bandwidth usage and resource provisioning which may result in degraded service provision.

- *Pollution attacks* [SSNRR10, BdAMNCdSBV13] or *index poisoning* [LNR06] aims at the P2P system's integrity and its DS asset by adding incorrect information to the P2P system. Consequences of pollution attacks are the proliferation of polluted content resulting in service impairments.

- *White washing* [BdAMNCdSBV13, CLB09] or *censorship attacks* aim at the availability or integrity of P2P systems. This includes either illicit changing of, deletion of, or denying access to data. Thereby, these attacks endanger the DS asset.

- *Collusion attacks* aim at the availability, integrity, or confidentiality of P2P networks. Collusion refers to the fact that a sufficiently large subset of peers colludes to carry out a strategy which targets at the P2P network's services and thereby negatively affects the Op asset. Usually this is done to override controlling mechanisms, e.g., for reputation or trust management, or bandwidth provisioning.

- *Routing attacks* aim at compromising the availability or integrity of P2P networks. Routing attacks play an important role in composite attacks, such as the Eclipse attack. In routing attacks, a malicious peer undermines the message passing mechanism, e.g., by dropping or delaying messages. Another routing attack variant is *routing table poisoning* [NR06]. In this attack, an attacker deliberately modifies its own or other peers' routing tables, e.g., by returning bogus information to benign peers' lookup requests. *Attraction and repulsion* [WZR08] are specific variants of routing attacks which either increase (attraction) or decrease (repulsion) the attractiveness of peers, e.g., during path selection or routing table maintenance tasks. These attacks negatively affect the DS asset.

- *Sybil attacks* [Dou02] aim at the availability or confidentiality of P2P networks and can be regarded as a specific version of *node/peer insertion attacks*. They consider the insertion of peers into the overlay which are controlled by a colluding adversarial party or a single adversary. This could happen at specific or arbitrary locations on the overlay's topology, depending on the attacker's aim. Furthermore, P2P applications may consider system users as legal entities and consequently restrict the amount of peers per user to the amount of allowed votes for that entity. Hence, a disbalance is brought to the system in terms of expected amount of peers

Table 2.4: P2P attacks combined with harmed security goals and assets.

Attack	Availability	Integrity	Confidentiality	Assets
DoS/DDoS	✓	✗	✗	Op
Pollution	✗	✓	✗	DS
White washing & censorship	✓	✓	✗	DS
Collusion	✓	✓	✓	Op
Routing	✓	✓	✗	DS
Sybil	✓	✗	✓	Op
Buffer map cheating	✓	✓	✗	Op
Eclipse	✓	✓	✓	DS, Op

per user. Sybil attacks may be a precursor for many of the previously described attacks. Sybil attacks affect the Op asset of the system.

- *Buffer map cheating attacks* [LWC09] aim to decrease the availability of P2P networks. Through this attack, the adversary reduces the outgoing traffic load of his peer(s) by lying about its data provisioning which is also an integrity infringement and affects the Op system asset. This attack is especially applied to streaming media P2P applications which rely on the collaboration of peers.
- *Eclipse attacks* [SNDW06, GRSS14] aim to decrease the availability, integrity, and confidentiality of P2P networks. These are composite attacks that may involve routing table poisoning, DoS/DDoS, Sybil attacks, collusion, white washing, or censorship. Consequently, these attacks have an impact on both system assets. Due to the large problem space created by this attack, we propose a unified terminology and taxonomy in Section 2.5.

Summary

The presented attacks consider modifications of the P2P system to either impair or abuse system services. The difference to comparable attacks in client/server system architectures is that P2P overlay networks may grow very large and adversaries have to adapt their efforts as well, i.e., scale the malicious peer fraction accordingly, thereby requiring a substantial amount of coordination to be done for sophisticated collusion strategies. Nevertheless, attacks vary a lot from client/server architectures in a sense that an attacker has probably no physical or local network access to the resources to be attacked. Therefore, malicious peers need for example proper announcement in the overlay network before they may launch their adversarial behavior. This results occasionally in complex attack patterns with plenty of variations to achieve the adversarial goal. We provide an in-depth exposition on different varieties of the Eclipse attack in the subsequent section in order to draw the reader's attention to the complexity of P2P attacks.

2.5 Eclipse Attacks - Definition & Taxonomy

In this section, we provide a detailed discussion of the Eclipse attack (EA), which is hard to detect due to its sophisticated behavior. It affects all three security goals and also both system assets. To

this end, we provide a concise description of the attack, discuss three prominent EA variants as well as detailed attack mechanics of the attack's different phases. We highlight the complexity by providing a taxonomy which characterizes different attack configurations as well as requirements on the adversary. We close this section with an overview of the related work in the EA field with special focus on the mitigation techniques.

2.5.1 Definition

The goal of an EA adversary is to *eclipse* one or more victim peers, i.e., to obstruct the victim's service provision by preventing benign peers from delivering their service requests to the victims. Basically, an EA consists of two phases: firstly, the placement of malicious resources in the overlay network in conjunction with routing table poisoning and, secondly, the adversarial actions carried out on intercepted messages. We use the following terminology to describe the different types of peers in an overlay under an EA. Benign peers $b \in B$, malicious peers $m \in M$, and victim peers $v \in V$, with $B \cup M \cup V = P$, $B \cap M = \emptyset$, $B \cap V = \emptyset$, and $M \cap V = \emptyset$.

Furthermore, we differentiate three EA types:

1. Generic EAs – malicious peers are scattered over the overlay network, and every non-malicious peer is considered as a victim, i.e., $V = P \setminus M$. The overlay is attacked in its entirety.

2. Localized EAs (LEA) – malicious peers are scattered over the overlay network and the amount of victim peers is significantly smaller compared to the total amount of peers on the overlay, i.e., $|V| \ll |P|$. A LEA targets the service provision of a specific peer subset only.

3. Topology-aware LEAs (taLEA) – malicious peers are located at specific topology locations. Consequently, less malicious peers are required than for the LEA case, i.e., $|V| < |M| \ll |P|$. Therefore, a taLEA might be favored by adversaries for long-term attacks.

Examples for the EA, LEA, and taLEA in a structured overlay's address space are given in Figures 2.1 through 2.3. In these examples, the address space is assumed to be a one-dimensional scale, and peers are mapped to unique identifiers on that scale. Malicious peers m_i are illustrated by red circles with the victim selection shown in square brackets. The asterisk in Figure 2.1 represents a generic selection, in contrast to Figure 2.2 where the victim selection contains only peer v, which corresponds to a LEA scenario. Moreover, Figure 2.3 shows a taLEA in which the topology awareness consists of placing malicious peers as direct neighbors of the victim peer. A taLEA usually requires a small and constant amount of malicious peers. For LEAs, on the other hand, previous studies suggest that malicious resources in the range of 10% to 20% are required.

In the next subsection, we describe techniques to launch EAs.

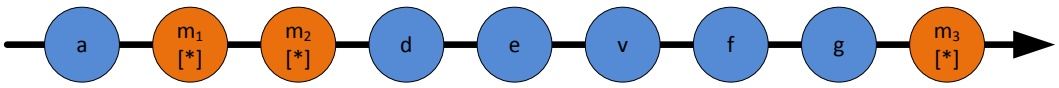


Figure 2.1: Generic EA, malicious peers m_i attack any benign peer.

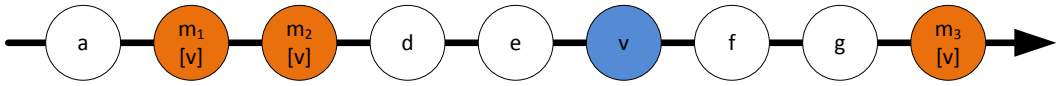


Figure 2.2: LEA, malicious peers m_i attack peers $v \in V$ only.

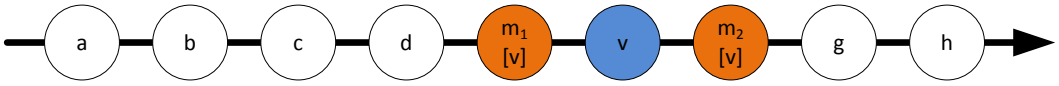


Figure 2.3: taLEA, malicious peers m_i attack peers $v \in V$ only, and make use of topology to decrease $|M|$ in contrast to the LEA scenario.

2.5.2 EA Phase 1 - Activation of Malicious Peers and Routing Table Poisoning

In order to create a malicious peer fraction, an attacker could take over existing benign peers and change their behavior from benign to malicious. Another possibility is the introduction of new malicious peers, which could be joined in the overlay network, either at arbitrary locations or, in the case of a taLEA, at specific locations in the overlay topology. In order to prevent victim peers from servicing requests, malicious peers need to get hold of request messages. As most P2P protocols make use of direct communication between mutually known peers, the sole introduction of malicious resources is insufficient for launching an EA. Peers are mutually known in case they have stored contact information of each other in their contact lists or routing tables (cf. Section 2.3). For the activation of an EA, the contact information of malicious peers is propagated among the benign peer population instead of the victim peers' correct underlay network addresses. In order to achieve this, a routing table poisoning (RTP) attack is conducted, thereby violating the integrity of benign peers' routing tables. Consequently, benign peers that make use of *poisoned* contact information are sending their request messages unknowingly to a malicious peer instead of the benign destination. This is also referred to as a *man-in-the-middle* (MitM) attack. Contact information is usually retrieved using the lookup mechanism in structured protocols and via searching in unstructured protocols. The following examples discuss benign and malicious lookup cases.

Benign Lookup Case

For the benign lookup depicted in Figure 2.4, we assume peer a has no contact information about peer v in its routing table and initiates a lookup in order to resolve v's contact information. A common lookup mechanism design pattern is to send the lookup request to a known peer closest to the destination. We assume a knows b and sends a lookup request message "v?". As b cannot resolve v, it forwards a's request to c, which is closer to v and able to resolve v. Finally, c returns contact information "v!" to a. The dashed pointer depicts a's communication with v, though this is not part of the lookup process.

Malicious Lookup Case

Besides benign peers, Figure 2.5 also shows two malicious peers m_1 and m_2. During a's lookup request for the destination v, peer m_1 is being queried as well and returns wrong contact information to a, i.e., m_1 states that v has the underlay network address of m_2. Consequently, a's routing table

gets poisoned and a initiates communication with m_2 instead of v (dashed pointer, not part of the lookup process).

Note that wrong contact information may also be propagated through benign peers which were poisoned beforehand. Once the routing tables are poisoned and malicious peers start receiving messages intended for other peers, the EA is initiated. Due to RTP, EAs threaten both assets, i.e., operations and data structures, in regard of the integrity security goal. Various adversarial actions are discussed in the next subsection.

Summary

The first EA phase describes how malicious resources become known in the overlay as a prerequisite for partaking in an EA. In contemporary structured P2P protocols, this is achieved by attacking the lookup mechanism, which is required for maintaining the peers' routing tables.

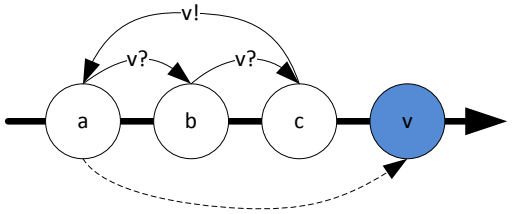


Figure 2.4: Overlay lookup, benign case.

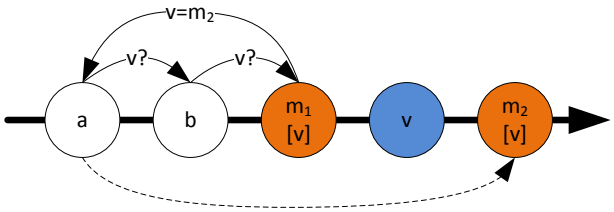


Figure 2.5: Overlay lookup, malicious case.

2.5.3 EA Phase 2 - Carrying out Adversarial Actions

As a consequence from routing table poisoning, malicious peers may take diverse actions. Malicious peers m_i are receiving messages instead of the legitimate victim peers. Then, m_i may launch malicious actions for each illegitimately received message. A straight forward action is service denial, which threatens the operations asset in terms of the availability security goal.

Malicious actions may also require to send reply messages to the benign requesters. These replies may be of the following types: *correct*, *modified*, or *forged*. A reply message is correct if it is subject to benign protocol behavior, whereas information has been deliberately changed by the attacker in a modified message. Finally, forged messages are synthetically created and sent; these are messages which do not correspond to a message in the benign case. In some cases, request message inspection

is required in order to create an adequate reply message. If reply message creation does not depend on the contents of the request message, but, e.g., requires data stored by the victim peer, the malicious peer forwards the original request message (or a forged copy) to the eclipsed victim peer in order to receive a legitimate reply message, which can be subsequently inspected to create a malicious reply message.

The previously described adversarial actions complicate attack detection in contrast to comparably simple denial of service actions.

Summary

The second EA phase describes the actual adversarial action, e.g., dropping or delaying messages. The EA behavior may be consistent or volatile over time, and decides on detectability as well as the collusion induced communication overhead. Clearly, we cannot discuss all potential adversarial actions exhaustively, yet we provide a taxonomic discussion of the different aspects of EAs in the next subsection. This taxonomy does not only help to classify a specific attack but also to plan countermeasure development and threat analyses.

2.5.4 Eclipse Attack Taxonomy

The intention of this taxonomy is to provide a structured view on Eclipse attacks. Thereby, it systematically presents the causes behind an attack, an attacker's intentions and capabilities, as well as the actual technical approach to conduct an attack. Our structure is depicted in Figure 2.6. Each of the tree's nodes represents a qualitative entity. In order to facilitate a quantitative comparison, selected nodes can be augmented with a scale system. The ascertained scale values can be aggregated in a vector notation at each parent node and finally allow for a quantitative comparison of different EA instances by considering the root node's vector. Such comparisons are helpful to prioritize mitigation planning, to estimate damage, or to assess the likelihood of an attack to occur.

Various quantitative security assessment techniques have been proposed, such as the attack surface and its related metrics [MW11, YMR14]. Sometimes, a plain algebraic approach is unfavorable or infeasible. In such cases, simple scale systems that propose a partial order may be sufficient, e.g., scale systems that assign given attribute values on an integer scale from 1 to 10 or the characteristics *high*, *medium*, *low*. There exists the *DREAD* threat classification method [SS04] and a newer, related approach called STRIDE method [HL09].

The taxonomy's main categories are *threat*, *requirements*, and *strategy* and are described in the following subsections.

Threat

The chance that an attacker exploits a vulnerability is called a *threat* to the system. It is composed of *vulnerability*, *assets*, *impact* and *consequences*. Vulnerabilities refer to exploitable weaknesses in the system and may enable access to assets. Once such a weakness is exploited, it has an impact on at least one of the system's security goals (cf. Section 2.4.1).

Assets require protection in terms of the three security goals availability, confidentiality, and integrity. The technical consequence of a threat is that the system's service provision could turn incorrect. In the second place, incorrect service may lead to venue loss, suffered reputation, defamation,

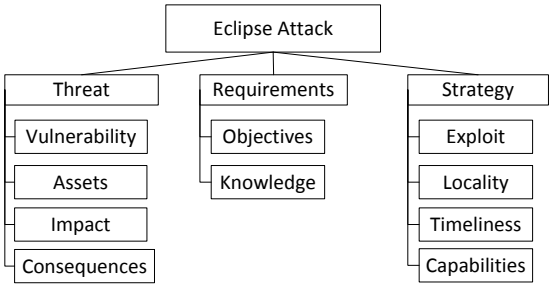


Figure 2.6: EA taxonomy overview.

or safety hazards, to name but a few. In terms of P2P systems, we have identified two assets, namely data structures and operations (cf. Section 2.4.2).

Requirements

This category describes an attacker's *objectives* and *knowledge*.

The objective addresses application specific features in terms of the system's services or business processes which are tied to selected services.

In order to carry out an attack, knowledge about the system to be attacked as well as the system's operational context is required. Knowledge is pivotal for the attack's efficiency, scalability, and detectability. Efficiency relates to effectivity and cost, i.e., the amount of resources an attacker has to spend to achieve a certain impact. Deeper understanding of the P2P application or protocol may decrease attack cost because less resources are required than in a zero knowledge brute force attack scenario. On the other hand, a cost-effective and sophisticated attack approach could be of limited applicability to other application or protocol scenarios due to its specialization. Many P2P protocols are scalable. Hence, the attack is required to scale with an increasing amount of peers in (or requests to) the system. Again, frugality and cost effectiveness require knowledge about the protocol and application. Better knowledge of the protocol or application allows for attacks which are harder to detect, e.g., an attacker could modify message payloads based on application knowledge to bring harm over the system rather than launching a denial-of-service attack, which is straight forward observable by many peers in the overlay. Besides knowing the protocol and application to be attacked, an attacker could consider knowledge about underlying operating systems, the networking infrastructure, and hardware of the system in order to design a specific attack.

Strategy

The strategy defines an EA's technical modus operandi. It is essentially based upon an *exploit* which materializes a threat. Furthermore, the exploit is applied to meet an attacker's objectives by activating undesired system behavior that unveils misuse potential.

The *locality* of where to apply the exploit is derived from the attacker's objectives and knowledge. For instance, it could be applied to all or just a subset of peers. Moreover, a peer subset does not need to be static and could change at any time or periodically.

Timeliness refers to the time range when the attack should be active, i.e., the starting and stopping points. Also, in case the attack's effectivity decays over time, e.g., due to P2P protocol maintenance or churn, timeliness describes in which intervals the exploit needs to be reapplied.

Capabilities refer to the resource pool and accessibility features for the system. The resource pool can vary in size over time and not all resources in the pool might be suitable for an attack. For example, an attacker's resource pool could include tens of thousands of networked machines, but only a small portion of them have suitable features (e.g., IP addresses) that would be mapped to a specific location in the address space in order to efficiently launch an attack. Accessibility features relate to the attacker's knowledge and include, for example, passwords or certificates to surround perimeter protection measures.

Summary

The intention of providing this taxonomy is two-fold. Firstly, we want to motivate the reader to comprehend and assess specific EAs in a structured way to improve on the attack understandability and also the comparability of different EAs. Overall, this may be helpful to understand common causes for different EA variants and to subsequently improve the P2P protocol design and remedy the shortcomings. Augmenting the taxonomy with quantitative models for risk and impact estimation allows, for example, to prioritize during mitigation planning.

2.6 Eclipse Attack Mitigation Techniques

We classify the existing work based on their proposed mitigation techniques. We identified five prevalent mitigation technique classes, namely *redundancy*, *diversity*, *admission*, *authentication & reputation*, and *constraints*. Now, we describe the individual classes alongside with the existing work in each class and discuss their advantages and shortcomings. Moreover, we underline the difficulty of designing techniques that mitigate all three previously introduced EA types (EA, LEA, and taLEA). Table 2.5 at the end of this section gives a structured overview on the techniques and their mitigation potential for EA variants.

2.6.1 Redundancy

Redundancy can refer to both data storage and message exchange. The works below target overlay messaging and apply spatial redundancy to increase the resilience against peer crashes or failures in the underlying networking infrastructure. In EA and LEA scenarios, spatial redundant messaging is an adequate mitigation under the assumption of uniformly distributed malicious peers in the overlay address space.

Overview on existing techniques

In [BM07] and [OC01], the authors propose overlay routing based on multiple and independent paths in order to increase the robustness. Due to the fact that these approaches rely on convergence across various paths or dimensions, the susceptibility to taLEAs remains, although with potentially higher cost.

SALSA [NW06] establishes circuits comparable to the TOR protocol [DMS04] in order to obfuscate the requesting peer. It furthermore protects against EAs and LEAs by performing redundant lookups.

In [HK03], the authors propose wide paths for redundant message forwarding. This technique is a suitable EA and LEA mitigation.

In [AS06], the authors propose redundant messaging and data storage to overcome EAs and LEAs.

Concluding remarks

The aforementioned techniques refer to redundant message sending to overcome failures of individual peers. Therefore, different paths are chosen with specific requirements for the peer selection process, e.g., to disallow duplicate peers for all paths of a given message.

These mitigation techniques are subject to the assumption that malicious resources are uniformly distributed within the overlay network. Consequently, they show good mitigation rates for non-topology-aware attacks (EA and LEA), have an acceptable network overhead, and promote scalability.

In order to also mitigate taLEAs by applying redundancy, the peer selection process could consider additional requirements, such as, distances and sojourn times, to avoid malicious peers. The challenge would be to maintain a comparably good network overhead despite the additional selection requirements.

2.6.2 Diversity

Some attacks, such as the taLEA, cannot be mitigated through existing redundant messaging approaches because the behavior of the message sending algorithm itself represents the weakness. For such attacks, design diversity can be a suitable mitigation, i.e., using different messaging algorithms to overcome, for example, deterministic behavior.

Overview on existing techniques

In [GRSS14], divergent lookups are proposed to mitigate taLEAs for different Kademlia protocol variants with iterative and recursive routing. Divergent lookups do not converge towards the destination peer whose contact information should be resolved and instead declare the proximity around the destination as a no-go area for the lookup process in order to avoid contacting malicious peers in that area.

Concluding remarks

Diversity is a generic pattern for achieving fault-tolerant computing. The previously discussed approach focuses on algorithmic design diversity. Therefore, different algorithm implementations are based on the same specification and combined in a framework to assess each algorithm's result using an acceptance test or voter mechanism. Ideally, each algorithm returns the same result; otherwise, the acceptance test or voter has to cope with differing results in a reliable way and either needs to repeat or proceed with the execution.

Consequently, not only benign failures but also attacks that exploit deterministic behavior in a specific algorithm design may be mitigated with design diversity.

In the context of EA variants, the combination of diverse designs and redundancy techniques could provide for a comprehensive EA, LEA, and taLEA mitigation. Probably, this applies as well to other attack classes and their variants.

A challenge in design diversity is clearly the higher cost for different implementations. Usually, further requirements are tied to design diversity, e.g., runtime complexity and network overhead should be similar, because voter/acceptance tests would otherwise impose a waiting time on those algorithms with better execution times.

2.6.3 Admission

Many EA variants require a set of malicious peers to join the overlay network. To this end, mitigation techniques have been proposed to enforce admission control during the join procedure. For example, peers need to provide a proof for their legitimicy, e.g., by using certificates.

Overview on existing techniques

In [SENB07], Sybil attacks are launched as a preparatory step for LEAs against KAD, which is a Kademlia-based file sharing system. The proposed mitigation makes use of a certificate authority to enable strong encryption for the secure admission and authentication of peers.

In [ZZC+11], the authors propose a computational puzzle in the key generation process to delimit malicious parties from choosing specific or too many keys as it may be the case in EAs, LEAs, or taLEAs.

Concluding remarks

Admission techniques require peers to provide valid credentials during the join procedure to certify their legitimicy.

Previously discussed works either require an additional infrastructure to verify the credentials or apply a decentralized scheme, which mainly focuses on limiting the amount of join attempts per time and per machine.

Depending on the amount of attacker resources, this technique may prevent malicious peers from joining in general, delimit the amount of joining peers, or it does not mitigate at all. The latter case may occur for a very powerful adversary that has a plethora of resources at his disposal including appropriate credentials. Secondly, the attacker could exploit a weakness in the admission system to bypass the limitation. Finally, the adversary could hijack peers that have already joined the overlay network and use these for further adversarial steps.

2.6.4 Authentication & Reputation

The message exchange among peers can be secured using cryptographic methods to maintain the integrity or confidentiality of messages and the authenticity of the sender. Furthermore, reputation systems help peers to assess the trustworthiness of their opposites.

Overview on existing techniques

In [CDG+02], the authors propose secure routing for P2P overlays which is an adequate mitigation technique for EAs, LEAs, and probably also taLEAs – the latter attack case was not assessed by the authors. Disadvantageously, this requires cryptographic primitives and a certificate authority. Consequently, this approach neglects anonymity or openness requirements of some applications.

Shadowwalker [MB09] proposes a verification scheme that demands a certain number of peers to confirm the identity of a peer before messages are exchanged. However, the approach has been shown to be vulnerable to selected denial of service attacks [SDH+10].

In [LYL14], the authors propose using a decentralized quorum to decide on peers' reputations. This helps to mitigate EAs and has been validated for the KAD system.

Concluding remarks

Authentication systems allow for securing overlay service calls between peers by digitally signing or encrypting messages to be exchanged. Reputation systems let a group of peers decide if a specific peer should be allowed for service requests or if its reply should be accepted.

Consequently, an adversarial peer with the intent to eavesdrop/emit service requests or replies would be required to obtain matching credentials, e.g., public and private cryptographic keys which have been registered in a public-key infrastructure, as well as session keys generated among peer groups. As in the previous discussion about admission techniques, an adversary could hijack legitimate peers in order to acquire access to those credentials. Alternatively, the public key infrastructure could be attacked as well. In any case, this requires a high proficiency level of the adversary.

These approaches require a public-key infrastructure which is a limitation for openness since potential participants are required to register with the public-key infrastructure beforehand. Privacy is limited as well since the registration might impose giving up on pseudonymity and anonymity.

2.6.5 Constraints

Selected peer features can be used to define constraints as a mitigating measure. The underlay network address can be considered as such a feature, e.g., to avoid adversaries from using a large range of IP addresses from the same network. Also, for overlay networks which map peers into the address space by using hash functions, an average distance between any two peer IDs can be assumed. In case the average distance is undershot, an ongoing localized attack can be assumed and the given peer is excluded from routing tables. Another assumption can be made about the average and maximum node degree in the overlay graph. Peers which unveil an unexpectedly high node degree might be malicious since some EA variants try to attract as many messages as possible.

Overview on existing techniques

The first mention of LEAs appears as a subsidiary remark in [SNDW06]. Moreover, the authors focus – as in their previous work [SCDR04] – on generic EAs and propose a mitigation through degree bounding, i.e., limiting the amount of routing table entries for each peer. Adherence to this limitation requires periodic checks using an auditing scheme. The authors conclude that degree bounding is probably not a suitable countermeasure for LEAs or taLEAs.

In [CKGM04], the authors propose induced churn to mitigate several attacks based on routing table poisoning. This technique flushes routing tables in periodic intervals in order to gain fresh and ideally benign contact information.

In [CCF09], a Sybil attack prepares a LEA against KAD to eclipse a specific address space range in order to intercept all KAD search requests destined to the victim peer. The proposed countermeasure aims at IP address restriction and a flooding protection scheme.

In [WTCT+08], the authors propose a backpointer hijacking method in order to launch a LEA in the KAD network. The authors propose to mitigate this using routing table policies which disallow peers from reclaiming the same overlay address in case they reconnect with a different IP address.

In [KLR09], two LEA variants against the KAD protocol are presented, where the first variant resembles the approach in [SENB07, CCF09]. The second variant prevents legitimate peers from publishing new content in the KAD file sharing system by placing malicious peers close to the legitimate one. Mitigations in this chapter refer to structural routing table constraints according to the assumption that attacker resources originate from a single – or few – underlay network domains.

In [CCFD13], the authors focus on the KAD network and discuss localized attacks such as the LEA in a publicly accessible file sharing network. They propose a reactive mitigation technique which explores the overlay address space to determine deviations in the distribution of peers' IDs and thereby deduce the existence of malicious peers. The approach has been validated for the KAD file sharing system and shows a small false-negative rate.

Concluding remarks

Constraining techniques focus on characteristics which an adversary could modify and exploit to its advantage. Therefore, degree bounding, routing table refreshing, minimum distance between peers, or specific underlay network characteristics have been proposed as a basis for constraints definition.

While most of these constraints can be monitored and enforced locally, i.e., without requiring message exchange among peers or with an additional infrastructure, constraints unfortunately impose some disadvantages, e.g., a higher network overhead, no guarantees that constraints are satisfiable, and constraints may result in a deterministic structureness of the overlay graph that could represent a new weakness.

2.6.6 Existing Work without EA Mitigation Proposals

In [LMSW10], the authors show the LEA susceptibility of the KAD file sharing network, which is based on a Kademlia protocol modification. In their LEA discussions, KAD's tolerance zone, which can be regarded as an application level proximity concept, is populated with malicious peers. The evaluation focuses on the attack's impact over time.

In [GLKS12], we discuss the systematic taLEA susceptibility from an application independent view and refer to accountable design choices of many structured P2P protocols. The authors conducted an analytical and experimental study to underline the potentially high impact of the attack for the Chord [SMLN+03], Pastry [RD01], and Kademlia [MM02] protocols.

2.6.7 Summary

We have identified five different mitigation technique classes, namely redundancy, diversity, admission, authentication & reputation, and constraints. These classes reduce the impact of attacks to

the system. For example in a pro-actively manner by only admitting peers with credentials as a first hurdle to avoid malicious peers from becoming part of the overlay network. Other classes focus on diversity and redundancy to ensure survivability and availability of data and services in faulty or adversarial scenarios.

As can be taken from Table 2.5, most of the related work addresses EA or LEA scenarios. An important question is how effective the various LEA mitigation strategies are in mitigating taLEA scenarios. The mere fact that they remain unconsidered from a model perspective in various works does not imply ineffectiveness. On the other hand, a reassessment regarding the efficiency of LEA mitigation techniques for taLEA scenarios would be helpful.

After all, a comprehensive mitigation of all introduced EA variants would require a combination of various mitigation techniques.

Table 2.5: Related work in regard of attack coverage and mitigation technique class.

Citation	EA	LEA	taLEA	Mitigation Class
[OC01]	✓	✓	✗	Redundancy
[CDG+02]	✓	✓	✗	Authentication
[HK03]	✓	✓	✗	Redundancy
[CKGM04]	✓	✗	✗	Constraints
[SCDR04]	✓	✓	✗	Constraints
[AS06]	✓	✓	✗	Redundancy
[NW06]	✓	✓	✗	Redundancy
[SNDW06]	✓	✓	✗	Constraints
[BM07]	✓	✓	✗	Redundancy
[SENB07]	✗	✓	✗	Authentication, Constraints
[WTCT+08]	✗	✓	✗	Constraints
[CCF09]	✗	✓	✗	Constraints
[KLR09]	✗	✓	✗	Constraints
[MB09]	✓	✗	✗	Authentication
[LMSW10]	✗	✓	✗	✗
[ZZC+11]	✓	✓	✓	Admission
[GLKS12]	✗	✗	✓	✗
[CCFD13]	✗	✓	✗	Constraints
[GRSS14]	✗	✗	✓	Diversity
[LYL14]	✓	✗	✗	Reputation

2.7 Summary

This chapter presents an overview of P2P protocols, weaknesses, attacks, and mitigations. Therefore, related surveys as well as articles on attacks and mitigations are discussed and categorized from different viewpoints. The Eclipse attack has been chosen as an example for a taxonomic dissection of P2P attacks with the goal to improve the understandability and guide the mitigation design.

P2P has become an established architectural approach for a wide range of recreational but also critical distributed applications. While P2P's scalability and decentralization unveil benefits not only in terms of fault-tolerance and cost effectiveness, P2P protocols are susceptible to a variety of attacks.

Attacks against P2P are often composite attacks, i.e., different weaknesses are exploited at once. Hence, a holistic mitigation technique may be sophisticated, expensive, and potentially sacrifices some of the P2P benefits.

Perimeter protection for service providers, as it is done for client/server systems, cannot be applied to P2P systems as all peers take part in service provision and are scattered across different physical network domains, which does not allow for a concise perimeter definition. Also, service state is distributed among many peers and often replicated in addition.

A good mitigation technique should allow to keep application requirements, such as privacy or anonymity, and furthermore support scalability and heterogeneity. For example, authentication through cryptographic primitives is feasible for selected application scenarios, but it might be undesired in open system scenarios which possibly involve peers with limited computational or energy resources. On the other hand, mitigation techniques which are desirable for decentralized and anonymous approaches may be infeasible for scalability or application timeliness requirements reasons.

An application scenario independent, scalable, heterogeneity fostering, and anonymous solution requires various, intertwined mitigation techniques. Most of the proactive mitigation techniques demand abandoning anonymity or scalability since peers are required to authenticate their service requests and replies. Other techniques put efforts on spatial redundancy to outnumber adversarial resources or diversity to overcome attackers that exploit deterministic protocol behavior. Reactive mitigation techniques audit overlays regarding their compliance to distance metrics and desist from further communicating with peers that display suspicious threshold violations. All in all, a combination of proactive and reactive P2P mitigation techniques is imperative to achieve adequate P2P security in large-scale application secenarios.

3 System & Attack Model

This chapter introduces technical descriptions and notions of the overlay network, the P2P protocol, and the attack that is central to our investigations – the taLEA. Subsequent chapters build upon these foundations.

3.1 Overlay Network Model

The P2P overlay network is modeled as a directed graph $D = (P, E)$ with peers $p \in P$. Each peer p maintains a routing table whose entries point to other peers $o \in P$, $o \neq p$. Whenever such a pointer exists, an edge $(p, o) \in E$ exists. We differentiate between incoming $E^-(p)$ and outgoing edges $E^+(p)$ of a peer p. Furthermore, we split the peer set P into malicious peers M, benign peers B, and victim peers V. $P = B \cup V \cup M$ and $B \cap M = \emptyset$, $B \cap V = \emptyset$, as well as $M \cap V = \emptyset$, and $N = |P|$.

3.2 Structured P2P Protocol Model

Next, we outline seven features that characterize an abstract view on many structured P2P protocols.

3.2.1 Address space

This is the reference point of all resources to be managed in an overlay. The resources are the peers and the data items associated to peers; both are mapped to the same address space. Contemporary P2P protocols use an integer scale range of $[0, 2^w - 1]$ with typically $w \in \{128, 160\}$ as an address space definition. Resources obtain an overlay key κ with length w which is mapped onto the address space. Therefore, an external identifier p_{ext} is passed as parameter to a hash function: $\kappa = h(p_{ext})$. κ therefore represents a unique identifier (moreover, at least with a low collision probability) and should be unmodified for the resource's lifetime. Examples for external identifiers are the peer's IP or MAC address, the hash of a file for a data item, or a random number.

3.2.2 Distance function

P2P protocols implement a distance function for address space traversal as it is required for example by routing, lookups, responsibility and replication mechanisms. Two overlay keys $\kappa_1 \neq \kappa_2$ yield a distance $dist(\kappa_1, \kappa_2) > 0$. Depending on the P2P protocol's topology and address space organization, different types of distance functions are applied, e.g., euclidean distance metrics, linear distance metrics, or the XOR distance metric.

Most of this work focuses on protocols that make use of the XOR distance metric which calculates distances based on the common prefix length (CPL) of two distinct peers. To compute the distance of two peers, a XOR operation is performed on the bitstrings of both keys and the resulting bitstring represents the distance.

3.2.3 Routing table

A routing table is a data structure managed individually by each peer. It contains tuples that relate a peer's overlay key to a tangible underlay network contact information, e.g., an IP address and port combination. Direct communication between peers imperatively demands a contact information. Usually, $c\ logN$ pointers are stored in a routing table and c is a protocol specific constant. Furthermore, the structure of routing tables relates to the address space and distance notion, i.e., peers store k contact information tuples about peers whose key is in a distance range of $[2^i, 2^{i+1})$ with $i = 0 \ldots w-1$ and k constant. With decreasing distance between any two peers, the amount of common bits of their overlay keys increases. Some P2P protocols manage different routing tables, e.g., to store special peers like the closest neighbors or the set of replicating peers independent of the contact information about the generic peer population. To resolve contact information of unknown peers, P2P protocols implement a lookup mechanism.

3.2.4 Message Passing

Peers implement an overlay messaging scheme and therefore provide *send* and *receive* operations. These operations allow to forward messages to peers whose contact information is stored in the routing table and to process those messages. P2P protocol maintenance tasks are based upon the message passing scheme, e.g., the lookup mechanism which is described in the next subsection. Furthermore, the message exchange operations are accessible as a foundation for defining P2P-based applications.

3.2.5 Lookup Mechanism

In case a peer a has to send a message to another peer b but a has no contact information about b in its routing table, then a *lookup* call is performed to resolve b's contact information. To support scalability, lookup mechanisms in most structured P2P protocols are subject to a design best practice which we call *convergent lookups*. The mechanics are very simple: peer a chooses from its routing table peer(s) with the smallest distance to b and sends the(m) lookup message(s). This approach is iteratively or recursively repeated, until either b is resolved or a timeout occurs.

3.2.6 Proximity

The proximity of a peer p describes an address space region around p which is sparsely populated by other peers. The average proximity size varies and depends for example on key length, the number of peers in the overlay, and the address space mapping function.

Under the assumption of a uniformly distributed peer population, the average proximity size can be approximated either by considering overlay parameters, or by analyzing routing tables of a subset of peers. The approximation using overlay parameters takes the the average distance between peers $\lambda = {}^{2^w}/_N$ into account. A naive interpretation for the proximity range in a linear address space could be $\pm{}^{\lambda}/_2$. In the second approach, routing tables have to be analyzed in regard of contact information to peers with smallest distance, e.g., which is the lowest non-empty list index i for peers at distance $[2^i, 2^{i+1})$.

3.2.7 Short & Long Distance Edges

We further introduce two categories of edges $e \in E^{-}$. Firstly, short distance edges (SDE) which point from within a peer p's proximity towards itself. Secondly, long distance edges (LDE) which point from peers outside p's proximity to it.

3.3 Attack Model for taLEAs

The goal of a taLEA is to *eclipse* a set of victim peers, i.e., degrade their service provision to the bulk of benign peers. We pick up the high-level taLEA notion that has been introduced in Section 2.5 and provide further details. In a taLEA, malicious peers are located at specific topology locations and less malicious peers are required than for example in the LEA case, i.e., $|V| < |M| \ll |P|$ and w.l.o.g. $|M| = \{1, \ldots, 64\}$. Moreover, the amount of malicious peers required by the adversary to launch a taLEA is constant and independent of N.

The topology-aware placement of malicious peers exploits a design weakness of convergent lookup calls as depicted in Figure 3.1 and previously discussed in Section 2.5.2. In taLEAs, attackers have to place their malicious peers m_i close by a victim peer v; a good attack strategy is a placement closer than any other benign peer to the victim, i.e., population of the victim's proximity. Once m_i receive a lookup call about v, an attacker has different options on how to further process that call, e.g., dropping the request, forwarding it to malicious or non-existent peers, or resolve incorrect contact information, as it is suggested by the arrow labeled "$v = m_2$" in Figure 3.1. Consequently, the lookup initiator a is misled by the wrong contact information and will initiate a message exchange with m_2 instead of v. We assume that peers cannot verify the correctness of assignments between keys and contact information.

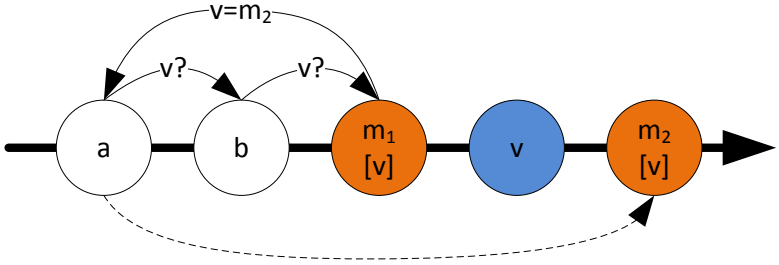


Figure 3.1: taLEA attack against overlay lookup mechanism.

The preparation of both, LEA and taLEA, requires a thorough analysis of the application that is deployed on the target overlay network in order to identify the victim peers. Yet, the taLEA preparation additionally requires:

- analysis of the P2P protocol's topology characteristics and design weaknesses (e.g., the lookup mechanism).
- acquisition of peers that are suitable for the placement, e.g., due to key generation constraints, which may stem from inadequate external features for key generation.

Nevertheless, once the placement has been finished, taLEAs are naturally less expensive, as they require only a small amount of peers to achieve an impact that is similar to the severity of a LEA which requires about 10% to 20% malicious peers.

3.4 Summary

We have introduced models for overlay graph, protocol, and taLEAs. These models serve as the foundation for our technical contributions that are presented throughout the subsequent chapters. The models in this chapter are intentionally kept abstract in order to address different protocols from the class of structured P2P protocols.

4 taLEA Impact Evaluation on Structured P2P Protocols

This chapter refers to contribution (C1) of this thesis. The goal is the impact assessment of taLEAs on structured P2P protocols. Therefore, we propose a two-fold approach that involves heuristics and simulation experiments.

4.1 Introduction

The overlay lookup mechanisms found in many structured P2P protocols show a convergent behavior which represents a security weakness. A specific exploit for that weakness is the taLEA which has been described in Section 3.3, and will be central to our investigations. We want to assess the impact of taLEAs on the victim peers' service provision. Therefore, we took the approach of evaluating the amount of lookups that are resolved from peers within the victim peer's proximity, which is considered malicious in taLEA scenarios. In order to come up with a quantitative assessment, we firstly propose an abstract heuristic that reflects the overlay routing mechanism of structured P2P protocols. After having established the abstract heuristic, we detail it for three different protocols, namely Chord [SMLN+03], Pastry [RD01], and Kademlia [MM02]. The heuristics provide the probability that lookup messages are sent to the victim's maliciously populated proximity. They are validated in a simulation case study which considers three different taLEA strategies. The study shows heuristic accuracy of up to 90% for one of the strategies.

At next, we detail technical preliminaries before introducing the abstract and protocol-specific heuristics. The chapter concludes with the case study and the subsequent result interpretation.

4.2 Approach: Lookup Susceptibility Estimation

The heuristic represents the basis for a systematic investigation of structured P2P protocol taLEA susceptibility. Therefore, we propose an abstract heuristic to approximate the characteristics of convergent overlay message routing as it is the case for many structured P2P protocols. The abstraction serves as a template for subsequently derived protocol-specific heuristics.

In order to set up the context for the heuristic, we first present some key notions for overlay routing and detail it for the specific protocols under our consideration.

4.2.1 Proximity & Lookup Definitions for Chord, Pastry, and Kademlia

Our definitions are based upon the proximity and convergent lookup mechanism definitions made in Sections 3.2.6 and 3.2.5.

The size, shape, and expected amount of peers located in a proximity area depends besides generic overlay parameters (such as key length or number of peers) also on the specific P2P protocol. Chord [SMLN+03] and Pastry [RD01] include the proximity concept in their routing table

structures and lookup mechanisms. Kademlia [MM02] has no proximity concept explicitly specified. At next, we detail our considerations for these three protocols as a prerequisite for the heuristics' definitions.

Chord & Pastry

Chord and Pastry consider the proximity as an integral part of their two-tier lookup mechanisms. Chord terms this proximity as a successor list; in Pastry it is called a leaf set. Successor list and leaf set are fixed size data structures and separately managed from the routing table. As a first lookup step, the algorithms check if the destination peer is located within the proximity. If that is not the case, then the routing table is considered for selecting the next peers for the lookup call.

Chord and Pastry's routing mechanisms are recursive by default, i.e., the lookup initiator passes the call away and hands off control to subsequent peers in the call chain.

Kademlia

Kademlia's routing algorithm follows a one-tier approach and its routing table consists of a single data structure which is interpreted as a tree. Leaves of the tree are so called buckets which are lists that store contact information of other peers. With increasing depth from the root of the routing table tree, buckets contain contact information of further distant peers. Distance is measured using the XOR metric and specified via the CPL (cf. Section 3.2.2).

We define the Kademlia proximity as the set of peers in the three highest and non-empty buckets in the routing table tree. Lookups in Kademlia are usually iterative and spatially redundant for timeliness and fault tolerance reasons.

4.3 Definition of the Abstract Heuristic for Lookup Susceptibility Estimation

On the previously sketched background, the proposed abstract heuristic develops a probabilistic basis to distinguish if a lookup is resolved by peers in the destination's proximity or by more distant peers. We chose a probabilistic approach since deterministic solutions are not feasible due to the dynamic nature of P2P overlays as caused by user behavior and/or peer or infrastructure perturbations.

In overlay lookups, lookup messages are either forwarded or resolved. Resolving implies that the destination's contact information has been found and will be returned to the lookup initiator. Moreover, forwarding implies that the lookup message is passed to a peer that is usually closer to the destination than the current peer. Messages can be resolved by two classes of peers: (i) peers in the proximity which maintain an SDE towards the destination, or (ii) peers outside the proximity which maintain an LDE towards the destination (cf. Section 3.2.7).

We observed through experiments that the probability of being resolved by an SDE is significantly higher than via an LDE. Our abstract heuristic reflects this correlation.

The Overlay Distance Class (ODC) concept allows an analysis independent of routing paths lengths. This abstraction reflects overlay message passing in an abstract structured overlay. In Figure 4.1, three ODCs are depicted. The ODC with index l denotes the class of peers which require exactly $l + 1$ message passing hops towards a destination peer (bold arrows). The ODC rectangle height illustrates the size of classes. Peers k, m and j are located in the proximity of destination peer v. Unlabeled arrows across ODCs denote lookup message forwarding, and arrows pointing to peer

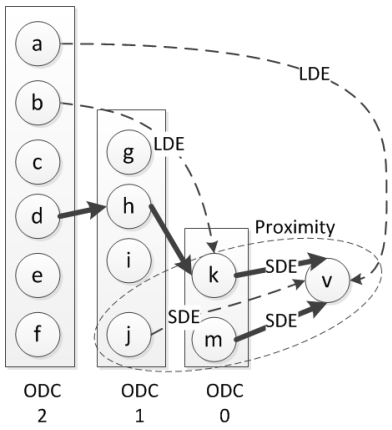


Figure 4.1: Overlay Distance Classes (ODCs) for an overlay with not more than 3 hops and with v as destination peer.

v denote resolved lookups. Resolved lookups by proximate peers occur along SDEs (edges (k,v), (m,v) and (j,v)) and by non-proximate peers via LDEs (edge (a,v)). Three special cases of edges may occur, these are indicated by dashed arrows in Figure 4.1 and represent above average lookup convergence: (i) LDEs from ODC ≥ 1 pointing to the destination (edge (a,v)), (ii) LDEs that point to an ODC which is not neighbored (edge (b,k)), and (iii) SDEs from an ODC ≥ 1 (edge (j,v)).

At next, we propose the abstract heuristic. It is essentially defined by probabilities which are explained in the following 5 steps. Each of these abstract 5 steps can be specifically instantiated for target protocols as will be shown in Section 4.4.

Step 1: $p_{SDE}(v,l)$ is the probability that a peer in ODC l has an SDE to v. This reflects the likelihood of a resolved lookup by peers belonging to ODC l which are also close to the destination.

Step 2: $p_{LDE}(v,l)$ is the probability that a peer in ODC l has an LDE to v. This reflects the likelihood of resolved lookups by distant peers belonging to ODC l.

Step 3: $\overline{p}_{SDE|LDE}(v,l)$ is the probability that a peer in ODC l has neither an SDE nor an LDE to peer v. This reflects the likelihood of lookup forwarding by peers in ODC l to an intermediate peer on the path towards the destination.

Thus, in a message loss free scenario, the following holds:

$$p_{SDE}(v,l) + p_{LDE}(v,l) + \overline{p}_{SDE|LDE}(v,l) = 1 \tag{4.1}$$

Moreover:

Step 4: $p^*_{SDE}(v,l)$ is the probability that a lookup message whose originator is located in ODC l will be resolved at some point with an SDE.

A compact representation of $p^*_{SDE}(v,l)$ is given by Equation 4.2:

$$\begin{aligned} p^*_{SDE}(v,l) = \quad & p_{SDE}(v,l) \\ + \quad & \sum_{i=1}^{l} \Big(p_{SDE}(v,l-i) \cdot \prod_{j=0}^{i-1} \overline{p}_{SDE|LDE}(v,l-j) \Big) \end{aligned} \tag{4.2}$$

Step 5: $p_{SDE}(v)$ is the probability that a lookup message sent by an arbitrary peer will be at some point resolved by an SDE. This represents the final step of the abstract heuristic.

In order to compute the overall probability $p_{SDE}(v)$, the ODCs' $p^*_{SDE}(v,l)$ for $l = 0 \ldots l_{max}$ need to be weighted with respect to the ODC class size. The protocol-specific fraction of peers belonging to each ODC is given by the weight function $g(l)$. Thus, we modify Equation 4.2 to consider the ODC size distribution of the specific target P2P protocol:

$$p_{SDE}(v) = \sum_{l=0}^{l_{max}} g(l) \cdot p^*_{SDE}(v,l) \tag{4.3}$$

The parameter l_{max} specifies the ODC with highest distance to target peer v ($l_{max} = 2$ in Figure 4.1).

4.4 Protocol-Specific Heuristics

For the heuristic steps of p_{SDE}, p_{LDE} and $\overline{p}_{SDE|LDE}$ developed in Section 4.2, we now provide specific formulae for Chord, Pastry and Kademlia. Based on this, it is possible to compute the probability of malicious message interception from a peer's proximity for varied LEA strategies.

Heuristics are based upon the P2P protocol specifications and are subject to three assumptions: (i) peers in the proximity have propagated to other peers in their address space region, (ii) intermediate peers always decrease the distance towards the destination, (iii) routing tables are sufficiently populated with contact information. These assumptions allow us to simplify the heuristics definitions by dropping the requirements to model special cases.

4.4.1 Chord

We assume an arbitrary peer p_i that belongs to ODC i. $N(i)$ denotes the expected number of peers between p_i and v in Chord's address space, i.e., small values of i result in small values returned by $N(i)$ which we define as:

$$N(i) = \frac{N}{s(i)} \tag{4.4}$$

The distance decrease on each subsequent hop during lookup message forwarding towards the message's destination is reflected by dividing N by $s(i)$ which we define as:

$$s(i) = 2^{2(l_{max}-i+1)} \tag{4.5}$$

Furthermore, we define

$$p_{SDE}(v,i) = \begin{cases} 1 & \text{if } N(i) \leq \kappa \\ \frac{\kappa^2}{N(i)^2} & \text{otherwise} \end{cases} \tag{4.6}$$

This formula approximates the probability of peer p_i having a SDE towards v. This probability decreases for increasing distances between the two peers. Thus, equation 4.6 returns:

- 1 if $N(i)$ is smaller or equal to the number of SDEs ($=\kappa$)

- a quadratic decrease if $N(i)$ is larger than κ

To compute $\overline{p}_{SDE|LDE}(v,i)$, a definition of the probability function $p_{LDE}(v,i)$ is required:

$$p_{LDE}(v,i) = \frac{(\log_2(N) - 2 \cdot (l_{max} - i)) \cdot s(i)}{3 \cdot N} \tag{4.7}$$

The expected amount of LDEs stored in each peer's routing state is $\log_2(N)$. On each hop towards the destination the amount of LDEs possibly pointing at v decreases by 2 on average while the amount of peers decreases by factor $s(i)$.

On subsequent message passing hops, the amount of peers decreases as well as the number of LDEs towards the destination. The relation between these two is required to estimate the probability of having an LDE to the destination. Therefore, the number of remaining LDEs is divided by the number of remaining peers. The division in Equation 4.7 by N results in the probability of an LDE pointing at destination v. The factor of $\frac{1}{3}$ is a calibration parameter that has been derived from simulation experiments.

Thus, for Chord $\overline{p}_{SDE|LDE}(v,i)$ is obtained as:

$$\overline{p}_{SDE|LDE}(v,i) = (1 - p_{LDE}(v,i)) \cdot (1 - p_{SDE}(v,i)) \tag{4.8}$$

$g(i)$ is a weight function for the $p_{SDE}(v)$. It specifies the number of peers for the i-th ODC. We present the corresponding values in Figure 4.2 and set:

$$g(i) = \frac{3}{s(i)} \tag{4.9}$$

The amount of peers in ODC i is one quarter the amount of peers in ODC $i+1$. Consequently, ODCs further away from v tend to hold exponentially more peers than closer ones. This guarantees a number of hops that is logarithmically dependent on N, as each message forwarding hop traverses at least one ODC according to our assumption.

With the previous definitions, all terms of Equation 4.3 are defined. The heuristic computations for other overlay network sizes are presented in Section 4.6.

4.4.2 Pastry

Equation 4.10 calculates the probability of an arbitrary peer's membership in ODC i. The distribution has been calculated for different overlay network sizes and is also depicted in Figure 4.3.

$$g(i) = \frac{2^b - 1}{2^{b \cdot (l_{max} - (i+1))}} \tag{4.10}$$

Similar to Chord, the address space distance decreases on each hop. According to Pastry's specification, the distance to a destination peer v decreases by a factor of $s(i) = 2^{b \cdot (l_{max} - (i+1))}$.

Pastry's prefix based distance notion which is encoded with a hexadecimal number system typically of base 16, i.e., $b = 4$. All peers share at least a prefix of length 0, therefore $N(l_{max})$ yields the number of all peers in the overlay.

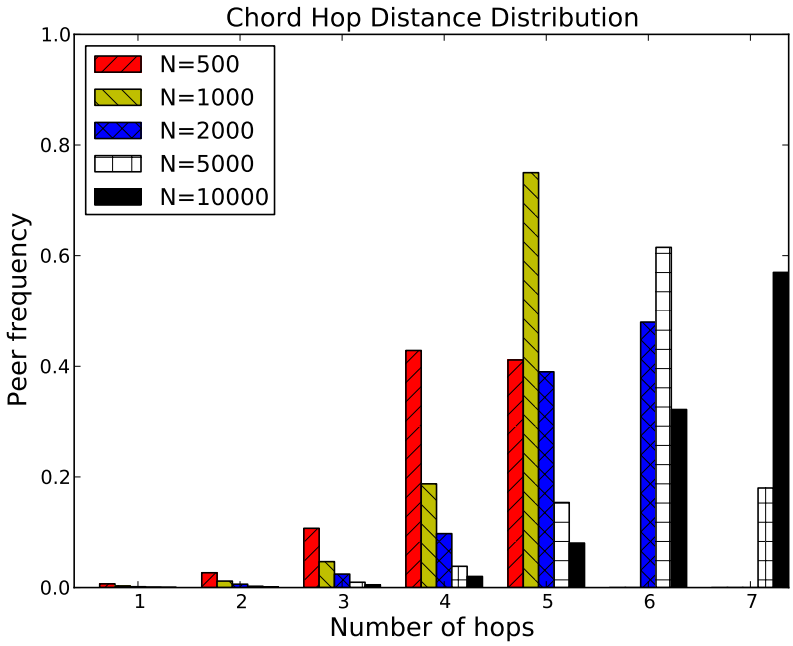


Figure 4.2: Chord ODC size distribution.

Peers with identical shared prefix lengths are in the same ODC. With increasing shared prefix length, the number of peers in an ODC decreases exponentially. $N(i)-N(i-1)$ denotes the number of peers in ODC i. For Pastry, $N(i)$ is defined as:

$$N(i) = \frac{N}{2^{b\cdot(l_{max}-(i+1))}} \tag{4.11}$$

To calculate the probability if an arbitrary peer of ODC i is located in v's proximity, the probability of an SDE towards v in the respective ODC needs to be known. Therefore, we define $T(i)$ as the expected number of peers which have both, an SDE towards v and an ODC i membership. The sum of probabilities of v belonging to ODC i such that not all of the SDEs to v are located in the same ODC i is multiplied by the number of SDE neighbors which are located outside of ODC i. The result is denoted as $T(i)$:

$$T(i) = \begin{cases} \frac{2}{N(i-1)} \cdot \sum_{j=1}^{\frac{\kappa}{2}} (\kappa - j) + \kappa \frac{N(i)-\kappa}{N(i)} \\ \quad \text{for } N(i) \geq \frac{\kappa}{2} + 1, \\ N(i) \quad \text{otherwise.} \end{cases} \tag{4.12}$$

If $N(i) \leq \frac{\kappa}{2} + 1$, two conclusions follow. First, $T(i) = N(i)$, i.e., the whole ODC i is located in v's proximity. Secondly, $p_{SDE}(v,i) = 1$ because all peers in ODC i will have an SDE to v. $T(i)$ is used for the definition of $p_{SDE}(v,i)$. This definition includes three different cases to deal with varying amounts of SDEs for different ODCs.

We define $p_{SDE}(v,i)$ for Pastry as follows:

$$p_{SDE}(v,i) = \begin{cases} 1 & \text{if } N(i-1) \leq \frac{\kappa}{2}+1 \\ q_{SDE}(v,i) & \text{otherwise} \end{cases}$$
$$q_{SDE}(v,i) = \begin{cases} \frac{\kappa - T(l_{max})}{N(i)-N(i-1)} & \text{for } i = l_{max} \\ \frac{T(i)-T(i+1)}{N(i)-N(i-1)} & \text{otherwise} \end{cases} \tag{4.13}$$

The two cases in $q_{SDE}(v,i)$ reflect the probability that a randomly chosen peer from ODC i is one of the $T(i)$ peers. The probability of a peer having an LDE to v increases exponentially with decreasing distance to peer v, as exponentially less peers exist with this shared prefix length.

$$p_{LDE}(v,i) = \frac{2^{b(l_{max}-i+1)}}{N} \tag{4.14}$$

$\overline{p}_{SDE|LDE}(v,i)$ can be computed using the probability function of step 3 in Section 4.2. The formula for the heuristic of the overall probability of a peer resolving the lookup via an SDE follows from Equation 4.3.

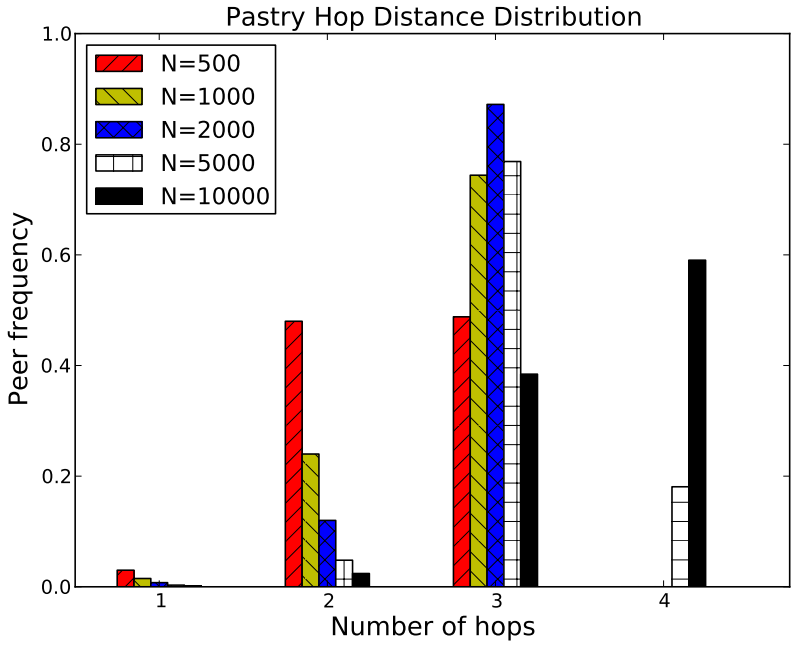


Figure 4.3: Pastry ODC size distribution.

4.4.3 Kademlia

Kademlia differs from previous protocols with respect to the following aspects. Firstly, the message passing mechanism does not explicitly differentiate between SDEs and LDEs. Despite of a concise proximity definition (see Section 4.2.1), on a conceptual level lookups are forwarded via LDEs until

the destination can be resolved. Secondly, message passing is per default iterative and spatially redundant. Kademlia's ODC size distribution (i.e., values for $g(i)$) is shown in Figure 4.4.

Besides these difference aspects, Kademlia's message passing mechanism is similar to Pastry's. Therefore, the Kademlia heuristic is closely related to the Pastry heuristic; we highlight the differences below.

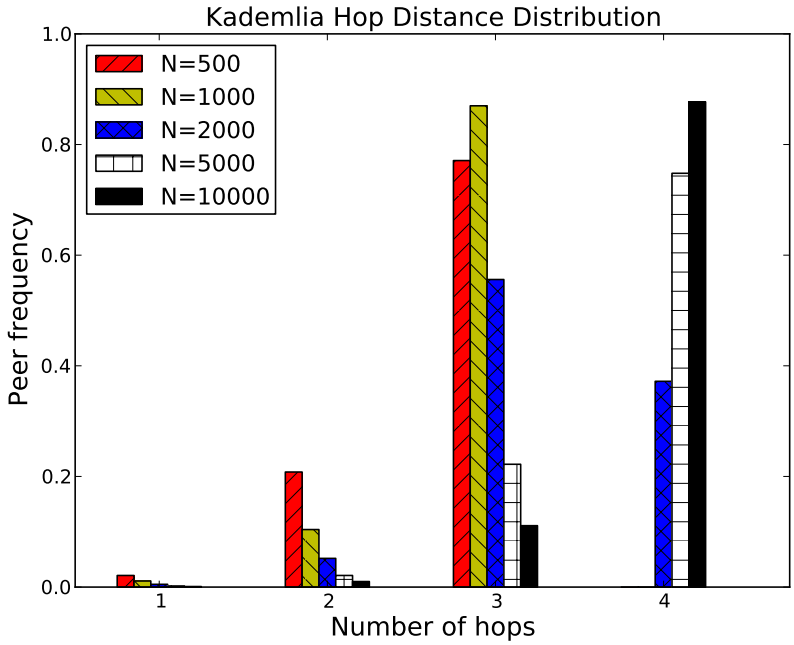


Figure 4.4: Kademlia ODC size distribution.

As Kademlia selects the most appropriate peers for the next hop from a peer set with a higher CPL than the current peer, order statistics are applied to determine the expectation $E\{Y_{max}\}$ for the highest CPL as follows. We assume a Kademlia overlay with buckets of size 8, threefold spatial redundant messaging, key length 128. The peers' keys in a bucket can be seen regarded as 8 random instances of a bitstring X with dimension 128. For the coordinates of a string

$$X_i = \left(X_{i,1}, X_{i,2}, \ldots, X_{i,128}\right), \quad i \in \{1, 2, \ldots, 8\}$$

holds:

$$\mathbf{P}\left[X_{i,j} = 0\right] = \frac{1}{2} = \mathbf{P}\left[X_{i,j} = 1\right], \quad j \in \{1, 2, \ldots, 128\}\,.$$

We consider a random variable Y_i that denotes the CPL of a bitstring with the destination peer. Therefore, Y_i has values from $\{1, 2, \ldots, 128\}$ with a probability distribution:

$$\mathbf{P}\left[Y_i = k\right] = \begin{cases} \left(\frac{1}{2}\right)^{k+1} & \text{for } k \in \{1, 2, \ldots, 126\} \\ \left(\frac{1}{2}\right)^{128} & \text{for } k \in \{127, 128\} \end{cases}$$

for all $i \in \{1,\ldots,8\}$. Let $\left(Y_{(1)},\ldots,Y_{(8)}\right)$ be the sample ordered by size, the so called order statistics. The expectation for the highest three (i.e., Y_8, Y_7, Y_6) needs to be found. For simplicity, the CPL of previous hops and one (from the bucket) is not considered in the example.

Equation 4.15 is based upon the Pastry heuristic, but references to b have been removed. $idLength$ represents the length of a key, typical values are 128, 160, or 192 bits. $setSize$ stands for the number of keys the highest CPL peers are chosen from, and max for the best peer to be considered, so $max = setSize$ yields the expectation for the peer with the maximum CPL.

$$\begin{aligned} &E\{Y_{max}\} = \\ &\textstyle\sum_{k=0}^{idLength} k \cdot \sum_{m=max}^{setSize} \binom{setSize}{m} (P[Y_{max} \leq k]^m \\ &\cdot P[Y_{max} > k]^{setSize-m} - P[Y_{max} \leq k-1]^m \\ &\cdot P[Y_{max} > k-1]^{setSize-m}) \end{aligned} \tag{4.15}$$

The Kademlia proximity has been defined by 24 peers which are stored in the three buckets with the lowest indexes. Therefore, $p_{SDE}(v,i)$ is defined as in Equation 4.13 with $\kappa = 24$.

We define Equation 4.16 to calculate the probability of an arbitrary peer's membership in ODC i.

$$g(i) = \frac{1}{2^{E\{Y_{max}\} \cdot i}} \tag{4.16}$$

The spatially redundant lookup mechanism in Kademlia returns over the first iteration a peer set with potentially unequal CPLs.

The first iteration of $p_{LDE}(v,i)$ has to be calculated with z different values for b and with z equal to the degree of parallelism. This is because the expected CPL of the first parallel hops chosen from the sender's routing table is smaller than the CPL of subsequent hops, as the set of peers stemming from the responses is larger. Therefore, we introduce b_0 and the formula changes accordingly to:

$$p_{LDE}(v,i) = \begin{cases} \frac{2^{b_0}}{N(i)} & \text{for } i = l_{max}, \frac{2^{b(l_{max}-i+1)}}{N(l_{max})} \quad \text{otherwise} \end{cases} \tag{4.17}$$

Equation 4.2 needs to be calculated individually for the different values of b_0 as Kademlia takes three different paths to the target, and the average of these will represent the value.

4.5 taLEA Strategies

This section introduces three taLEA strategies that will be used in subsequent simulation experiments to validate the previously defined heuristics. The strategies are called *proximity hijacking*, and *proximity insertion* with high and low distances. Our heuristics approximate the proximity hijacking strategy. Thus, we deduce in the subsequent simulation case study the accuracy of our heuristics from simulation runs according to the hijacking strategy.

4.5.1 Strategy 1: Proximity Hijacking

In the case of proximity hijacking, no malicious peers are joined into the overlay network, but an attacker overtakes (i.e. hijacks) benign peers in the victim peers' proximities. An attacker could achieve this, for example, by exploiting security weaknesses to gain administrative control over the peers' machines.

4.5.2 Strategies 2 & 3: Proximity Insertion

Using the proximity insertion strategy, new malicious peers are inserted into the victim peers' proximities. Malicious peers require a placement closer to the victim than any other benign proximate peer; the keys of the malicious peers have to be chosen accordingly prior to the insertion. Due to the self-organization in structured P2P overlays, victim peers finally exchange their benign proximities with the malicious peers after some time.

We differentiate between the INSERT-low and INSERT-high proximity insertion strategies. INSERT-low implies that malicious peers are inserted very close to the victim peer, whereas INSERT-high relaxes this constraint a bit and requires the attacker to undercut the address space range of the closest benign proximity peers. Therefore, INSERT-high usually results in a wider range of malicious keys and INSERT-low accumulates malicious keys nearby the victim.

4.6 Case Study: Heuristics Validation

In this section, we validate our three heuristics by evaluating a taLEA simulation case study that shows an accuracy of 90% for the proximity hijacking strategy. The proximity insertion strategies reveal a higher deviation. Furthermore, we present simulation settings and performance metrics.

4.6.1 Simulation Settings

The simulation study was conducted with Chord, Pastry and Kademlia implementations in OverSim [BHK09]. Parameters of our simulation case studies are shown in Table 4.1. Simulations are conducted for varying overlay network sizes between 500 and 10000 benign peers. For benign peers, we consider a uniform workload, i.e., each peer sends periodically a message to the victim peer v. Benign peers $b \in B$ and malicious peers $m \in M$ are subject to churn, and the victim peer v is present in the overlay for the whole simulation period.

4.6.2 Metrics

Our simulations investigate two performance aspects, namely the accuracy of our heuristics and the degree of malicious lookup interception by varied taLEA strategies. The model accuracy is compared to the experimental results of the hijack strategy. The attack severity is the ratio of messages that an attacker can intercept to the total number of lookups that are addressed to v.

Parameters	Values
Simulated time	up to 16 hours, depending on overlay size
Overlay sizes	500, 1000, 2000, 5000, 10000
% ID space (λ)	0.2%, 0.1%, 0.05%, 0.02%, 0.01%
Churn	1h lifetime with exponentially distributed probability
Chord proximity size (κ)	8
Chord average proximity width (ε)	1.6%, 0.8%, 0.4%, 0.16%, 0.08%
Chord high width inserted peer distance from v or one another	0.2%/κ, 0.1%/κ, 0.05%/κ, 0.02%/κ, 0.01%/κ
All overlays, low width inserted peer distance from v or one another	0.000001% for all sizes
Pastry proximity size (κ)	16
Pastry average proximity width (ε)	3.2%, 1.6%, 0.8%, 0.32%, 0.16%
Pastry high width inserted peer distance from v or one another	0.4%/κ, 0.2%/κ, 0.1%/κ, 0.04%/κ, 0.02%/κ
Kademlia proximity size (κ)	24
Kademlia average proximity width (ε)	5%, 2.5%, 1.25%, 0.625%, 0.3125%
Kademlia high width inserted peer distance from v or one another	0.4%/κ, 0.2%/κ, 0.1%/κ, 0.04%/κ, 0.02%/κ

Table 4.1: Simulation Parameters.

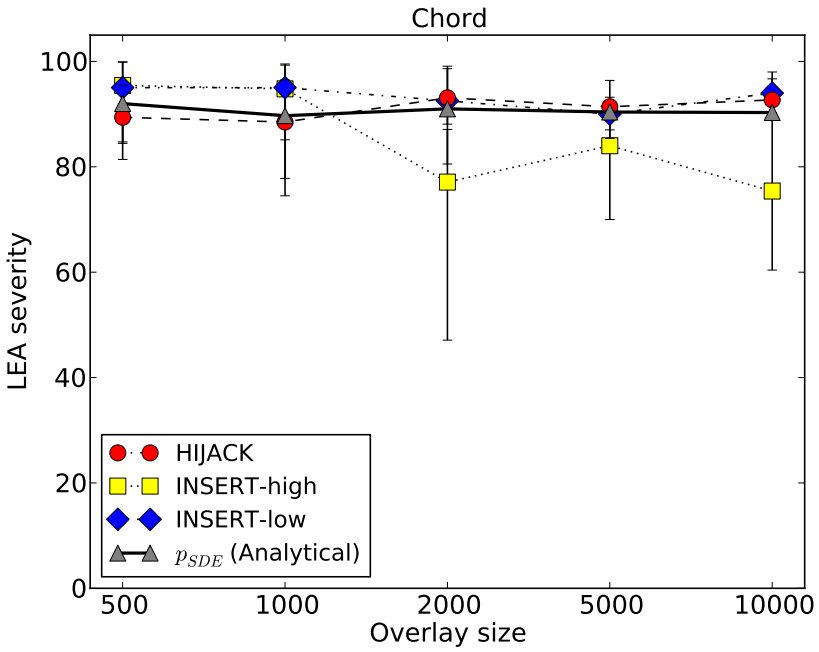


Figure 4.5: taLEA severity (Chord).

4.6.3 Heuristics Accuracy Assessment

We now investigate the impact of different overlay network sizes, and the different taLEA strategies on the accuracy of our heuristics. Hereby, we fix the number of malicious peers to $|M| = \kappa$, i.e., $\kappa = 8$ for Chord, $\kappa = 16$ for Pastry which reflects the default settings according to the protocols' specifications and $\kappa = 24$ for Kademlia according to our notion of Kademlia's proximity.

We present experiment results for the different strategies in Figures 4.5 (Chord), 4.6 (Pastry) and 4.7 (Kademlia) for various overlay network sizes, the three taLEA strategies as well as our heuristics.

The p_{SDE} curve denotes the computation of our heuristics and the hijack curves denotes the Proximity hijacking taLEA strategy that has been presented in Section 4.5. Deviations between p_{SDE} and the hijack curves are small, i.e., ±5%, which yields an accuracy of our heuristics of 90%. The p_{SDE} taLEA severity is about 90% for Chord, up to 80% for Pastry, and up to 82% for Kademlia. The severity is equal to the degree of message interception, in case the victim peer's proximity has been hijacked or expelled through the newly inserted malicious peers. An important observation is that the taLEA severity is almost independent of the overlay size.

Also, results for the proximity insertion taLEA strategies are presented. The severity of these two strategies is higher for Chord and Pastry compared to the proximity hijacking simulation experiments. Kademlia displays lower susceptibility for the Proximity Insertion taLEA strategy.

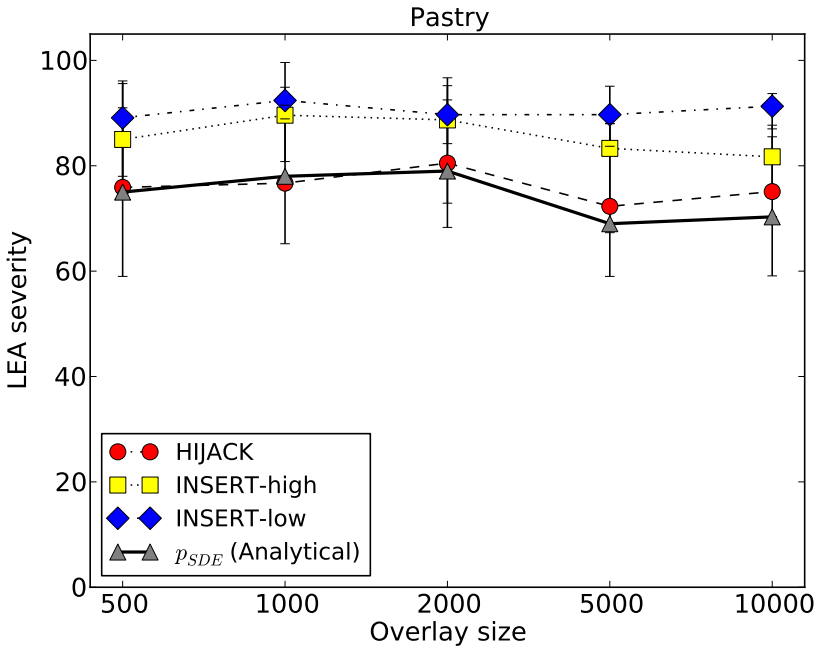


Figure 4.6: taLEA severity (Pastry).

The difference between INSERT-low and INSERT-high is because of overlay maintenance effects that consider changes of LDEs that pointed towards v before the taLEA was started.

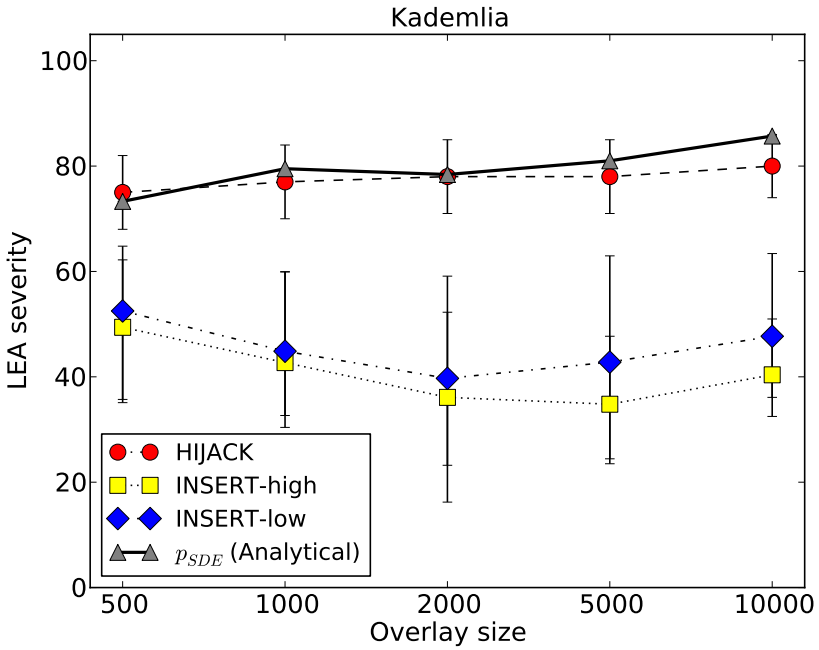


Figure 4.7: taLEA severity (Kademlia).

4.7 Limitations

The proposed heuristics approach shows a high accuracy compared to simulation experiments, yet the applicability may be limited in case one of the following aspects is not fulfilled:

- The abstract heuristic assumes an overlay message passing scheme that is decreases the distance towards the destination on every iteration. This is the case for structured P2P protocols that make use of prefix-based message exchange, or overlay networks with ring topologies.
- The validation process showed a very high accuracy for the so called hijacking taLEA strategy during which an attacker picks benign peers and transforms these into malicious peers. But, the insertion strategies show a higher deviation to the heuristics and we believe that our heuristics are not adequately addressing churn effects.
- Moreover, the large parameter landscape including workload, taLEA, protocol, and churn models has not been entirely addressed in this case study.

4.8 Summary & Conclusion

Maintaining a structured overlay topology is both, a key feature of structured P2P protocols and at the same time a key security weakness that facilitates taLEAs because structured topologies favor deterministic algorithm behavior, e.g., for exchanging messages.

We have presented the significant impact of taLEAs as a first contribution in this thesis using our heuristic approach and through simulation experiments. The advantage of a heuristic approach is

that impact estimations can be computed instantly (even for very large overlay networks), compared to lengthy simulation experiment series (which are limited by computational resources and time). Nevertheless, heuristics tend to evolve to rather complex models for a wide system parameter landscape. In the following chapter, we are focusing on the development of a taLEA countermeasure. In order to assess the mitigation efficiency for a wide range of parameters including dynamic effects in the overlay, we solely focus on simulation experiments. Nevertheless, for very specific and static system settings, the approach using heuristics may be an adequate choice.

An important result of this chapter is the development of a heuristic template that can be extended to protocol-specific heuristics. These allow to approximate the amount of potentially intercepted lookup messages by malicious peers that are located in the proximity of a victim peer. The impact estimation computed by the three protocol-specific heuristics has shown an accuracy of up to 90% compared to results gathered through simulation experiments. Both, the analytical and simulation case studies have shown that the last hop of a route significantly favors (with a probability higher than 70%) SDEs to deliver messages to their destination. The proposed taLEA strategies utilize only 8 (Chord), 16 (Pastry), or 24 (Kademlia) malicious peers and cause a significant impact on the overlay.

We have made another interesting observation during experiments using the insertion taLEA strategies in Chord and Pastry overlays: maintenance sometimes replaces LDEs incident to the victim by LDEs incident with malicious peers. Consequently, this increases the taLEA severity. This results in message interception of more than 90% for Chord and Pastry. Contrary, in Kademlia the proximity insertion strategies achieve a message interception of up to 50% which is clearly below the proximity hijack strategy's severity. One possible reason is the convergence of routing paths towards frequently addressed peers and the synthetic design of our simulation workload. We focus on a wider parameter landscape in subsequent chapters of this work using simulation experiments.

These results are crucial not only to conduct taLEAs, but mainly to justify the development of requisite countermeasures for mitigating taLEAs.

5 taLEA Mitigation: Divergent Lookups

5.1 Introduction

This chapter provides our taLEA mitigation approach. The susceptibility analysis in Chapter 4 presented the significant impact of taLEAs on the service provision of victim peers. We have identified convergent lookups as a design weakness that allows to launch taLEAs. To this end, we propose in this chapter *divergent lookups* as a mitigation measure which refers to contributions (C2) and (C3) of this thesis.

5.2 Motivation: Susceptibility Analysis of Convergent Lookups

In a taLEA, malicious peers are placed closer to victim peers than any benign peer in the overlay, i.e., they populate the victim's proximity (cf. Section 3.3). As a consequence, malicious peers receive a significant amount of convergent lookup calls from benign peers and are then able to launch further attacks such as returning bogus information or denial of service.

One way to prevent malicious peers from conducting a taLEA is to restrict the lookup calls of benign peers to non-proximity address space regions of the destination peer. In that region, we expect under taLEA assumptions no malicious peers. This leads to

Hypothesis 1. *In order to increase the lookup mechanism's taLEA resilience, a lookup has to avoid the destination's proximity.*

In the next subsection we provide a discussion about a convergent lookup's algorithm. This helps to understand the established lookup concepts and it functions as an algorithmic basis for divergent lookups.

5.2.1 Pseudocode Discussion

The pseudocode for convergent lookups is presented in Algorithm 1. Now, we describe the parameters as well as the algorithm's instructions. Four parameters are provided, the key κ of the peer to be looked up, the routing table rt of the lookup initiator, α denotes the degree of spatial redundancy, and i_{max} defines the maximum amount of lookup iterations in order to limit the network overhead and to prevent a lookup from accidentally running forever. The convergent behavior can be observed in line 4 of the algorithm where at most α closest peers to the destination are picked from the queue Q and queried using the *query* primitive. This primitive sends lookup request messages to other peers picked from the queue, requesting contact information for key κ. If queried peers do not know about κ, these return closer contacts towards κ as a proposal for further requests in the next iteration. All query results are written back to the queue in line 5 of the algorithm. Finally, line 6 checks if κ has been resolved and is contained in the queue, otherwise the algorithm loops into the next iteration.

Algorithm 1 shows the iterative variant, which is by default used in Kademlia. Iterative lookup variants usually impose a higher network overhead as each iteration is coordinated by the lookup

initiator again. Also, recursive algorithms exist and we propose iterative and recursive pseudocode for divergent lookups at next.

Usually, convergent lookups show good performance with $O(\log N)$ steps. One of their major drawbacks is the taLEA susceptibility.

Algorithm 1 Convergent Lookup (iterative)

```
1:  procedure LOOKUP(κ, rt, α, i_max)
2:      Q ← rt
3:      for i = 1 to i_max do
4:          C ← query ≤ α closest peers in Q for κ
5:          Q ← Q ∪ C
6:          if C contains c with c.κ == κ then
7:              return c.IPaddress
8:          end if
9:      end for
10:     return NOTFOUND
11: end procedure
```

5.3 (C2) Divergent Lookups - Random Walks

We define two essential lookup requirements for mitigating taLEAs which will be considered in our divergent lookup implementations:

1. Avoid the destination's proximity.
2. Drop the lookup convergence criterion.

The first requirement stems from Hypothesis 1 and it is realized by the definition of a proximity threshold parameter which may not be overrun by the lookup. The second requirement is addressed by replacing the criterion that only closest peers to the destination are queried. This ensures that lookups do not converge towards the proximity threshold and potentially starve. In our second contribution's case study, we make no assumptions about the distribution of peers in the overlay that store an LDE to the destination in their routing table. Consequently, we apply a random walk search strategy.

At next, we discuss the pseudocode of our iterative divergent lookup variant that makes use of a random walk search strategy.

5.3.1 Pseudocode Discussion

We now detail the divergent lookup with random walk strategy (divRW), the respective pseudocode is presented in Algorithm 2. The divRW algorithm has an additional parameter t_p which defines the proximity threshold in terms of a common prefix length. The threshold depends on the key length and the amount of peers on the overlay network. The noteworthy differences to the convergent algorithm can be seen in lines 4 and 5, respectively. Also, these modifications satisfy the previously introduced two requirements. Firstly, in line 4, peers from κ's proximity, i.e., those sharing a higher

Algorithm 2 Divergent Lookup using random walk search strategy (divRW) - iterative

```
1:  procedure LOOKUP(κ, rt, α, i_max, t_p)
2:      Q ← rt
3:      for i = 1 to i_max do
4:          Q ← Q \{sharing CPL > t_p with κ}
5:          C ← query ≤ α random peers in Q for κ
6:          Q ← Q ∪ C
7:          if C contains c with c.κ == κ then
8:              return c.IPaddress
9:          end if
10:     end for
11:     return NOTFOUND
12: end procedure
```

CPL than t_p with κ, are removed to avoid malicious peers in Q. Moreover, in line 5 peers are selected in a *random* fashion which replaces the convergent distance decreasing strategy.

We expect from divRW a significant lookup reliability improvement over the convergent variant under taLEA conditions. Clearly, the message complexity and timeliness of divRW will exceed those of convergent ones. Therefore, we aim to improve on the performance by a more advanced divergent lookup search strategy, which is detailed in the next section.

5.4 (C3) Divergent Lookups - P2P Address Space Slicing (PASS)

Due to the higher network overhead of divRW especially for large overlay networks, we propose the P2P address space slicing (PASS) technique as a performance improvement. The PASS approach does not abandon the random walk search strategy, but considers a smaller region on the address space than $[0, t_p]$. Hence, PASS further segregates the address space according to assumptions in which regions κ can be found with high chance and few algorithm iterations.

Firstly, we show an example for PASS before providing analytical insights about adequate slice selection.

5.4.1 PASS: Slicing Example

This example makes use of the CPL notion (cf. Section 3.2.2) in view of segregating an overlay's address space. For illustration reasons, we refer to a smaller address space which has three peers assigned to it.

We assume for this example a key length $w = 5$ which results in an address space range $[0, 31]$. Furthermore, we assign to three peers p_1, p_2, and p_3 the decimal keys $\kappa_1 = 2$, $\kappa_2 = 9$, and $\kappa_3 = 10$. Figure 5.1 shows the address space with the three peers and the CPL regions on the address space in terms of p_3. The key ranges for a given CPL slice depend on the keys' binary prefixes of length 0 through 4, as shown in Table 5.1. In this table, the binary key pattern is augmented with boxes which encapsulate the prefixes and underscore characters show mutable bits. CPLs 0 through 3 require a further fixed bit q next to the prefix because for $\neg\, q$ the pattern would be classified as the next higher CPL. The table's rightmost column denotes the CPL slice's size, i.e., their share for potentially

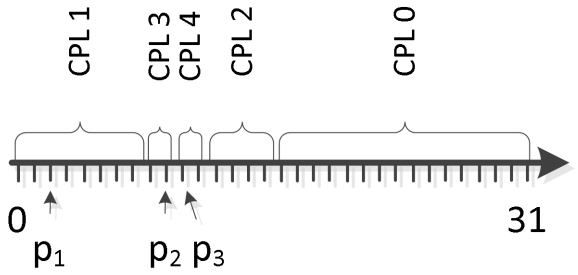


Figure 5.1: Address space and CPL regions for peer p_3.

Figure 5.2: CPL space from peer p_3's perspective.

mappable peers; obviously longer prefixes induce a smaller size. The actual slice assignment from peer p_3's perspective is depicted in Figure 5.2.

5.4.2 Analytical Discussion

The previous example provided insights on the address space organization in regard of CPL slices and their sizes. Now, we focus on the partial view each peer maintains about the overlay network's address space in order to understand which CPL slice should be selected for performant divergent lookups. Therefore, we propose Lemma 1:

Lemma 1. *For two arbitrary peers, the likelihood of one peer having stored contact information about the other increases for decreasing distance.*

We provide a proof for Lemma 1 under consideration of two assumptions:

1. The P2P protocol uses CPL as a distance notion.
2. Message exchange among peers is subject to a uniform distribution with a reasonable frequency.

CPL	Binary key pattern	Decimal key range for slice	Share w.r.t. address space
0	1 _ _ _ _	[16, 31]	0.5
1	0 0 _ _ _	[0, 7]	0.25
2	0 1 1 _ _	[12, 15]	0.125
3	0 1 0 0 _	[8, 9]	0.0625
4	0 1 0 1 _	[10, 11]	0.0625

Table 5.1: CPL slices example for peer p_3 with key $\kappa_3 = 10_{10} = 01010_2$.

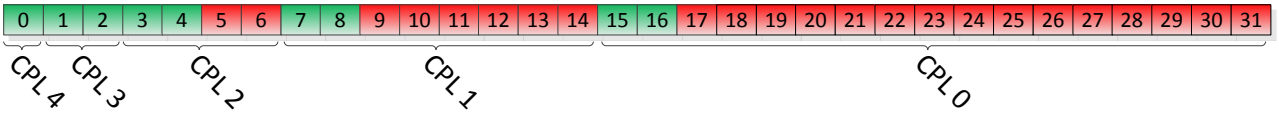


Figure 5.3: Simplified example to illustrate routing table storage capacity w.r.t. CPL slices, $w = 5$ and $k = 2$.

Proof. We assume the existence of peers p, and q in an address space of size 2^w. p and q share a CPL of $w_{p,q} = w - \theta$ with w being the key length, and $\theta \in \{1, \ldots, w-1\}$. Clearly, the CPL is larger for small values of θ. Routing tables of peers consist of w lists with usually $k \in \{1, \ldots, 20\}$ entries storage capacity for contact information of peers at a distance of $[2^i, 2^{i+1})$ with $i = 0, \ldots, w-1$. Therefore, q is stored in list $i = \theta$. Hence, for a smaller CPL (i.e., a larger value for θ) the range of potentially selectable keys for storage increases exponentially whereas the lists' storage space size k remains constant. Consequently, the likelihood of storing a peer with small values for θ is higher than for a peer with a large value for θ. □

The essence of Lemma 1 is depicted in Figure 5.3 as another example for $w = 5$ and $k = 2$. This example does not relate to the perspective of a specific key, but visualizes the two storage slots (greed shaded keys) per CPL for each of the w routing table lists. For a CPL of 3 and 4, a routing table could store all potentially mappable keys in the respective slices, and the storage capacity decreases exponentially with decreasing CPL.

5.4.3 PASS Design Rationale

Lemma 1 supports the claim that searching non-proximity slices closer towards the proximity threshold of the destination peer is more performant. We now provide insights on the different CPL regions, in order to motivate the CPL slice selection of the PASS approach. Figure 5.4 depicts the CPL scale on behalf of an arbitrary destination peer. We define three CPL thresholds, i.e., t_p is the proximity threshold, t_l is the lower threshold for PASS searches, and t_u is PASS's upper threshold. Moreover, $0 \leq t_l \leq t_u \leq t_p < w$. The following three CPL slice ranges are considered inadequate:

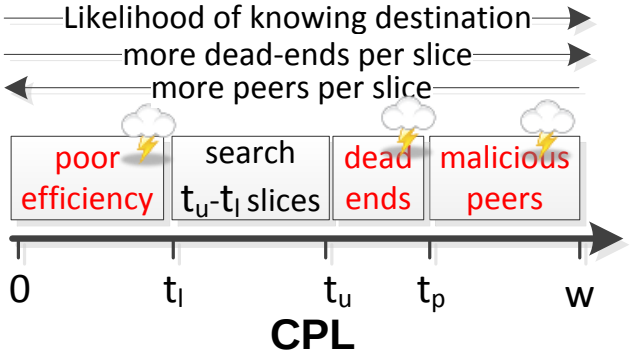


Figure 5.4: Address space slicing illustration on a CPL scale.

- The subrange $[0, t_l)$ contains a very large amount of peers which also have – compared to the other subranges – a low chance of knowing the destination peer. Searching this region would negatively affect the divergent lookup's performance.
- The subrange $(t_u, t_p]$ contains a high amount of *dead ends*. These are peers, which know the destination neither directly nor transitively, i.e., they cannot find a path to a peer in the same slice which has contact information about the destination peer. Dead ends exist in all slices, but slices close-by the proximity threshold are sparsely populated; based on the assumption of a uniform workload distribution, these few peers have a lower chance of knowing the destination peer compared to a slice that contains more peers. Few peers and a high amount of dead ends would impair the divergent lookup's reliability.
- The subrange $(t_p, w]$ is by taLEA assumptions populated with malicious peers and should not be searched in order to prevent an attack from being activated.

As a consequence, PASS selects the CPL slice range $[t_l, t_u]$ for performing divergent lookups. Clearly, a good choice for thresholds decides on reliability and performance. Therefore two conditions should be satisfied:

1. Slices should contain peers that know the destination peer with a high probability.
2. Peer population size of the selected slices must allow for exhaustive searching in the worst case.

Naturally, the thresholds depend on P2P protocol parameters and the amount of peers in the overlay. We provide experimental insights that support the threshold selection in Section 5.5.7.

At next, we discuss the iterative and recursive pseudocode of divPASS.

5.4.4 Pseudocode Discussion

We propose PASS for divergent lookups (which we abbreviate as divPASS) as an optimization of divRW with special focus on reducing the network overhead. It is subject to Hypothesis 1 and we provide insights on the iterative (Algorithm 3) and recursive (Algorithm 4) divPASS pseudocode and explain their differences to divRW (Algorithm 2).

Iterative divPASS

In divPASS, the proximity threshold parameter t_p has been replaced by the lower t_l and upper t_u threshold parameters which denote the slice range in which κ's contact information is searched for (lines 3, 5, and 12). Because a lookup initiating peer could be of considerable CPL distance to κ, the lower bound condition for t_l is increasingly relaxed in case no suitable candidates can be found (lines 4 through 7 in Algorithm 3). After an initial peer candidate set was found (line 8), divPASS searches in the given CPL slices for κ (lines 9 through 17).

Recursive divPASS

The recursive divPASS algorithm is presented in Algorithm 4. It consists of two procedures, i.e., lookup-initiator() which is executed on the lookup initiating peer and lookup() for the recursion,

Algorithm 3 Divergent Lookup (iterative) - PASS

```
1:  procedure LOOKUP(κ, rt, α, i_max, t_l, t_u)
2:      j ← 1
3:      Q ← rt \{sharing t_l > CPL > t_u with κ}
4:      while |Q| == 0 and j ≤ t_l do
5:          Q ← rt \{sharing (t_l − j) > CPL > t_u with κ}
6:          j ← j + 1
7:      end while
8:      if |Q| > 0 then
9:          for i = 1 to i_max do
10:             C ← query ≤ α random peers in Q for κ
11:             Q ← Q ∪ C
12:             Q ← Q \{sharing t_l > CPL > t_u with κ}
13:             if C contains c with c.κ == κ then
14:                 return c.ip-address
15:             end if
16:         end for
17:     end if
18:     return NOTFOUND
19: end procedure
```

which corresponds to a remote procedure call. Moreover, the recursive variant uses a time-to-live variable TTL_{max} instead of a maximum iteration counter as it is the case for the iterative version. For each of the α recursions a counter variable TTL is maintained to delimit the maximum recursion depth. The initial candidate selection with lower threshold relaxation can be found in lines 4 through 8 similar to the iterative divPASS algorithm. Lines 10 through 14 initiate α parallel independent lookups by calling the lookup procedure which is implemented in lines 21 through 33. The latter procedure contacts only a single peer in the recursive step, in order to bind the parallelism degree to α.

5.5 Evaluation

To complement the analytic basis behind divRW and divPASS, we have conducted extensive simulations to validate the reliability and performance of divergent lookups. Specifically, three distinct simulation case studies have been conducted:

1. **taLEA case study** - underlines the significant taLEA impact on convergent lookups. This case study is considered as a baseline to experimentally substantiate the mitigation potential of divergent lookups in the third case study.

2. **PASS threshold selection** - provides experimental insights how suitable slices can be identified for PASS and we show that overlay networks with varying peer population sizes reveal similar suitability characteristics.

3. **divPASS vs. divRW** - presents the assessment of divRW and divPASS. Furthermore, it displays quantitative results in terms of reliability, performance, and timeliness of our approaches.

Algorithm 4 Divergent Lookup (recursive) - PASS

```
1: procedure LOOKUP-INITIATOR(κ, rt, α, TTL_max, t_l, t_u)
2:     j ← 1
3:     r ← NULL
4:     Q ← rt \{sharing t_l > CPL > t_u with κ}
5:     while |Q| == 0 and j ≤ t_l do
6:         Q ← rt \{sharing (t_l − j) > CPL > t_u with κ}
7:         j ← j + 1
8:     end while
9:     if |Q| > 0 then
10:        for i = 1 to α do
11:            p ← random peer from Q
12:            R ← p.lookup(κ, p, α, TTL_max, t_l, t_u)
13:            Q ← Q \{p}
14:        end for
15:        if R contains r with r.κ == κ then
16:            return r.ip-address
17:        end if
18:    end if
19:    return NOTFOUND
20: end procedure
21: procedure LOOKUP(κ, p, α, TTL, t_l, t_u)
22:    if p.rt contains κ then
23:        return entry in p.rt for κ
24:    else if TTL > 0 then
25:        TTL ← TTL − 1
26:        Q ← p.rt \{sharing t_l > CPL > t_u with κ}
27:        c ← pick random peer from Q
28:        Q ← Q \{c}
29:        return c.lookup(κ, c, α, TTL, t_l, t_u)
30:    else
31:        return NOTFOUND
32:    end if
33: end procedure
```

Before discussing results of these three evaluations, we first introduce our simulation environment, models, metrics, and parameters.

5.5.1 Simulations Overview

Experiments are performed using the discrete event simulator OMNeT++ [Pon93] and the OverSim [BHK09] extension that provides P2P protocol implementations. The simulator and the extension have a good reputation in the P2P research community for conducting experimental research case studies.

Each experiment run is simulated for 8 hours simulation time and is repeated at least 20 times. The average metric measurements are provided along with 95% confidence intervals. The measurements are recorded for the last 8000 seconds before the end of each run. This way, we ensure that peers have stored contact information of others in their routing tables due to the exposure to simulated workload and churn effects. Experiments using iterative and recursive lookups have been conducted by extending OverSim's Kademlia and R/Kademlia implementations correspondingly. Table 5.2 lists the most important simulation experiment parameters.

Parameter	**Value**	**Parameter**	**Value**
α (iterative)	10	α (recursive)	3
i_{max} (iterative)	50	TTL_{max} (recursive)	50
t_p (divRW)	80	w	128
t_l (divPASS)	4	t_u (divPASS)	6

Table 5.2: Simulation parameters.

5.5.2 Simulation Churn Models

Churn denotes entering and leaving peers in the overlay. These dynamic effects are often imposed by user behavior, benign failures of peers, or failures of the underlying networking infrastructure. Churn models define an average lifetime of a peer, i.e., the amount of time units it remains on the overlay network after joining it. Moreover, an average dead time is also part of our churn models to denote the waiting period after a peer leaves before a new one is joining the overlay. Using this approach, the peer population fluctuates only slightly once the initial overlay construction has been completed. Peer lifetimes are modeled using the Pareto distribution [YLWL06]. Three churn models are used throughout our studies:

- **NoChurn** Refers to a static peer population. Peers join at the specified rate at the beginning of the simulation and remain until the end. While victim peers $v \in V$ and malicious peers $m \in M$ are in our experiments subject to the *NoChurn* model, this does not always hold for the benign peer population $b \in B$. Benign peers can also be subject to the two Pareto models described at next.

- **P-500** This churn model is subject to a Pareto distribution with average peer lifetime and dead time of 500 seconds. Given the experiment runtime, this results in approximately 50 full exchanges of the benign peer population.

- **P-7200** This model is similar to the previous one, but with an average lifetime and dead time of 7200 seconds which results in approximately 4 full exchanges of the benign peer population.

5.5.3 Simulation Workload Models

The simulation experiments are subject to two different application workload models. Peers are sending application messages on average every 10 seconds to a randomly chosen peer, the sending interval is subject to a uniform distribution with a standard deviation of 5 seconds. Thereby, peers trigger lookup calls in case the random destination peer is unknown for the originator. The two workload classes differ in how the destination peers are chosen.

- **(W1) - Fully Distributed Application** This workload class selects the destination peer uniformly distributed [GDS+03] from $V \cup B$.
- **(W2) - Service Overlay Network** This workload class selects in 90% of the cases the destination peer uniformly distributed from V and in the remaining 10% of the cases from B.

5.5.4 Simulation taLEA Model

Varying amounts of malicious peers $|M|$ are introduced into the overlay throughout our taLEA experiments. To model the adversarial behavior, their lookup mechanism has been modified such that reply messages contain incorrect contact information of the destination peer, if the destination is a victim peer $v \in V$. This behavior can be detected by the lookup success rate metric which counts the aforementioned behavior as an unsuccessful lookup. Evaluation metrics are defined in the next subsection.

5.5.5 Evaluation Metrics

Three metrics steer our reliability and performance assessments for the different lookup variants:

- *Lookup Success Rate* (**LSR**) - this is our primary reliability metric. It specifies the average percentage of successful lookup calls of benign peers $b \in B$ which request contact information of victim peers $v \in V$. Moreover, we consider LSR as a taLEA resilience indicator in the divergent lookup validation process. A lookup fails once it receives contact information from a malicious peer $m \in M$ or when the maximum number of iterations i_{max} or the time-to-live TTL_{max} has exceeded and NOTFOUND is returned.
- *Message Complexity* (**MC**) - this is our primary performance metric. It allows to assess the network overhead of a given lookup method. It is the average number of messages sent of all successful lookup calls initiated by $b \in B$ and destined to $v \in V$.
- *Number of Iterations* (**NoI**) - this is our secondary performance metric. It offers an estimate for divergent lookup latencies. NoI is the average amount of algorithm iterations or recursions until it receives the first successful lookup reply, i.e., a reply by $b \in B$ that points at the destination $v \in V$.

5.5.6 taLEA Case Study

We conduct taLEAs with varied amounts of malicious peers $|M|$ in order to assess the lookup message loss, i.e., whenever a malicious peer receives a lookup message destined to a peer $v \in V$, it is indicated as a lost message. Furthermore, we assess the message complexity for convergent lookups to compare them with the divergent variants. Experiment results for iterative convergent routing are presented in Figure 5.5 for $N = \{5000, 10000, 50000\}$ peers. Both workloads show a message loss of at most 89% for the P-500 churn model with $|M| = 64$, and close to 45% (W1) or 40% (W2) with $|M| = 8$. Message loss decreases for increasing peer lifetimes in both workload scenarios. MC measurements are in the range from 10.3 to 11.3 for W1 and 4.9 to 5.3 with the W2 workload. Figure 5.6 presents message loss for the recursive convergent lookup in the W1 workload scenario. The average message loss for $|M| = 64$ is – when compared to the iterative variant – with about 61% to 11% lower and yields rates between 28% and 78%. MC measurements are in the range of 2.9 to 4.3.

Interpreting the Results

Peers in the NoChurn scenario are exposed for the longest time to the workload, therefore they have a higher chance of storing the correct contact information of victim peers which may be replied to other benign peers in subsequently received lookups. Hence, in P-500 and P-7200 scenarios it is more likely that peers request contact information for $v \in V$ and retrieve incorrect contact information. Moreover, the provision of bogus contact information about $v \in V$ is increasing with $|M|$ and consequently message loss increases as well.

5.5.7 PASS Threshold Selection

The reliability and message complexity of divPASS depends on choosing good lower and upper thresholds t_l and t_u. Insights about the selection are provided through an experiment set whose results are shown in Figures 5.7a through 5.7d. These present two ratios: (i) peers with LDEs to the destination peers (blue squares) and (ii) no-dead-ends (red diamonds). The number of peers per slice (dashed line) denotes the y-axis on the right hand side. All x-axes show the CPL range $[0, 20]$. The depicted measurements reflect simulations with the W1 workload and the NoChurn model for different values of N and w; the choices led to different address space load factors $\rho = \frac{N}{2^w}$, which have been considered in this experiment series to compare dense to sparsely populated address spaces.

Interpreting the Results

Similar key characteristics of Figures 5.7a through 5.7d show an increase of peers with LDE ratio up to CPL 12 (or 14 for Figure 5.7d) whereas the no-dead-end ratio decreases. As a consequence, the upper threshold t_u should be chosen such that the chance of contacting a dead-end can be neglected. The selection of the lower threshold t_l must ensure that the amount of peers populated in the lookup slice range $[t_l, t_u]$ is not excessively large, such that α parallel lookups with search depth i_{max} (TTL_{max} respectively for the recursive variant) have a high probability of finding an LDE to the destination. We conjecture that the intersection of the LDE curve and the no-dead-end curve is a good choice for the lower threshold. Thus, for the set of experiments using $w = 128$ and $N = \{5000, 10000, 20000\}$ in the next subsection, we have selected $t_l = 4$ and $t_u = 6$ as divPASS parameters.

We believe due to the measurement similarities that overlays with both, high load factors and longer key lengths, can also apply divPASS in regard of the expected reliability and performance, i.e., overlays hosting millions of peers

5.5.8 Divergent Lookup Reliability and Performance Case Study

In this subsection, divPASS is compared to divRW using our three metrics LSR, MC, and NoI.

Iterative, W1

Figures 5.8a through 5.8c show experiments for the iterative divergent lookup comparison using workload W1. In terms of LSR, a pairwise comparison of divPASS to divRW for similar (churn, N) configurations always shows a better LSR measurement for divPASS; its measurements are in the range of 90% to 100% successful lookups, whereas divRW results are in the range of 60% to 90%. Figure 5.8b depicts MC, and shows for all divPASS configurations a significant lower overhead for the same divRW configuration. Clearly, divPASS MC outranks divRW MC by factor 3 to 6 depending on the considered (churn, N) configuration. The iterative divPASS experiments have been conducted using $\alpha = 10$, yet the average MC measurements show 10 messages or less for some of the configurations. This happens, as PASS searches slices that reveal a high destination knowledge ratio but are sparsely populated which may result in a successful divergent lookup call after querying less than α peers. Figure 5.8c shows the NoI; comparable to the previous MC discussion, we find for NoI significantly better divPASS measurements, i.e., in the range between 1.3 to 1.6 iterations. The corresponding divRW NoI measurements are in the range between 2.2 and 3.4 iterations. Clearly, divPASS shows better results which makes us believe that divPASS would provide better latencies than divRW for successful lookups in real networks.

Iterative, W2

Figures 5.9a through 5.9c show experiment results to compare the iterative divergent lookups using workload W2. Average LSR in Figure 5.9a shows equally good results for divPASS like in the W1 workload case, yet divRW has a higher LSR in W2 compared to W1. Overall, all LSR results are in the range between 98% to 100%. Average MC results are shown in Figure 5.9b. The maximum MC improvement of divPASS over divRW is about factor 2 in the W2 case. Although, the MC difference between divPASS and divRW is lower for W2 than for W1, it is important to consider the different MC scale ranges of W2 (Figure 5.9b) and W1 (Figure 5.8b). divPASS MC measurements are in the range of 5.8 to 10.2 and divRW in the range of 11 to 13. NoI measurements in Figure 5.9c show in the pairwise (churn, N) configuration comparison that divPASS measurements do not outrank those of divRW like it is the case for W1. Because in W2 more contact information among benign peers about $v \in V$ exists, divRW scores better as it can select without slice range restriction easily α peers during the first iteration with a high probability of a successful lookup. As divPASS searches a subset of slices, it may occur that during the first iteration less than α peers for the given slice range are found. If these peers from the first iteration cannot resolve $v \in V$, a second iteration is required. All measurements are close to each other in the range of 1 to 1.4 iterations.

Recursive, W1

Figures 5.10a through 5.10c show experiment results for the comparison of the recursive divergent lookup variants using workload W1. LSR measurements in Figure 5.10a show success rates in the range of 95% to 100% and divRW shows slightly better LSR results than divPASS. On the other hand, MC for divPASS is around 10 messages while divRW is in the range of 18 to 39 messages, as can be taken from Figure 5.10b. NoI measurements are in the range of 1.4 to 2.5 iterations, as depicted in Figure 5.10c.

5.5.9 Interpreting the Results

The first part of our case study shows that taLEA may cause message loss of 60% up to 89%. We propose PASS for divergent lookups to mitigate taLEAs and propose a PASS threshold selection method that considers two ratios: peers knowing the destination and no-dead-ends per slice. Experiment data supports the claim that PASS is a suitable taLEA mitigation approach for very large overlay networks with hundreds of thousands to potentially millions of peers. The actual divPASS validation yields a worst case result of 90% LSR. Failing divPASS lookups can be ascribed to dead-ends or lack of suitable candidate peers, especially for churn models with short average peer lifetimes. The pairwise comparison of iterative divPASS to iterative divRW shows divPASS's superiority for the W1 workload model. The W2 workload model shows similar and good LSR results for both approaches. Furthermore, divPASS either significantly reduces MC or is at worst on par with divRW. Also, NoI for divPASS shows an improvement over divRW in case of workload W1. In case of W2, correct contact information of the victim peers is widely available in the overlay, which allows divRW to retrive it faster than divPASS which has to select an appropriate set of peers for the smaller PASS search region.

5.6 Limitations

Divergent lookups are adequate for taLEA mitigation in various contexts, yet they are limited in their applicability if one of the following aspects is not fulfilled:

- Divergent lookups require LDEs in the overlay graph. The creation of LDEs is usually a side effect of message exchange among peers and the mandated lookup calls beforehand. Therefore, divergent lookups require in general overlays with a reasonable application workload frequency.
- Malicious peers conducting the taLEA may not have joined the overlay from the beginning or (in churn scenarios) must join with a reasonable delay – which is workload dependent – after the victim peer has joined the overlay. Otherwise, either no LDEs or only incorrect LDEs will exist in the overlay graph. In other words, benign information about the victim peers has to be available.
- Divergent lookups have a slightly higher cost in terms of network overhead compared to convergent ones, this may not be desired in some systems.

5.7 Summary & Conclusion

We have presented a novel lookup algorithm for structured P2P protocols. The lookups use two different search strategies and show high resilience to taLEAs, i.e., the taLEA mitigation rate is on

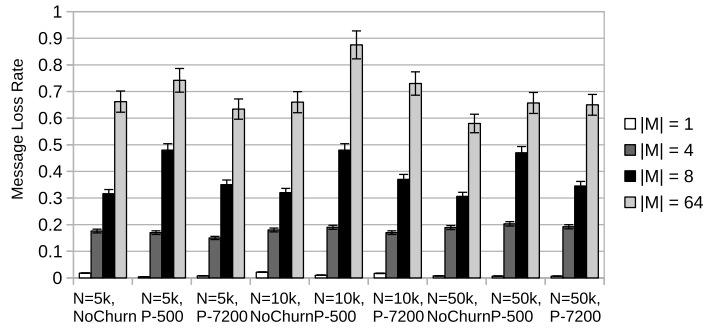
average 90% to 100%. The new lookups have been validated using simulations for the P2P protocols Kademlia and R/Kademlia which are subject to iterative and recursive lookup message exchanges. Divergent lookups fulfill requirements that address a potentially large field of application cases and also pertain fundamental P2P features:

- Reliability: divergent lookups mitigate taLEAs with a high probability.
- Scalability: searches should be applicable irrespective of the overlay's size or the application running on it.
- Decentralized operation: searches should be satisfiable with data stored in routing tables and do not require an external oracle.
- Timeliness: should be similar to convergent lookups, such that divergent lookups can be applied for a wide field of established P2P applications as well.
- Anonymity: peers do not require a certificate or public key infrastructure in order to authenticate themselves for joining and other operations on the overlay network.

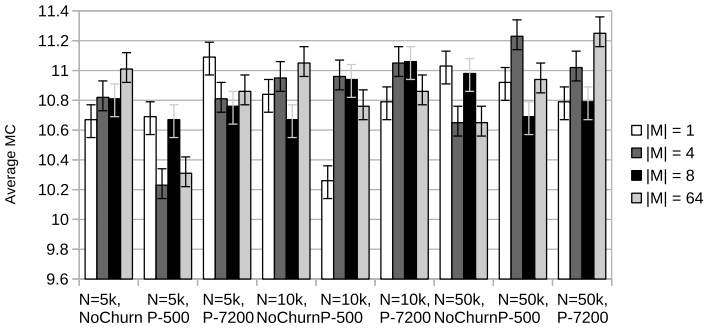
The achievements relate mainly to contributions (C2) and (C3) of this thesis. As an extension to contribution (C1), we have provided in this chapter a more detailed taLEA baseline of convergent lookup susceptibility than in the previous chapter. The difference is mostly in the wider parameter landscape, such as total amount of peers, varying amounts of malicious peers, more churn scenarios, two workload models, and both iterative and recursive routing. The workloads have been defined to be similar to fully distributed applications, such as data discovery using a DHT, and also to service overlay networks which usually contain a few sources and sinks that are considered more important than the majority of peers.

We have introduced three metrics which allow for a reliability and message complexity cross comparison of convergent and divergent lookups. An additional metric measures the number of iterations in divergent lookup simulations, its average serves as a rough estimate for timeliness, especially to validate divPASS's superiority over divRW. Moreover, we believe that divPASS is applicable for overlay networks with hundreds of thousands to millions of peers based on experimental results to support the slice selection process of PASS. The analysis showed similar characteristics in terms of two ratios (per slice LDE ratio and no-dead-end ratio) independent of different network sizes and overlay densities.

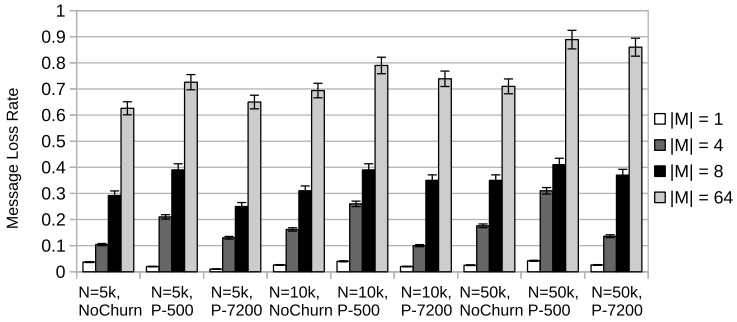
The implementation overhead for divergent lookups requires only small changes to convergent lookups, nevertheless, a change of the protocol API is required. We highlight that divergent lookups should not replace convergent ones, hence we propose n-version-lookups which uses the n-version-programming framework [Avi85] to build a combined lookup consisting of convergent and divergent variants in order to achieve both, highly reliable and low latency operation.



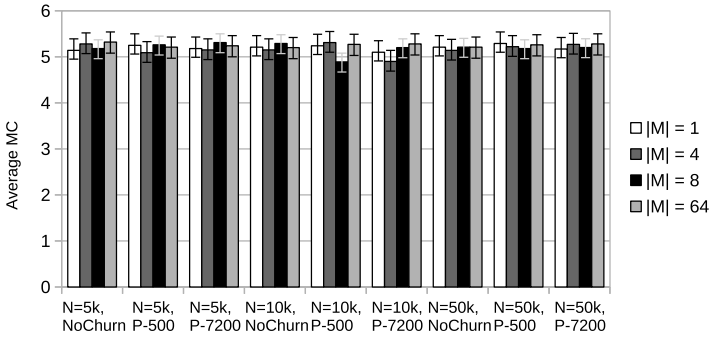
(a) W1, average message loss



(b) W1, average MC

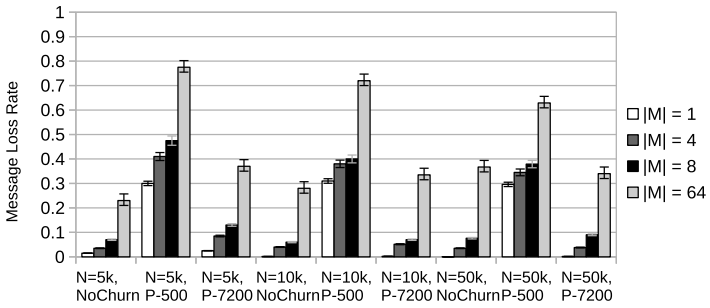


(c) W2, average message loss

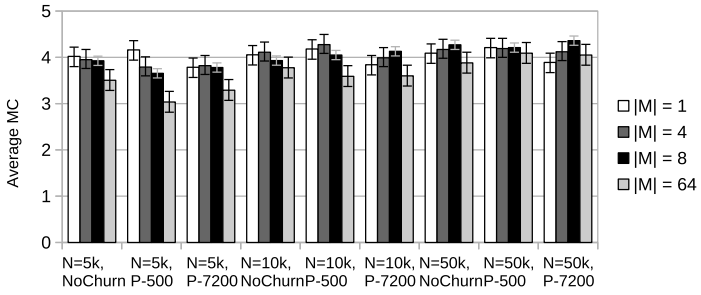


(d) W2, average MC

Figure 5.5: Convergent iterative lookup during taLEA.

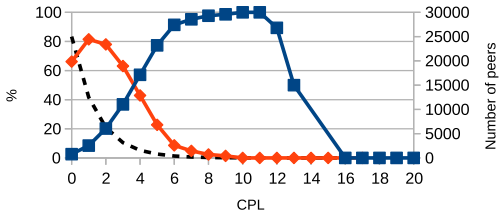


(a) W1, average message Loss

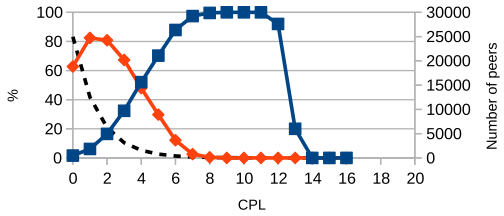


(b) W1, average MC

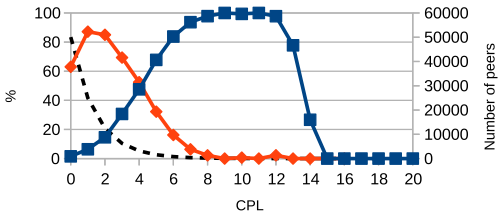
Figure 5.6: Convergent recursive lookup during taLEA.



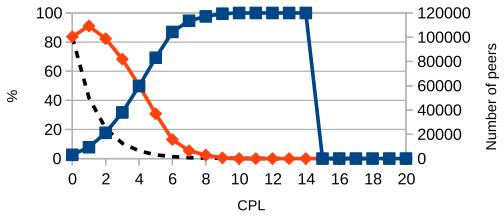
(a) $N = 50.000$, $w = 128$, $\rho \approx 1,92 \cdot 10^{-34}$



(b) $N = 50.000$, $w = 16$, $\rho \approx 0,762$

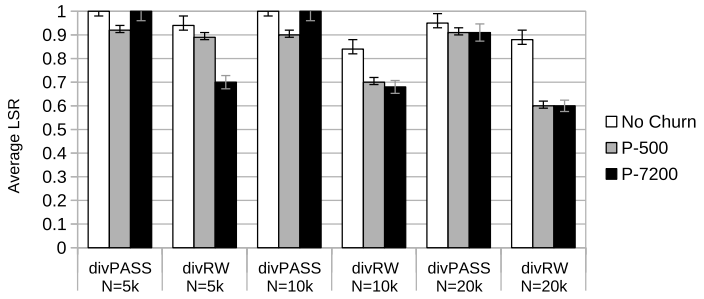


(c) $N = 100.000$, $w = 20$, $\rho \approx 0,095$

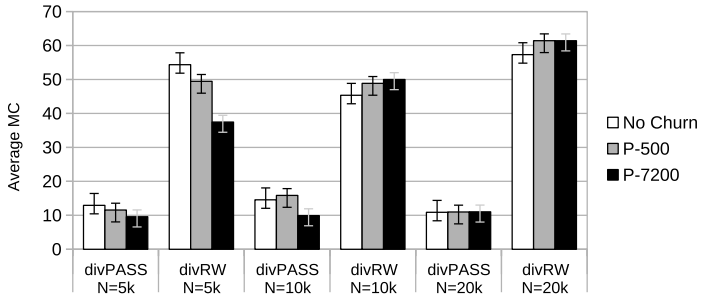


(d) $N = 200.000$, $w = 20$, $\rho \approx 0,190$

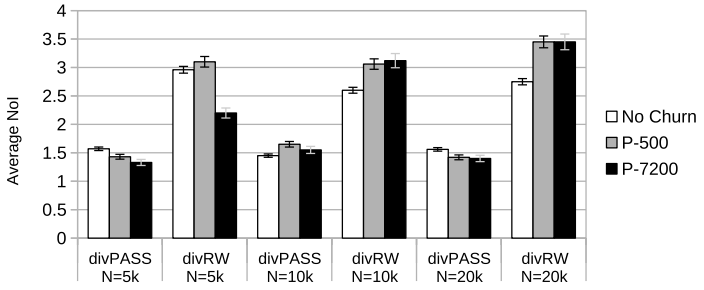
Figure 5.7: Ratio of peers with LDEs to destination peer (blue boxes), no-dead-end ratio (red diamonds), number of peers (dashed).



(a) Lookup success rate (LSR)

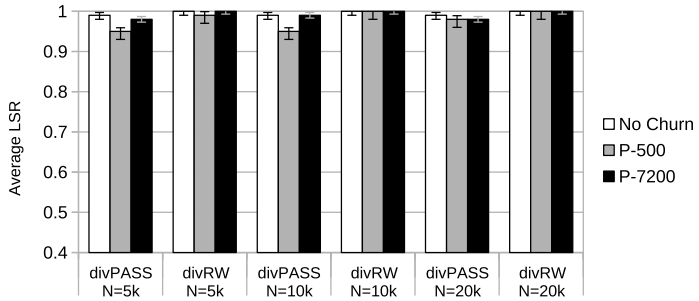


(b) Message complexity (MC)

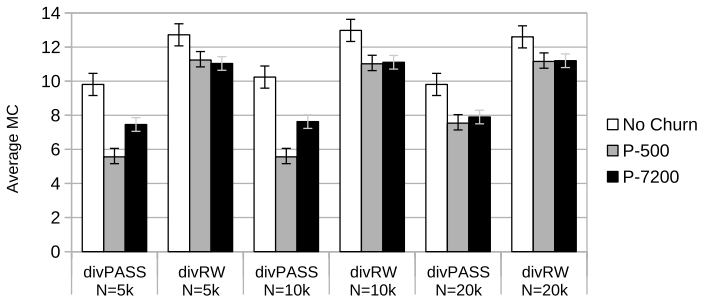


(c) Number of iterations (NoI)

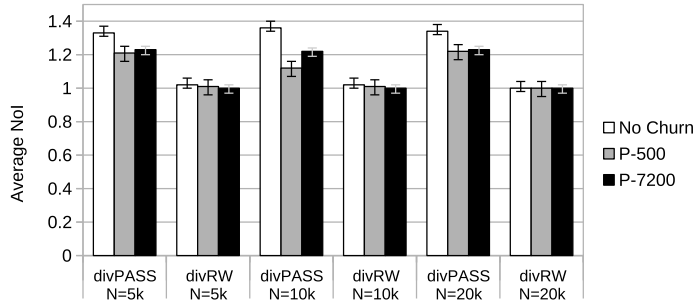
Figure 5.8: Comparison of iterative divPASS vs. divRW, W1.



(a) Lookup success rate (LSR)

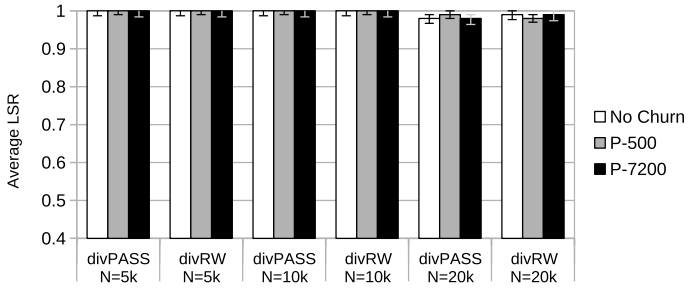


(b) Message complexity (MC)

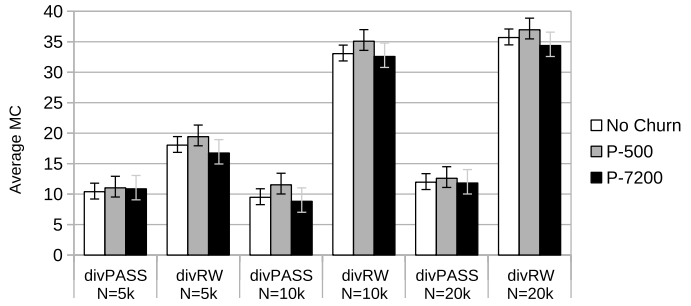


(c) Number of iterations (NoI)

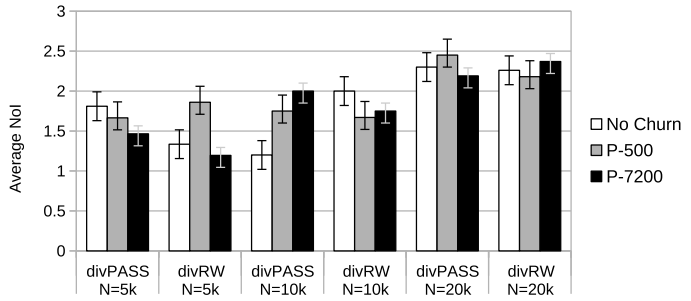
Figure 5.9: Comparison of iterative divPASS vs. divRW, W2.



(a) Lookup success rate (LSR)



(b) Message complexity (MC)



(c) Number of iterations (NoI)

Figure 5.10: Comparison of recursive divPASS vs. divRW, W1.

6 Concluding Remarks

6.1 Summary & Conclusion

Contemporary large-scale applications increasingly apply P2P technology as it provides features such as scalability, fault-tolerance, and decentralization. P2P protocols are often applied in data discovery and data dissemination applications, e.g., file sharing, instant messaging, Car-2-Car communication, or the more general case of machine-to-machine (M2M) communication. M2M will expectedly grow up to 11.5 billion mobile, M2M-enabled devices by 2019 [Cis15]. P2P provides inherent fault-tolerance and hence allows for dependable service provision in perturbed operational environments. Yet, this is just one side of the coin, as critical applications additionally require secure operation. An adversary could for example violate the right to informational self-determination, induce venue loss, or cause losses of life or equipment.

P2P architectures are essentially different from classical client/server architectures in a sense that each peer usually has to fulfill client and server duties. Moreover, the data maintained by peers is stored widely distributed. Both, system design and security requirements of P2P protocols are consequently different from client/server architectures. Hence, new security weaknesses exist that demand mitigating responses which are desired to pertain the beneficial features of P2P designs.

Firstly, as a ahead of our technical contributions, we discuss P2P security issues, including design weaknesses, corresponding attacks, and mitigation techniques in Chapter 2 as contribution (C0). In more detail, we focus on the so called Eclipse attack (EA) and specific localized variants. In an EA, a set of malicious peers in the overlay network prevents benign peers from proper service provision. Relating back to established security notions, an EA can be considered as a combination of man-in-the-middle attacks, routing table poisoning, and distributed denial-of-service attacks. As a consequence, EAs have the ability to affect all three security protection goals, i.e., availability, confidentiality, and integrity. Moreover, one of the EA variants, the topology-aware localized Eclipse attack (taLEA) is highly efficient. taLEAs achieve severe impacts using only a limited amount of peers irrespective of the overlay network size. This leads to the first research question of this thesis:

(R1) What is the impact of taLEAs?

In order to investigate the impact of taLEAs on structured P2P protocols, we are focusing on the lookup mechanism's design weakness that exists in various P2P protocol implementations. Our first technical contribution (C1) proposes a set of heuristics that reflect the lookup mechanisms of different P2P protocols (cf. Chapter 4). The heuristics allow to estimate the amount of lookup calls which can be intercepted by an attacker as a prerequisite action for launching a taLEA. The heuristics have been validated using simulation experiments and our investigations yielded a significant taLEA impact, i.e., on average 50% to 95% of the lookup calls could be intercepted.

After having established this initial baseline, we were interested in taLEA effects for a wider parameter landscape:

- Varied amounts of malicious peers.

- P2P protocols with recursive lookup mechanism.
- Different application workload scenarios.
- Several churn scenarios.
- Up to five times larger overlay networks.

The previous parameter landscape has been considered in our second technical contribution (C2) which includes a profound baseline measurement case study, too. It shows the significant impact of taLEAs against various P2P protocols in a multitude of scenarios. Using only 4 malicious peers for a taLEA results in roughly 20% lookup call interception for the W1 workload case and iterative routing. In case for 64 malicious peers, lookup interception rates drop with only few exceptions below 60%. For all baseline results the interception rate does not appear to drop for increasing overlay sizes, which underlines the criticality of taLEAs. This observation leads to our second research question:

(R2) Is there a taLEA mitigation technique that does not sacrifice P2P's benefits?

We have proposed divergent lookups as a taLEA mitigation technique. Our second technical contribution (C2) introduces a divergent lookup with a simple random walk search strategy (cf. Chapter 5). The assessment of divergent lookups has been conducted using simulation experiments for a parameter subset of the baseline measurements. Moreover, the same metrics from the baseline study were utilized for assessing divergent lookups to allow for a concise comparison in terms of reliability, message complexity, and timeliness.

While the reliability of divergent lookups with random walks reaches 70% to 100%, the message complexity results are about factor 5 higher compared to convergent lookups. Hence, we have proposed a P2P address space slicing framework as our third technical contribution (C3), that can be integrated with divergent lookups (cf. Chapter 5). The framework narrows down the search space of the divergent lookup. To do so, the framework considers structural routing table properties to identify address space regions that should be looked up due to their high amount of peers that know the destination peer. The result of performing lookups in these specific regions only is a decrease in message complexity compared to (C2) to a factor range of 1.5 to 2.5 compared to convergent lookups. Furthermore, (C3) increases the reliability towards the range of 90% to 100%.

Overall, divergent lookups promote scalability, openness, anonymity, resource frugality, decentralization, and heterogeneity. Therefore, the solution is applicable for a large set of scenarios. The implementation overhead for divergent lookups is expected to be low.

6.2 Limitations

The analyses and techniques presented throughout this thesis are subject to the system model and attack model which have been laid out in Chapter 3. Moreover, the following limitations are of importance:

- We concentrate in our work on P2P protocols which use convergent lookup mechanisms, i.e., decrease on each iteration the distance towards the destination peer based on an address space scale system.

- The existence of shortcuts towards the victim peers (LDEs) in the topology is essential for divergent lookups. In order for LDEs to be present, we expect that the overlay is not idle and peers exchange messages at a reasonable rate.
- Malicious and victim peers may not join simultaneously, but are required to join with a given time-wise delta. Otherwise, victim peers will not be exposed to the application workload and consequently victim LDEs are unlikely to be established among victim and benign peers.
- The cost of divergent lookups using the address space slicing technique – in contrast to the convergent variant – is slightly higher in terms of latencies and message complexity.

6.3 Future Research

The following directions for future research are currently envisaged:

- Advanced attack models: up to now, we were considering constant attacker behavior. This will be changed in future work as a means of decreasing the probability of detecting malicious peers. Moreover, attackers may on lookup message interception mount various malicious actions, we will also focus on different of these variants. Moreover, we are assessing divergent lookups for non topology-aware localized scenarios.
- Malicious information decay: an experimental case study about the decay of malicious information over time that has been stored in the routing tables of benign peers for a wide parameter landscape will be helpful to assess the efficiency of different attack and mitigation strategies.
- Decentralized detection: in more sophisticated attack models, the differentiation based on a lookup result if the responding peer is either poisoned or malicious is not trivial. This problem is a prerequisite for the next future research aspect:
- Routing table sanitizing: once a benign peer has detected a malicious peer, it should be removed from as many routing tables as possible in order to decrease malicious peers attack capabilities. This requires a carefully defined protocol in order to allow for secure operation, such that it cannot be misused by an attacker.

7 Publications

7.1 First Author

- Security Aspects of Peer-to-Peer Protocols, Daniel Germanus and Neeraj Suri, submitted to ACM Computing Surveys (CSUR), 2015
- PASS: An Address Space Slicing Framework for P2P Attack Mitigation, Daniel Germanus, Hatem Ismail, Neeraj Suri, submitted to the 34th IEEE Symposium on Reliable and Distributed Systems (SRDS), 2015
- Mitigating Eclipse Attacks in Peer-to-Peer Networks, Daniel Germanus, Stefanie Roos, Thorsten Strufe, Neeraj Suri, in Proceedings of IEEE Conference on Communications and Network Security (CNS), 2014
- Coral: Reliable and Low-latency P2P Convergecast for Critical Sensor Data Collection, Daniel Germanus, Johannes Schwandke, Abdelmajid Khelil, Neeraj Suri, in Proceedings of IEEE SmartGridComm Symposium, 2013
- Susceptibility Analysis of Structured P2P Systems to Localized Eclipse Attacks, Daniel Germanus, Robert Langenberg, Abdelmajid Khelil, Neeraj Suri, in Proceedings of IEEE Symposium on Reliable Distributed Systems (SRDS), 2012
- Leveraging the Next-Generation Power Grid: Data Sharing and Associated Partnerships, Daniel Germanus, Ionna Dionysiou, Harald Gjermundrød, Abdelmajid Khelil, Neeraj Suri, David E. Bakken, and Carl Hauser, in Proceedings of IEEE Conference on Innovative Smartgrid Technologies (ISGT) Europe, 2010
- Increasing the Resilience of Critical SCADA Systems Using Peer-to-Peer Overlays, Daniel Germanus, Abdelmajid Khelil, Neeraj Suri, in Proceedings of International Symposium on Architecting Critical Systems (ISARCS), 2010
- Threat Modeling and Dynamic Profiling, Daniel Germanus, Andréas Johansson, Neeraj Suri, in Annals of Emerging Research in Information Assurance, Security and Privacy Services, Elsevier, 2009

7.2 Coauthor

- Attacking and Hardening Divergent Lookups against Localized Attacks in P2P Systems, Hatem Ismail, Daniel Germanus, Neeraj Suri, submitted to IEEE Conference on Communications and Network Security (CNS), 2015
- Robust and Real-time Communication on Heterogeneous Networks for Smart Distribution Grid, Kubilay Demir, Daniel Germanus, Neeraj Suri, in Proceedings of IEEE SmartGridComm Symposium, 2014

- Espousing Peer-to-Peer Communication for the Smart Grid (chapter in the Book Smart Grid Communications Vision for 2015, 2020, and 2030), Abdelmajid Khelil, Daniel Germanus, Neeraj Suri, in Wiley Press, 2012
- Protection of SCADA Communication Channels (Chapter 9 in the Book Critical Infrastructure Protection: Advances in Critical Infrastructure Protection: Information Infrastructure Models, Analysis, and Defense), Abdelmajid Khelil, Daniel Germanus, Neeraj Suri, Springer, ISBN-13: 978-3642289194, 2012
- Computer Dependability and Security Evaluation - (Chapter 8 in the Book Critical Infrastructure Security: Assessment, Prevention, Detection, Response), Stefan Winter, Daniel Germanus, Hamza Ghani, Thorsten Piper, Abdelmajid Khelil, Neeraj Suri, in WIT Press, ISBN-13: 978-1845645625, 2012
- The human role in tools for improving robustness and resilience of critical infrastructures, Aladino Amantini, Michal Choras, Salvatore D'Antonio, Elyoenai Egozcue, Daniel Germanus, Reinhard Hutter, in Journal of Cognition, Technology & Work, 2012
- Towards Benchmarking of P2P Technologies from a SCADA Systems Protection Perspective, Abdelmajid Khelil, Sebastian Jeckel, Daniel Germanus, Neeraj Suri, in Proceedings of the 2nd International Conference on Mobile Lightweight Wireless Systems, 2010

Curriculum Vitae

Daniel Germanus was born in Frankfurt am Main, Germany, on January 27th 1979. At age 5 he was confronted for the first time with the BASIC programming language on a Commodore 64 home computer. From then on, his fascination of the computing machinery grew steadily. After finishing high school, Daniel worked as a software engineer for different companies and became an entrepreneur. He received the Bachelor of Science degree in computational engineering at Technical University Darmstadt in 2006 and the Master of Science degree in computer science, also at Technical University Darmstadt, in 2009. In the same year, he started his Ph.D. research on dependability and security of distributed systems, especially P2P protocols.

Bibliography

[ALRL04] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, Jan 2004.

[AS06] Baruch Awerbuch and Christian Scheideler. Towards a scalable and robust dht. In *Proc. ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '06, pages 318–327, New York, NY, USA, 2006.

[ATS04] Stephanos Androutsellis-Theotokis and Diomidis Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Comput. Surv.*, 36(4):335–371, 2004.

[Avi85] A. Avizienis. The N-Version Approach to Fault-Tolerant Software. *IEEE Transactions on Software Engineering*, SE-11(12):1491–1501, 1985.

[BdAMNCdSBV13] R. Barra de Almeida, J.A. Miranda Natif, A.P. Couto da Silva, and A. Borges Vieira. Pollution and whitewashing attacks in a p2p live streaming system: Analysis and counter-attack. In *Proc. Communications (ICC)*, pages 2006–2010, 2013.

[BHK09] I. Baumgart, B. Heep, and S. Krause. Oversim: A scalable and flexible overlay framework for simulation and real network applications. *In Proc. P2P*, pages 87–88, 2009.

[Bit08] BitTorrent. Bittorrent protocol specification. Technical report, 2008.

[BM07] I. Baumgart and S. Mies. S/Kademlia: A practicable Approach towards Secure Key-based Routing. *In Proc. ICPADS*, pages 1–8, 2007.

[CCF09] Thibault Cholez, Isabelle Chrisment, and Olivier Festor. Evaluation of Sybil Attacks Protection Schemes in Kad. *Scalability of Networks and Services*, 5637:70–82, 2009.

[CCFD13] Thibault Cholez, Isabelle Chrisment, Olivier Festor, and Guillaume Doyen. Detection and Mitigation of Localized Attacks in a widely Deployed P2P Network. *Peer-to-Peer Networking and Applications*, 6(2):155–174, 2013.

[CDF+14] S. Cirani, L. Davoli, G. Ferrari, R. Leone, P. Medagliani, M. Picone, and L. Veltri. A scalable and self-configuring architecture for service discovery in the internet of things. *Internet of Things Journal, IEEE*, 1(5):508–521, 2014.

[CDG+02] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. Secure Routing for Structured Peer-to-Peer Overlay Networks. *SIGOPS Oper. Syst. Rev.*, pages 299–314, 2002.

[Cis15] Cisco Systems. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014-2019. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html, 2015. (Accessed on 2015-04-20).

[CKGM04] T. Condie, S.D. Kamvar, and H. Garcia-Molina. Adaptive Peer-to-Peer Topologies. In *Proc. P2P*, pages 53 – 62, 2004.

[CLB09] Jianguo Chen, Huijuan Lu, and S.D. Bruda. A solution for whitewashing in p2p systems based on observation preorder. In *Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC'09*, volume 2, pages 547–550, 2009.

[CRB+03] Yatin Chawathe, Sylvia Ratnasamy, Lee Breslau, Nick Lanham, and Scott Shenker. Making gnutella-like p2p systems scalable. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 407–418, New York, NY, USA, 2003.

[DM04] F. DePaoli and L. Mariani. Dependability in peer-to-peer systems. *Internet Computing, IEEE*, 8(4):54–61, 2004.

[DMS04] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor : The Second-Generation Onion Router. *In Proc. USENIX Security Symposium*, 2004.

[Dou02] John R. Douceur. The sybil attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 251–260, London, UK, UK, 2002.

[GDS+03] Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, and John Zahorjan. Measurement, Modeling, and Analysis of a Peer-to-peer File-sharing Workload. In *SOSP '03*, pages 314–329, New York, NY, USA, 2003.

[GIP+13] João V. Gomes, Pedro R. M. Inácio, Manuela Pereira, Mário M. Freire, and Paulo P. Monteiro. Detection and classification of peer-to-peer traffic: A survey. *ACM Comput. Surv.*, 45(3):30:1–30:40, 2013.

[GKSS13] D. Germanus, A. Khelil, J. Schwandke, and N. Suri. Coral: Reliable and low-latency p2p convergecast for critical sensor data collection. In *Proc. SmartGridComm*, pages 300–305, 2013.

[GLKS12] D. Germanus, R. Langenberg, A. Khelil, and N. Suri. Susceptibility Analysis of Structured P2P Systems to Localized Eclipse Attacks. In *Proc. Reliable Distributed Systems (SRDS), 2012*, pages 11–20, 2012.

[GRSS14] Daniel Germanus, Stefanie Roos, Thorsten Strufe, and Neeraj Suri. Mitigating Eclipse attacks in Peer-To-Peer networks. In *Proc. Communications and Network Security (CNS), 2014*, pages 400–408, 2014.

[HK03] Kirsten Hildrum and John Kubiatowicz. Asymptotically efficient approaches to fault-tolerance in peer-to-peer networks. In FaithEllen Fich, editor, *Distributed Computing*, volume 2848 of *Lecture Notes in Computer Science*, pages 321–336. 2003.

[HL09] Michael Howard and Steve Lipner. *The security development lifecycle*. O'Reilly Media, Incorporated, 2009.

[KKHY13] Timo Koskela, Otso Kassinen, Erkki Harjula, and Mika Ylianttila. P2p group management systems: A conceptual analysis. *ACM Comput. Surv.*, 45(2):20:1–20:25, 2013.

[KLR09] Michael Kohnen, Mike Leske, and ErwinP. Rathgeb. Conducting and Optimizing Eclipse Attacks in the Kad Peer-to-Peer Network. *LNCS*, 5550:104–116, 2009.

[KT13] Anne-Marie Kermarrec and Peter Triantafillou. Xl peer-to-peer pub/sub systems. *ACM Comput. Surv.*, 46(2):16:1–16:45, 2013.

[Kwo09] Yu-Kwong Kwok. Autonomic peer-to-peer systems: Incentive and security issues. In Yan Zhang, Laurence Tianruo Yang, and Mieso K. Denko, editors, *Autonomic Computing and Networking*, pages 205–236. 2009.

[LCP+05] Eng Keong Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys Tutorials, IEEE*, 7(2):72–93, 2005.

[LMJV13] Daniel Lazaro, Joan Manuel Marques, Josep Jorba, and Xavier Vilajosana. Decentralized resource discovery mechanisms for distributed computing in peer-to-peer environments. *ACM Comput. Surv.*, 45(4):54:1–54:40, 2013.

[LMSW10] Thomas Locher, David Mysicka, Stefan Schmid, and Roger Wattenhofer. Poisoning the Kad network. *LNCS*, 5935:195–206, 2010.

[LNR06] Jian Liang, N. Naoumov, and K.W. Ross. The index poisoning attack in p2p file sharing systems. In *Proc. INFOCOM*, pages 1–12, 2006.

[LWC09] Dan Li, Jianping Wu, and Yong Cui. Defending against buffer map cheating in donet-like p2p streaming. *IEEE Transactions on Multimedia*, 11(3):535–542, 2009.

[LYL14] Qiang Li, Jie Yu, and Zhoujun Li. An enhanced kad protocol resistant to eclipse attacks. In *Proc. Networking, Architecture, and Storage (NAS)*, pages 83–87, 2014.

[MB09] Prateek Mittal and Nikita Borisov. ShadowWalker: Peer-to-peer Anonymous Communication using Redundant Structured Topologies. *In Proc. CCS*, pages 161–172, 2009.

[MM02] P. Maymounkov and D. Mazières. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. *In Proc. IPTPS*, pages 53 – 65, 2002.

[MNR02] Dahlia Malkhi, Moni Naor, and David Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In *Symposium on Principles of Distributed Computing*, PODC '02, pages 183–192, 2002.

[MW11] P.K. Manadhata and J.M. Wing. An attack surface metric. *IEEE Transactions on Software Engineering*, 37(3):371–386, 2011.

[NR06] Naoum Naoumov and Keith Ross. Exploiting p2p systems for ddos attacks. In *Proc. Scalable Information Systems*, InfoScale '06, New York, NY, USA, 2006.

[NW06] Arjun Nambiar and Matthew Wright. Salsa: A Structured Approach to Large-scale Anonymity. *In Proc. CCS*, pages 17–26, 2006.

[OC01] Eunseuk Oh and Jianer Chen. Parallel Routing in Hypercube Networks with Faulty Nodes. *In Proc. ICPADS*, pages 338–345, 2001.

[Pas12] Andrea Passarella. A survey on content-centric technologies for the current Internet: CDN and P2P solutions. *Computer Communications*, 35(1):1 – 32, 2012.

[Pon93] György Pongor. OMNeT: Objective Modular Network Testbed. In *In Proc. MASCOTS*, 1993.

[RD01] Antony Rowstron and Peter Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. *In Proc. Middleware*, pages 329–350, 2001.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. *SIGCOMM Comput. Commun. Rev.*, 31(4):161–172, 2001.

[RHB08] R. Ranjan, A. Harwood, and R. Buyya. Peer-to-peer-based resource discovery in global grids: a tutorial. *Communications Surveys Tutorials, IEEE*, 10(2):6–33, 2008.

[Rip01] M. Ripeanu. Peer-to-peer architecture case study: Gnutella network. In *Proc. Peer-to-Peer Computing*, pages 99–100, 2001.

[SCDR04] Atul Singh, Miguel Castro, Peter Druschel, and Antony Rowstron. Defending against Eclipse Attacks on Overlay Networks. In *Proc. SIGOPS*, pages 115–120, 2004.

[SDH+10] Max Schuchard, Alexander W. Dean, Victor Heorhiadi, Nicholas Hopper, and Yongdae Kim. Balancing the Shadows. *In Proc. Workshop on Privacy in the Electronic Society*, pages 1–10, 2010.

[SENB07] Moritz Steiner, Taoufik En-Najjary, and Ernst W. Biersack. Exploiting KAD : Possible Uses and Misuses. *Computer Communication Review*, 37(5):65–69, 2007.

[SMLN+03] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Networking*, 11(1):17–32, 2003.

[SNDW06] A. Singh, T.-W. Ngan, P. Druschel, and D.S. Wallach. Eclipse Attacks on Overlay Networks: Threats and Defenses. *In Proc. INFOCOM*, pages 1–12, 2006.

[SS04] Frank Swiderski and Window Snyder. *Threat modeling*. Microsoft Press, 2004.

[SSNRR10] J. Seibert, Xin Sun, C. Nita-Rotaru, and Sanjay Rao. Towards securing data delivery in peer-to-peer streaming. In *Communication Systems and Networks (COMSNETS), 2010 Second International Conference on*, pages 1–10, 2010.

[STFG13] M. Steinheimer, U. Trick, W. Fuhrmann, and B. Ghita. P2p-based community concept for m2m applications. In *Future Generation Communication Technology (FGCT)*, pages 114–119, 2013.

[UPS11] Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. A Survey of DHT Security Techniques. *ACM Computing Surveys*, pages 8:1–8:49, 2011.

[VdPVA10] Ricardo Villanueva, Maria del Pilar Villamil, and Mile Arnedo. Secure routing strategies in dht-based systems. *LNCS*, 6265:62–74, 2010.

[Wal03] Dan S. Wallach. A survey of peer-to-peer security issues. In *Software Security - Theories and Systems*, volume 2609 of *Lecture Notes in Computer Science*, pages 42–57. 2003.

[WTCT+08] Peng Wang, James Tyra, Eric Chan-Tin, Tyson Malchow, Denis Foo Kune, Nicholas Hopper, and Yongdae Kim. Attacking the Kad Network. *SecureComm*, pages 1–10, 2008.

[WZR08] A Walters, D. Zage, and C.N. Rotaru. A framework for mitigating attacks against measurement-based adaptation mechanisms in unstructured multicast overlay networks. *Networking, IEEE/ACM Transactions on*, 16(6):1434–1446, 2008.

[YK13] Amir Yahyavi and Bettina Kemme. Peer-to-peer architectures for massively multiplayer online games: A survey. *ACM Comput. Surv.*, 46(1):9:1–9:51, 2013.

[YLWL06] Zhongmei Yao, D. Leonard, Xiaoming Wang, and D. Loguinov. Modeling Heterogeneous User Churn and Local Resilience of Unstructured P2P Networks. *In Proc. ICNP*, pages 32–41, 2006.

[YMR14] A.A. Younis, Y.K. Malaiya, and I. Ray. Using attack surface entry points and reachability analysis to assess the risk of software vulnerability exploitability. In *High-Assurance Systems Engineering (HASE)*, pages 1–8, 2014.

[ZHS+04] B.Y. Zhao, Ling Huang, J. Stribling, S.C. Rhea, AD. Joseph, and J.D. Kubiatowicz. Tapestry: a resilient global-scale overlay for service deployment. *Selected Areas in Communications, IEEE Journal on*, 22(1):41–53, 2004.

[ZZC+11] Ren Zhang, Jianyu Zhang, Yu Chen, Nanhao Qin, Bingshuang Liu, and Yuan Zhang. Making eclipse attacks computationally infeasible in large-scale dhts. In *Performance Computing and Communications Conference (IPCCC), 2011 IEEE 30th International*, pages 1–8, 2011.