ConTED: Towards Content Trust for the Decentralized Web

Valentin Siegert
Distributed and Self-organizing Systems
Chemnitz University of Technology
Chemnitz, Germany
0000-0001-5763-8265

Arved Kirchhoff
Distributed and Self-organizing Systems
Chemnitz University of Technology
Chemnitz, Germany
0000-0002-6002-8414

Martin Gaedke
Distributed and Self-organizing Systems
Chemnitz University of Technology
Chemnitz, Germany
0000-0002-6729-2912

***Abstract*—The initiatives for redecentralization of the Web such as SoLiD aim to enhance users' privacy by enforcing transparency about the data used by Web applications. However, it is a challenge for a Web application acquiring data from third-party sources to trust data originating from many or even hidden parties. A decentralized web application requires to evaluate trust and take trust-aware decisions autonomously without relying on a centralized infrastructure. While many related trust models consider direct or reputation-based trust for making trust-aware decisions, in decentralized web applications content and context factors (called content trust) become critical due to the arbitrary number of potential data providers and the contextual nature of trust. Besides, the dynamic nature of the decentralized web necessitates trust-aware decisions that are made autonomously by the machine in a collaborative environment without further human intervention. To address these challenges, we present ConTED, a content trust evaluation framework for enabling decentralized Web applications to evaluate content trust autonomously. We also describe the architecture concept, which makes it feasible to integrate content trust models for decentralized Web applications. To demonstrate the feasibility, ConTED is integrated with aTLAS testbed, a web-based test bed to examine trust for a redecentralized web. Finally, we evaluate ConTED in terms of scalability and accuracy through a set of experiments.**

***Index Terms*—Trust, Decentralization, Web security and privacy, Web application modelling and engineering.**

I. Introduction

In recent years, a large portion of user data has been held in a small number of centralized online platforms dictated by large companies resulting in data silos and walled gardens [1]. This has raised severe privacy concerns [1], [2] and started the formation of Web redecentralization initiatives like SoLiD [2], or the EU's Next Generation Internet[1]. One way to achieve this redecentralization is by creating universal, open protocols and application interfaces, and allowing users to be independent of central authorities or entities [1]–[4]. SoLiD [2] is a linked data platform proposed for supporting the redecentralization of the web. Storing and handling personal data in decentralized pod-like structures as proposed by SoLiD enforces transparency over the data an application uses. This enhances user privacy by providing users with the information necessary for making an informed decision on their application usage.

[1]https://digital-strategy.ec.europa.eu/en/policies/next-generation-internet-initiative

One imminent challenge of the web's redecentralization is enabling decentralized web applications to interact trustworthily [1], [3]. SoLiD-based web applications integrate potentially large amounts of data stored in third-party pods, in-which the data's source might be unknown or hidden. Yet, the included data should not abuse the web application's functionality, threaten its security or compromise its trustworthiness. In the current web, interaction partners' trust relationships are controlled and created by central authorities, which consider a third party trustworthy based on predetermined artifacts or human-given permissions [3].

Accordingly, decentralized web applications need an own *trust evaluation* and *trust-aware decision making* [5]. Trust models based on direct trust use directly observed or experienced historical interaction outcomes as evidence for evaluating the trustworthiness of potential interaction partners [5], [6]. However, this means that trust testimonials made by other participants in the network are not taken into account. In contrast, reputation-based trust describes a form of trust management based on trust testimonials [3], [6]. Many related trust models in the literature [7]–[10] make use of a combination of direct and reputation-based trust. Due to an arbitrary number of potential interaction partners in the decentralized web, an autonomous solution is required for determining whether a given piece of information should be trusted or not [3]. Additionally, decentralized web applications have to handle trust dynamically as a change of foreign data may impacts previously established trust relationships [3]. Using a trust model beyond direct or reputation-based trust by also analyzing content and context [11] potentially improves the trust evaluations and thus the trust-based decision making.

The literature highly acknowledges contentual and contextual trust factors as in the work presented by Gil and Artz [11], which trust factors harmonize with the ones of the survey from Granatyr et al. [6]. However, Gil and Artz focus on search engines where the target group is intended for users who should finalize the pre-evaluated content trust and its subsequented trust-aware decisions. Given the importance of decentralied Web applications and the advantages provided by contentual and contextual trust factors, the challenge lies in the integration of contentual trust within decentralized Web applications for autonomous trust evaluation. In particular,:

(1) the consideration of autonomy and dynamics of trust relationships leads to additional calculations depending on the number of Web applications, which can take a significant amount of time and computational power, (2) without a human user in the loop for finalizing the trust evaluation, the accuracy of the trust calculations becomes critical due to the lack of an external entity, (3) depicting an architecture on how to integrate content trust evaluations and trust-aware decision making into web applications.

Therefore, we propose ConTED, a **Con**tent **T**rust **E**valuation framework for the **D**istributed web, which enables autonomous content trust for decentralized web application. ConTED evaluates a final trust value per received message communicated between Web applications to make a trust-aware decision possible. To the best of our knowledge, no previous work has incorporated content trust for decentralized web applications. We implemented ConTED prototypically[2] with focus on the trust evaluation itself. Our contributions are:

1) We present an autonomous content trust evaluation framework for decentralized web applications, which is independent on the used trust scale to subsequently make a trust-aware decision.
2) We implement a prototype[2] of our solution within aTLAS [3], [12] which is a web-based testbed to examine trust for a redecentralization of the web.
3) Finally, we conduct extensive evaluations within a realistic experiment setup to demonstrate ConTED's scalability and accuracy.

The remainder of the paper is structured as follows. The concept of ConTED with architecture, process, and factor realization is presented in section II. Section III details the evaluations for ConTED performed on aTLAS [3], [12]. The related work on trust evaluation models is presented in Section IV. Finally, section V concludes the paper and provides a prospect on the future work.

II. ConTED

In this section we first describe its architecture and where to place ConTED within a web application. Subsequently, we introduce ConTED's trust evaluation process.

Trust awareness means to make a trust-aware decision on whom or what to trust based on a trust evaluation. For Content Trust, we thus take the received resource or message, and the interaction partner's meta data as input to result in aforesaid decision. ConTED has to be placed between the HTTP message receiving and the web application's controller processing it. The message is required to include a resource from e.g. a SoLiD pod, and ConTED then helps the controller to only include trustworthy received resources. Figure 1 shows such a framework with ConTED as the trust evaluation component and the remaining components as a possible structure to provide and receive data for and from ConTED. ConTED needs input from the trust environment to be seen at (1), and input from three analysis components given at (2). Out

[2]available at https://vsr.informatik.tu-chemnitz.de/projects/2020/atlas/

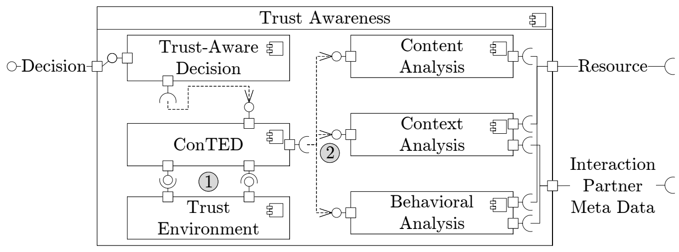


Fig. 1: Trust Awareness with ConTED

of all this information it presents the final trust value as well as the updated cooperation threshold to the *Trust-Aware Decision* component, which makes the decision to finish the trust awareness process.

The trust environment represents a data structure or exchange approach containing recommendations from other web applications regarding the evaluated resource, as well as the evaluation history and trust preferences of the web application. The evaluation history contains the results for past trust evaluations. It tracks the final trust value and the values for every implemented content trust factor, as well as data for identifying and potentially contacting the web application that provided the resource. To adapt the trust evaluation as needed, a set of trust preferences is used. ConTED implements a simple configuration system which allows an application to decide beforehand, which content trust factors should be used within the evaluation and how they should be weighted towards each other. This configuration also allows to specify the mode of operation and the assessment standard of some factors by applications' owners and is called *trust preferences*.

The three analysis components deliver information based on the resource and interaction partner initiating the trust awareness. While the *content analysis* is required to parse the resource based on content factors like topics, authors, or publication date, it also is required to analyse the specificity and the likelihood of being correct. The *context analysis* serves the meta data about the interaction partner as well as the identified criticality level, and the *behavioral analysis* gives ConTED input on bias, deception and incentive.

The final trust value calculated by ConTED is a weighted average of the content trust factor values:

$$final\,trust(R) = \frac{\sum_{n=1}^{c}(w(f_{R,n}) * v(f_{R,n}))}{\sum_{n=1}^{c} w(f_{R,n})}$$

where $w(f_{R,n})$ is the weight of the n-th factor about the resource R and $v(f_{R,n})$ is its value. This allows fine-tuning of the resulting trust value according to application-specific weights. By default, the weights are all set to 1 if not changed by an expert setting up ConTED in his web application. All values, be it the final or factor sub-values, are expressed on the trust scale by Marsh and Briggs [13].

The factors are implemented based on the definitions in the work of Gil et al. [11]. Figure 2 illustrates the process of evaluating trust and indicates the data flow during the evaluation. Beginning at the starting event *Message received*,

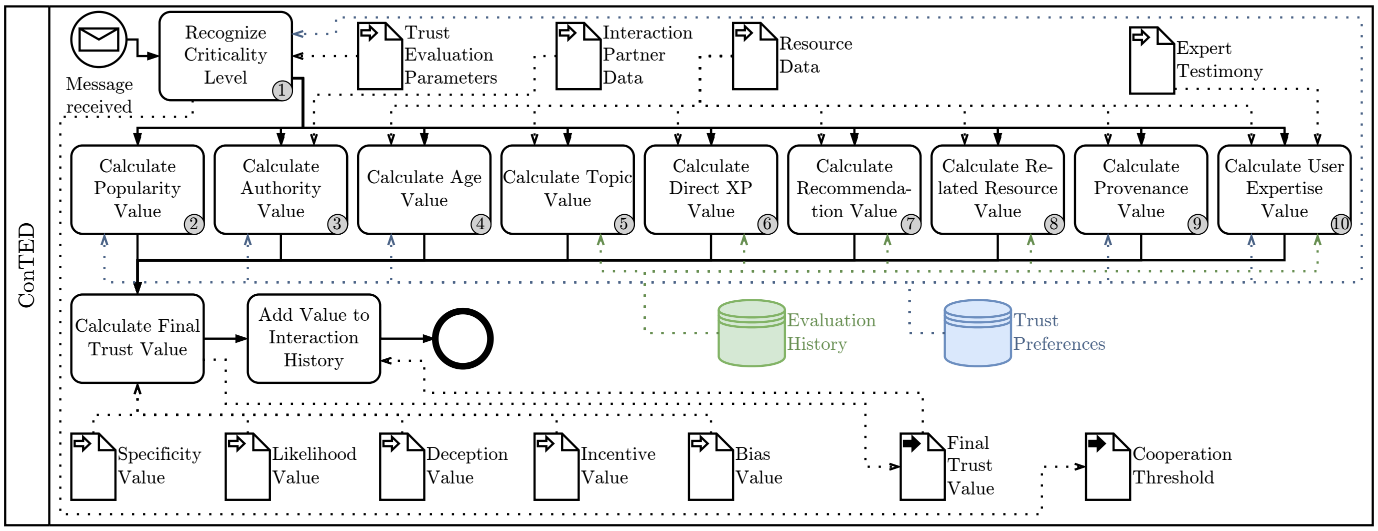

ConTED
Message received
Recognize Criticality Level
1
Trust Evaluation Parameters
Interaction Partner Data
Resource Data
Expert Testimony
Calculate Popularity Value
2
Calculate Authority Value
3
Calculate Age Value
4
Calculate Topic Value
5
Calculate Direct XP Value
6
Calculate Recommendation Value
7
Calculate Related Resource Value
8
Calculate Provenance Value
9
Calculate User Expertise Value
10
Calculate Final Trust Value
Add Value to Interaction History
Evaluation History
Trust Preferences
Specificity Value
Likelihood Value
Deception Value
Incentive Value
Bias Value
Final Trust Value
Cooperation Threshold

Fig. 2: Trust evaluation process in ConTED

the trust evaluation starts after a resource has been sent by an interaction partner and was parsed by the underlying framework. This data, as well as information about the interaction partner, is represented as *Resource Data* and *Interaction Partner Data* respectively.

In the following we present all content trust factors implemented in ConTED. Those factors being represented in figure 2 with a task are highlighted in the figure with the number in parentheses following their names in the text.

Recency prevents the usage of obsolete data, which is on the evaluation history. It is implemented by applying an age filter which simulates "forgetting" evaluation results after a certain time. The maximum age for interaction entries is defined as recency limit within the trust preferences. This age filter is part of the evaluation history and therefore not explicitly included in figure 2.

Context and criticality (1) does not produce a trust value, but changes the cooperation threshold of the interaction based on the criticality parameter provided during the trust evaluation call. The respective values for each level of criticality are configured in the trust preferences. Such that the cooperation threshold can be used in other factors in the current context, this factor is preliminary to all others.

Popularity (2) calculation may differs in its process. For our prototype, we used a simplified, machine-accessible social network. For each resource that is being evaluated, the evaluating application is sending requests to its peer group to find out how popular the resource is in a group of others that are assumed to have similar requirements and cultural imprints. Each peer then checks its evaluation history for the specified resource and returns a boolean value that signals whether its history contains at least one entry for this resource that resulted in a trust value above the peer's cooperation threshold and does not exceed the recency limit. After collecting the messages from its peers, the popularity value calculates as the proportion of peers that responded with *true*. The resulting popularity value will be within the interval $[0, 1]$ to ensure that resources are not penalized too heavily for their lack of popularity, as it is not necessarily an indication of high information quality. It should be noted further, that it does not matter whether a peer actually interacted with the resource, only the final result of the trust evaluation is important in this step.

Authority (3) determines whether a resource was shared by a trusted authority. It receives an identifier from the interaction input, as well as a list of trusted authorities that have been configured in the trust preferences. If the interaction partner is considered a trusted authority, the authority value is set to 1, otherwise the factor is omitted from the final trust value calculation.

Age (4) operates in two different modes: (1) resources are rewarded if they were released recently and neither rewarded or penalized for exceeding the configured maximum resource lifetime, or (2) for resources that have exceeded the maximum lifetime, the trust evaluation will be skipped and the final trust value is set to -1. While the first mode is used for evaluating resources in e.g. an academic setting where recent resources are often more accurate than historic ones, the second mode is used for use cases in which resources lose their validity after a certain time, e.g. weather reports for the current week are not valid anymore two weeks later. Regardless of which mode of operation is used, an Age Value is calculated. This value is assigned 1, if the lifetime of the resource has not yet exceeded the configured maximum. Otherwise, a configured *grace* timespan is applied. If x represents the number of seconds by which the maximum lifetime has been exceeded and the grace timespan is given in seconds, it is calculated as $Age = max(1 - \frac{x}{grace}, 0)$.

Topic (5) calculates the mean average of topic trust values that the truster has towards the trustee regarding all topics detected in the resource. Topic trust is composed of two kinds of values, both aggregated and used for the Topic value calculation: (1) historic final evaluation results for the same trustee with similar resource topics, and (2) predefined topic trust values for others which are meant to be used in environments with few attempted interactions. It should be noted, that ConTED expects the content analysis component to correctly detect all topics that are contained in the resource. Alternatively, should the detection of topics with absolute certainty not be possible, each topic could be provided and later be stored with an associated confidence value by which the respective trust score saved in the evaluation history can be weighted for calculating the Topic value as weighted average instead.

Direct Experience (6) is derived from the local evaluation history by calculating the average of all entries.

Recommendation (7) describes, similarly to reputation-based trust models, a mechanism which aggregates and analyzes third-party testimony to derive a statement about the resource‘s trustworthiness. ConTED receives witness testimony based on the evaluation history of other participants of the network, where the others have to be above the updated cooperation threshold updated in Context and Criticality. Each witness forms its testimony by calculating the mean of the final trust values for the currently evaluated resource. This testimony is filtered by first weighting each response with the trust value for the respective application, and then eliminating statistical outliers by using a median approach for calculating the resulting average instead of the arithmetic mean.

Related Resources (8) describes that the trustworthiness of a resource is influenced by the given references. This content trust factor is flawed because giving a reference to a high quality resource does not guarantee a high information quality or trustworthiness of the referencing resource. However, resources that cite sources that are deemed reputable are considered more trustworthy than resources that are based on dubious or no references at all. The corresponding trust value for this factor is based on a list of referenced resources provided by the *Resource Data* that are then evaluated based on the evaluation history to determine which references should be considered reputable.

Provenance (9) is implemented similarly to the calculation of the authority value. First, the task receives identifiers for the original resource authors, as well as a list of configured trusted sources. If one or more of the resource authors are trusted, the Provenance value is set to 1, otherwise this factor is omitted from the final trust value calculation. The interaction partner may not be the original resource author, but instead e.g. an information aggregator or social network component that hosts the requested resource.

User expertise (10) aims at aggregating expert testimony about the evaluated resource. To achieve this, the task requires the topics and the identifier of the resource, additionally to the local evaluation history. During the next step the resources with similar topics to the one currently evaluated are chosen, if their trust values are above the current cooperation threshold. For these trusted resources, the saved original authors are gathered and their witness testimony regarding the currently evaluated resource is requested. Finally, the user expertise value will be output as the average of the received expert opinions. The detected resource authors may be applications that can be contacted via the network, but they could also be natural entities that have no direct interface to the network. If an author can not be contacted autonomously via the network, their expert opinion will be excluded from the evaluation.

Some content trust factors are not directly calculated within ConTED itself, but derived by other components of the *Trust Awareness* system and represent as additional input data in figure 2. These components are not further discussed in this paper but will be mocked by aTLAS [3], [12] for the evaluation. The *Behavioral Analysis* component provides the respective values for the content trust factors **Bias**, **Deception**, and **Incentive**. The values for **Specificity** and **Likelihood** are supplied by the *Content Analysis* component. While most of these values are simply injected directly into the final trust value calculation, the Deception value is used differently to counter active lying. A resource is deemed deceptive if the provided value is above the deception threshold that is allowed for the current criticality level by the trust preferences. If an attempted deception is detected, instead of using the deception value during the final trust value calculation, the current trust evaluation is skipped and the final trust value is assigned -1.

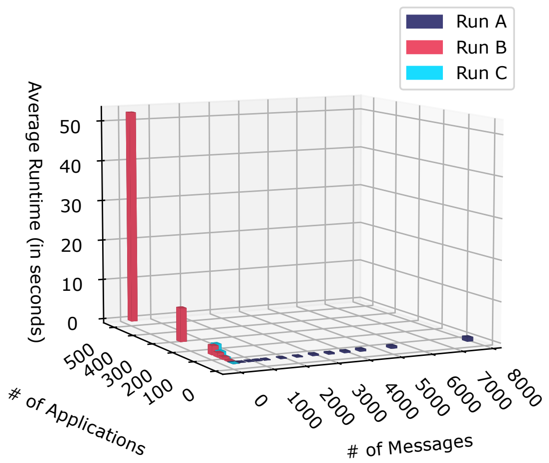
Based on the use case context of web applications, not every content trust factor was implemented in ConTED. Three factors were excepted from the trust evaluation: *Limited Resources*, *Agreement*, and *Appearance*. Limited Resources and Agreement are part of the trust-aware decision and thus are not part of ConTED. While Appearance may affect perceived trustworthiness of a resource by a user, it is not relevant for the autonomous communication between web applications.

III. Evaluation

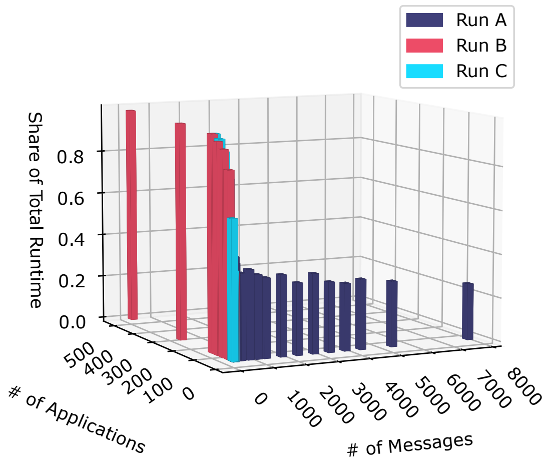
In this section, we explore the scalability by scaling web applications as well as exchanged messages, and determine the accuracy of ConTED based on the defined metric by Jelenc et al. [14]. We execute the experiments all on our ConTED prototype[2] within aTLAS [3], [12]. The additional required input components for ConTED will be mocked in the test scenarios for the purpose of this paper.

A. Scalability

We carried out a series of scalability experiments of ConTED within aTLAS [3], [12]. The experiments were carried out on up to 17 hosts having memory sizes of 4, 8, 16 or 32 GB and using on of the following CPUS: AMD Ryzen 7 5800X, AMD A8-6500, Intel Xeon E5-2620, Intel Core i7-4770, or Intel Core i7-7700. Per host one supervisor is executed, and one of the hosts runs in parallel the aTLAS server, which distributes the applications evenly over all available supervisors. To measure the runtime of each trust evaluation, an aTLAS built-in function is used to record the time elapsed during the current trust evaluation. The execution time of the whole scenario is measured by this built-in as well, and can be used to calculate the runtime share of the trust evaluation time. To mitigate the statistical effect of fluctuations based on



(a) Average Runtime



(b) Runtime Share

Fig. 3: ConTED's Scalability

processor scheduling or CPU load, each scenario is run ten times and the arithmetic mean is calculated.

To test a range of varying combinations of applications and messages exchanged between them, three test runs are prepared. Each test run uses every implemented content trust factor and consists of scenarios that are randomly generated based on fixed application and exchange message size constraints. All scenarios scaled up until aTLAS current implementation came to issues of handling big scenario data, which was noted by these tests and will be addressed in future work.

Test Run A fixes the amount of web applications at 10, while it scales up the exchanged messages with the values 5, 10, 25, 50, 75, 100, 250, 500, 750, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 5000, 7500.

Test Run B fixes the amount of exchanged messages at 10, while it scales up the web applications with the values 5, 10, 25, 50, 75, 100, 250, 500.

Test Run C is the combination of test run A and B and thus scales up both values in the same steps, but with the values 5, 10, 25, 50, 75, 100.

The scalability test results are displayed in figure 3, while the raw data of the scalability tests are accessible online[2]. As subfigure 3a shows, the average runtime per trust evaluation scaled by a factor of 26.23, while the messages scaled by a factor of 1500. In stark contrast to this, test run B's average runtime scaled by a factor of 1922, while the web applications scaled by a factor of 100. Finally, test run C's average runtime grew by a factor of 75.33, while the messages and the applications scaled by 20. This similar behavior of B and C shows, that the number of applications in a scenario has a more significant impact on the average execution time of trust evaluations than the number of messages. The growing number of required network connections due to more applications to be queried is demonstrably more time-consuming than running many messages within one scenario. Subfigure 3b highlights this finding again in the view of the trust evaluations runtime share. Our scalability tests show that the impact of the past messages on the runtime is low in comparison to the amount of other applications. Additionally, the overall timings are promising but showcase the need for a limitation of interaction with other applications during the trust evaluation as Test Run B has a high average runtime at 500 applications.

B. Accuracy

The metric defined by Jelenc et al. [14] can be used to evaluate trust models that do not have a decision making mechanism, which is the case for ConTED. In short, *Accuracy* expresses the similarity between the ranking of web applications by trust value as calculated by the evaluated trust model, and the ranking of web applications by capability. The capability of an application describes its quality as interaction partner, which is precisely the value that the trust evaluation process attempts to estimate. The capabilities are a predefined synthetic value for the accuracy evaluation and do obviously not exist in practice as otherwise the trust evaluation would be trivial. Using an already established metric allows for a comparison of ConTED with different existing trust models based on the results presented by Jelenc et al. [14] and their Alpha testbed.

To mitigate the Alpha testbeds focus on one truster called agent α, we use aTLAS scenarios with exactly one truster application, called A. In addition, 50 trustee applications with randomly generated capabilities are added to each scenario, and A does not know the capabilities. All applications in the scenario are connected to each other and have an opinion about others. To simulate each web application providing the same service, which is posed by Jelenc at al. as another requirement, the message itself, as well as the topics of the message are similar between all messages exchanged in the scenario. Every trustee in the scenario sends within one message cycle exactly one message to A, which then computes the trust value for this message and writes it to the evaluation history. This data is extracted and the Accuracy for the current cycle is calculated based on the application capabilities.

While the Alpha testbed generates opinions on the fly, the opinions used in this evaluation are based on history data specified in the aTLAS scenario. To generate history data that is based on the capability of one web application, but contains sufficient noise to be realistic, a pseudo-random generator based on a truncated normal distribution is used. This normal distribution is parameterized with the capability and an arbitrary standard deviation. The size of the standard deviation is responsible for the range of the resulting noise in the generated data, and for the entirety of the evaluation it is $\sigma = 0.10$. Jelenc at al. [14] use the same deviation for generating interaction outcomes but lower it to $\sigma = 0.05$ for generating opinions. However, in our evaluation, opinions are calculated based on the existing history, as described in section II for the recommendation content trust factor. All provided content trust factor values (e.g., Bias, Specificity, Likelihood, etc.) are randomly generated.

As ConTED allows customization, we used four scenarios to evaluate the accuracy, which differ from each other in the following aspects:

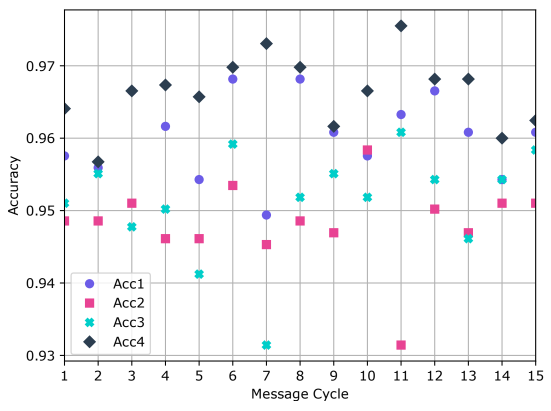
Scenario Acc1: It uses all content trust factor of ConTED for its trust evaluation, but there are no authorities, trusted authors or trusted topics configured. Each service provider specifies only its own name as author and does not give any references to related resources. This simulates a situation in which the application does not know its peers and the network does not contain any authorities known to A.

Scenario Acc2: To test how including more content trust factors affects the resulting accuracy, A uses in this scenario only the content trust factors Direct Experience and Recommendation. Each message contains a Specificity value, which is supposed to give the model per message input, similar to the effect of an interaction experience value of the Alpha testbed.

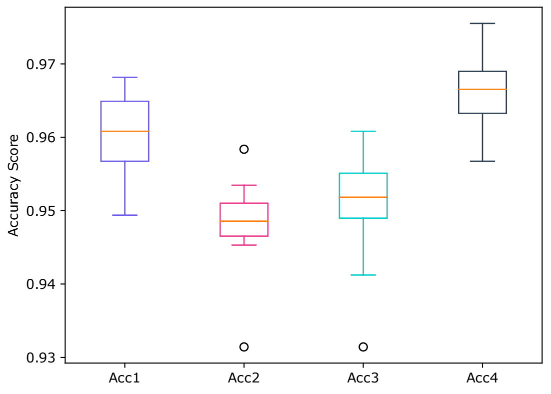
Scenario Acc3: Each content trust factor of ConTED is in use. In the configuration of A, 5 randomly selected applications are assigned as authorities and the 5 applications with the highest internal capabilities are set as trusted authors. Finally, the described pseudo-random number generator is used to assign A's topic trust values for each trustee in the scenario. This scenario simulates an advanced and more dynamic situation with possibly untrustworthy authorities, already formed topic trust values and well chosen trusted authors.

Scenario Acc4: This scenario is mostly equivalent to scenario Acc3, with the only difference being that no authorities are chosen. Having no artificially created hurdles, this scenario should be considered the main benchmark for evaluating ConTED's Accuracy.

The resulting dataset of the *Accuracy* evaluation is displayed in figure 4. Subfigure 4a shows a zoomed in view of the *Accuracy* over time in message cycles on a granular y-scale. In addition to the graphical representation of the *Accuracy* scores over time, subfigure 4b illustrates the results of the experiment as boxplot. ConTED evaluates trust with high *Accuracy*, which is supported by the results of the evaluation experiment. It is evident that scenario Acc4 is the scenario for which the model achieved the highest Accuracy. ConTED shows the lowest *Accuracy* for scenario Acc2. Based on the results of the trust models examined by Jelenc et al. [14], ConTED is in its prototypical state, and under the circumstances and assumptions of the conducted experiments, more accurate than the trust models examined by Jelenc et al. in 2013, which are Beta Reputation [8], Travos [15], EigenTrust [10], the models from Yu et al. [16], and from Abdul-Rahman and Hailes [7].



(a) Accuracy over time in message cycles



(b) Accuracy per scenario

Fig. 4: ConTED's Accuracy

IV. Related Work

There is no existing work that considers contentual trust factors for decentralized applications as ConTED is first of it's kind, but several trust models exist to tackle specific use cases of trust relationships or to establish trust with a specific approach [5], [6]. To systematically analyze the trust models, we identify a set of four requirements based on the motivation and problem presented in the Introduction. These requirements are: *decentralization*, *autonomy*, *situational awareness* and numerical *trust scale*. The requirements are used to assess the related work. The trust models are selected based on being well-known in the literature of trust models or as they are recently published. The trust models in the literature can be categorized according to Direct, Reputation, Socio-Cognitive and Organization Trust, as also acknowledged in the literature [5]. All requirements are mapped onto a four-level assessment scheme: not satisfied ○, partially satisfied ◔, mostly satisifed ◕, fully-satisfied ●.

Direct trust-based models aggregate and analyze historical interaction outcomes to collect evidence for the trustworthiness

TABLE I: Qualitative comparison between trust models

	Trust Model	D	A	S	T
Direct	Jiang et al. [17]	●	◕	○	●
	SD-TDQL [18]	●	●	○	●
Reputation	Abdul-Rahman et al. [7]	●	●	◕	◔
	Alemneh et al. [19]	●	◕	◔	●
	Azad et al. [20]	◔	◕	◕	○
	Beta Reputation [8]	●	◕	◕	●
	DTMS [21]	◔	●	◕	●
	EigenTrust [10]	◔	◕	○	◔
	FIRE [9]	●	●	◔	●
	Rathee et al. [22]	●	●	◔	●
	Tm-IIoT [23]	○	●	◕	●
	Travos [15]	●	●	◕	◔
	Yu et al. [16]	●	●	◕	●
Socio-Cognitive	DiffTrust [24]	◔	◕	◕	●
	Mathas et al. [25]	○	◔	◕	●
	ReGreT [26]	◔	◕	◕	●
Organization	Ibáñez et al. [1]	●	●	-	-
	Hermoso et al. [27]	○	●	○	◔
Content	**ConTED**	●	◕	●	●

D, A, S, T respectively stand for **D**ecentralization, **A**utonomy, **S**ituational awareness, **T**rust scale.

of a potential interaction partner [5]. The type of collected evidence depends on the trust model. For example, SD-TDQL [18] logs the experienced network packet forwarding behavior of a communication partner. The model presented by Jiang et al. [17], however, focuses on the general interaction success. The techniques used for aggregating and comparing the evidence vary.

Reputation-based trust models describe the aggregation of trust-related application information based on recommendations, opinions and witness testimonies given by third-party applications [6]. However, witness information is less reliable than direct experience [5], [6] and a number of different attack scenarios exploit the reliance on other applications [19]. For this reason, trust models often combine direct and reputation-based evidence using sophisticated filtering methods like a subjective logic-based system [19], or a stateful decision tree [22]. Older and commonly known trust models like Beta Reputation [8], EigenTrust [10], Travos [15] and the two models from Abdul-Rahman et al. [7] and Yu et al. [16] bring in different ways to actively counter liars. They propose also different types of reputations and how to emerge them, or even add popularity scores as FIRE [9] does with its certified reputations. Some models of this group are designed to solve additional problems like protecting the privacy of the truster [20], storing testimony irreversibly while keeping the trust evaluation transparent [21] or effectively handling a large number of devices [23]. From table I, it is clear that reputation-based trust is the prevalent trust management solution in recent publications [19]–[23], despite the existing attack vectors and their lack of contentual factors.

Socio-cognitive trust models analyze the intrinsic properties of their potential interaction partners, as well as external factors that are likely to allow conclusions about the future behavior of an application [5]. They utilize sociological information mostly acquired through social network analysis, stereotypical assumptions or prejudice [5], [6]. In their trust model, Mathas et al. [25] demonstrate the use of prejudice and social network analysis, while DiffTrust [24] applies the social diffusion theory with focus on social proximity. The commonly known ReGreT [26] in contrast works with fuzzy rules to determine social relationships in the three categories of witness, neighbour and system reputations. However, it stays unclear on how to create the social relations without a user, as even the most known approach from this group, ReGreT, expects the social information to be given within the system. This makes them inappropriate for the decentralized web in terms of decentralization and autonomy.

Organizational trust models create and maintain trust by introducing an organizational structure that applications delegate their trust evaluation to [5]. The trust model by Hermoso et al. [27] works by generating a central coordination artifact that aggregates trust values from an underlying direct trust-based model and groups the members of the network into roles according to the data provided by its peers. In contrast to this, Ibáñez et al. [1] present a trust model based on smart contracts stored in a distributed ledger. Thus, applications do not have to trust each other, but trust the validity and functionality of the distributed ledger instead. Most organizational trust models undermine the principal of decentralization completely. The idea of using an distributed ledger is solving the decentralization issue, but does not support the missing situational awareness of such trust models.

Most of the analysed approaches consider decentralization, autonomy and the usage of a numerical trust scale. However, in terms of autonomy, these models do not consider how trust is initialized in the starting point when there is no trust between the Web applications. Therefore, it remains also questionable for the analysed models how a new Web application can join an existing network of trust relationships or how to introduce the model in the beginning. From table I, it is also evident that none of the existing solutions consider the situational awareness requirement fully as none consider the contentual information of an interaction during the trust evaluation. Many of the trust models perform an extensive context analysis, e.g. behavior analysis or meta-analysis of ratings, to counter especially active lying, but only Abdul-Rahman et al. [7] differentiates between different services, as ConTED does by distinguishing the web resources. In contrast to the existing solutions, ConTED integrates contentual factors into the trust evaluation. Further, none of the trust models consider their integration within a web application, as they are designed for Multi-Agent Systems or a very specific scenario without a clear integration into the web stack. One way of realising trust with contentual factors are the content policies of Wikipedia [28], however alot of effort is required by editors to establish policies, which is not suitable for making autonomous trust aware decisions.

In contrast, our solution aims at performing trust evaluations for decentralized web applications by considering decentralization, autonomy, situational awareness and a numerical trust scale. Moreover, we have not found any approach that considers contentual trust for decentralized web applications. With ConTED we solve this gap by creating a framework, which makes use of the content trust model described by Gil and Artz [11] and which can be integrated into the message flow of a web application. It serves as a framework for web applications within the decentralized web due to its decentralized design and especially can integrate linked data as in the data distribution model of SoLiD [2]. Due to the content trust factors it is ahead of situational awareness including not only contextual information as several trust models do already, but also contentual information.

V. Conclusion

Making trust-aware decisions that consider content trust factors can enhance trust in the redecentralized Web. In this paper, we introduced ConTED for decentralized Web applications to interact trustworthily. It supports the trust awareness of decentralized web applications with a focus on the realization of contentual and contextual factors. ConTED aims to provide a final trust evaluation and an updated cooperation threshold, thus executing the trust evaluation. We demonstrated its feasibility by implementing it prototypically[2] within aTLAS [3], [12], and evaluated its scalability and accuracy within a realistic experiment setup.

Overall, the evaluation suggested that ConTED can be helpful for evaluating trust-aware decision for decentralized Web applications. In the future work, we plan to work on the content, context and behavioral analysis components. Those components will only extend the required time as well as decrease the accuracy of the overall trust evaluation due to imperfect realizations. Conceptualizing them in the, for the accuracy and scalabiltiy of the model, least detrimental manner will be the subject of our next steps. Additionally, a trust-aware decision making component is in the future required to finalize the trust awareness process and thus making web applications fully capable of trustworthy interactions with ConTED. Furthermore, it remains to be clarified in the future how ConTED approaches the initialization of trust or how different trust preferences are handled more autonomously in their setup. Unfortunately, in its current state ConTED requires a lot of setup information to initialize a running state, which creates the need for an automatic setup in the future. This includes the fact that ConTED is currently open to many trust preferences, including the possibility to change the weights of the final trust's weighted sum. In summary ConTED improves content trust awareness for decentralized web applications.

References

[1] L. D. Ibáñez *et al.*, "Redecentralizing the web with distributed ledgers," *IEEE Intelligent Systems*, vol. 32, no. 1, pp. 92–95, jan 2017.

[2] A. V. Sambra *et al.*, "Solid: A Platform for Decentralized Social Applications Based on Linked Data," MIT CSAIL & Qatar Computing Research Institute, Tech. Rep., 2016.

[3] V. Siegert, M. Noura, and M. Gaedke, "aTLAS: A testbed to examine trust for a redecentralized web," in *Proceedings of WI-IAT 2020*, 2020, pp. 411–416.

[4] M. Noura, V. Siegert, and M. Gaedke, "Wat: Autonomous hypermedia-driven web agents for web of things devices." in *Proceedings of the All the Agents Challenge co-located with the 20th International Semantic Web Conference*, 2021, pp. 38–43.

[5] H. Yu *et al.*, "A Survey of Multi-Agent Trust Management Systems," *IEEE Access*, vol. 1, pp. 35–50, 2013.

[6] J. Granatyr *et al.*, "Trust and Reputation Models for Multiagent Systems," *ACM Computing Surveys*, vol. 48, no. 2, pp. 1–42, oct 2015.

[7] A. Abdul-Rahman and S. Hailes, "Supporting trust in virtual communities," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*. IEEE, 2000.

[8] A. Jøsang and R. Ismail, "The beta reputation system," in *Proceedings of the 15th Bled Electronic Commerce Conference*, 2002, pp. 2502–2511.

[9] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt, "Fire: An integrated trust and reputation model for open multi-agent systems," in *ECAI 2004: 16th European Conference on Artificial Intelligence*, 2004, pp. 18–22.

[10] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *WWW '03: Proceedings of the 12th international conference on World Wide Web*, 2003, pp. 640–651.

[11] Y. Gil and D. Artz, "Towards content trust of web resources," *Journal of Web Semantics*, vol. 5, no. 4, pp. 227–239, dec 2007.

[12] V. Siegert and M. Gaedke, "Wta: Towards a web-based testbed architecture," in *International Conference on Web Engineering*, 2021, pp. 115–123.

[13] S. Marsh and P. Briggs, "Examining Trust, Forgiveness and Regret as Computational Concepts," in *Computing with Social Trust*, J. Golbeck, Ed. Springer, 2009, pp. 9–43.

[14] D. Jelenc *et al.*, "Decision making matters: A better way to evaluate trust models," *Knowledge-Based Systems*, vol. 52, pp. 147–164, nov 2013.

[15] W. T. L. Teacy *et al.*, "Travos: Trust and reputation in the context of inaccurate information sources," in *Autonomous Agents and Multi-Agent Systems*, vol. 12, 3 2006, pp. 183–198.

[16] B. Yu, M. P. Singh, and K. Sycara, "Developing trust in large-scale peer-to-peer systems," in *IEEE 1st Symposium on Multi-Agent Security and Survivability*, 2004, pp. 1–10.

[17] S. Jiang, J. Zhang, and Y. S. Ong, "A Multiagent Evolutionary Framework based on Trust for Multiobjective Optimization," in *Proceedings of AAMAS'12*, vol. 2, 2012, pp. 299–306.

[18] D. Zhang *et al.*, "Software-Defined Vehicular Networks With Trust Management: A Deep Reinforcement Learning Approach," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, 2020.

[19] E. Alemneh *et al.*, "A two-way trust management system for fog computing," *Future Generation Computer Systems*, vol. 106, pp. 206–220, 2020.

[20] M. A. Azad *et al.*, "Decentralized Self-Enforcing Trust Management System for Social Internet of Things," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2690–2703, apr 2020.

[21] X. Chen, J. Ding, and Z. Lu, "A Decentralized Trust Management System for Intelligent Transportation Environments," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2020.

[22] G. Rathee *et al.*, "A trust management scheme to secure mobile information centric networks," *Computer Communications*, vol. 151, no. December 2019, pp. 66–75, feb 2020.

[23] C. Boudagdigue *et al.*, "Trust Management in Industrial Internet of Things," *IEEE Transactions on Information Forensics and Security*, vol. 15, no. 4, pp. 3667–3682, apr 2020.

[24] H. Fang, J. Zhang, and N. M. Thalmann, "A Trust Model Stemmed from the Diffusion Theory for Opinion Evaluation," in *Proceedings of AAMAS'13*, 2013, pp. 805–812.

[25] C.-M. Mathas, C. Vassilakis, and N. Kolokotronis, "A Trust Management System for the IoT domain," in *2020 IEEE World Congress on Services (SERVICES)*. IEEE, oct 2020, pp. 183–188.

[26] J. Sabater and C. Sierra, "Reputation and social network analysis in multi-agent systems," in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems part 1 - AAMAS '02*. ACM Press, 2002, pp. 475–482.

[27] R. Hermoso, H. Billhardt, and S. Ossowski, "Role Evolution in Open Multi-Agent Systems as an Information Source for Trust," in *Proceedings of AAMAS'10*, 2010, pp. 217–224.

[28] P. Ayers, C. Matthews, and B. Yates, *How Wikipedia works: And how you can be a part of it*. No Starch Press, 2008.